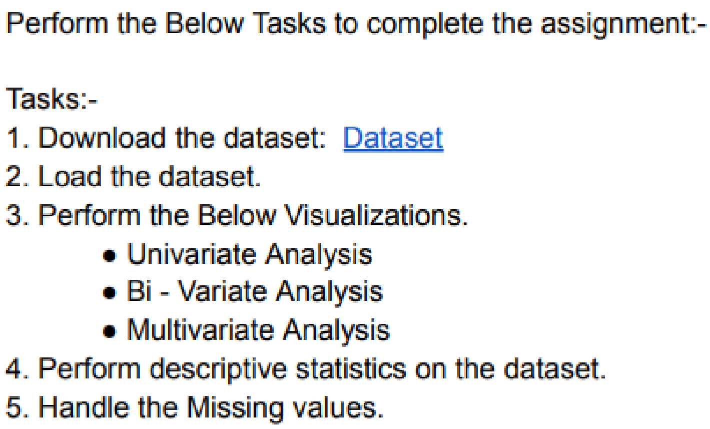
Assignment 2

Charvi Upreti

[charvi.upreti2021@vitstudent.ac.in (mailto:charvi.upreti2021@vitstudent.ac.in)](mailto:charvi.upreti2021@vitstudent.ac.in)

21BCE1440



Perform the Below Tasks to complete the assignment:-

Tasks:-

1. Download the dataset: Dataset
2. Load the dataset.
3. Perform the Below Visualizations.
    - Univariate Analysis
    - Bi - Variate Analysis
    - Multivariate Analysis
4. Perform descriptive statistics on the dataset.
5. Handle the Missing values.

Import the required libraries

In [1]:

```
#import required libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from matplotlib import rcParams
```

Task 1 and 2: Dataset was Downloaded and then loaded

link of dataset: https://www.kaggle.com/datasets/mohamedafsal007/house-price-dataset-of-india (https://www.kaggle.com/datasets/mohamedafsal007/house-price-dataset-of-india)

In [2]:

```
df=pd.read_csv("C:/Users/Charvi Upreti/Desktop/Assignments/Assignment 2/House Price India.csv")
```

Some observations

In [3]:

```
df.shape
```

Out[3]:

```
(14620, 23)
```

In [4]:

```
df.head().T
```

Out[4]:

	0	1	2	3	4
id	6.762810e+09	6.762811e+09	6.762811e+09	6.762813e+09	6.762813e+09
Date	4.249100e+04	4.249100e+04	4.249100e+04	4.249100e+04	4.249100e+04
number of bedrooms	5.000000e+00	4.000000e+00	5.000000e+00	4.000000e+00	3.000000e+00
number of bathrooms	2.500000e+00	2.500000e+00	2.750000e+00	2.500000e+00	2.000000e+00
living area	3.650000e+03	2.920000e+03	2.910000e+03	3.310000e+03	2.710000e+03
lot area	9.050000e+03	4.000000e+03	9.480000e+03	4.299800e+04	4.500000e+03
number of floors	2.000000e+00	1.500000e+00	1.500000e+00	2.000000e+00	1.500000e+00
waterfront present	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
number of views	4.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
condition of the house	5.000000e+00	5.000000e+00	3.000000e+00	3.000000e+00	4.000000e+00
grade of the house	1.000000e+01	8.000000e+00	8.000000e+00	9.000000e+00	8.000000e+00
Area of the house(excluding basement)	3.370000e+03	1.910000e+03	2.910000e+03	3.310000e+03	1.880000e+03
Area of the basement	2.800000e+02	1.010000e+03	0.000000e+00	0.000000e+00	8.300000e+02
Built Year	1.921000e+03	1.909000e+03	1.939000e+03	2.001000e+03	1.929000e+03
Renovation Year	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
Postal Code	1.220030e+05	1.220040e+05	1.220040e+05	1.220050e+05	1.220060e+05
Lattitude	5.286450e+01	5.288780e+01	5.288520e+01	5.295320e+01	5.290470e+01
Longitude	-1.145570e+02	-1.144700e+02	-1.144680e+02	-1.143210e+02	-1.144850e+02
living_area_renov	2.880000e+03	2.470000e+03	2.940000e+03	3.350000e+03	2.060000e+03
lot_area_renov	5.400000e+03	4.000000e+03	6.600000e+03	4.284700e+04	4.500000e+03
Number of schools nearby	2.000000e+00	2.000000e+00	1.000000e+00	3.000000e+00	1.000000e+00
Distance from the airport	5.800000e+01	5.100000e+01	5.300000e+01	7.600000e+01	5.100000e+01
Price	2.380000e+06	1.400000e+06	1.200000e+06	8.380000e+05	8.050000e+05

In [5]:

```
print(df.isnull().sum())
```

```
id                                        0
Date                                      0
number of bedrooms                        0
number of bathrooms                       0
living area                               0
lot area                                  0
number of floors                          0
waterfront present                        0
number of views                           0
condition of the house                    0
grade of the house                        0
Area of the house(excluding basement)     0
Area of the basement                      0
Built Year                                0
Renovation Year                           0
Postal Code                               0
Lattitude                                 0
Longitude                                 0
living_area_renov                         0
lot_area_renov                            0
Number of schools nearby                  0
Distance from the airport                 0
Price                                     0
dtype: int64
```

In [6]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14620 entries, 0 to 14619
Data columns (total 23 columns):
 #   Column                                 Non-Null Count  Dtype  
---  ------                                 --------------  -----  
 0   id                                     14620 non-null  int64  
 1   Date                                   14620 non-null  int64  
 2   number of bedrooms                     14620 non-null  int64  
 3   number of bathrooms                    14620 non-null  float64
 4   living area                            14620 non-null  int64  
 5   lot area                               14620 non-null  int64  
 6   number of floors                       14620 non-null  float64
 7   waterfront present                     14620 non-null  int64  
 8   number of views                        14620 non-null  int64  
 9   condition of the house                 14620 non-null  int64  
 10  grade of the house                     14620 non-null  int64  
 11  Area of the house(excluding basement)  14620 non-null  int64  
 12  Area of the basement                   14620 non-null  int64  
 13  Built Year                             14620 non-null  int64  
 14  Renovation Year                        14620 non-null  int64  
 15  Postal Code                            14620 non-null  int64  
 16  Lattitude                              14620 non-null  float64
 17  Longitude                              14620 non-null  float64
 18  living_area_renov                      14620 non-null  int64  
 19  lot_area_renov                         14620 non-null  int64  
 20  Number of schools nearby               14620 non-null  int64  
 21  Distance from the airport              14620 non-null  int64  
 22  Price                                  14620 non-null  int64  
dtypes: float64(4), int64(19)
memory usage: 2.6 MB
```

Task 3: Visualizations.

In [7]:

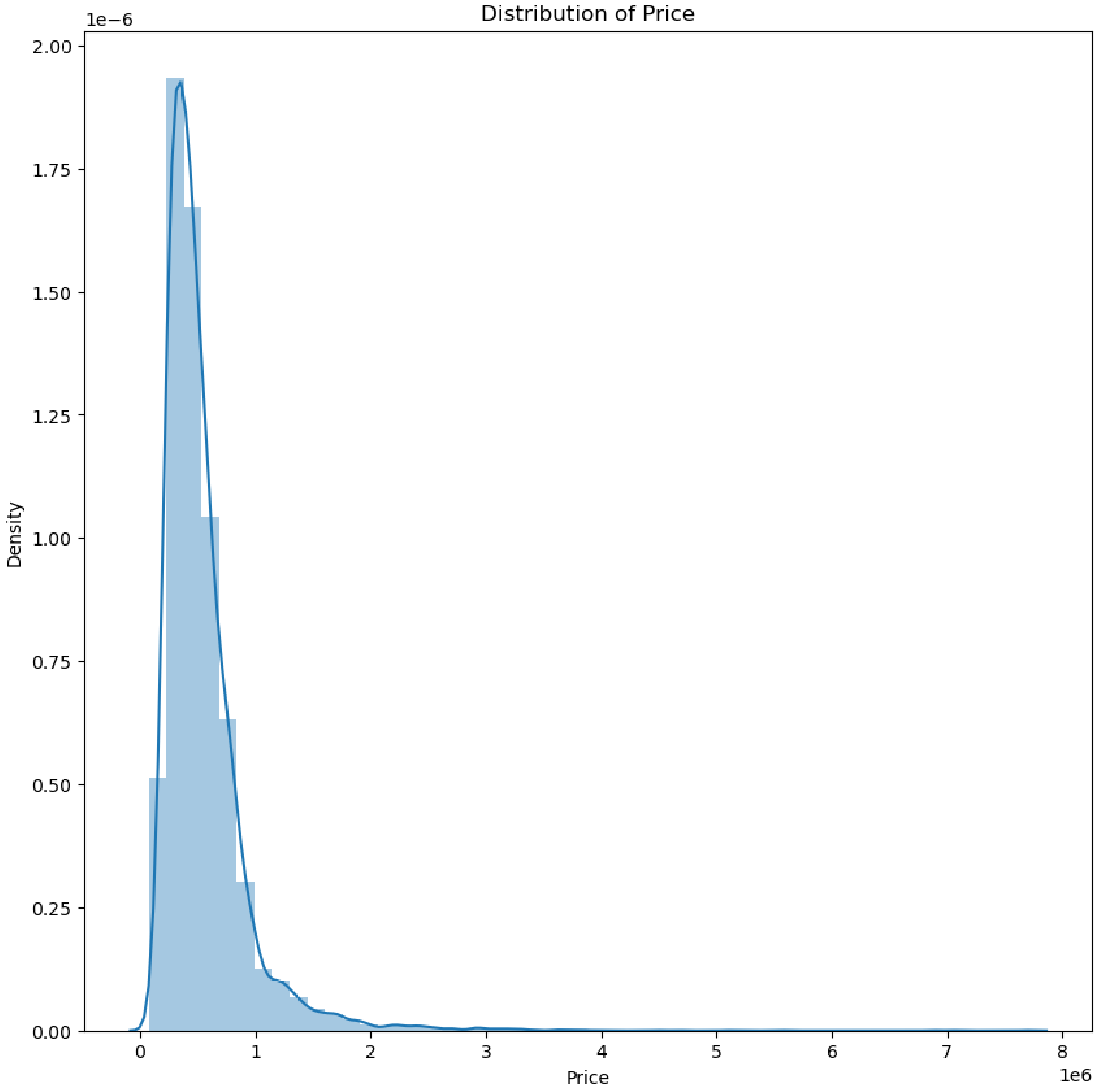
```
rcParams['figure.figsize']=10,10
```

Univariate analysis

In [8]:

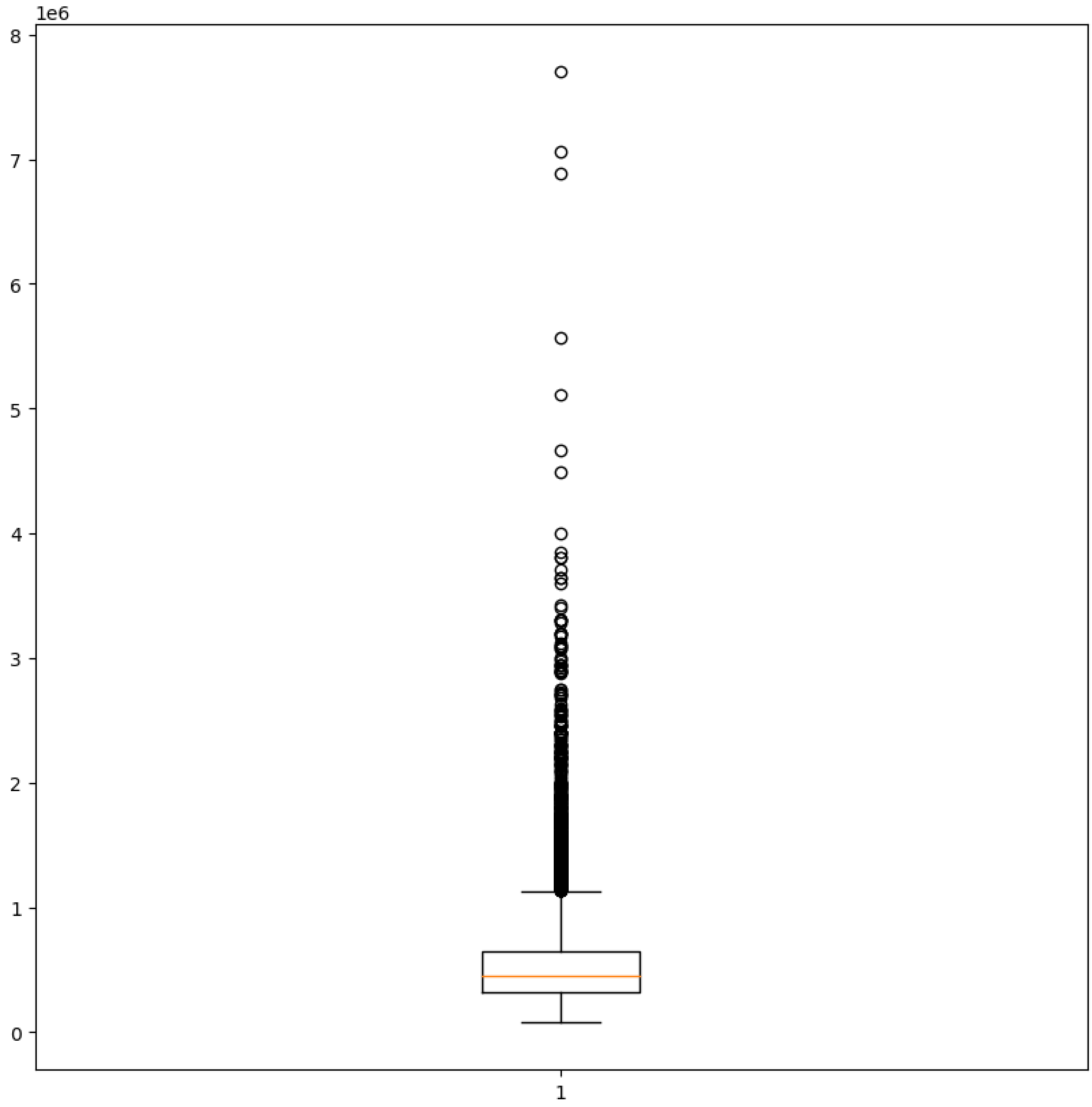
```
# Distribution of Price
sns.distplot(df["Price"])
plt.title("Distribution of Price")
plt.show()
```

```
C:\Users\Charvi Upreti\AppData\Local\Temp\ipykernel_11220\1667094637.py:2: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751 (https://gist.github.co
m/mwaskom/de44147ed2974457ad6372750bbe5751)

  sns.distplot(df["Price"])
```



In [9]:

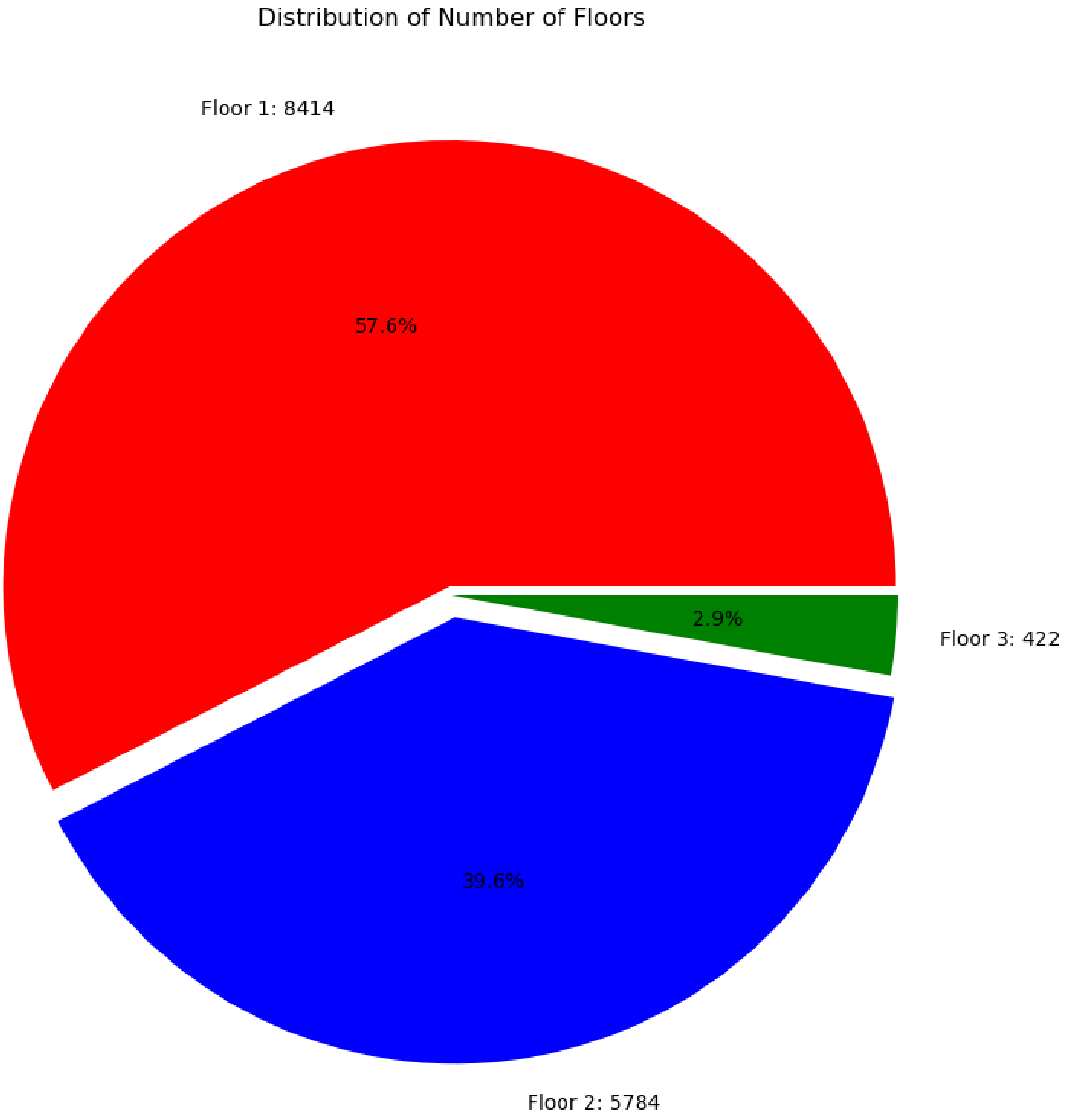
```
plt.boxplot(df.Price)
plt.show()
# Many outliers, ie many observations are away from normal observation
# we can replace them with median or remove them while processing
```



In [10]:

```
# Convert "number of floors" to integers, as some floors were marked in decimal points (eg 1.5) whic
df['number of floors'] = df['number of floors'].astype(int)

# Finding the number of unique floors
unique_floor_values = df['number of floors'].unique()
print(unique_floor_values)

# Distribution of the number of floors
floor_counts = df["number of floors"].value_counts()
floor_values = floor_counts.index
floor_occurrences = floor_counts.values

labels = [f"Floor {floor}: {count}" for floor, count in zip(floor_values, floor_occurrences)]

plt.pie(floor_occurrences,[0.02,0.05,0],labels=labels, autopct='%1.1f%%',colors=['red','blue','gree
plt.title('Distribution of Number of Floors')
plt.show()

#Clearly there are maximum houses with 1 floors.

```

```
[2 1 3]
```



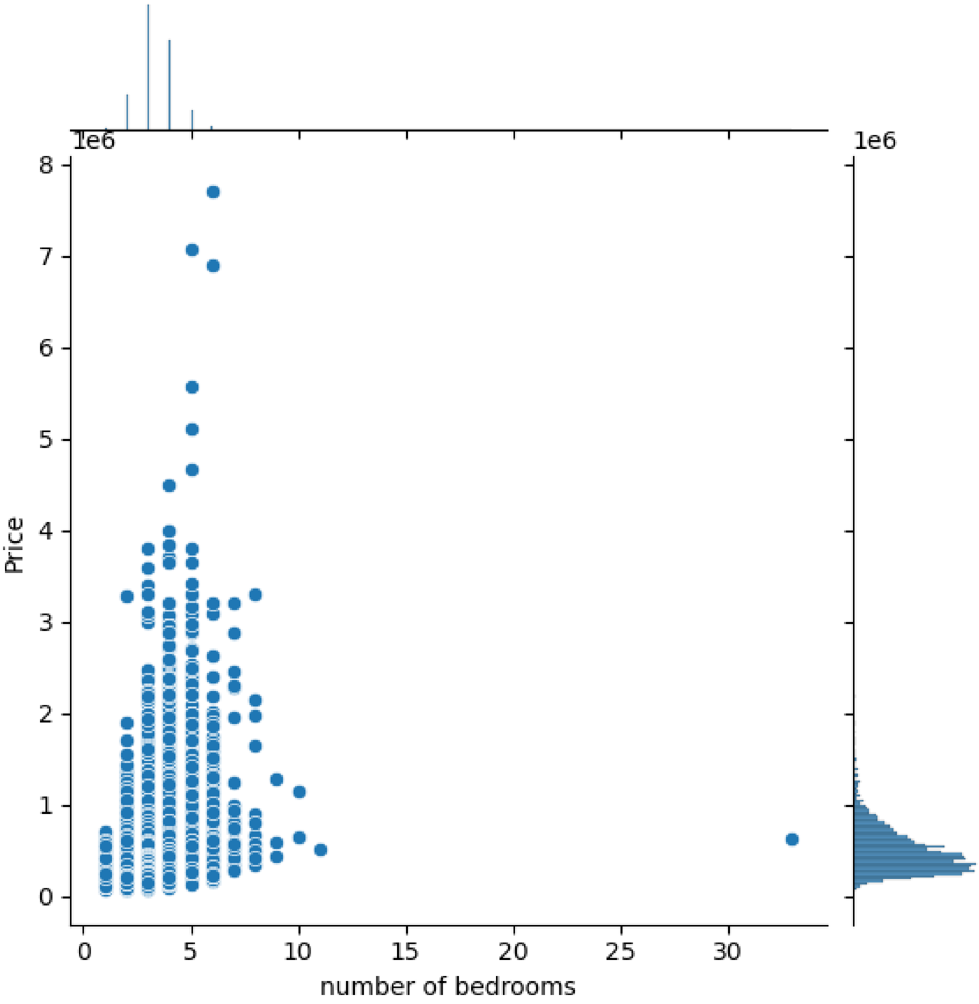
Bi - variate analysis

In [11]:

```
sns.jointplot(x='number of bedrooms',y='Price',data=df)
```

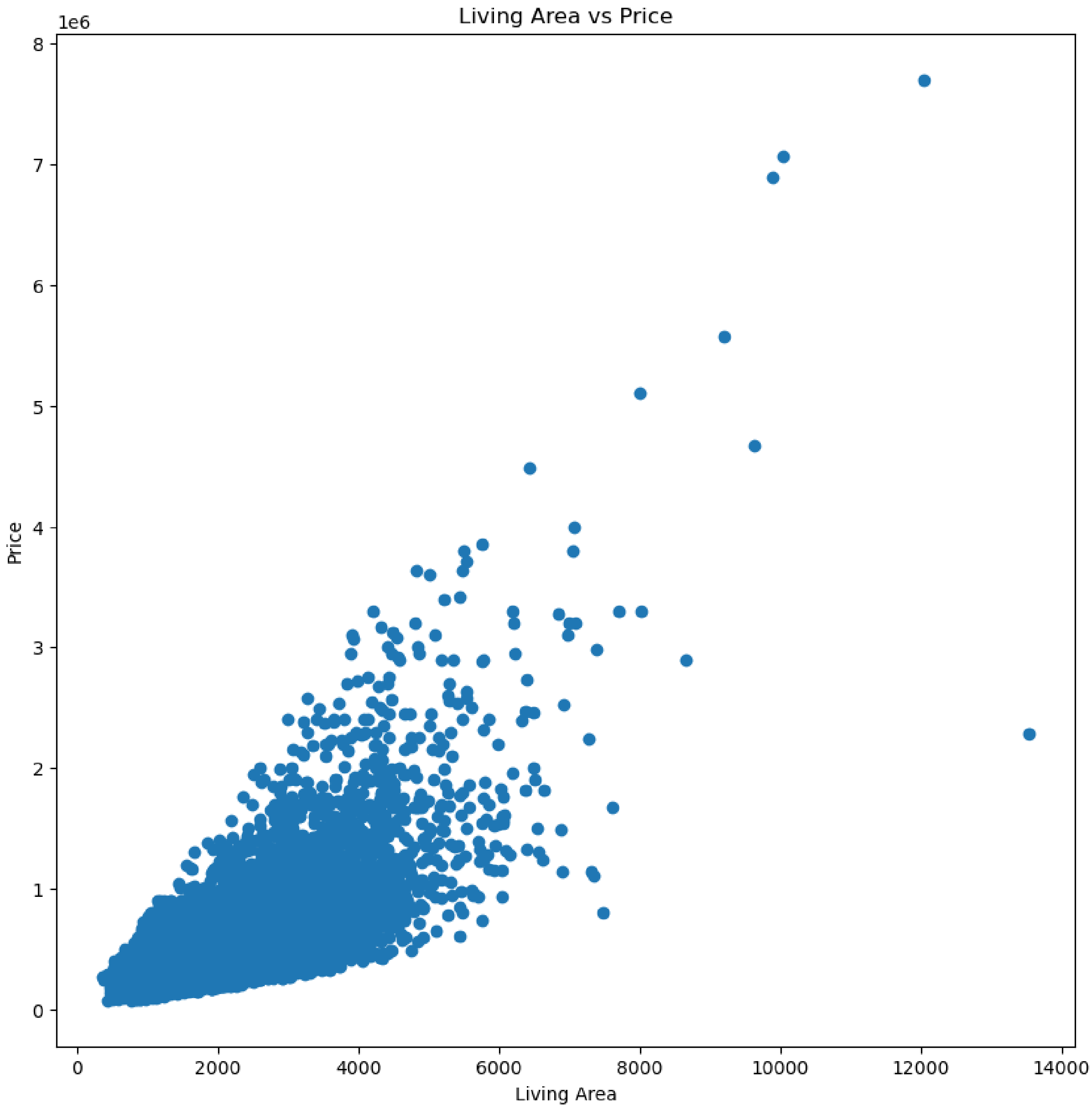
Out[11]:

```
<seaborn.axisgrid.JointGrid at 0x1f7fa2368b0>
```



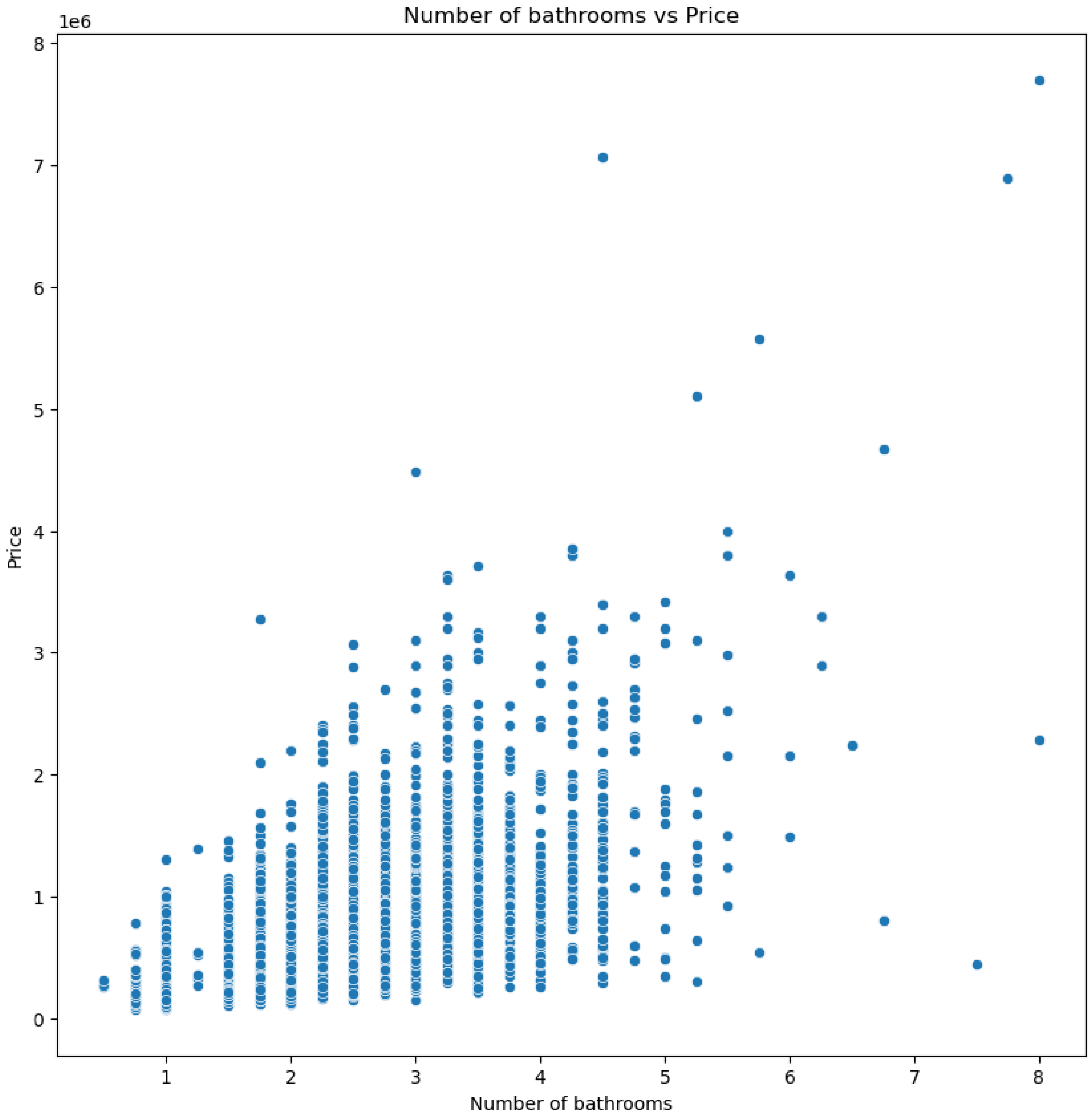
In [12]:

```python
plt.scatter(df['living area'], df['Price'])
plt.xlabel('Living Area')
plt.ylabel('Price')
plt.title('Living Area vs Price')
plt.show()
```



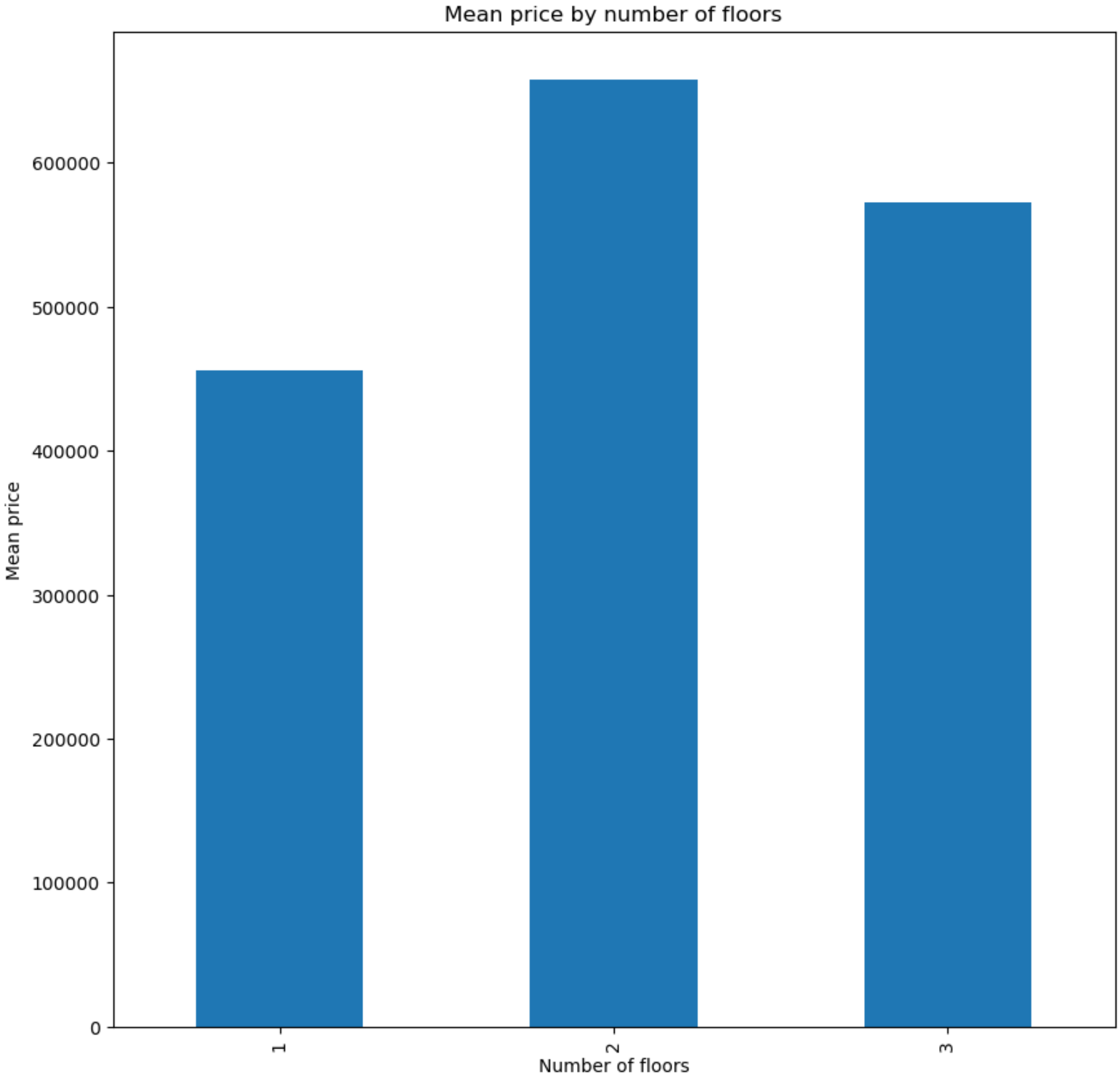
In [13]:

```
sns.scatterplot(x='number of bathrooms',y='Price',data=df)
plt.xlabel('Number of bathrooms')
plt.ylabel('Price')
plt.title('Number of bathrooms vs Price')
plt.show()
```



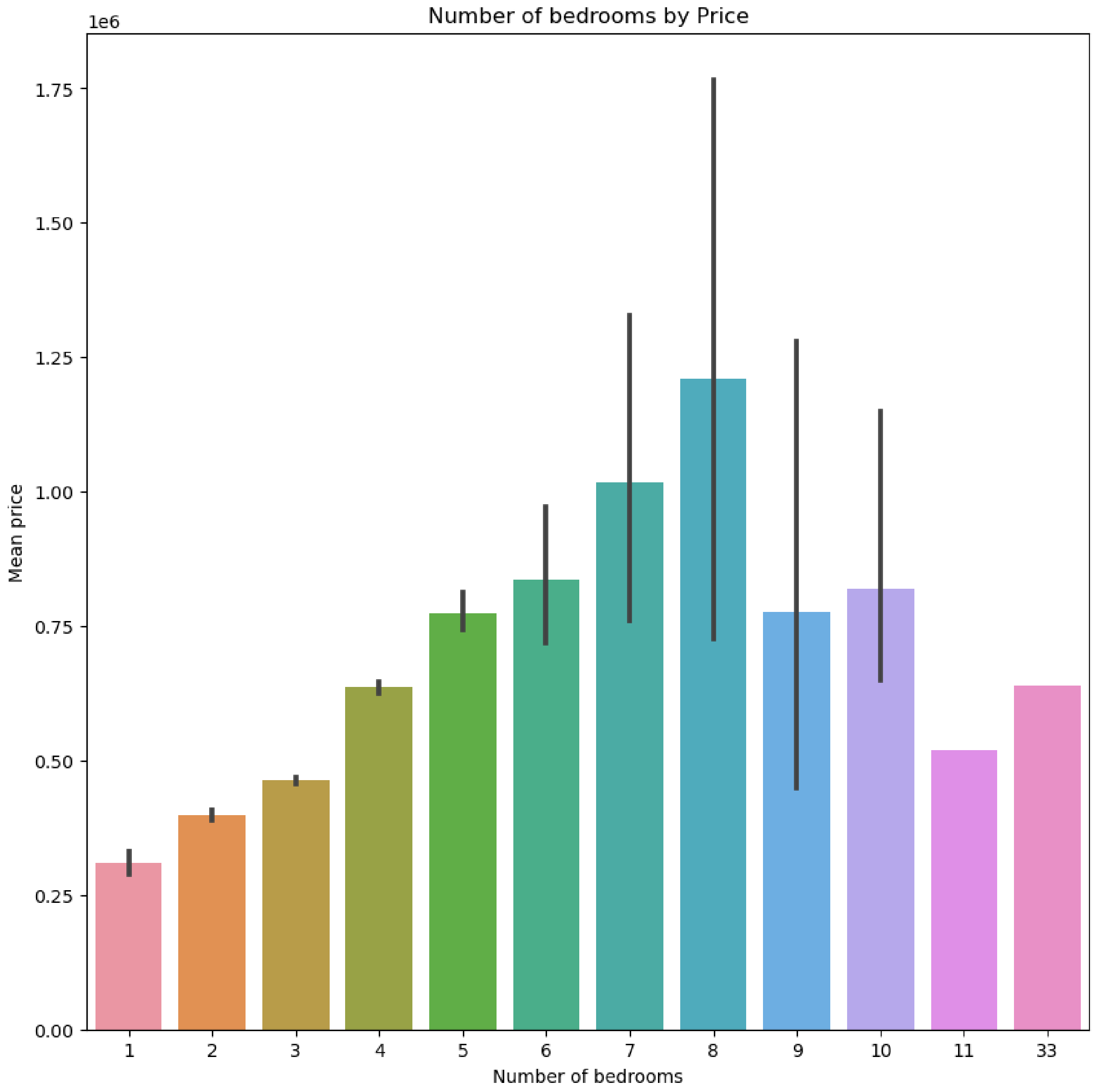
In [14]:

```
mean_price_by_floors = df.groupby('number of floors')['Price'].mean()
mean_price_by_floors.plot(kind='bar')
plt.xlabel('Number of floors')
plt.ylabel('Mean price')
plt.title('Mean price by number of floors')
plt.show()
#mean price of 2 floors is maximum
```



In [15]:

```
sns.barplot(x='number of bedrooms',y='Price',data=df)
plt.xlabel('Number of bedrooms')
plt.ylabel('Mean price')
plt.title('Number of bedrooms by Price')
plt.show()
```



Multivariate analysis

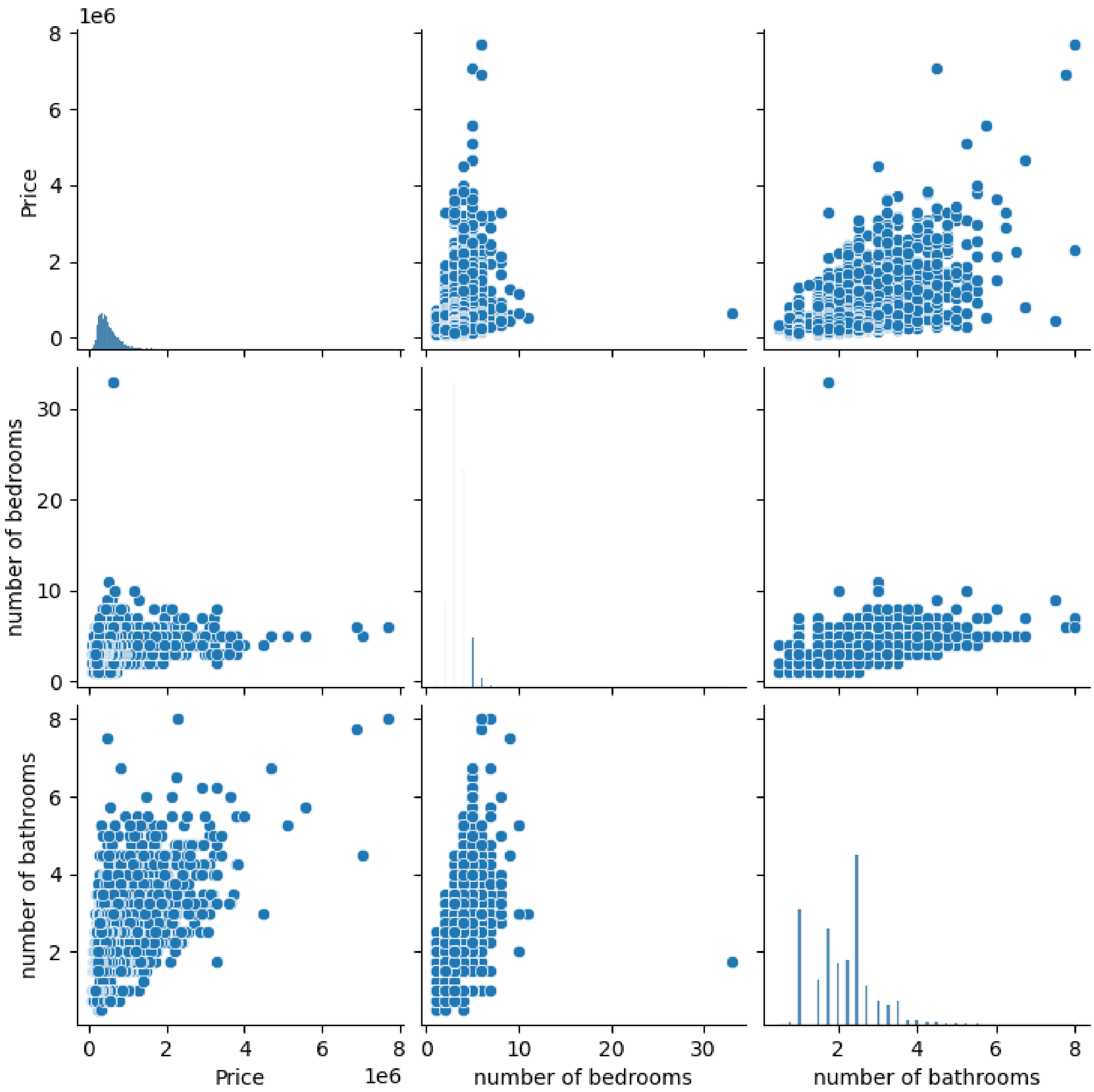
In [16]:

```
columns_to_plot = ["Price", "number of bedrooms", "number of bathrooms"]
sns.pairplot(df[columns_to_plot])
#also sns.pairplot(df)
```

```
C:\Users\Charvi Upreti\anaconda3\lib\site-packages\seaborn\axisgrid.py:118: UserWarning:
The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```

Out[16]:

```
<seaborn.axisgrid.PairGrid at 0x1f7804785b0>
```

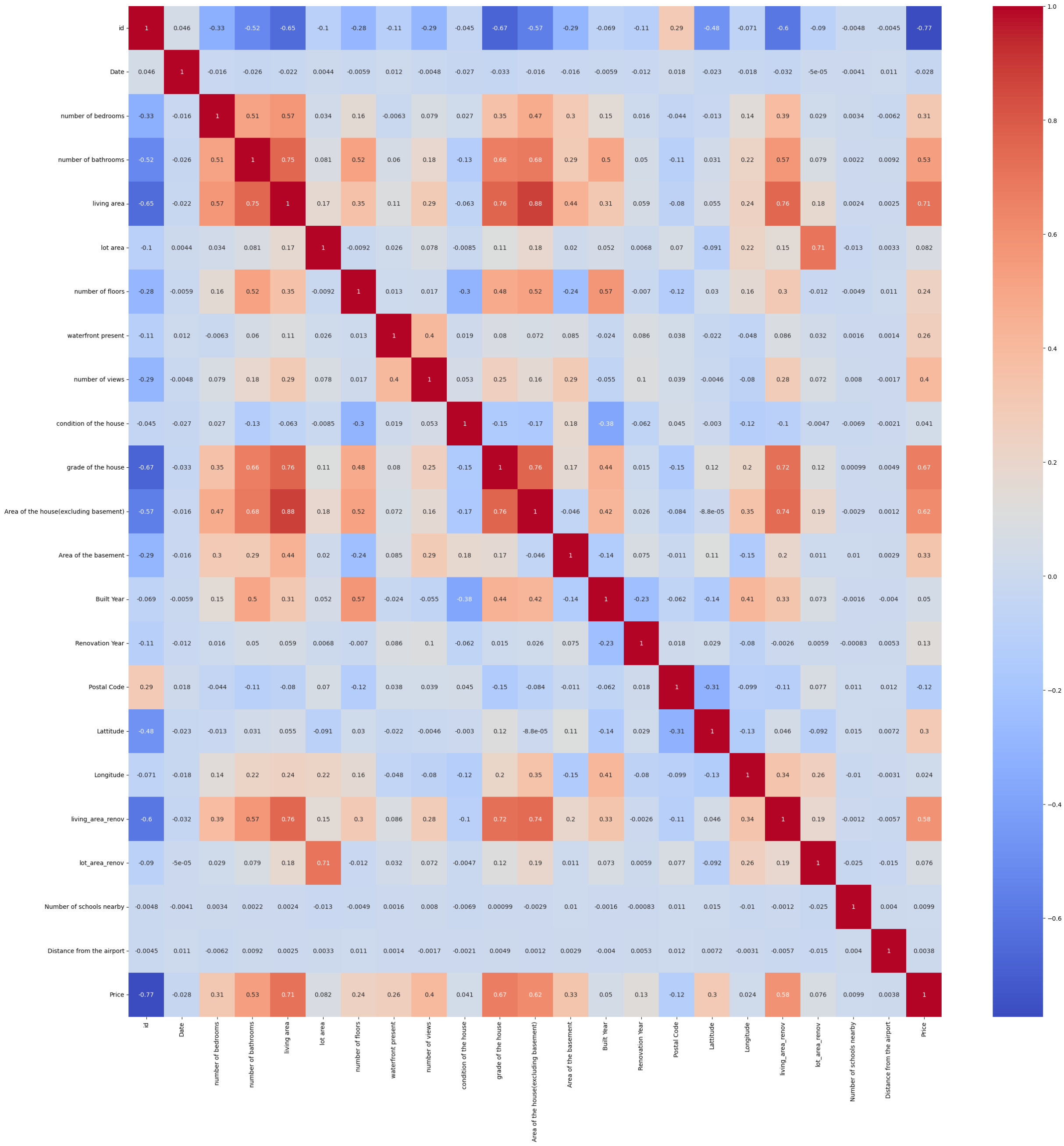


In [17]:

```
corr = df.corr()
plt.figure(figsize=(30,30))
sns.heatmap(corr,annot=True,cmap="coolwarm")
```

Out[17]:

```
<Axes: >
```



Task 4: Perform descriptive statistics on the dataset

In [18]:

```
df.describe().transpose()
```

Out[18]:

	count	mean	std	min	25%	50%	
id	14620.0	6.762821e+09	6237.574799	6.762810e+09	6.762815e+09	6.762821e+09	6.762826
Date	14620.0	4.260454e+04	67.347991	4.249100e+04	4.254600e+04	4.260000e+04	4.266200
number of bedrooms	14620.0	3.379343e+00	0.938719	1.000000e+00	3.000000e+00	3.000000e+00	4.000000
number of bathrooms	14620.0	2.129583e+00	0.769934	5.000000e-01	1.750000e+00	2.250000e+00	2.500000
living area	14620.0	2.098263e+03	928.275721	3.700000e+02	1.440000e+03	1.930000e+03	2.570000
lot area	14620.0	1.509328e+04	37919.621304	5.200000e+02	5.010750e+03	7.620000e+03	1.080000
number of floors	14620.0	1.453352e+00	0.552787	1.000000e+00	1.000000e+00	1.000000e+00	2.000000
waterfront present	14620.0	7.660739e-03	0.087193	0.000000e+00	0.000000e+00	0.000000e+00	0.000000
number of views	14620.0	2.331053e-01	0.766259	0.000000e+00	0.000000e+00	0.000000e+00	0.000000
condition of the house	14620.0	3.430506e+00	0.664151	1.000000e+00	3.000000e+00	3.000000e+00	4.000000
grade of the house	14620.0	7.682421e+00	1.175033	4.000000e+00	7.000000e+00	7.000000e+00	8.000000
Area of the house(excluding basement)	14620.0	1.801784e+03	833.809963	3.700000e+02	1.200000e+03	1.580000e+03	2.240000
Area of the basement	14620.0	2.964791e+02	448.551409	0.000000e+00	0.000000e+00	0.000000e+00	5.800000
Built Year	14620.0	1.970926e+03	29.493625	1.900000e+03	1.951000e+03	1.975000e+03	1.997000
Renovation Year	14620.0	9.092401e+01	416.216661	0.000000e+00	0.000000e+00	0.000000e+00	0.000000
Postal Code	14620.0	1.220331e+05	19.082418	1.220030e+05	1.220170e+05	1.220320e+05	1.220480
Lattitude	14620.0	5.279285e+01	0.137522	5.238590e+01	5.270760e+01	5.280640e+01	5.290890
Longitude	14620.0	-1.144040e+02	0.141326	-1.147090e+02	-1.145190e+02	-1.144210e+02	-1.143150
living_area_renov	14620.0	1.996702e+03	691.093366	4.600000e+02	1.490000e+03	1.850000e+03	2.380000
lot_area_renov	14620.0	1.275350e+04	26058.414467	6.510000e+02	5.097750e+03	7.620000e+03	1.012500
Number of schools nearby	14620.0	2.012244e+00	0.817284	1.000000e+00	1.000000e+00	2.000000e+00	3.000000
Distance from the airport	14620.0	6.495096e+01	8.936008	5.000000e+01	5.700000e+01	6.500000e+01	7.300000
Price	14620.0	5.389322e+05	367532.380804	7.800000e+04	3.200000e+05	4.500000e+05	6.450000

Task 5: Handling missing values

In [19]:

```
df.isnull().all()
```

Out[19]:

```
id                                       False
Date                                     False
number of bedrooms                       False
number of bathrooms                      False
living area                              False
lot area                                 False
number of floors                         False
waterfront present                       False
number of views                          False
condition of the house                   False
grade of the house                       False
Area of the house(excluding basement)    False
Area of the basement                     False
Built Year                               False
Renovation Year                          False
Postal Code                              False
Lattitude                                False
Longitude                                False
living_area_renov                        False
lot_area_renov                           False
Number of schools nearby                 False
Distance from the airport                False
Price                                    False
dtype: bool
```

In [20]:

```
missing_values = df.isnull().sum()
print(missing_values)
```

```
id                                       0
Date                                     0
number of bedrooms                       0
number of bathrooms                      0
living area                              0
lot area                                 0
number of floors                         0
waterfront present                       0
number of views                          0
condition of the house                   0
grade of the house                       0
Area of the house(excluding basement)    0
Area of the basement                     0
Built Year                               0
Renovation Year                          0
Postal Code                              0
Lattitude                                0
Longitude                                0
living_area_renov                        0
lot_area_renov                           0
Number of schools nearby                 0
Distance from the airport                0
Price                                    0
dtype: int64
```

=> Hence, There are no missing values.

In case of null values we can:-

1. Drop the columns.

2 Imputation In case there were null values in categorical column we can replace null field with mode.

otherwise if in int/float column, we can replace the null field with

mean (if normal distribution.)

else with median.

we can check if normal distribution by

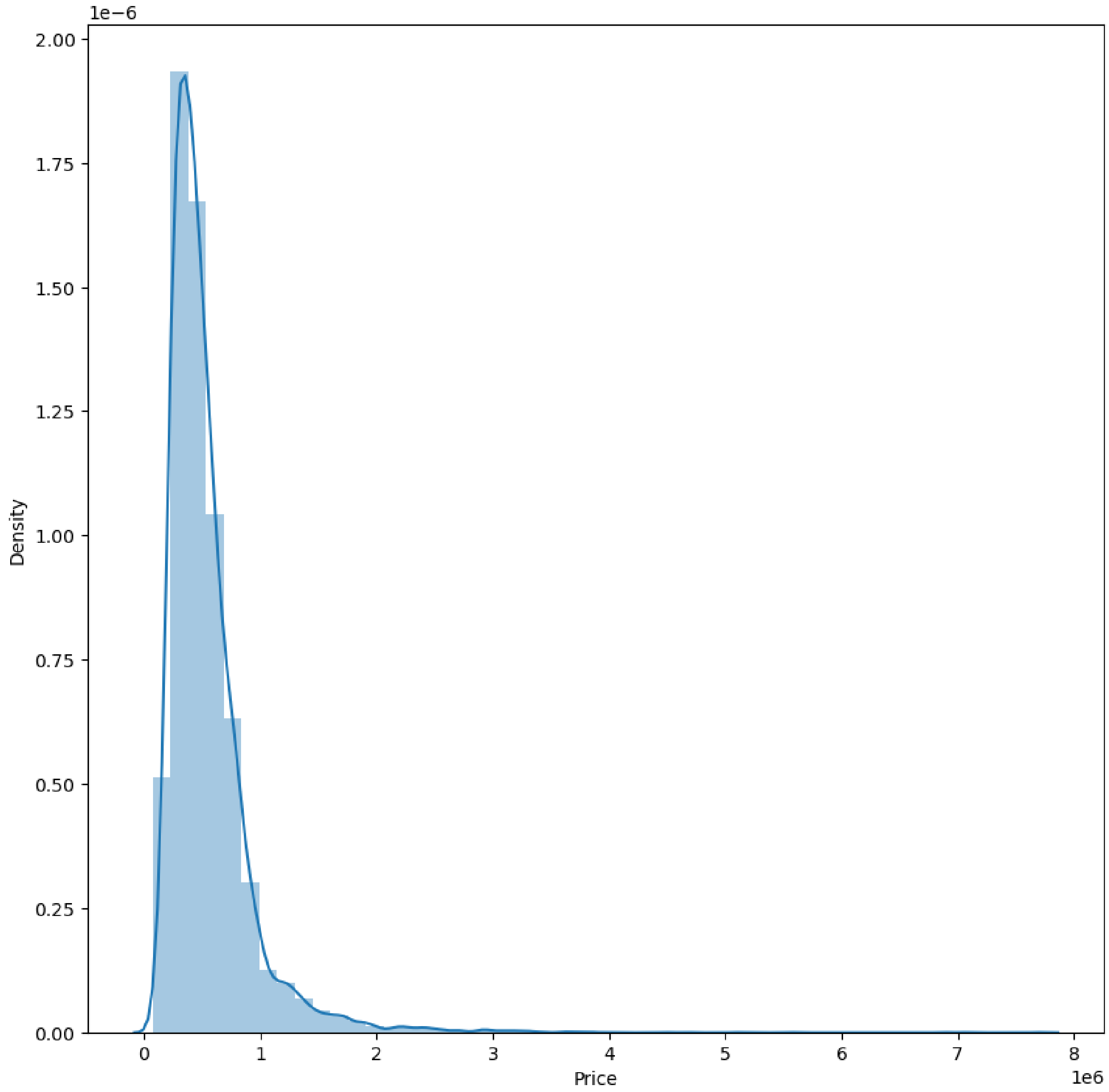
In [21]:

```
sns.distplot(df.Price)
#shape should be bell shaped curve if normal
#here we will take median
```

```
C:\Users\Charvi Upreti\AppData\Local\Temp\ipykernel_11220\1854958562.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751 (https://gist.github.co
m/mwaskom/de44147ed2974457ad6372750bbe5751)

  sns.distplot(df.Price)
```

Out[21]:

```
<Axes: xlabel='Price', ylabel='Density'>
```


1e−6
Density
Price
1e6