

In [3]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

In [4]:

```
house_data = pd.read_csv("bengaluru_house_data.csv")
```

In [5]:

```
house_data.shape
```

Out[5]:

```
(13320, 9)
```

In [6]:

```
house_data.columns
```

Out[6]:

```
Index(['area_type', 'availability', 'location', 'size', 'society',
      'total_sqft', 'bath', 'balcony', 'price'],
      dtype='object')
```

In [7]:

```
house_data.head()
```

Out[7]:

	area_type	availability	location	size	society	total_sqft	bath	balcony	price
0	Super built-up Area	19-Dec	Electronic City Phase II	2 BHK	Coomee	1056	2.0	1.0	39.0
1	Plot Area	Ready To Move	Chikka Tirupathi	4 Bedroom	Theanmp	2600	5.0	3.0	120.0
2	Built-up Area	Ready To Move	Uttarahalli	3 BHK	NaN	1440	2.0	3.0	62.0
3	Super built-up Area	Ready To Move	Lingadheeranahalli	3 BHK	Soiewre	1521	3.0	1.0	95.0
4	Super built-up Area	Ready To Move	Kothanur	2 BHK	NaN	1200	2.0	1.0	51.0

In [6]:

```
house_data.tail()
```

Out[6]:

	area_type	availability	location	size	society	total_sqft	bath	balcony	pr
13315	Built-up Area	Ready To Move	Whitefield	5 Bedroom	ArsiaEx	3453	4.0	0.0	23
13316	Super built-up Area	Ready To Move	Richards Town	4 BHK	NaN	3600	5.0	NaN	40
13317	Built-up Area	Ready To Move	Raja Rajeshwari Nagar	2 BHK	Mahla T	1141	2.0	1.0	6
13318	Super built-up Area	18-Jun	Padmanabhanagar	4 BHK	SollyCI	4689	4.0	1.0	48
13319	Super built-up Area	Ready To Move	Doddathoguru	1 BHK	NaN	550	1.0	1.0	1

In [8]:

```
house_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 13320 entries, 0 to 13319
Data columns (total 9 columns):
#   Column          Non-Null Count  Dtype
---  -
0   area_type       13320 non-null  object
1   availability     13320 non-null  object
2   location        13319 non-null  object
3   size            13304 non-null  object
4   society         7818 non-null   object
5   total_sqft      13320 non-null  object
6   bath            13247 non-null  float64
7   balcony         12711 non-null  float64
8   price           13320 non-null  float64
dtypes: float64(3), object(6)
memory usage: 936.7+ KB
```

In [9]:

```
house_data.describe()
```

Out[9]:

	bath	balcony	price
count	13247.000000	12711.000000	13320.000000
mean	2.692610	1.584376	112.565627
std	1.341458	0.817263	148.971674
min	1.000000	0.000000	8.000000
25%	2.000000	1.000000	50.000000
50%	2.000000	2.000000	72.000000
75%	3.000000	2.000000	120.000000
max	40.000000	3.000000	3600.000000

In [10]:

```
house_data.isnull().sum()
```

Out[10]:

```
area_type      0
availability    0
location       1
size           16
society        5502
total_sqft     0
bath           73
balcony        609
price          0
dtype: int64
```

In [11]:

```
house_data=house_data.drop(['area_type','availability','balcony',
                             'society'],axis=1)
```

In [12]:

```
house_data.head()
```

Out[12]:

	location	size	total_sqft	bath	price
0	Electronic CityPhase II	2 BHK	1056	2.0	39.07
1	Chikka Tirupathi	4 Bedroom	2600	5.0	120.00
2	Uttarahalli	3 BHK	1440	2.0	62.00
3	Lingadheeranahalli	3 BHK	1521	3.0	95.00
4	Kothanur	2 BHK	1200	2.0	51.00

In [13]:

```
house_data.isnull().sum()
```

Out[13]:

```
location      1
size          16
total_sqft    0
bath          73
price         0
dtype: int64
```

In [14]:

```
house_data = house_data.dropna()
```

In [15]:

```
house_data.shape
```

Out[15]:

```
(13246, 5)
```

In [16]:

```
house_data['BHK']=house_data['size'].apply(lambda x: int(x.split(' ')[0]))
```

In [17]:

```
house_data.head()
```

Out[17]:

	location	size	total_sqft	bath	price	BHK
0	Electronic CityPhase II	2 BHK	1056	2.0	39.07	2
1	Chikka Tirupathi	4 Bedroom	2600	5.0	120.00	4
2	Uttarahalli	3 BHK	1440	2.0	62.00	3
3	Lingadheeranahalli	3 BHK	1521	3.0	95.00	3
4	Kothanur	2 BHK	1200	2.0	51.00	2

In [18]:

```
house_data['BHK'].unique()
```

Out[18]:

```
array([ 2,  4,  3,  6,  1,  8,  7,  5, 11,  9, 27, 10, 19, 16, 43, 14, 12,
        13, 18], dtype=int64)
```

In [19]:

```
house_data['BHK'].value_counts()
```

Out[19]:

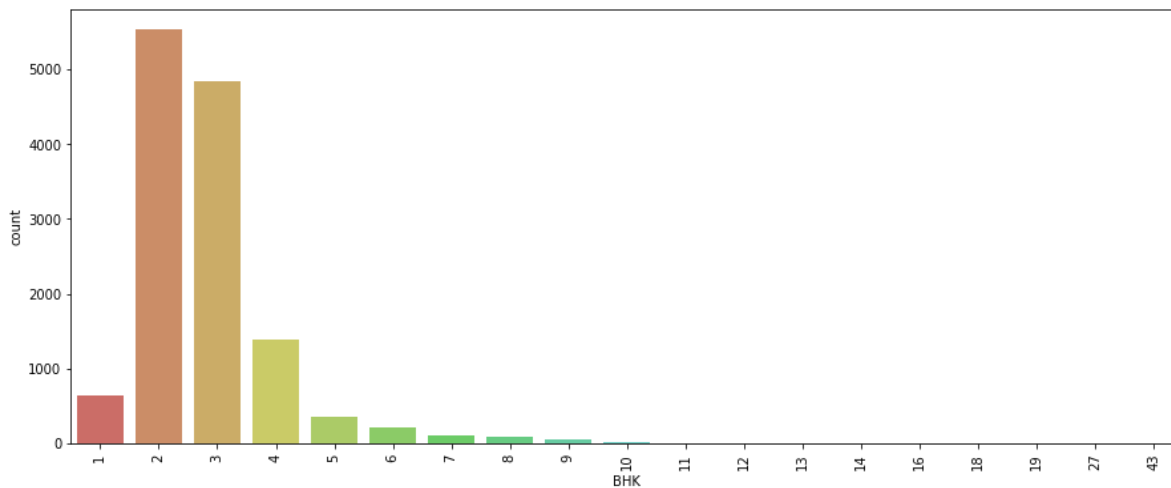
```
2      5527
3      4832
4      1395
1       649
5       353
6       221
7       100
8        89
9        54
10       14
11        4
27        1
19        1
16        1
43        1
14        1
12        1
13        1
18        1
```

```
Name: BHK, dtype: int64
```

In [20]:



```
plt.figure(figsize=(15,6))
sns.countplot('BHK', data = house_data, palette='hls')
plt.xticks(rotation = 90)
plt.show()
```



In [21]:



```
house_data['bath'].unique()
```

Out[21]:

```
array([ 2.,  5.,  3.,  4.,  6.,  1.,  9.,  8.,  7., 11., 10., 14., 27.,
        12., 16., 40., 15., 13., 18.])
```

In [22]:

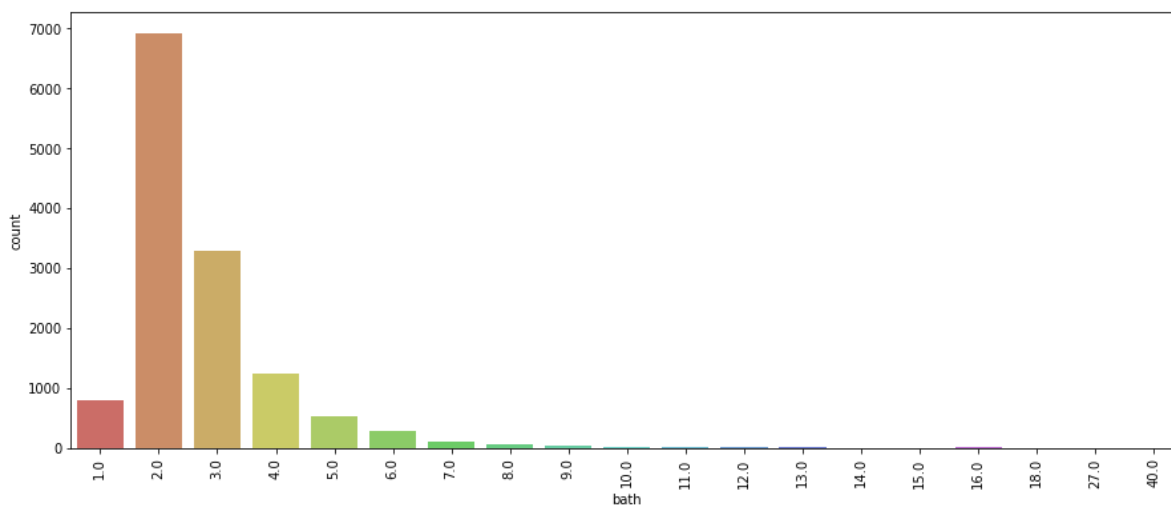
```
house_data['bath'].value_counts()
```

Out[22]:

```
2.0    6908
3.0    3285
4.0    1226
1.0     788
5.0     524
6.0     273
7.0     102
8.0      64
9.0      43
10.0     13
12.0      7
13.0      3
11.0      3
16.0      2
27.0      1
40.0      1
15.0      1
14.0      1
18.0      1
Name: bath, dtype: int64
```

In [23]:

```
plt.figure(figsize=(15,6))
sns.countplot('bath', data = house_data, palette='hls')
plt.xticks(rotation = 90)
plt.show()
```



In [24]:

```
house_data[house_data.BHK>15]
```

Out[24]:

	location	size	total_sqft	bath	price	BHK
1718	2Electronic City Phase II	27 BHK	8000	27.0	230.0	27
3379	1Hanuman Nagar	19 BHK	2000	16.0	490.0	19
3609	Koramangala Industrial Layout	16 BHK	10000	16.0	550.0	16
4684	Munnekollal	43 Bedroom	2400	40.0	660.0	43
11559	1Kasavanhalli	18 Bedroom	1200	18.0	200.0	18

In [25]:

```
def isfloat(x):
    try:
        float(x)
    except:
        return False
    return True
```

In [26]:

```
house_data[~house_data['total_sqft'].apply(isfloat)]
```

Out[26]:

	location	size	total_sqft	bath	price	BHK
30	Yelahanka	4 BHK	2100 - 2850	4.0	186.000	4
122	Hebbal	4 BHK	3067 - 8156	4.0	477.000	4
137	8th Phase JP Nagar	2 BHK	1042 - 1105	2.0	54.005	2
165	Sarjapur	2 BHK	1145 - 1340	2.0	43.490	2
188	KR Puram	2 BHK	1015 - 1540	2.0	56.800	2
...
12975	Whitefield	2 BHK	850 - 1060	2.0	38.190	2
12990	Talaghattapura	3 BHK	1804 - 2273	3.0	122.000	3
13059	Harlur	2 BHK	1200 - 1470	2.0	72.760	2
13265	Hoodi	2 BHK	1133 - 1384	2.0	59.135	2
13299	Whitefield	4 BHK	2830 - 2882	5.0	154.500	4

190 rows × 6 columns

In [27]:

```
def convert_sqft_tonum(x):
    token=x.split('-')
    if len(token)==2:
        return (float(token[0])+float(token[1]))/2
    try:
        return float(x)
    except:
        return None
```

In [28]:

```
house_data=house_data.copy()
house_data['total_sqft']=house_data['total_sqft'].apply(convert_sqft_tonum)
```

In [29]:

```
house_data.head()
```

Out[29]:

	location	size	total_sqft	bath	price	BHK
0	Electronic CityPhase II	2 BHK	1056.0	2.0	39.07	2
1	Chikka Tirupathi	4 Bedroom	2600.0	5.0	120.00	4
2	Uttarahalli	3 BHK	1440.0	2.0	62.00	3
3	Lingadheeranahalli	3 BHK	1521.0	3.0	95.00	3
4	Kothanur	2 BHK	1200.0	2.0	51.00	2

In [30]:

```
house_data.loc[20]
```

Out[30]:

```
location      Kengeri
size          1 BHK
total_sqft    600.0
bath          1.0
price         15.0
BHK           1
Name: 20, dtype: object
```

In [31]:

```
data1=house_data.copy()
data1['price_per_sqft']=data1['price']*1000000/data1['total_sqft']
data1.head()
```

Out[31]:

	location	size	total_sqft	bath	price	BHK	price_per_sqft
0	Electronic CityPhase II	2 BHK	1056.0	2.0	39.07	2	36998.106061
1	Chikka Tirupathi	4 Bedroom	2600.0	5.0	120.00	4	46153.846154
2	Uttarahalli	3 BHK	1440.0	2.0	62.00	3	43055.555556
3	Lingadheeranahalli	3 BHK	1521.0	3.0	95.00	3	62458.908613
4	Kothanur	2 BHK	1200.0	2.0	51.00	2	42500.000000

In [32]:

```
len(data1.location.unique())
```

Out[32]:

1304

In [33]:

```
data1.location=data1.location.apply(lambda x: x.strip())
location_stats=data1.groupby('location')['location'].agg('count').sort_values(ascending=
location_stats
```

Out[33]:

```
location
Whitefield          535
Sarjapur Road       392
Electronic City     304
Kanakpura Road      266
Thanisandra         236
...
1 Giri Nagar         1
Kanakapura Road,     1
Kanakapura main Road 1
Karnataka Shabarimala 1
whitefiled           1
Name: location, Length: 1293, dtype: int64
```

In [34]:

```
len(location_stats[location_stats<=10])
```

Out[34]:

1052

In [35]:

```
locationlessthan10=location_stats[location_stats<=10]  
locationlessthan10
```

Out[35]:

```
location  
Basapura          10  
1st Block Koramangala  10  
Gunjur Palya      10  
Kalkere           10  
Sector 1 HSR Layout 10  
..               ..  
1 Giri Nagar      1  
Kanakapura Road,  1  
Kanakapura main Road 1  
Karnataka Shabarimala 1  
whitefiled        1  
Name: location, Length: 1052, dtype: int64
```

In [36]:

```
len(data1.location.unique())
```

Out[36]:

1293

In [37]:

```
data1.location=data1.location.apply(lambda x: 'other' if x in locationlessthan10 else x)  
len(data1.location.unique())
```

Out[37]:

242

In [38]:

```
data1.head()
```

Out[38]:

	location	size	total_sqft	bath	price	BHK	price_per_sqft
0	Electronic CityPhase II	2 BHK	1056.0	2.0	39.07	2	36998.106061
1	Chikka Tirupathi	4 Bedroom	2600.0	5.0	120.00	4	46153.846154
2	Uttarahalli	3 BHK	1440.0	2.0	62.00	3	43055.555556
3	Lingadheeranahalli	3 BHK	1521.0	3.0	95.00	3	62458.908613
4	Kothanur	2 BHK	1200.0	2.0	51.00	2	42500.000000

In [39]:

```
data1[data1.total_sqft/data1.BHK<300].head()
```

Out[39]:

	location	size	total_sqft	bath	price	BHK	price_per_sqft
9	other	6 Bedroom	1020.0	6.0	370.0	6	362745.098039
45	HSR Layout	8 Bedroom	600.0	9.0	200.0	8	333333.333333
58	Murugeshpalya	6 Bedroom	1407.0	4.0	150.0	6	106609.808102
68	Devarachikkanahalli	8 Bedroom	1350.0	7.0	85.0	8	62962.962963
70	other	3 Bedroom	500.0	3.0	100.0	3	200000.000000

In [40]:

```
data2=data1[~(data1.total_sqft/data1.BHK<300)]
data2.head()
```

Out[40]:

	location	size	total_sqft	bath	price	BHK	price_per_sqft
0	Electronic CityPhase II	2 BHK	1056.0	2.0	39.07	2	36998.106061
1	Chikka Tirupathi	4 Bedroom	2600.0	5.0	120.00	4	46153.846154
2	Uttarahalli	3 BHK	1440.0	2.0	62.00	3	43055.555556
3	Lingadheeranahalli	3 BHK	1521.0	3.0	95.00	3	62458.908613
4	Kothanur	2 BHK	1200.0	2.0	51.00	2	42500.000000

In [41]:

```
data2.shape
```

Out[41]:

```
(12502, 7)
```

In [42]:

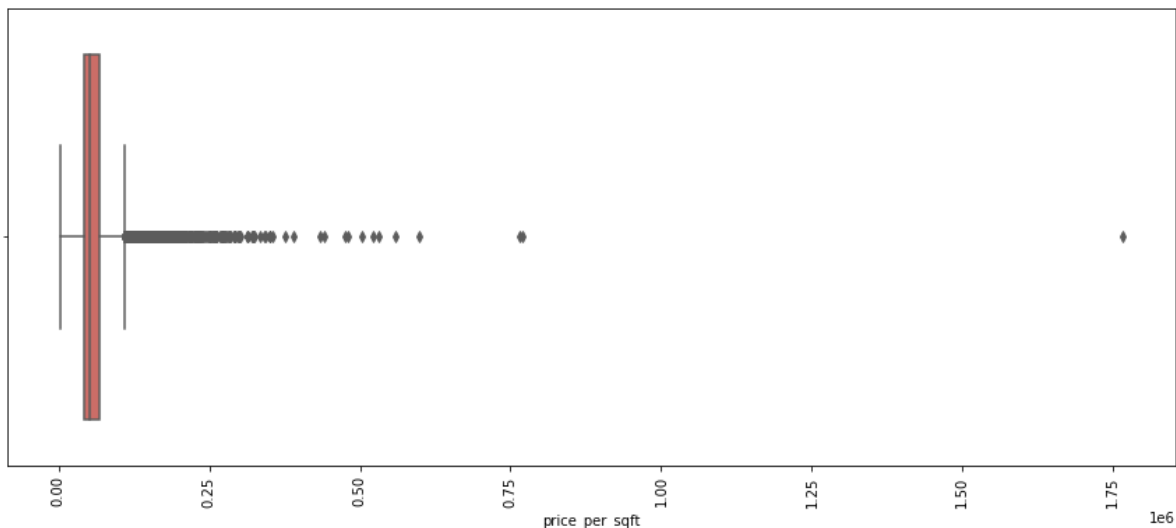
```
data2["price_per_sqft"].describe().apply(lambda x:format(x, 'f'))
```

Out[42]:

```
count      12456.000000
mean       63085.028260
std        41681.273385
min         2678.298133
25%        42105.263158
50%        52941.176471
75%        69166.666667
max       1764705.882353
Name: price_per_sqft, dtype: object
```

In [43]:

```
plt.figure(figsize=(15,6))
sns.boxplot('price_per_sqft', data = data2, palette='hls')
plt.xticks(rotation = 90)
plt.show()
```



In [44]:

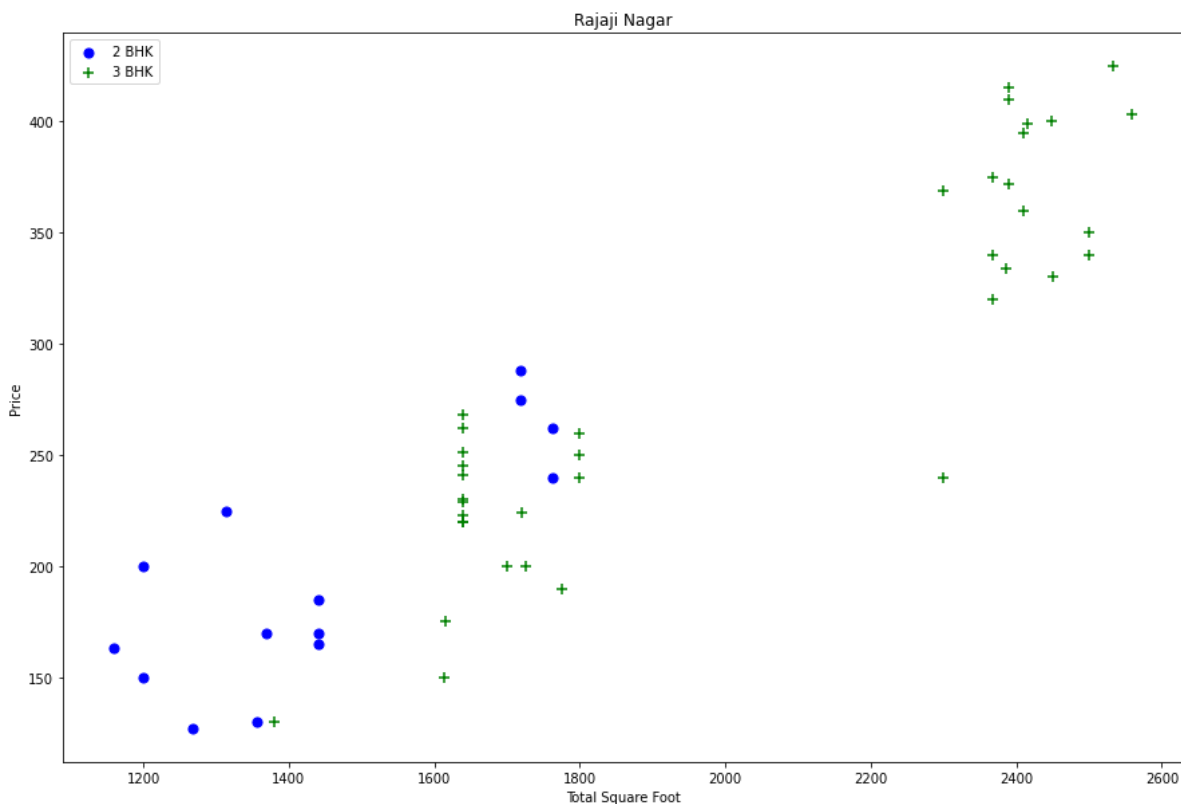
```
def remove_pps_outliers(df):
    df_out=pd.DataFrame()
    for key,subdf in df.groupby('location'):
        m=np.mean(subdf.price_per_sqft)
        st=np.std(subdf.price_per_sqft)
        reduced_df=subdf[(subdf.price_per_sqft>(m-st))& (subdf.price_per_sqft<(m+st))]
        df_out=pd.concat([df_out,reduced_df],ignore_index=True)
    return df_out
data3=remove_pps_outliers(data2)
data3.shape
```

Out[44]:

(10241, 7)

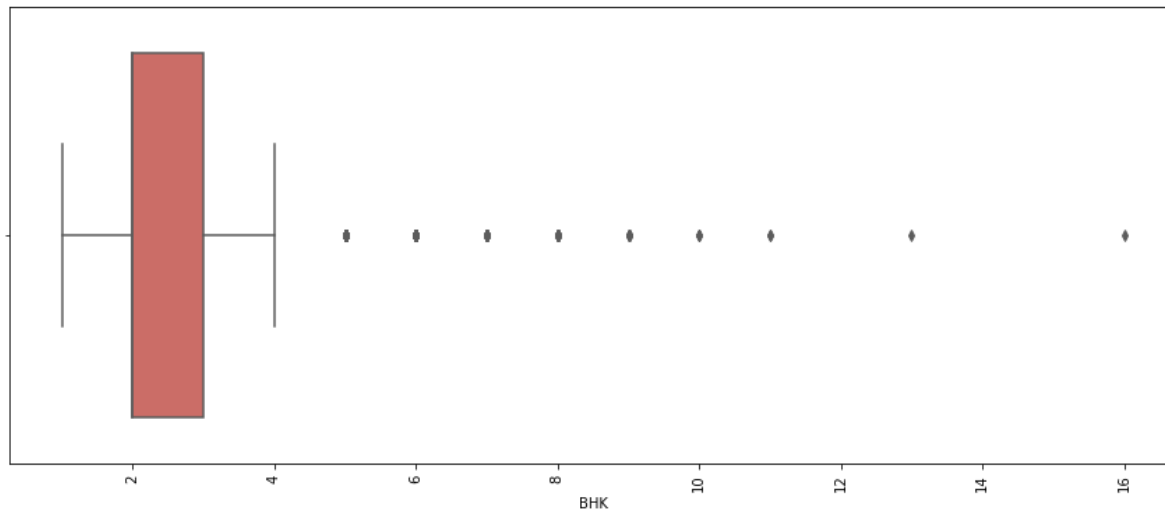
In [45]:

```
import matplotlib.pyplot as plt
def plot_scatter_chart(df,location):
    bhk2=df[(df.location==location)&(df.BHK==2)]
    bhk3=df[(df.location==location)&(df.BHK==3)]
    plt.rcParams['figure.figsize']=(15,10)
    plt.scatter(bhk2.total_sqft,bhk2.price,color='Blue',label='2 BHK',s=50)
    plt.scatter(bhk3.total_sqft,bhk3.price,color='green',marker='+',label='3 BHK',s=50)
    plt.xlabel('Total Square Foot')
    plt.ylabel('Price')
    plt.title(location)
    plt.legend()
plot_scatter_chart(data3,"Rajaji Nagar")
```



In [50]:

```
plt.figure(figsize=(15,6))
sns.boxplot('BHK', data = data3, palette='hls')
plt.xticks(rotation = 90)
plt.show()
```



In [51]:

```
def remove_bhk_outliers(df):
    exclude_indices=np.array([])
    for location, location_df in df.groupby('location'):
        bhk_sats={}
        for BHK,BHK_df in location_df.groupby('BHK'):
            bhk_sats[BHK]={
                'mean':np.mean(BHK_df.price_per_sqft),
                'std':np.std(BHK_df.price_per_sqft),
                'count':BHK_df.shape[0]
            }
        for BHK,BHK_df in location_df.groupby('BHK'):
            stats=bhk_sats.get(BHK-1)
            if stats and stats['count']>5:
                exclude_indices=np.append(exclude_indices,BHK_df[BHK_df.price_per_sqft<
            return df.drop(exclude_indices,axis='index')

data4=remove_bhk_outliers(data3)
data4.shape
```

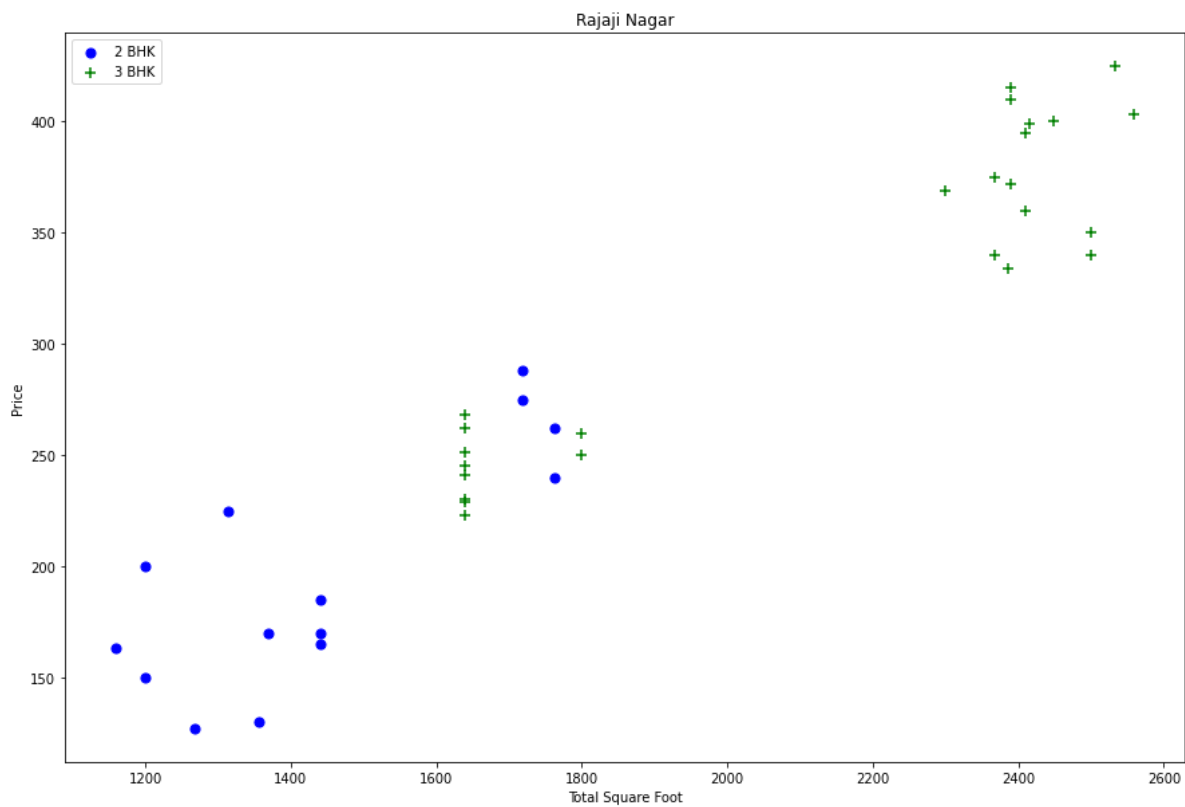
Out[51]:

(7329, 7)

In [52]:



```
plot_scatter_chart(data4, "Rajaji Nagar")
```

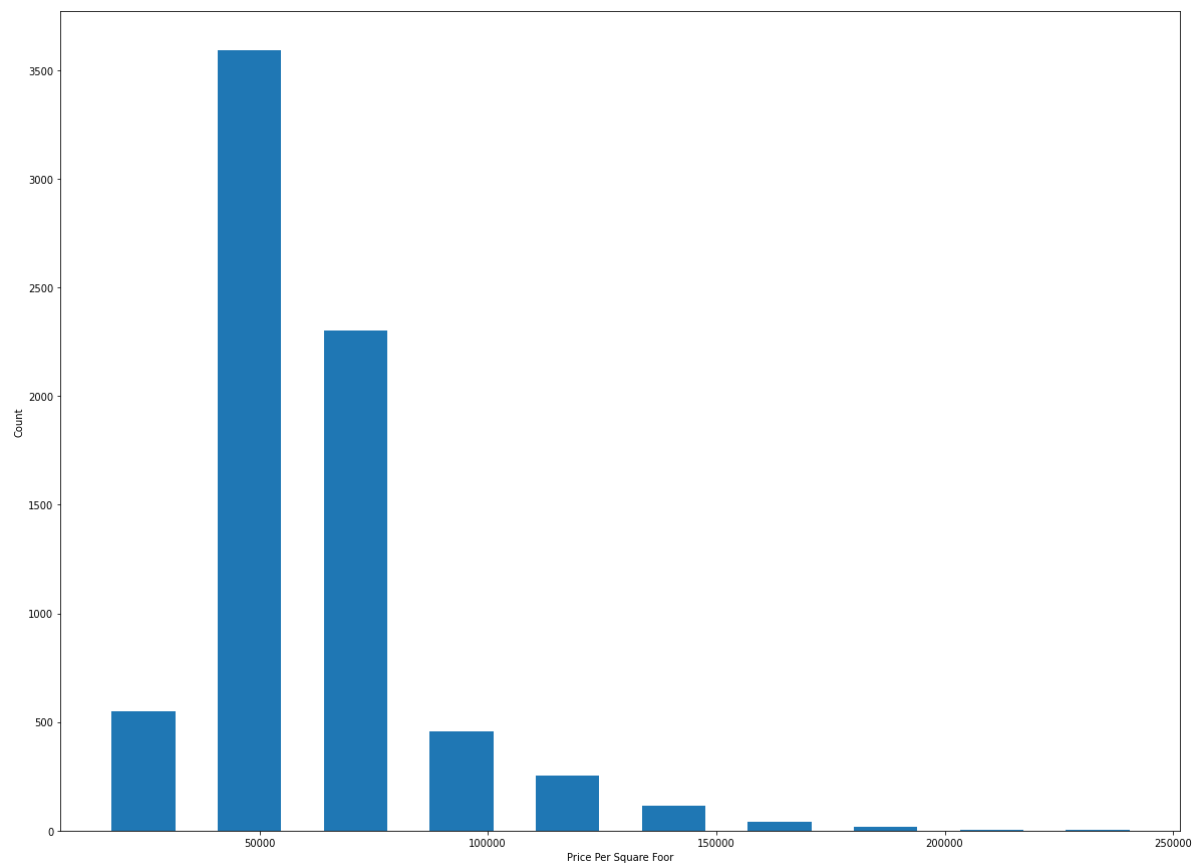


In [53]:

```
plt.rcParams['figure.figsize']=(20,15)  
plt.hist(data4.price_per_sqft,rwidth=0.6)  
plt.xlabel("Price Per Square Foot")  
plt.ylabel("Count")
```

Out[53]:

Text(0, 0.5, 'Count')



In [54]:



```
data4.bath.unique()
```

Out[54]:

```
array([ 4.,  3.,  2.,  5.,  8.,  1.,  6.,  7.,  9., 12., 16., 13.])
```

In [55]:



```
data4[data4.bath>10]
```

Out[55]:

	location	size	total_sqft	bath	price	BHK	price_per_sqft
5277	Neeladri Nagar	10 BHK	4000.0	12.0	160.0	10	40000.00000
8486	other	10 BHK	12000.0	12.0	525.0	10	43750.00000
8575	other	16 BHK	10000.0	16.0	550.0	16	55000.00000
9308	other	11 BHK	6000.0	12.0	150.0	11	25000.00000
9639	other	13 BHK	5425.0	13.0	275.0	13	50691.24424

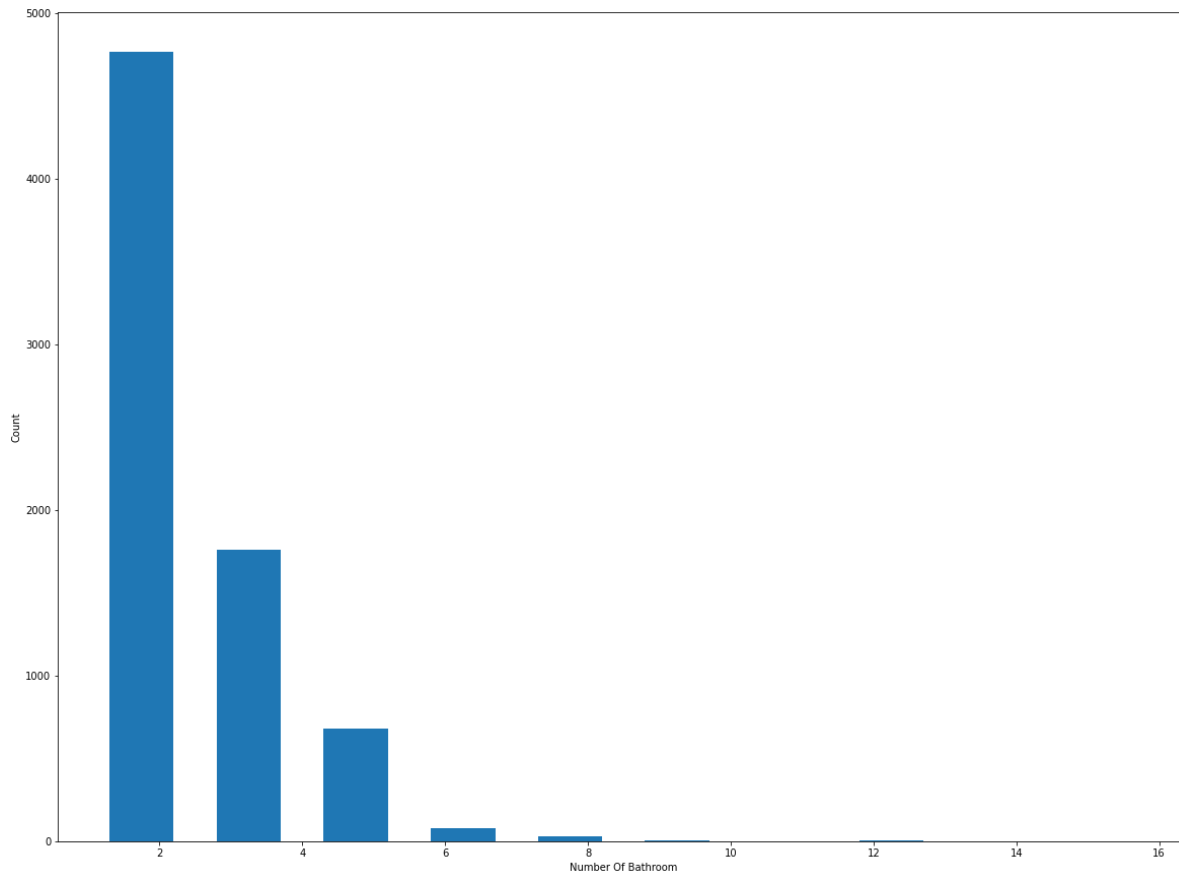
In [56]:



```
plt.rcParams['figure.figsize']=(20,15)
plt.hist(data4.bath,rwidth=0.6)
plt.xlabel("Number Of Bathroom")
plt.ylabel("Count")
```

Out[56]:

Text(0, 0.5, 'Count')



In [63]:



```
data4[data4.bath>data4.BHK+2]
```

Out[63]:

	location	size	total_sqft	bath	price	BHK	price_per_sqft
1626	Chikkabanavar	4 Bedroom	2460.0	7.0	80.0	4	32520.325203
5238	Nagasandra	4 Bedroom	7000.0	8.0	450.0	4	64285.714286
6711	Thanisandra	3 BHK	1806.0	6.0	116.0	3	64230.343300
8411	other	6 BHK	11338.0	9.0	1000.0	6	88198.976892

In [64]:

```
data5=data4[data4.bath<data4.BHK+2]  
data5.shape
```

Out[64]:

(7251, 7)

In [65]:

```
data6=data5.drop(['size','price_per_sqft'],axis='columns')  
data6
```

Out[65]:

	location	total_sqft	bath	price	BHK
0	1st Block Jayanagar	2850.0	4.0	428.0	4
1	1st Block Jayanagar	1630.0	3.0	194.0	3
2	1st Block Jayanagar	1875.0	2.0	235.0	3
3	1st Block Jayanagar	1200.0	2.0	130.0	3
4	1st Block Jayanagar	1235.0	2.0	148.0	2
...
10232	other	1200.0	2.0	70.0	2
10233	other	1800.0	1.0	200.0	1
10236	other	1353.0	2.0	110.0	2
10237	other	812.0	1.0	26.0	1
10240	other	3600.0	5.0	400.0	4

7251 rows × 5 columns

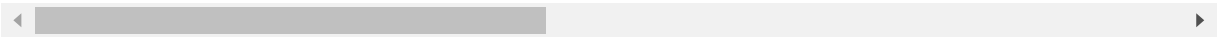
In [66]:

```
dummies=pd.get_dummies(data6.location)
dummies.head(10)
```

Out[66]:

	1st Block Jayanagar	1st Phase JP Nagar	2nd Phase Judicial Layout	2nd Stage Nagarbhavi	5th Block Hbr Layout	5th Phase JP Nagar	6th Phase JP Nagar	7th Phase JP Nagar	8th Phase JP Nagar	9th Phase JP Nagar	...	V
0	1	0	0	0	0	0	0	0	0	0	...	
1	1	0	0	0	0	0	0	0	0	0	...	
2	1	0	0	0	0	0	0	0	0	0	...	
3	1	0	0	0	0	0	0	0	0	0	...	
4	1	0	0	0	0	0	0	0	0	0	...	
5	1	0	0	0	0	0	0	0	0	0	...	
6	1	0	0	0	0	0	0	0	0	0	...	
8	0	1	0	0	0	0	0	0	0	0	...	
9	0	1	0	0	0	0	0	0	0	0	...	
10	0	1	0	0	0	0	0	0	0	0	...	

10 rows × 242 columns



In [67]:

```
data7=pd.concat([data6,dummies.drop('other',axis='columns')],axis='columns')
data7.head()
```

Out[67]:

	location	total_sqft	bath	price	BHK	1st Block Jayanagar	1st Phase JP Nagar	2nd Phase Judicial Layout	2nd Stage Nagarbhavi	5th Block Hbr Layout	...	
0	1st Block Jayanagar	2850.0	4.0	428.0	4	1	0	0	0	0	...	
1	1st Block Jayanagar	1630.0	3.0	194.0	3	1	0	0	0	0	...	
2	1st Block Jayanagar	1875.0	2.0	235.0	3	1	0	0	0	0	...	
3	1st Block Jayanagar	1200.0	2.0	130.0	3	1	0	0	0	0	...	
4	1st Block Jayanagar	1235.0	2.0	148.0	2	1	0	0	0	0	...	

5 rows × 246 columns

In [68]:

```
data8=data7.drop('location',axis='columns')
data8.head()
```

Out[68]:

	total_sqft	bath	price	BHK	1st Block Jayanagar	1st Phase JP Nagar	2nd Phase Judicial Layout	2nd Stage Nagarbhavi	5th Block Hbr Layout	5th Phase JP Nagar	...	Vijaya
0	2850.0	4.0	428.0	4	1	0	0	0	0	0	...	
1	1630.0	3.0	194.0	3	1	0	0	0	0	0	...	
2	1875.0	2.0	235.0	3	1	0	0	0	0	0	...	
3	1200.0	2.0	130.0	3	1	0	0	0	0	0	...	
4	1235.0	2.0	148.0	2	1	0	0	0	0	0	...	

5 rows × 245 columns

In [69]:

data8.shape

Out[69]:

(7251, 245)

In [70]:

```
X=data8.drop('price',axis='columns')
X.head()
```

Out[70]:

	total_sqft	bath	BHK	1st Block Jayanagar	1st Phase JP Nagar	2nd Phase Judicial Layout	2nd Stage Nagarbhavi	5th Block Hbr Layout	5th Phase JP Nagar	6th Phase JP Nagar	...	Vija
0	2850.0	4.0	4	1	0	0	0	0	0	0	...	
1	1630.0	3.0	3	1	0	0	0	0	0	0	...	
2	1875.0	2.0	3	1	0	0	0	0	0	0	...	
3	1200.0	2.0	3	1	0	0	0	0	0	0	...	
4	1235.0	2.0	2	1	0	0	0	0	0	0	...	

5 rows × 244 columns

In [71]:

y=data8.price

In [72]:

X_train = X.iloc[:5802]

In [73]:

y_train = y.iloc[:5802]

In [74]:

X_test = X.iloc[5802:7252]

In [75]:

y_test = y.iloc[5802:7252]

In [76]:



```
from sklearn.linear_model import LinearRegression
model=LinearRegression()
model.fit(X_train,y_train)
model.score(X_test,y_test)
```

Out[76]:

0.755747331402168

In [77]:



```
from sklearn.model_selection import ShuffleSplit
from sklearn.model_selection import cross_val_score

cv = ShuffleSplit(n_splits=5, test_size=0.2, random_state=0)

cross_val_score(LinearRegression(), X, y, cv=cv)
```

Out[77]:

array([0.82430186, 0.77166234, 0.85089567, 0.80837764, 0.83653286])

In [78]:

```

from sklearn.model_selection import GridSearchCV

from sklearn.linear_model import Lasso
from sklearn.tree import DecisionTreeRegressor

def find_best_model_using_gridsearchcv(X,y):
    algos = {
        'linear_regression' : {
            'model': LinearRegression(),
            'params': {
                'normalize': [True, False]
            }
        },
        'lasso': {
            'model': Lasso(),
            'params': {
                'alpha': [1,2],
                'selection': ['random', 'cyclic']
            }
        },
        'decision_tree': {
            'model': DecisionTreeRegressor(),
            'params': {
                'criterion' : ['mse','friedman_mse'],
                'splitter': ['best','random']
            }
        }
    }
    scores = []
    cv = ShuffleSplit(n_splits=5, test_size=0.2, random_state=0)
    for algo_name, config in algos.items():
        gs = GridSearchCV(config['model'], config['params'], cv=cv, return_train_score=False)
        gs.fit(X,y)
        scores.append({
            'model': algo_name,
            'best_score': gs.best_score_,
            'best_params': gs.best_params_
        })

    return pd.DataFrame(scores,columns=['model','best_score','best_params'])

find_best_model_using_gridsearchcv(X,y)

```

Out[78]:

	model	best_score	best_params
0	linear_regression	0.818354	{'normalize': True}
1	lasso	0.687429	{'alpha': 1, 'selection': 'cyclic'}
2	decision_tree	0.718892	{'criterion': 'mse', 'splitter': 'random'}

In [79]:



```
def price_predict(location,sqft,bath,BHK):  
    loc_index=np.where(X.columns==location)[0][0]  
    x=np.zeros(len(X.columns))  
    x[0]=sqft  
    x[1]=bath  
    x[2]=BHK  
    if loc_index >=0:  
        x[loc_index]=1  
    return model.predict([x])[0]
```

In [80]:



```
price_predict('1st Phase JP Nagar',1000,2,2)
```

Out[80]:

86.50537337722247

In [81]:



```
price_predict('1st Phase JP Nagar',1000,2,3)
```

Out[81]:

81.9696568636569

In [82]:



```
price_predict('5th Phase JP Nagar',1000,2,2)
```

Out[82]:

38.93415026548578

In [83]:



```
price_predict('Indira Nagar',1000,2,2)
```

Out[83]:

180.82820686320383