In [2]:
```
# import neccessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
```

In [3]:
```
# upload data:
df = pd.read_csv("F:\\archive (14)\\Instagram data.csv")
```

In [4]:
```
# To see the top five heads of the df
df.head(2)
```

Out[4]:

	Impressions	From Home	From Hashtags	From Explore	From Other	Saves	Comments	Shares	Likes	Profile Visits	Follows
0	3920	2586	1028	619	56	98	9	5	162	35	2
1	5394	2727	1838	1174	78	194	7	14	224	48	10

In [5]:
```
# To See the last five rows of data:
df.tail(2)
```

Out[5]:

	Impressions	From Home	From Hashtags	From Explore	From Other	Saves	Comments	Shares	Likes	Profile Visits	Follow
117	32695	11815	3147	17414	170	1095	2	75	549	148	21
118	36919	13473	4176	16444	2547	653	5	26	443	611	22

In [6]:
```
# To see the shape of data
df.shape
```

Out[6]:
```
(119, 13)
```

In [7]:
```
# To see the size of data
df.size
```

Out[7]:
```
1547
```

In [8]:
```
# To check the dtypes of data, there is all integers except caption and
Hashtags:
df.dtypes
```

Out[8]:
```
Impressions        int64
From Home          int64
From Hashtags      int64
From Explore       int64
From Other         int64
Saves              int64
Comments           int64
Shares             int64
Likes              int64
Profile Visits     int64
Follows            int64
Caption           object
Hashtags          object
dtype: object
```

In [9]:
```
# To get an infomation of data:
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 119 entries, 0 to 118
Data columns (total 13 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Impressions     119 non-null    int64
 1   From Home       119 non-null    int64
 2   From Hashtags   119 non-null    int64
 3   From Explore    119 non-null    int64
 4   From Other      119 non-null    int64
 5   Saves           119 non-null    int64
 6   Comments        119 non-null    int64
 7   Shares          119 non-null    int64
 8   Likes           119 non-null    int64
 9   Profile Visits  119 non-null    int64
 10  Follows         119 non-null    int64
 11  Caption         119 non-null    object
 12  Hashtags        119 non-null    object
dtypes: int64(11), object(2)
memory usage: 12.2+ KB
```

In [10]:
```
# To see all the columns in the data:
df.columns
```

Out[10]:
```
Index(['Impressions', 'From Home', 'From Hashtags', 'From Explore',
       'From Other', 'Saves', 'Comments', 'Shares', 'Likes', 'Profile Visits',
       'Follows', 'Caption', 'Hashtags'],
      dtype='object')
```

In [11]:
```
# To get a statistical summary of data:
df.describe().T
```

Out[11]:

	count	mean	std	min	25%	50%	75%	max
Impressions	119.0	5703.991597	4843.780105	1941.0	3467.0	4289.0	6138.0	36919.0
From Home	119.0	2475.789916	1489.386348	1133.0	1945.0	2207.0	2602.5	13473.0
From Hashtags	119.0	1887.512605	1884.361443	116.0	726.0	1278.0	2363.5	11817.0
From Explore	119.0	1078.100840	2613.026132	0.0	157.5	326.0	689.5	17414.0
From Other	119.0	171.092437	289.431031	9.0	38.0	74.0	196.0	2547.0
Saves	119.0	153.310924	156.317731	22.0	65.0	109.0	169.0	1095.0
Comments	119.0	6.663866	3.544576	0.0	4.0	6.0	8.0	19.0
Shares	119.0	9.361345	10.089205	0.0	3.0	6.0	13.5	75.0
Likes	119.0	173.781513	82.378947	72.0	121.5	151.0	204.0	549.0
Profile Visits	119.0	50.621849	87.088402	4.0	15.0	23.0	42.0	611.0
Follows	119.0	20.756303	40.921580	0.0	4.0	8.0	18.0	260.0

In [12]:
```
# To check the null values in the data or not, there is no null values so
treatment is not required;
df.isnull().sum()
```

Out[12]:
```
Impressions       0
From Home         0
From Hashtags     0
From Explore      0
From Other        0
Saves             0
Comments          0
Shares            0
Likes             0
Profile Visits    0
Follows           0
Caption           0
Hashtags          0
dtype: int64
```

In [13]:
```
# see the value counts of data with unstack
df.value_counts(ascending=True).unstack()
```

Out[13]:

Impressions	From Home	From Hashtags	From Explore	From Other	Saves	Comments	Shares	Likes	Profile Visits	Follows
1941	**1466**	**411**	**37**	**17**	**49**	**6**	**3**	**82**	**8**	**2**

Impressions	From Home	From Hashtags	From Explore	From Other	Saves	Comments	Shares	Likes	Profile Visits	Follows	
2064	1304	362	249	37	49	4	5	76	9	0	
2191	1308	809	45	18	35	2	1	72	18	0	
2218	1597	411	162	15	28	6	3	81	29	4	
2327	1774	435	59	35	45	3	3	85	7	2	

Impressions	From Home	From Hashtags	From Explore	From Other	Saves	Comments	Shares	Likes	Profile Visits	Follows	
...	...	...	...	...	...	...	...	...	...	...	
16062	3144	11817	564	468	252	6	20	416	330	94	
17396	1817	10008	5192	251	285	7	7	416	467	260	
17713	2449	2141	12389	561	504	3	23	308	70	96	
32695	11815	3147	17414	170	1095	2	75	549	148	214	
36919	13473	4176	16444	2547	653	5	26	443	611	228	

102 rows × 54 columns

In [14]:
```
# see unique values in the data
df.nunique()
```

Out[14]:
```
Impressions       101
From Home          97
From Hashtags     100
From Explore       95
From Other         84
Saves              84
Comments           15
Shares             28
Likes              85
Profile Visits     59
Follows            29
Caption            90
Hashtags           54
dtype: int64
```

In [15]:
```
# To get the two columns together to do comparison:
df[['Impressions', 'Likes']]
```

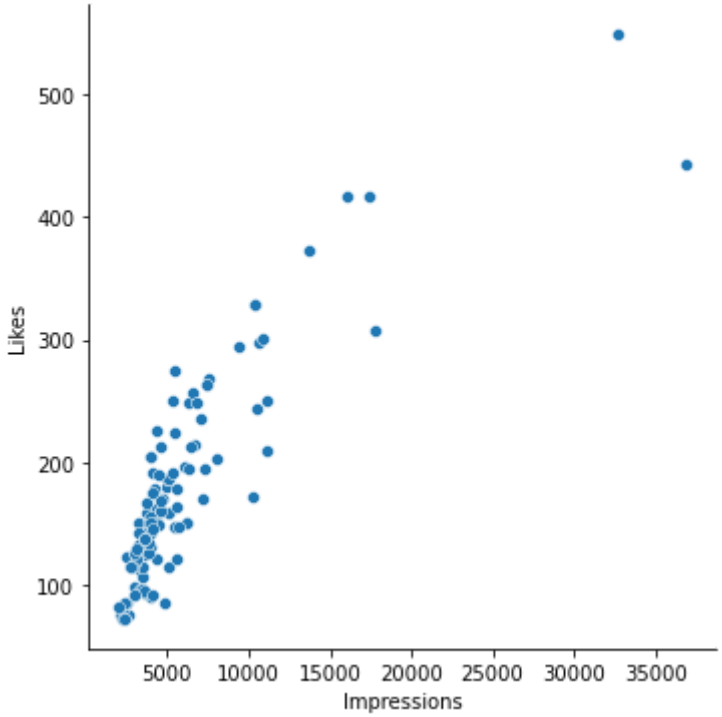
Out[15]:

	Impressions	Likes
0	3920	162
1	5394	224
2	4021	131
3	4528	213
4	2518	123
...	...	...
114	13700	373
115	5731	148
116	4139	92
117	32695	549
118	36919	443

119 rows × 2 columns

In [16]:
```
# make a relational plot in which x axis contains value of impressions
and y axis contains values of likes,
# this plot telling us the relationship between impressions and Likes,
how they are connected with each
# other ......there is more likes more impressions and less likes then
getting less impressions,
sns.relplot(data= df, x="Impressions", y="Likes")
```

Out[16]:
```
<seaborn.axisgrid.FacetGrid at 0x1beaddee2e0>
```



In [17]:
```
# To get a groupby of impressions and from home values:
df.groupby(['Impressions', 'From Home']).sum().T
```

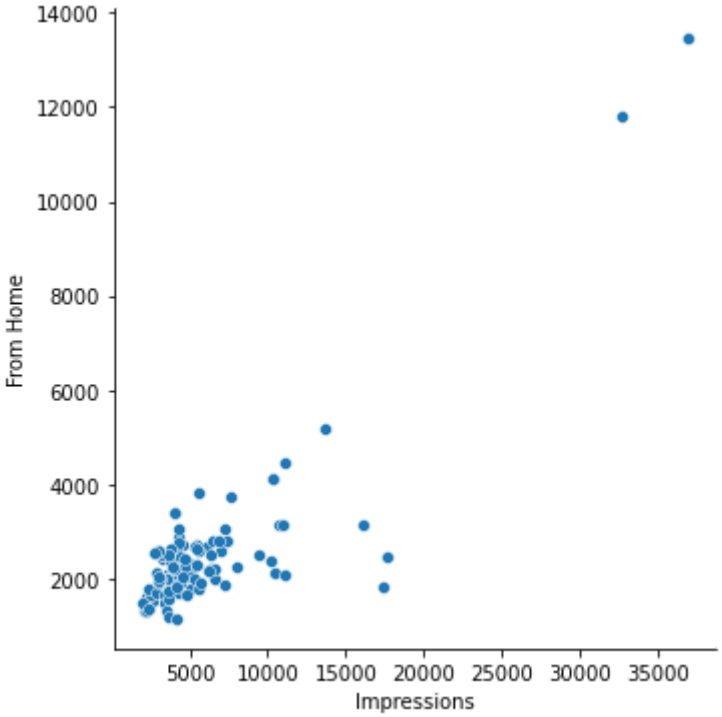
Out[17]:

Impressions	1941	2064	2191	2218	2327	2407	2518	2523	2621	2766	...	10667	10933	11
From Home	**1466**	**1304**	**1308**	**1597**	**1774**	**1338**	**1704**	**1659**	**1543**	**2541**	**...**	**3152**	**3152**	**2**
From Hashtags	411	362	809	411	435	1310	255	796	599	232	...	6564	6610	
From Explore	37	249	45	162	59	552	279	29	333	102	...	617	623	
From Other	17	37	18	15	35	78	37	21	25	18	...	187	334	
Saves	49	49	35	28	45	80	96	34	22	80	...	219	225	
Comments	6	4	2	6	3	16	5	6	5	20	...	13	13	
Shares	3	5	1	3	3	40	4	0	1	8	...	15	15	
Likes	82	76	72	81	85	144	123	86	76	228	...	297	301	
Profile Visits	8	9	18	29	7	20	8	4	26	22	...	306	347	
Follows	2	0	0	4	2	0	0	2	0	12	...	74	94	

9 rows × 102 columns

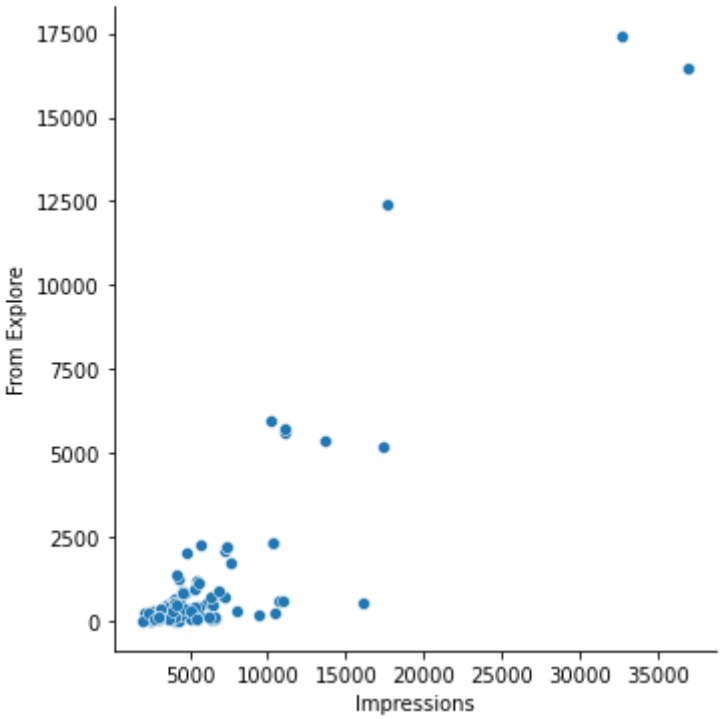
In [18]:
```
# To see relational plot of impressions and from home which is telling us
the relationship of both,
# more impressions when posts from home and the getting low...
sns.relplot(data= df, x="Impressions", y="From Home")
```

Out[18]: <seaborn.axisgrid.FacetGrid at 0x1beadf261f0>



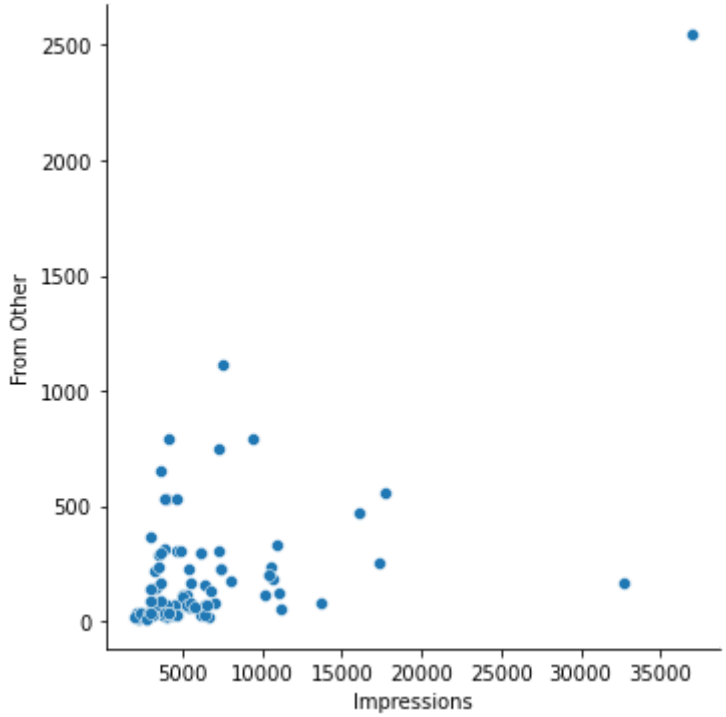
In [19]:
```
# Relational plot of impressions and 'from explore' values of data which is telling us their relationship
sns.relplot(data= df, x="Impressions", y="From Explore")
```

Out[19]: <seaborn.axisgrid.FacetGrid at 0x1beae700730>



In [20]:
```
# Relational plot of impressions and 'from other ', which telling us their relationship
sns.relplot(data= df, x="Impressions", y="From Other")
```

Out[20]: <seaborn.axisgrid.FacetGrid at 0x1beae76e280>



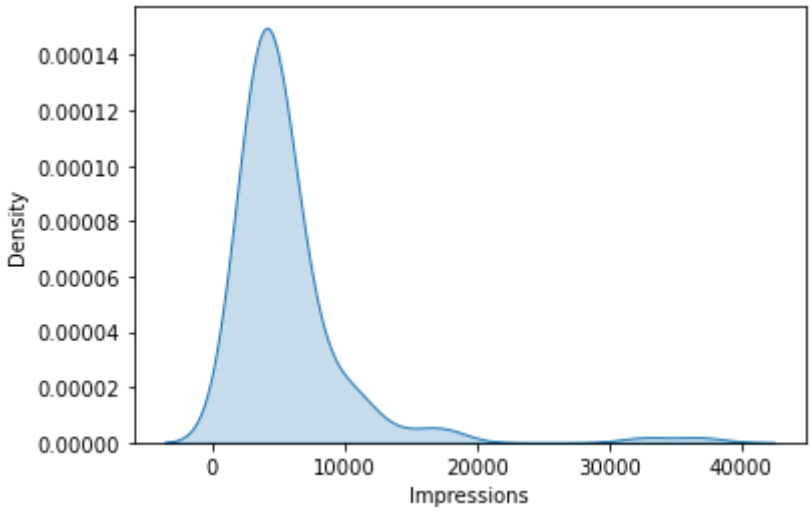
In [21]:
```
# to get the group by impressions with sum values of all varaiables with see the heads:
df.groupby('Impressions').sum().head(2).T
```

Out[21]:

Impressions	1941	2064
From Home	1466	1304
From Hashtags	411	362
From Explore	37	249
From Other	17	37
Saves	49	49
Comments	6	4
Shares	3	5
Likes	82	76
Profile Visits	8	9
Follows	2	0

In [22]:
```
# KDE is plot of 'Impressions'
sns.kdeplot(df['Impressions'], shade = True)
```

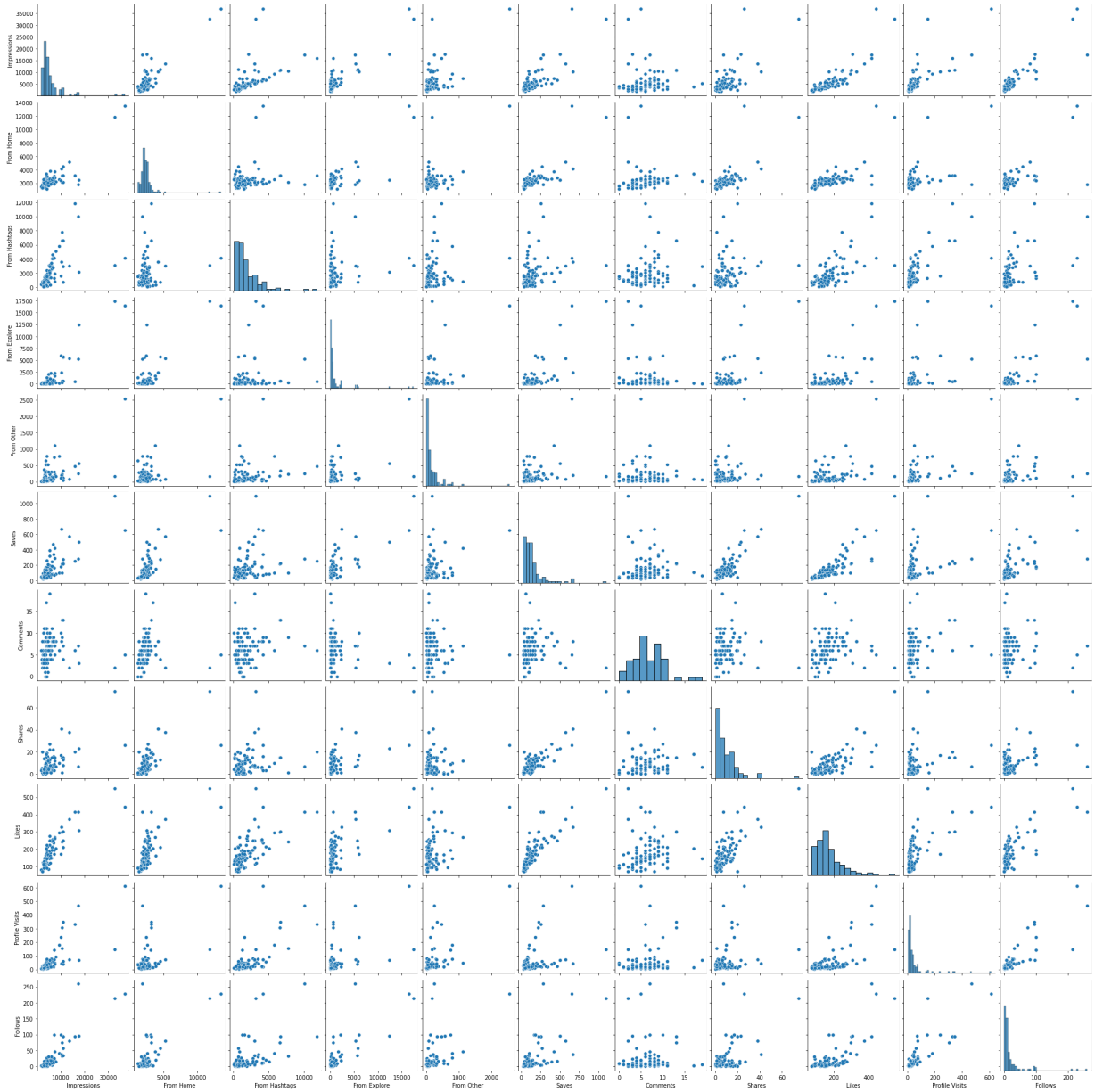
Out[22]:
```
<AxesSubplot:xlabel='Impressions', ylabel='Density'>
```



In [23]:
```
# Blank Cell
```

In [24]:
```
# Pairplot of data:
sns.pairplot(df)
```

Out[24]: <seaborn.axisgrid.PairGrid at 0x1beae7c8940>



In [25]:
```
# See the correlation of data
corr= df.corr()
```

In [26]:
```
# To see the relationship between the variables:
corr
```

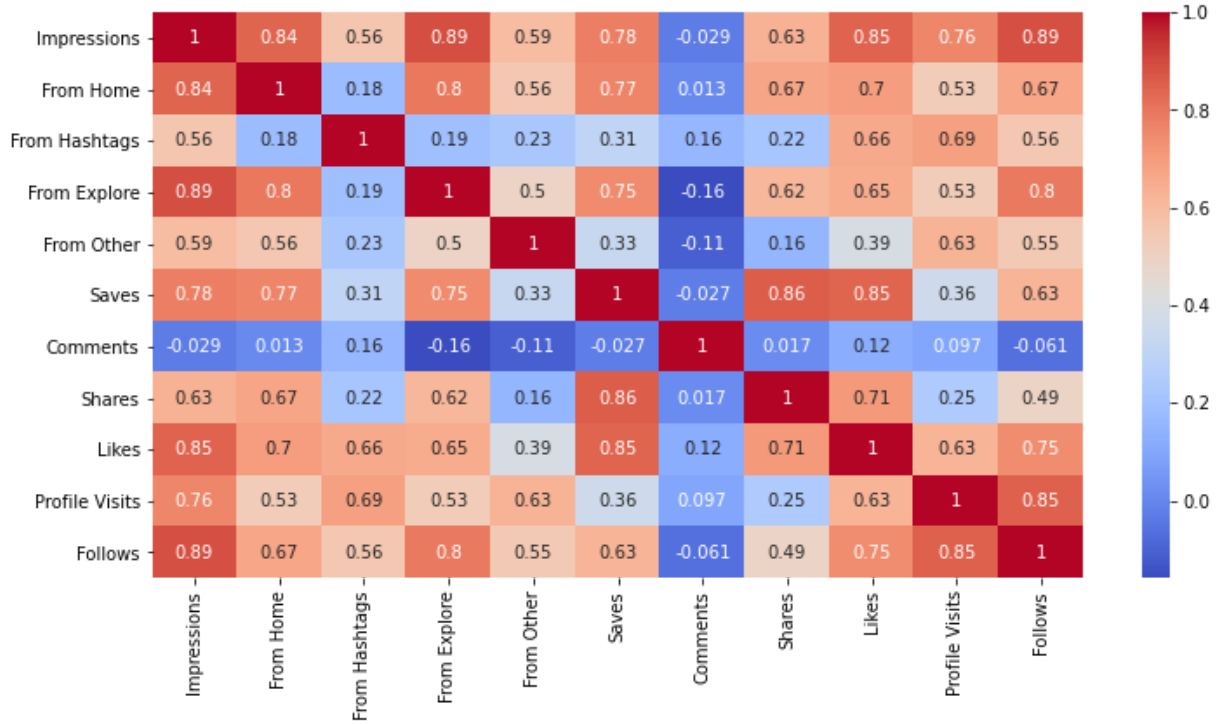
Out[26]:

	Impressions	From Home	From Hashtags	From Explore	From Other	Saves	Comments	Shares	
Impressions	1.000000	0.844698	0.560760	0.893607	0.592960	0.779231	-0.028524	0.634675	0.8
From Home	0.844698	1.000000	0.177516	0.800573	0.555666	0.768817	0.012716	0.674985	0.6
From Hashtags	0.560760	0.177516	1.000000	0.190453	0.229623	0.305929	0.161439	0.219511	0.6
From Explore	0.893607	0.800573	0.190453	1.000000	0.495685	0.747803	-0.158565	0.615731	0.6
From Other	0.592960	0.555666	0.229623	0.495685	1.000000	0.331907	-0.108703	0.156834	0.3
Saves	0.779231	0.768817	0.305929	0.747803	0.331907	1.000000	-0.026912	0.860324	0.8
Comments	-0.028524	0.012716	0.161439	-0.158565	-0.108703	-0.026912	1.000000	0.016933	0.1

	Impressions	From Home	From Hashtags	From Explore	From Other	Saves	Comments	Shares	
Shares	0.634675	0.674985	0.219511	0.615731	0.156834	0.860324	0.016933	1.000000	0.7
Likes	0.849835	0.698330	0.662124	0.653699	0.393510	0.845643	0.123586	0.707794	1.0
Profile Visits	0.760981	0.531076	0.691345	0.531850	0.633080	0.360628	0.096714	0.245361	0.6
Follows	0.889363	0.672675	0.555485	0.796019	0.546737	0.628461	-0.060631	0.493070	0.7

In [27]:
```
# Make Heat map of variables to check the higher and lower values
# Draw the heatmap which shows the relationship between the varaibles how they are related to each other,
# dark blocks shows higher values(strongly related) , medium color shows medium values and light color blocks shows lesser
#(weak relation)
# values..

plt.figure(figsize=(12,6))
sns.heatmap(corr, annot= True, cmap= 'coolwarm')
```

Out[27]: <AxesSubplot:>



In [28]:
```
# make a pivot table for Impressions and From home variables:
table = pd.pivot_table(data=df,index=['Impressions','From Home']).mean()
table
```

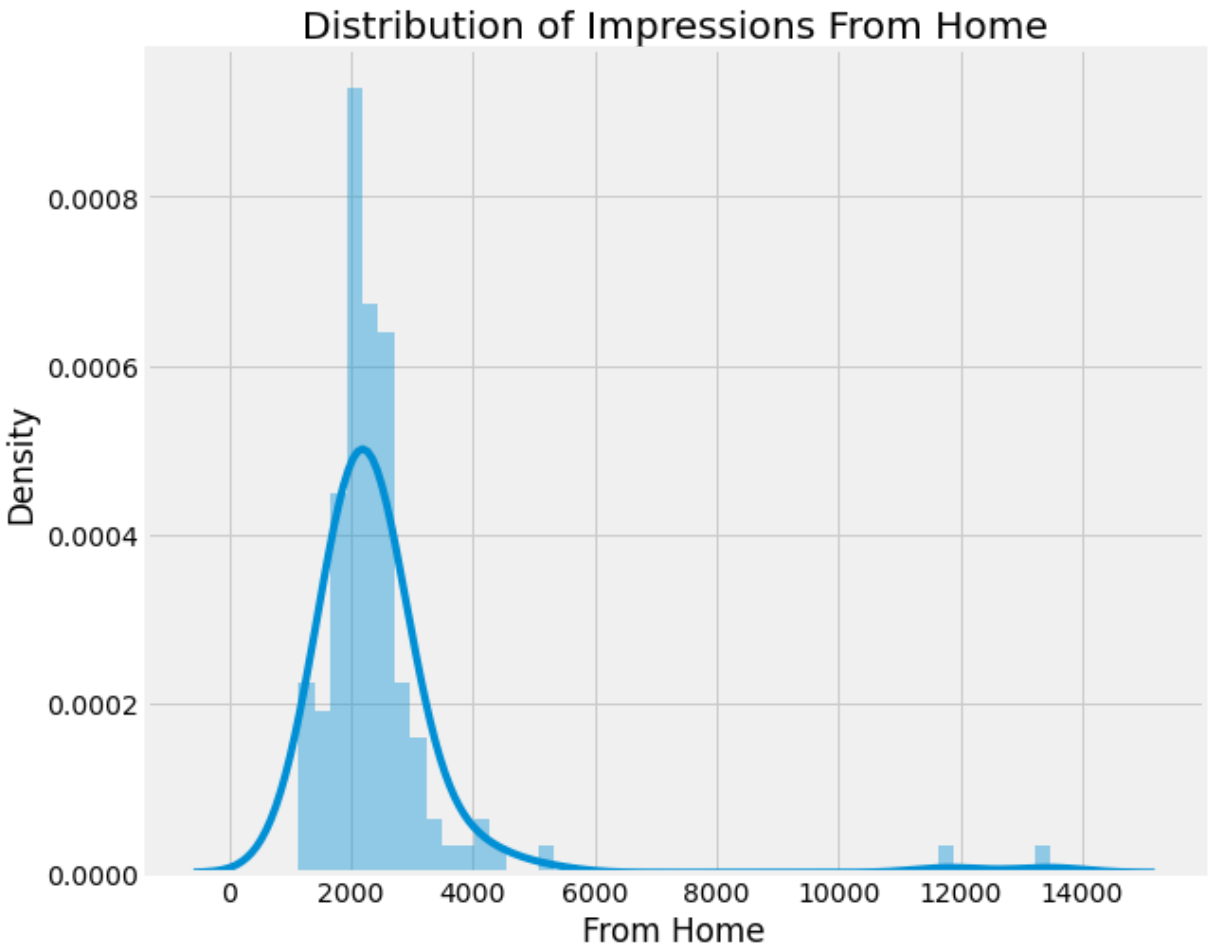
Out[28]:
```
Comments              6.352941
Follows              22.823529
From Explore       1178.568627
From Hashtags      1968.284314
```

```
From Other         184.549020
Likes              176.823529
Profile Visits      54.666667
Saves              156.549020
Shares               9.303922
dtype: float64
```

In [29]:

```
# By these two variables we are analyzing the reach of my instagram
posts,i will look at the distribution
# of impressions i have received from home, this graph is telling us how
much impressions values are
# receiveing from the varible "From Home", how much it is influencing the
'Impressions' for the posts.
plt.figure(figsize=(10, 8))
plt.style.use('fivethirtyeight')
plt.title("Distribution of Impressions From Home")
sns.distplot(df['From Home'])
plt.show()

# summary: this impressions we getting from the home section on instagram
which shows how much my posts
# reach my followers, by seeing this impressions from home, i can say
it's hard to reach all my followers
# daily.
```

```
C:\Users\Simmy\anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarnin
g: `distplot` is a deprecated function and will be removed in a future version. Please
adapt your code to use either `displot` (a figure-level function with similar flexibil
ity) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



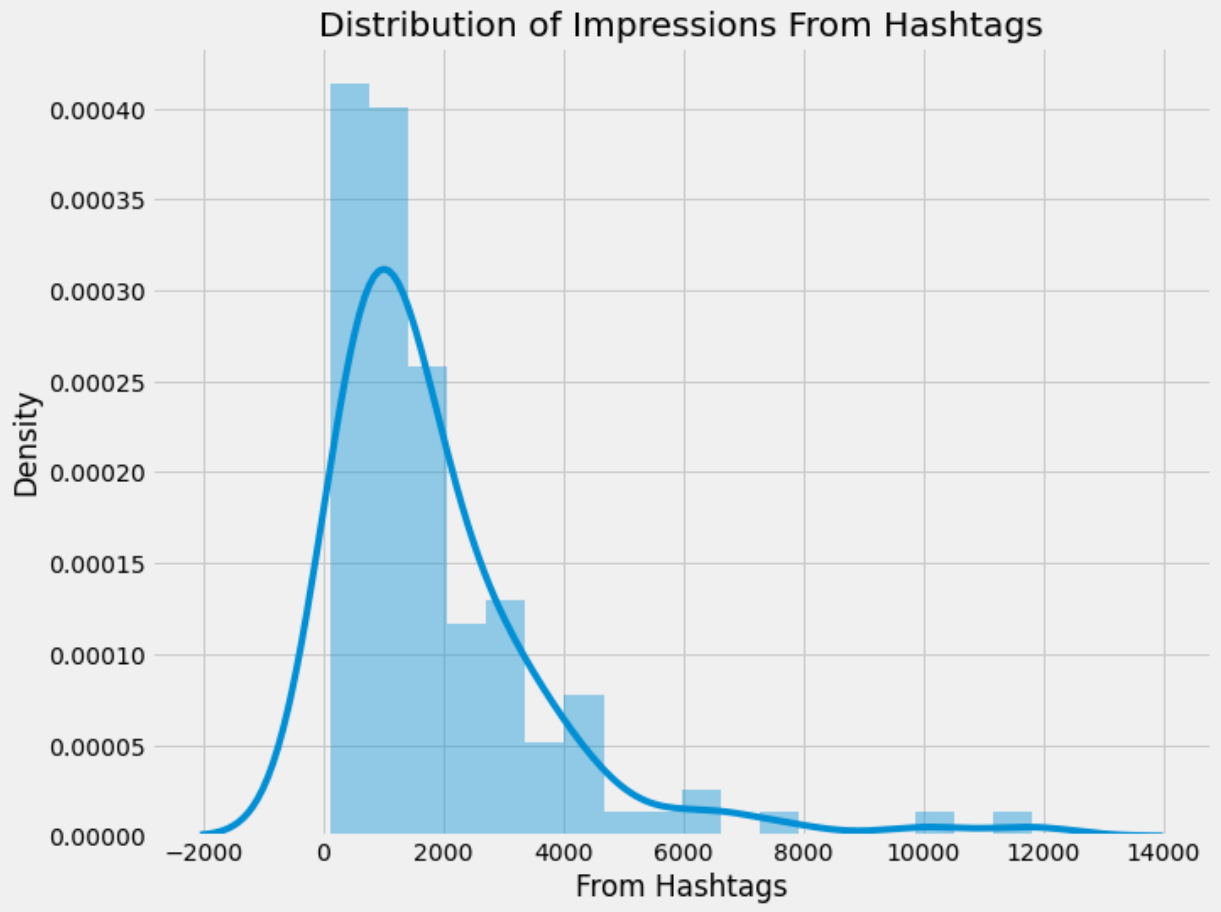
In [30]:
```
# making a pivot table for the "Impressions" and " From Hashtags" variables, which will further
# playing a role in distribution graph for these two and analyzing the reach of the instagram posts.
table = pd.pivot_table(data=df,index=['Impressions','From Hashtags']).mean()
table
```

Out[30]:
```
Comments            6.352941
Follows            22.823529
From Explore     1178.568627
From Home        2496.911765
From Other        184.549020
Likes             176.823529
Profile Visits     54.666667
Saves             156.549020
Shares              9.303922
dtype: float64
```

In [31]:
```
# By these two variables we are analyzing the reach of my instagram posts,i will look at the distribution
# of impressions i have received from home, this graph is telling us how much impressions values are
# receiveing from the variable "From Hshtags", how much it is influencing the 'Impressions' for the posts
# by the "From Hashtags" variable...
```

```
plt.figure(figsize=(10, 8))
plt.title("Distribution of Impressions From Hashtags")
sns.distplot(df['From Hashtags'])
plt.show()

#summary: now lets have a look on this distribution of impressions i
received from hashtags, by seeing this
# graph i m analyzing that impressions are stronger by hashtags section
later on , it is falling down,
# Hashtags are tools we use to categorize our posts on Instagram so that
we can reach more people based
# on the kind of content we are creating. Looking at hashtags impressions
shows that not all posts can be
# reached using hashtags, but many new users can be reached from
hastags.not all posts connected to
# hashtags but some of the posts connected to hashtags therefore, some of
the posts reaching to users,
# this is why , initially it is raising and influencing "Impressions" and
then later it is falling , now
# impressions are falling.
```

```
C:\Users\Simmy\anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarnin
g: `distplot` is a deprecated function and will be removed in a future version. Please
adapt your code to use either `displot` (a figure-level function with similar flexibil
ity) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



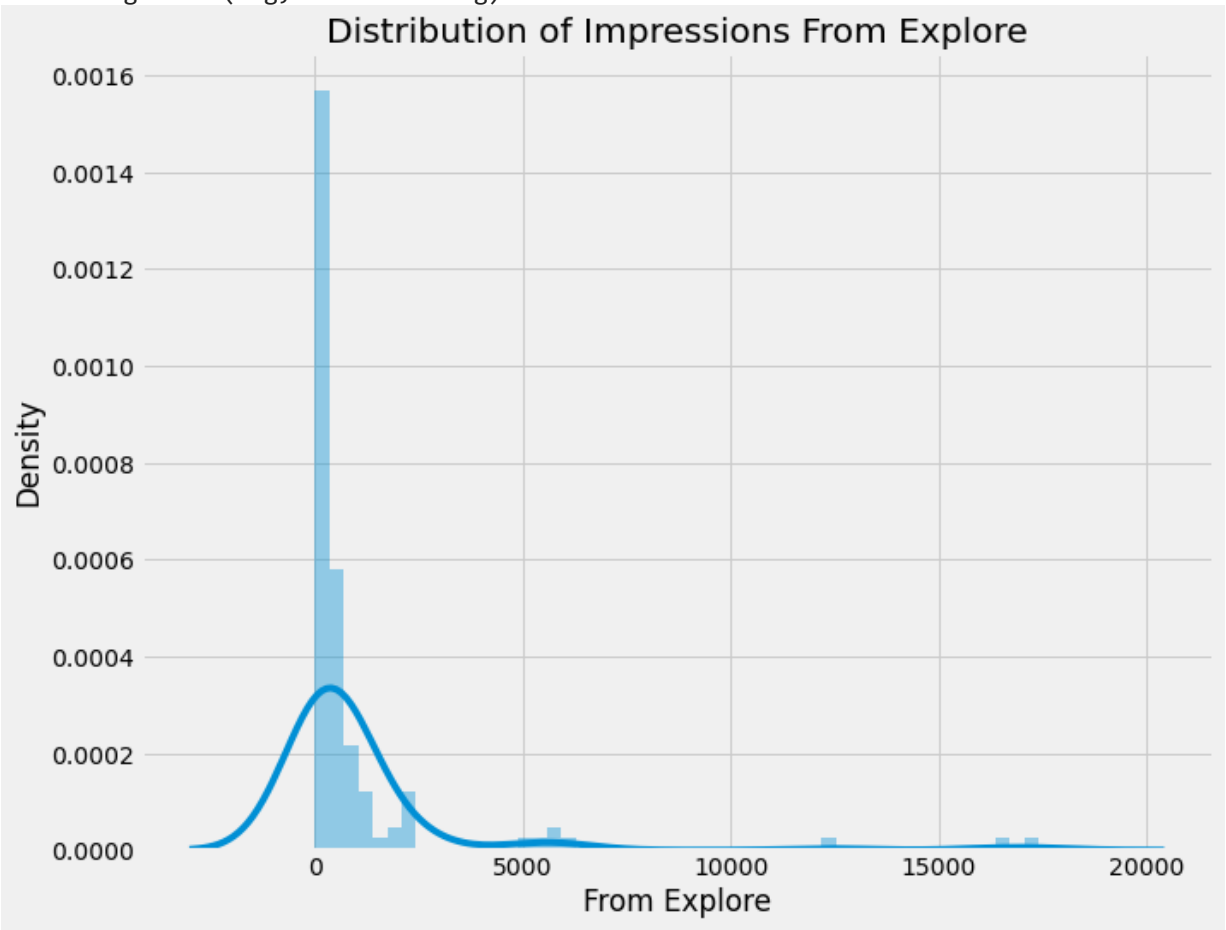
In [32]:
```
# making a pivot table for the "Impressions" and " From Explore"
variables, which will further
# playing  a role in  distribution graph for these two and analyzing the
reach of the instagram posts.
table = pd.pivot_table(data=df,index=['Impressions','From
Explore']).mean()
table
```

Out[32]:
```
Comments              6.352941
Follows              22.823529
From Hashtags      1968.284314
From Home          2496.911765
From Other          184.549020
Likes               176.823529
Profile Visits       54.666667
Saves               156.549020
Shares                9.303922
dtype: float64
```

In [33]:
```
# By these two variables we are analyzing the reach of my instagram
posts,i will look at the distribution
# of impressions i have received from home, this graph is telling us how
much impressions values are
# receiveing from the variable "From Explore", how much it is influencing
the 'Impressions' for the posts
# by the "From Explore" variable...
plt.figure(figsize=(10,8))
```

```
plt.title("Distribution of Impressions From Explore")
sns.distplot(df['From Explore'])
plt.show()

#summary: Now lets have a look on distribjution graph of "impressions"
and "From Explore", by this
#explore section of instagram is the reccomendation system of
instagram.It recommends posts to the users
#based on their preferences and interests.By seeing at the impressions i
have received from the explore
# section, i can say that instagram doest not reccomend our posts much to
mthe users.Some posts have
# reach from the explore section, but still very low as compared to the
reach i recieve from hastags.
```

```
C:\Users\Simmy\anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarnin
g: `distplot` is a deprecated function and will be removed in a future version. Please
adapt your code to use either `displot` (a figure-level function with similar flexibil
ity) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



In [34]:

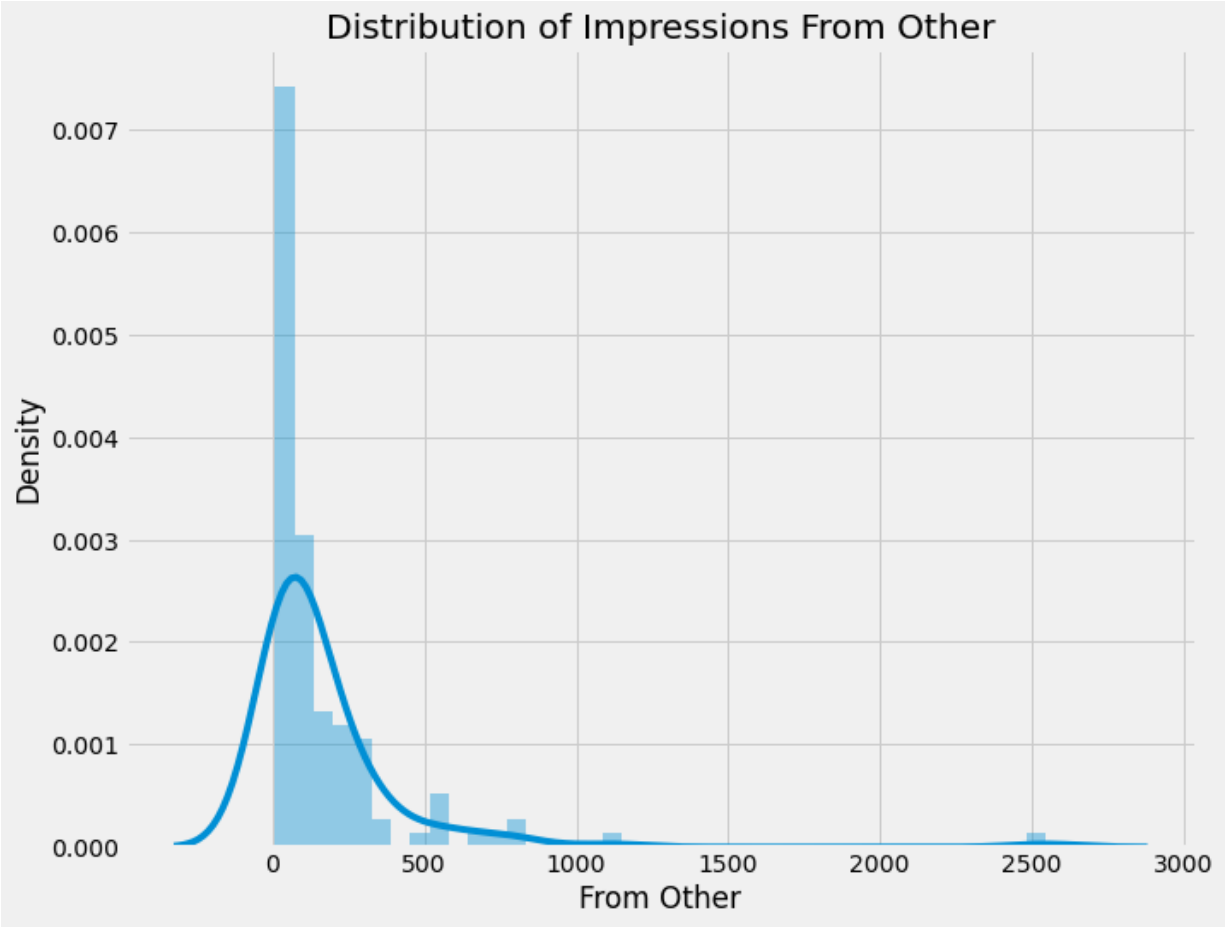
```
# making a pivot table for the "Impressions" and " From Other" variables,
which will further
# playing  a role in  distribution graph for these two and analyzing the
reach of the instagram posts.
```

```
table = pd.pivot_table(data=df,index=['Impressions','From Other']).mean()
table
```

Out[34]:
```
Comments              6.352941
Follows              22.823529
From Explore       1178.568627
From Hashtags      1968.284314
From Home          2496.911765
Likes               176.823529
Profile Visits       54.666667
Saves               156.549020
Shares                9.303922
dtype: float64
```

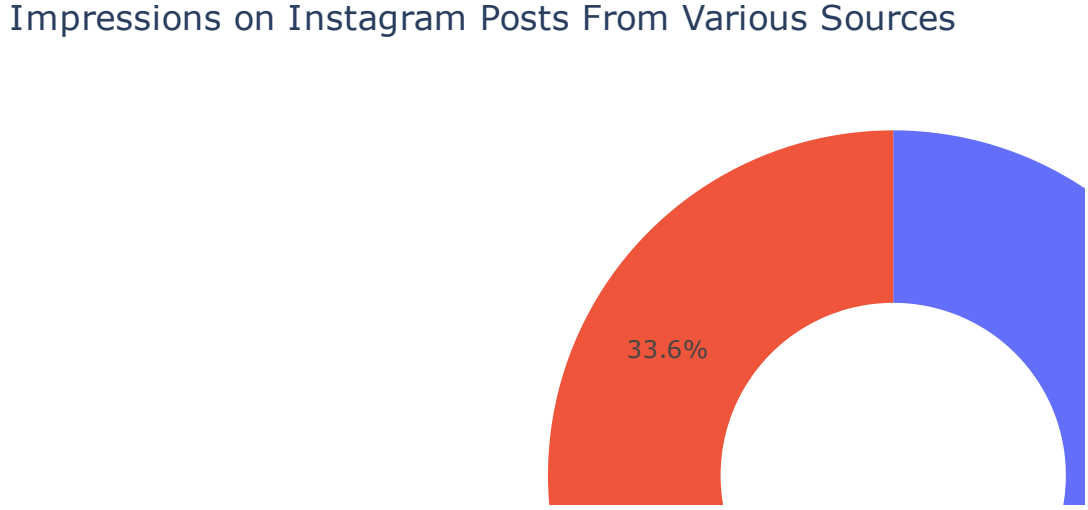
In [35]:
```
# making a distribution plot of "Impressions" and "From Other". By these two variables we are analyzing the reach of my instagram posts,i will look at the distribution
# of impressions i have received from home, this graph is telling us how much impressions values are
# receiveing from the variable "From Other", how much it is influencing the 'Impressions' for the posts
# by the "From Other" variable...
plt.figure(figsize=(10,8))
plt.style.use('fivethirtyeight')
plt.title("Distribution of Impressions From Other")
sns.distplot(df['From Other'])
plt.show()

# summary: From other section , posts are reaching to the users by the other resources, like other links,
# advertisement, frens of frens, followers of follwers or their following or in other ways, by seeing at r
# the other section ,some posts are reaching from the other section but it is very low as compared to other
# sections, it inflencing the impressions but in very low.
```

```
C:\Users\Simmy\anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



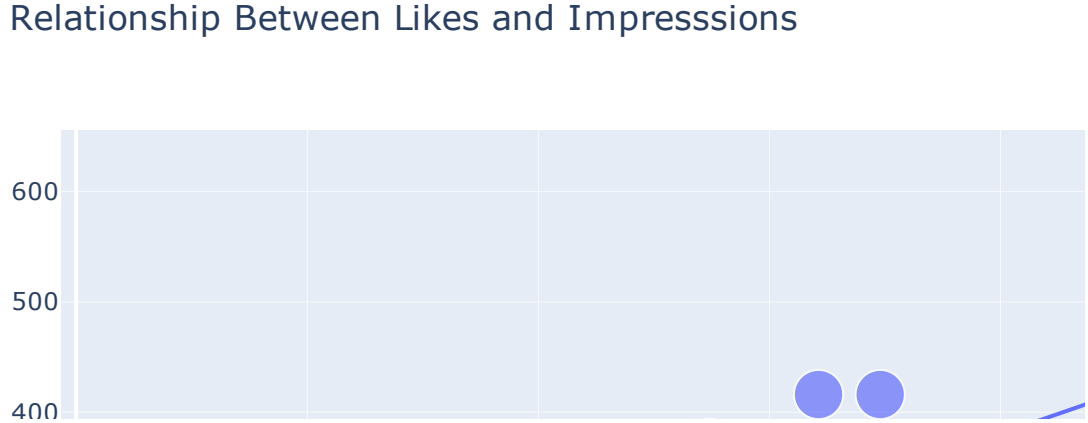
In [36]:

```
#Now let's have a look at the percentage of impressions I get from
various sources on Instagram, making a
# donut chart to get better understanding of all sections which is
required for the analyses.
home = df["From Home"].sum()
hashtags = df["From Hashtags"].sum()
explore = df["From Explore"].sum()
other = df["From Other"].sum()

labels = ['From Home','From Hashtags','From Explore','Other']
values = [home, hashtags, explore, other]

fig = px.pie(df, values=values, names=labels,
             title='Impressions on Instagram Posts From Various Sources',
hole=0.5)
fig.show()

#summary: So the above donut plot shows that almost 50 per cent of the
reach is from my followers,
#38.1 per cent is from hashtags, 9.14 per cent is from the explore
section,
#and 3.01 per cent is from other sources.
```



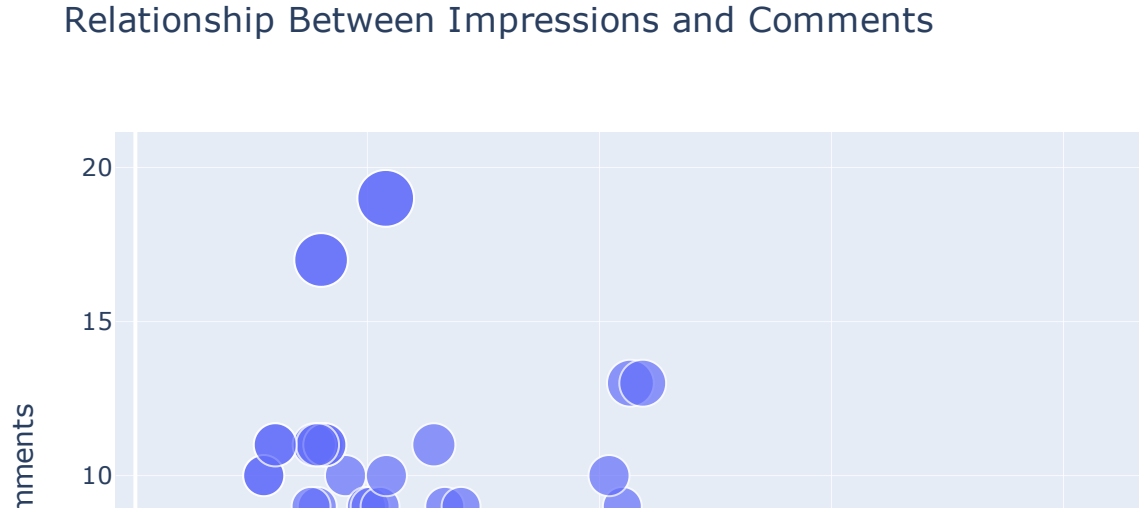
In [37]:

```
#Let's have a look at the relationship between the number of likes and
the number of
#impressions on my Instagram posts:
figure = px.scatter(data_frame= df, x = "Impressions",
                    y = "Likes", size= "Likes",trendline = 'ols',
                    title= "Relationship Between Likes and Impresssions")
figure.show()
#There is a linear relationship between the number of likes and the reach
I got on Instagram.
```

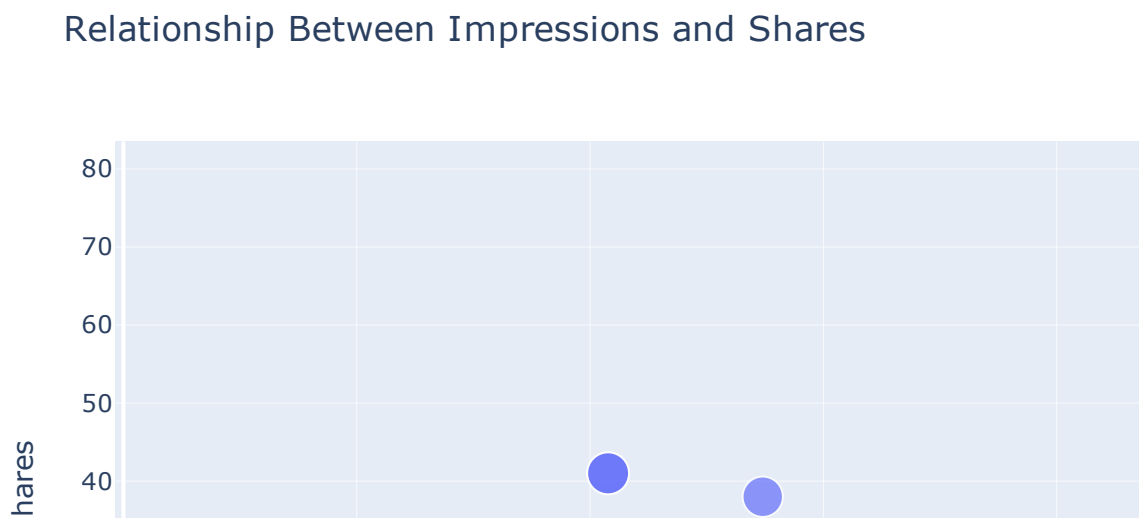


In [38]:

```
#Now let's see the relationship between the number of comments and the
number of impressions on
#my Instagram posts:
figure = px.scatter(data_frame= df, x = "Impressions", y= "Comments",
size = "Comments",trendline= 'ols',
                    title= "Relationship Between Impressions and
Comments")
figure.show()
#It looks like the number of comments we get on a post doesn't affect its
reach.
```

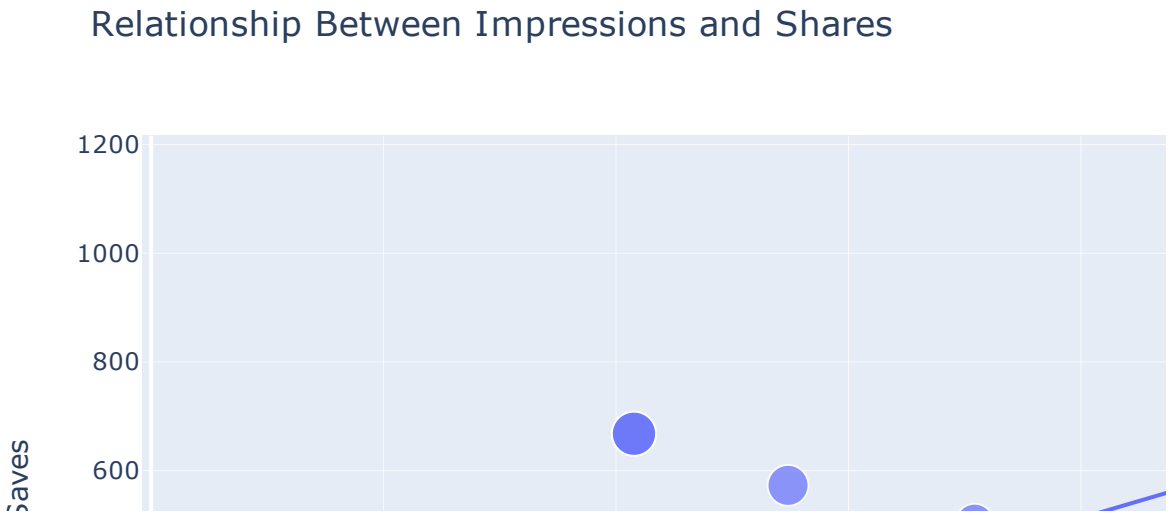


In [39]:
```
#Now let’s have a look at the relationship between the number of shares
and the number of impressions:
figure = px.scatter(data_frame= df, x="Impressions", y =
"Shares",trendline= "ols",
                    title= "Relationship Between Impressions and Shares",
size= "Shares")
figure.show()
#A more number of shares will result in a higher reach, but shares don’t
affect the reach of a post
#as much as likes do.
```



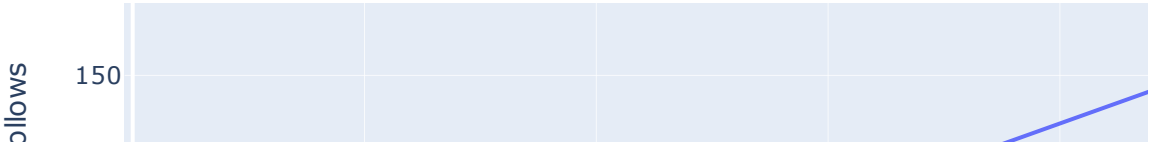
In [40]:
```
#Now let’s have a look at the relationship between the number of saves
and the number of impressions:
figure = px.scatter(data_frame= df, x="Impressions", y =
"Saves",trendline= "ols",
                    title= "Relationship Between Impressions and Shares",
size= "Saves")
```

```
figure.show()
#There is a linear relationship between the number of times my post is
#saved and the reach of my Instagram post.
```



In [41]:

```
#Now let's have a look at the relationship between the number of
impressions and the follows:
figure = px.scatter(data_frame= df, x="Impressions", y =
"Follows",trendline= "ols",
                    title= "Relationship Between Impressions and Shares",
size= "Follows")
figure.show()
```



In [42]:
```
# find out the correlation between the varibles.
correlation = df.corr()
print(correlation["Impressions"].sort_values(ascending = False))
```

```
Impressions       1.000000
From Explore      0.893607
Follows           0.889363
Likes             0.849835
From Home         0.844698
Saves             0.779231
Profile Visits    0.760981
Shares            0.634675
From Other        0.592960
From Hashtags     0.560760
Comments         -0.028524
Name: Impressions, dtype: float64
```

In [43]:
```
# correlation of likes columns with sorting with decending order:
print (correlation["Likes"].sort_values(ascending = False))
```

```
Likes             1.000000
Impressions       0.849835
Saves             0.845643
Follows           0.746333
Shares            0.707794
From Home         0.698330
From Hashtags     0.662124
From Explore      0.653699
Profile Visits    0.626107
From Other        0.393510
Comments          0.123586
Name: Likes, dtype: float64
```

In [44]:
```
# sorting values of some variables:
df.sort_values(by=['Likes','Follows'], inplace=True,
               ascending = [True, True])
print(df)
```

```
     Impressions  From Home  From Hashtags  From Explore  From Other  Saves  \
19          2407       1338            655           276          39     40
38          2191       1308            809            45          18     35
```

```
86          2407       1338           655           276          39     40
6           2621       1543           599           333          25     22
20          2064       1304           362           249          37     49
..           ...        ...           ...           ...         ...    ...
114        13700       5185          3041          5352          77    573
40         16062       3144         11817           564         468    252
107        17396       1817         10008          5192         251    285
118        36919      13473          4176         16444        2547    653
117        32695      11815          3147         17414         170   1095

     Comments  Shares  Likes  Profile Visits  Follows  \
19          8      20     72              10        0
38          2       1     72              18        0
86          8      20     72              10        0
6           5       1     76              26        0
20          4       5     76               9        0
..        ...     ...    ...             ...      ...
114         2      38    373              73       80
40          6      20    416             330       94
107         7       7    416             467      260
118         5      26    443             611      228
117         2      75    549             148      214

                                                Caption  \
19   Data Science Use Cases: Here's how Zomato is u...
38   Language detection is a natural language proce...
86   Data Science Use Cases: Here's how Zomato is u...
6    Learn how to analyze a candlestick chart as a ...
20   A boxplot is a statistical data visualization ...
..                                                 ...
114  Here are some of the best data science certifi...
40   280 Machine Learning Projects Solved & Explain...
107  Here is a list of 100+ Machine Learning Algori...
118  175 Python Projects with Source Code solved an...
117  Here are some of the best data science certifi...

                                              Hashtags
19   #data#datascience#dataanalysis#dataanalytic...
38   #data#datascience#dataanalysis#dataanalytic...
86   #data#datascience#dataanalysis#dataanalytic...
6    #stockmarket#investing#stocks#trading#mone...
20   #datavisualization#datascience#data#dataana...
..                                               ...
114  #datascience#datasciencejobs#datasciencetrai...
40   #data#datascience#dataanalysis#dataanalytic...
107  #machinelearning##machinelearningalgorithms#d...
118  #python##pythonprogramming##pythonprojects#py...
117  #datascience#datasciencejobs#datasciencetrai...

[119 rows x 13 columns]
```

In [45]:

```
# Analyzing the conversion rate:
#In Instagram, conversation rate means how many followers you are getting
from the number of
#profile visits from a post. The formula that you can use to calculate
conversion rate is
#(Follows/Profile Visits) * 100.
#Now let's have a look at the conversation rate of my Instagram account:


conversion_rate= (df["Follows"].sum()/df["Profile Visits"].sum()*100)
conversion_rate
```

Out[45]: 41.00265604249668

In [46]:
```
# Blank Cell:----------------------**********
```

In [47]:
```
#home = df["From Home"].sum()
#hashtags = df["From Hashtags"].mean()
#explore = df["From Explore"].mean()
#other = df["From Other"].mean()

#labels = ['From Home','From Hashtags','From Explore','Other']
#values = [home, hashtags, explore, other]

#fig = px.pie(df, values=values, names=labels,
             #title='Impressions on Instagram Posts From Various
Sources', hole=0.5)
#fig.show()
```

In [ ]:
```

```

In [48]:
```
#figure = px.scatter(data_frame= df, x = 'Profile Visits',
                    #y = "Follows", size = "Follows", trendline= "ols",
                    #title = "Relationship Between Profile Visits and
Followers Gained")
#figure.show()
```

In [49]:
```
#df.groupby(['Saves', 'Comments']).sum()
```

In [50]:
```
#df.groupby('Impressions').agg({'From Home': ['mean', 'min', 'max']})
```

In [ ]:
```
#****END PROJ
```

In [ ]:
```

```

In [ ]:
```

```

In [ ]:
```

```