HRISHIKESH G KULKARNI - 21BAI1660

hrishikesh.gkulkarni2021@vitstudent.ac.in

ASSIGNMENT - 02

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

Task1 -> downloaded the dataset (Done!)

Task2 -> Load the dataset into a dataframe

In [ ]:
```
df = pd.read_csv('House Price India.csv')
df.head()
```

Out[ ]:

	id	Date	number of bedrooms	number of bathrooms	living area	lot area	number of floors	waterfront present	num vie
0	6762810145	42491	5	2.50	3650	9050	2.0	0	
1	6762810635	42491	4	2.50	2920	4000	1.5	0	
2	6762810998	42491	5	2.75	2910	9480	1.5	0	
3	6762812605	42491	4	2.50	3310	42998	2.0	0	
4	6762812919	42491	3	2.00	2710	4500	1.5	0	

5 rows × 23 columns

In [ ]:
```
df.shape
```

Out[ ]:
```
(14620, 23)
```

In [ ]:
```
df["number of floors"].value_counts()
```

Out[ ]:
```
1.0    7103
2.0    5666
1.5    1311
3.0     418
2.5     118
3.5       4
Name: number of floors, dtype: int64
```

In [ ]:
```
df["number of bedrooms"].value_counts()
```

Out[ ]:
```
3     6612
4     4724
2     1844
5     1079
6      176
1      136
7       30
8       11
9        3
10       3
33       1
11       1
Name: number of bedrooms, dtype: int64
```

In [ ]:
```
df["Number of schools nearby"].value_counts()
```

Out[ ]:
```
3    4973
2    4853
1    4794
Name: Number of schools nearby, dtype: int64
```

In [ ]:
```
df["Built Year"].value_counts()
```

Out[ ]:
```
2014    404
2005    319
2006    300
2004    296
2003    295
       ...
1902     20
1935     18
1933     17
1934     15
2015     12
Name: Built Year, Length: 116, dtype: int64
```

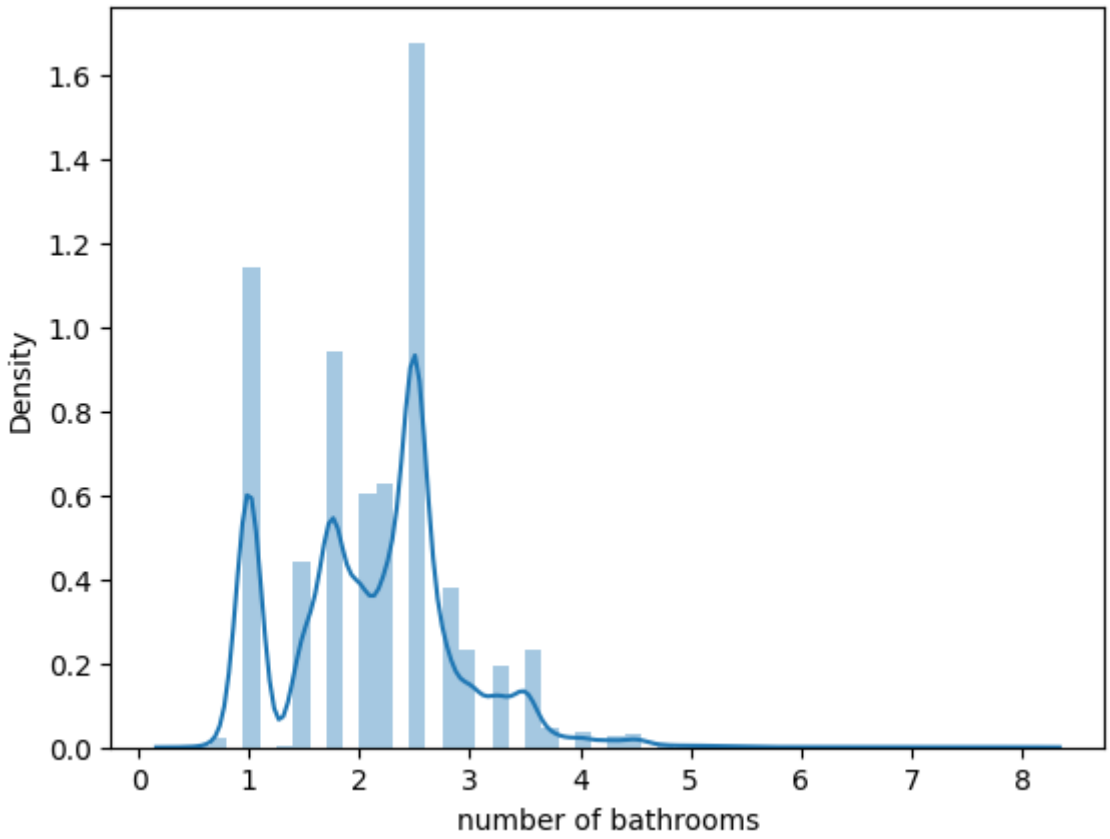
Task3 -> Doing the Visualisations (EDA)

i) Univariate Analysis

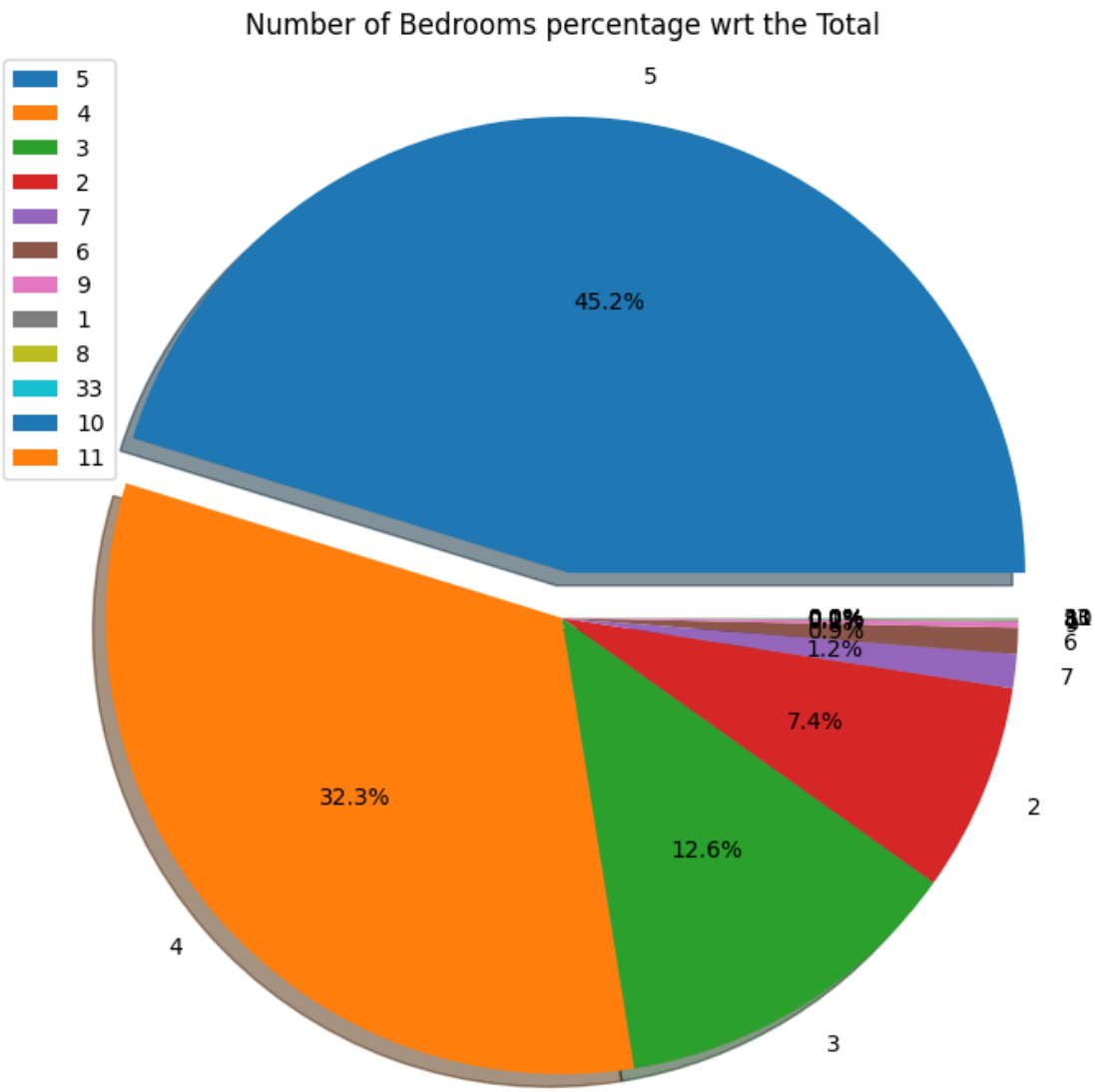
In [ ]:
```
sns.distplot(df["number of bathrooms"])
```

```
C:\Users\hrish\AppData\Local\Temp\ipykernel_11652\1889700575.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

  sns.distplot(df["number of bathrooms"])
```

Out[ ]:
```
<AxesSubplot: xlabel='number of bathrooms', ylabel='Density'>
```

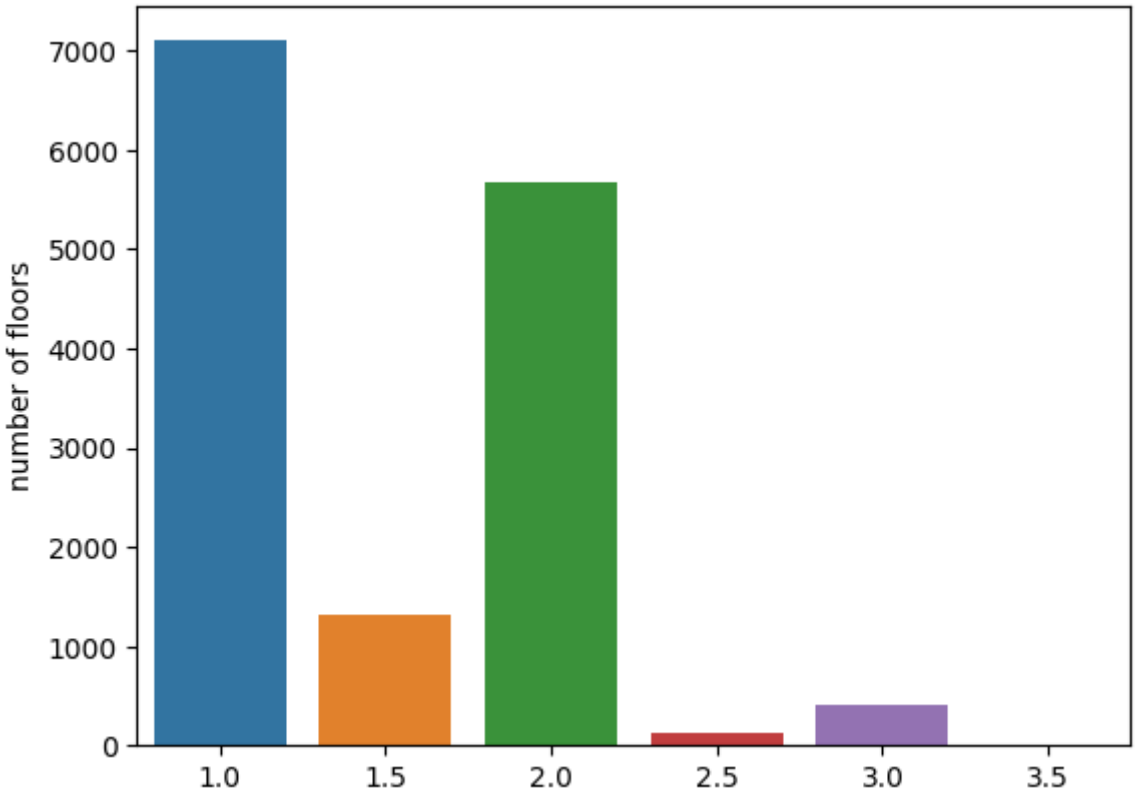


In [ ]:
```
plt.figure(figsize=(9,9))
plt.pie(df["number of bedrooms"].value_counts(),[0.1,0,0,0,0,0,0,0,0,0,0,0], lab
plt.title('Number of Bedrooms percentage wrt the Total')
plt.legend()
plt.show()
```



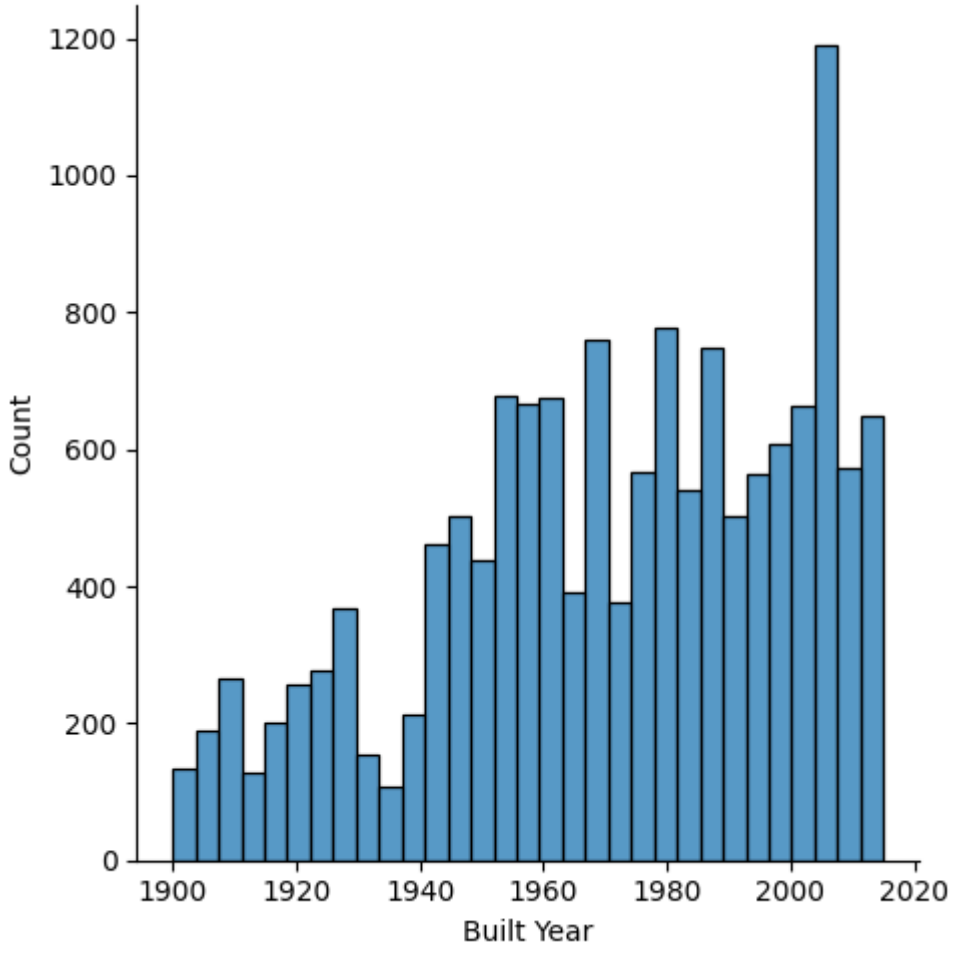
In [ ]:
```
sns.barplot(x =df["number of floors"].value_counts().index,y =df["number of floo
```

Out[ ]:
```
<AxesSubplot: ylabel='number of floors'>
```

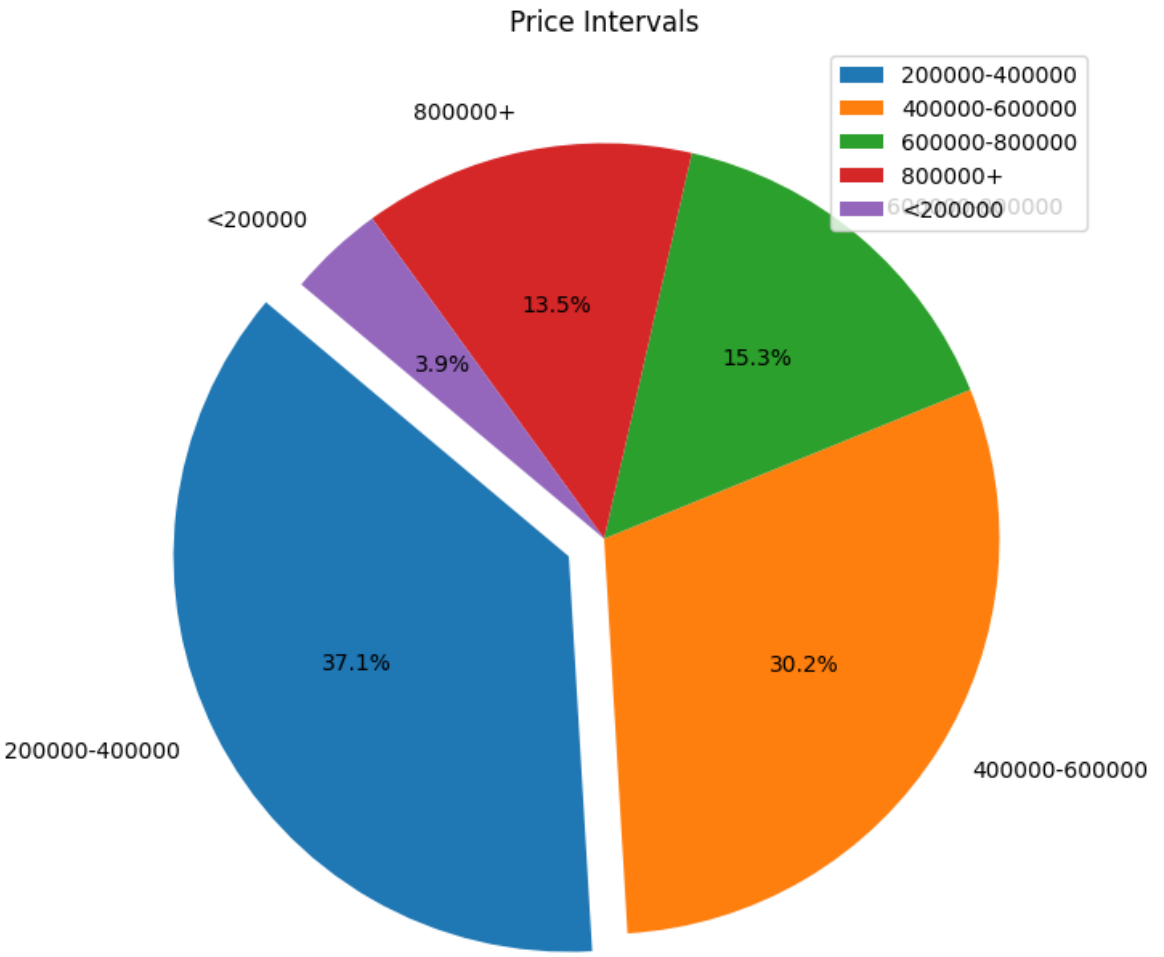


In [ ]:
```
sns.displot(df["Built Year"])
```

Out[ ]:
```
<seaborn.axisgrid.FacetGrid at 0x19da03ca2f0>
```



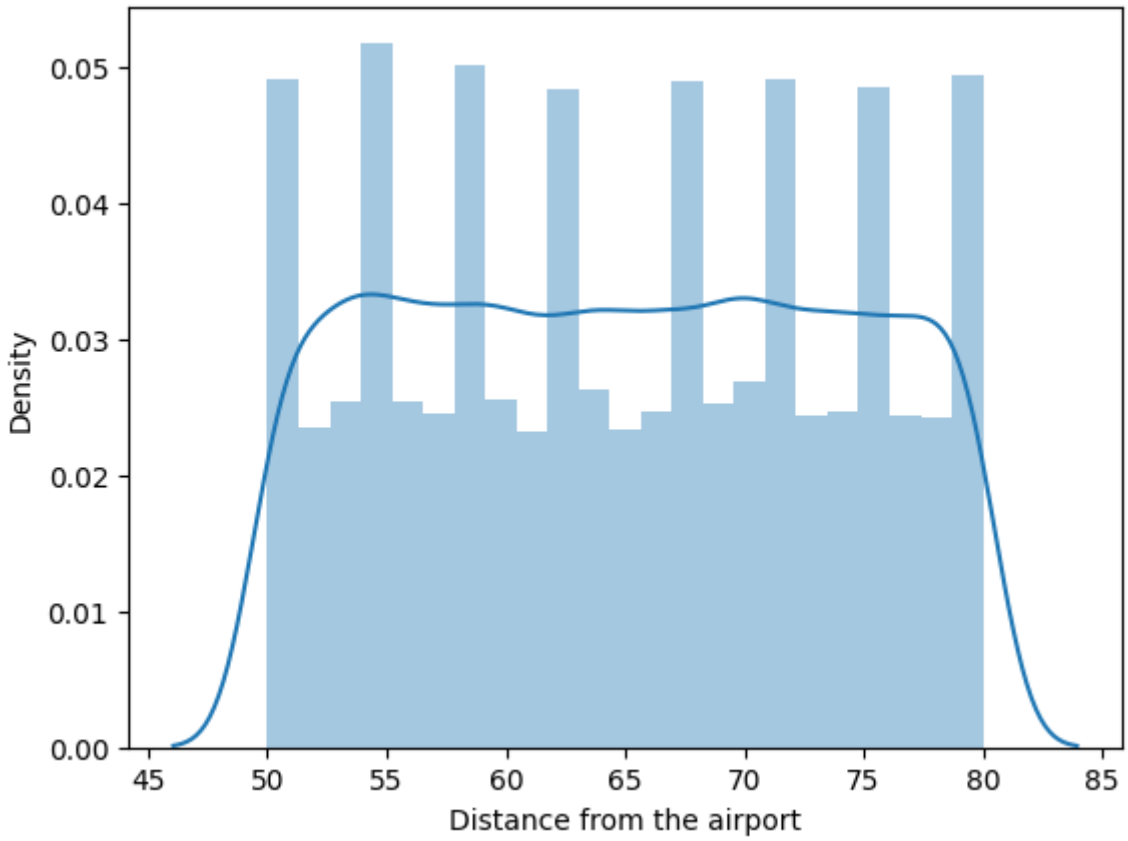
In [ ]:
```
# print(df["Price"].value_counts())
plt.figure(figsize=(8,8))
intervals = [0, 200000, 400000, 600000, 800000, float('inf')]
labels = ['<200000', '200000-400000', '400000-600000', '600000-800000', '800000+
df['Price Interval'] = pd.cut(df['Price'], bins=intervals, labels=labels)
interval_counts = df['Price Interval'].value_counts()
plt.pie(interval_counts, [0.1,0,0,0,0], labels=interval_counts.index, autopct='%
plt.title("Price Intervals")
plt.legend()
plt.show()
```



In [ ]:
```
sns.distplot(df["Distance from the airport"])
```

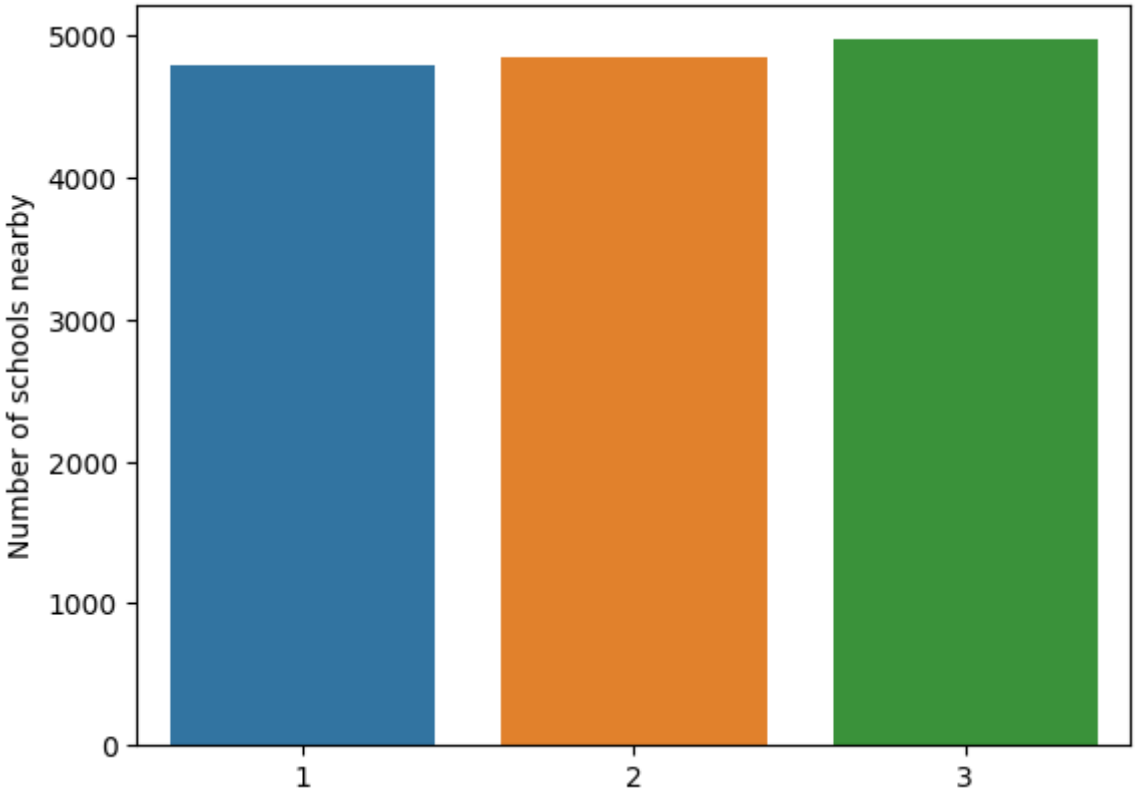
```
C:\Users\hrish\AppData\Local\Temp\ipykernel_11652\2743045996.py:1: UserWarning:

`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

  sns.distplot(df["Distance from the airport"])
```

Out[ ]: <AxesSubplot: xlabel='Distance from the airport', ylabel='Density'>



In [ ]:
```
sns.barplot(x =df["Number of schools nearby"].value_counts().index,y =df["Number
```

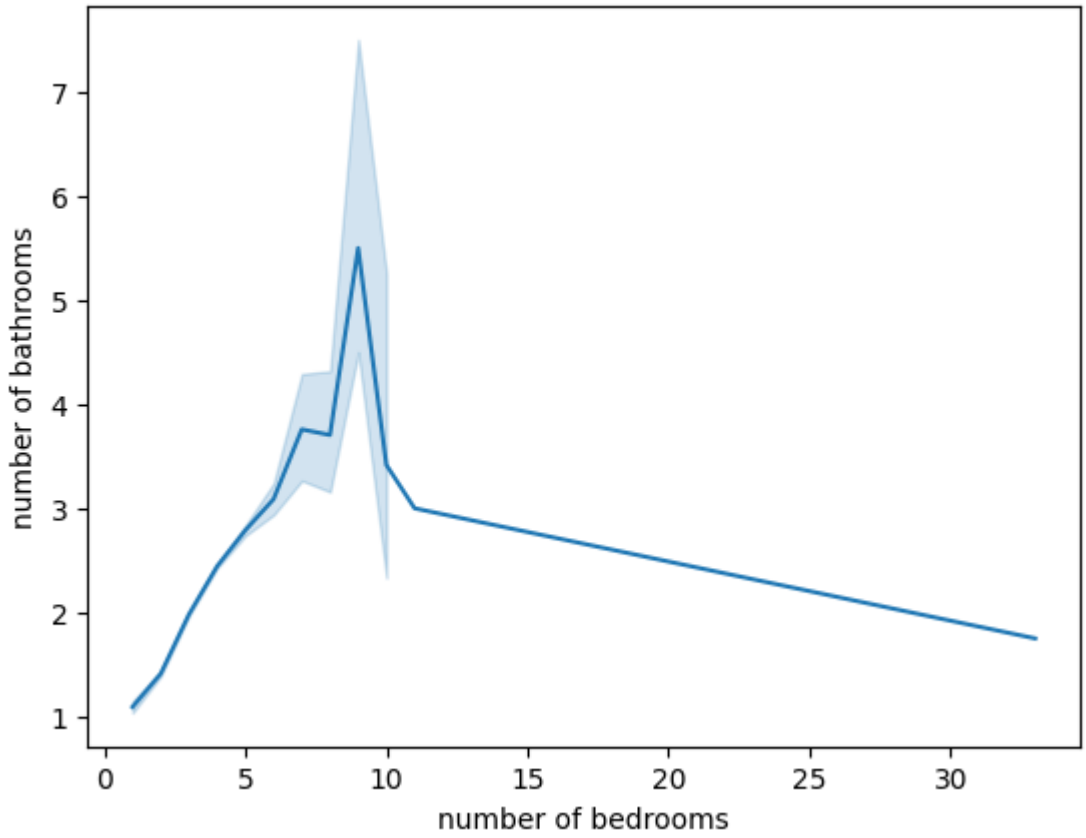
Out[ ]:
```
<AxesSubplot: ylabel='Number of schools nearby'>
```



ii) Bivariate Analysis

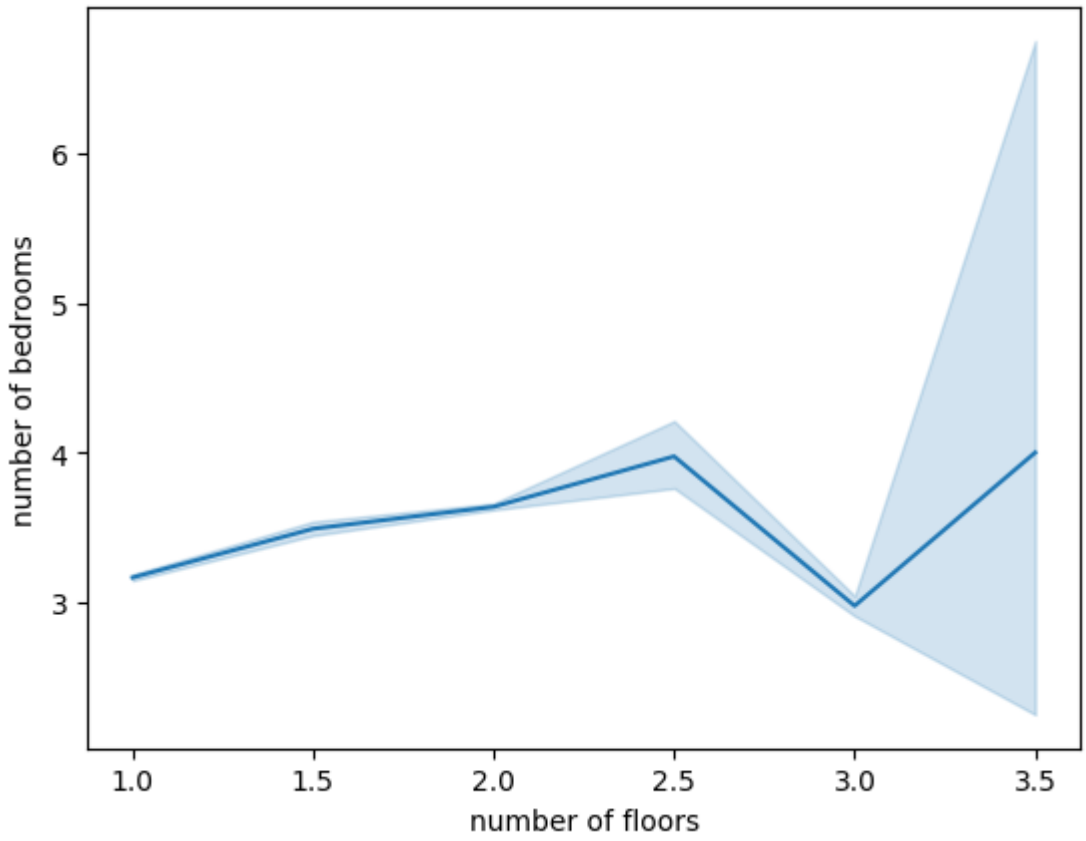
In [ ]:
```
sns.lineplot(x = df["number of bedrooms"],y=df["number of bathrooms"])
```

Out[ ]: <AxesSubplot: xlabel='number of bedrooms', ylabel='number of bathrooms'>



In [ ]:
```
sns.lineplot(x = df["number of floors"],y=df["number of bedrooms"])
```

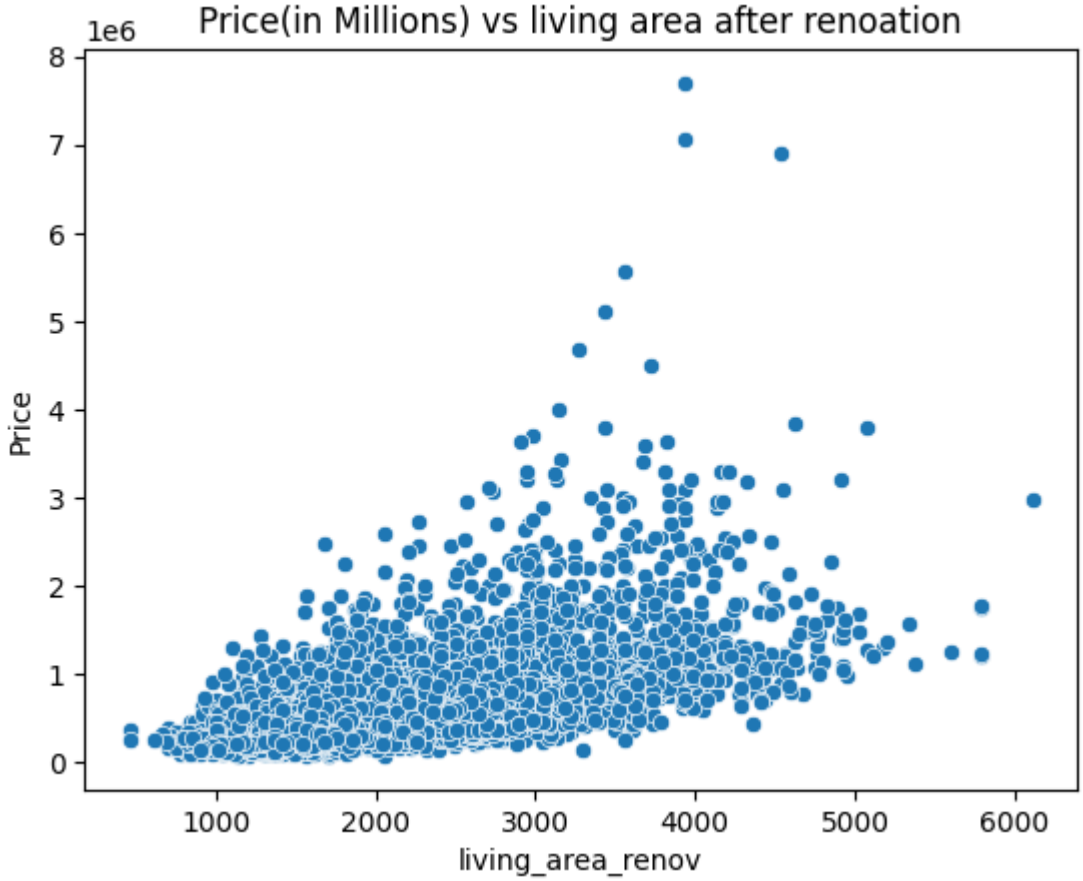
Out[ ]: <AxesSubplot: xlabel='number of floors', ylabel='number of bedrooms'>



In [ ]:
```
plt.title("Price(in Millions) vs living area after renoation")
```

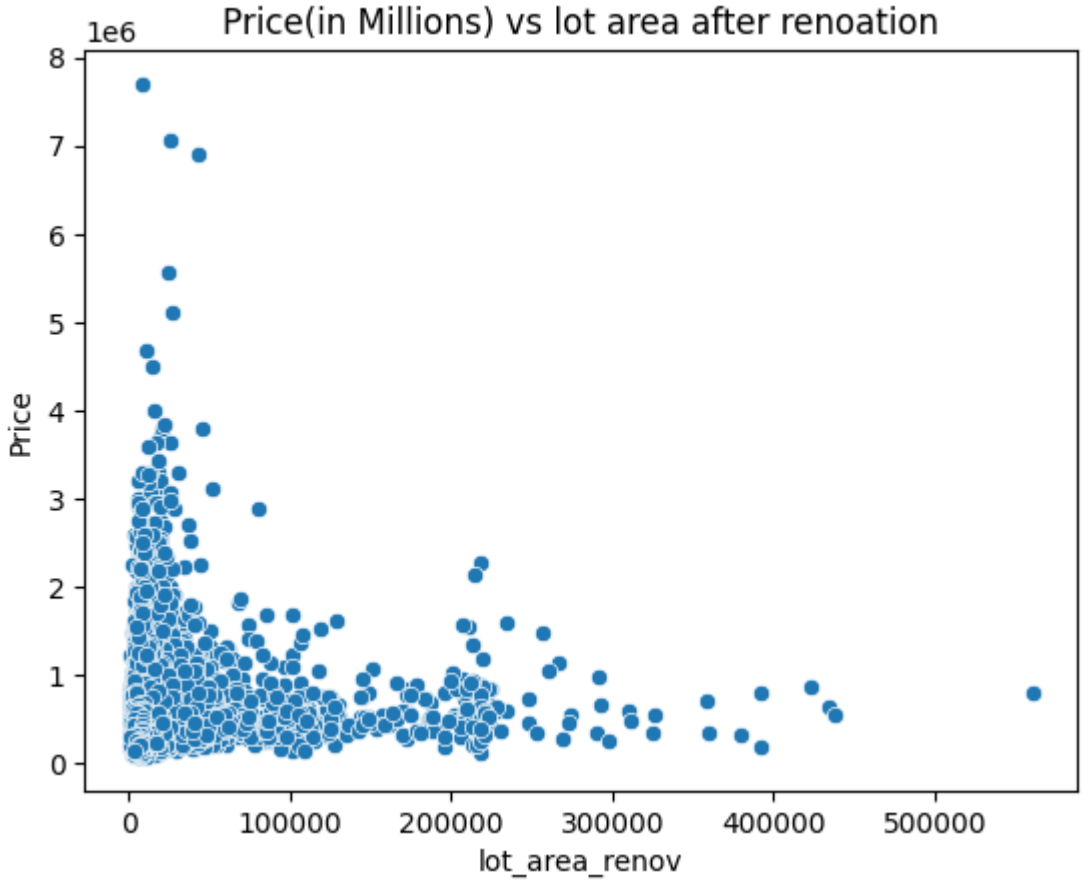
```
sns.scatterplot(x=df["living_area_renov"], y=df["Price"])
```

Out[ ]:
```
<AxesSubplot: title={'center': 'Price(in Millions) vs living area after renoati
on'}, xlabel='living_area_renov', ylabel='Price'>
```



In [ ]:
```
plt.title("Price(in Millions) vs lot area after renoation")
sns.scatterplot(x=df["lot_area_renov"], y=df["Price"])
```

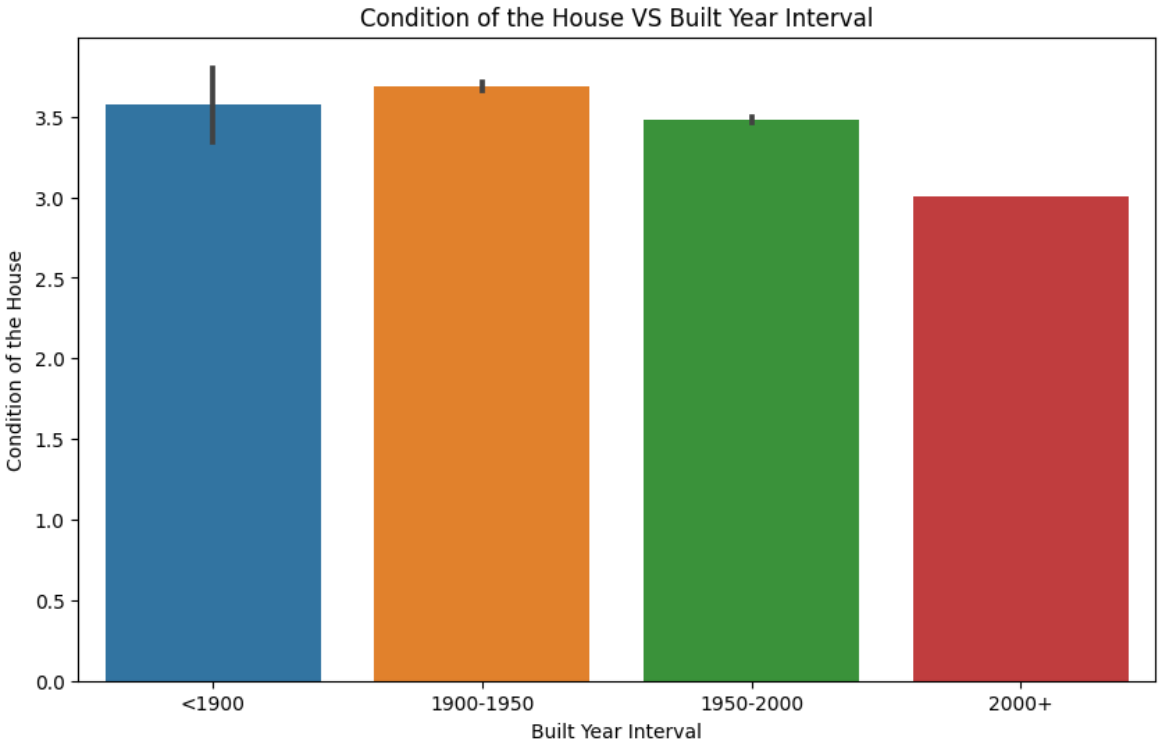
Out[ ]:
```
<AxesSubplot: title={'center': 'Price(in Millions) vs lot area after renoatio
n'}, xlabel='lot_area_renov', ylabel='Price'>
```



In [ ]:
```
plt.title("living area before renovation vs living area after renovation")
sns.scatterplot(x=df["living area"], y=df["living_area_renov"])
```

Out[ ]:
```
<AxesSubplot: title={'center': 'living area before renovation vs living area af
ter renovation'}, xlabel='living area', ylabel='living_area_renov'>
```

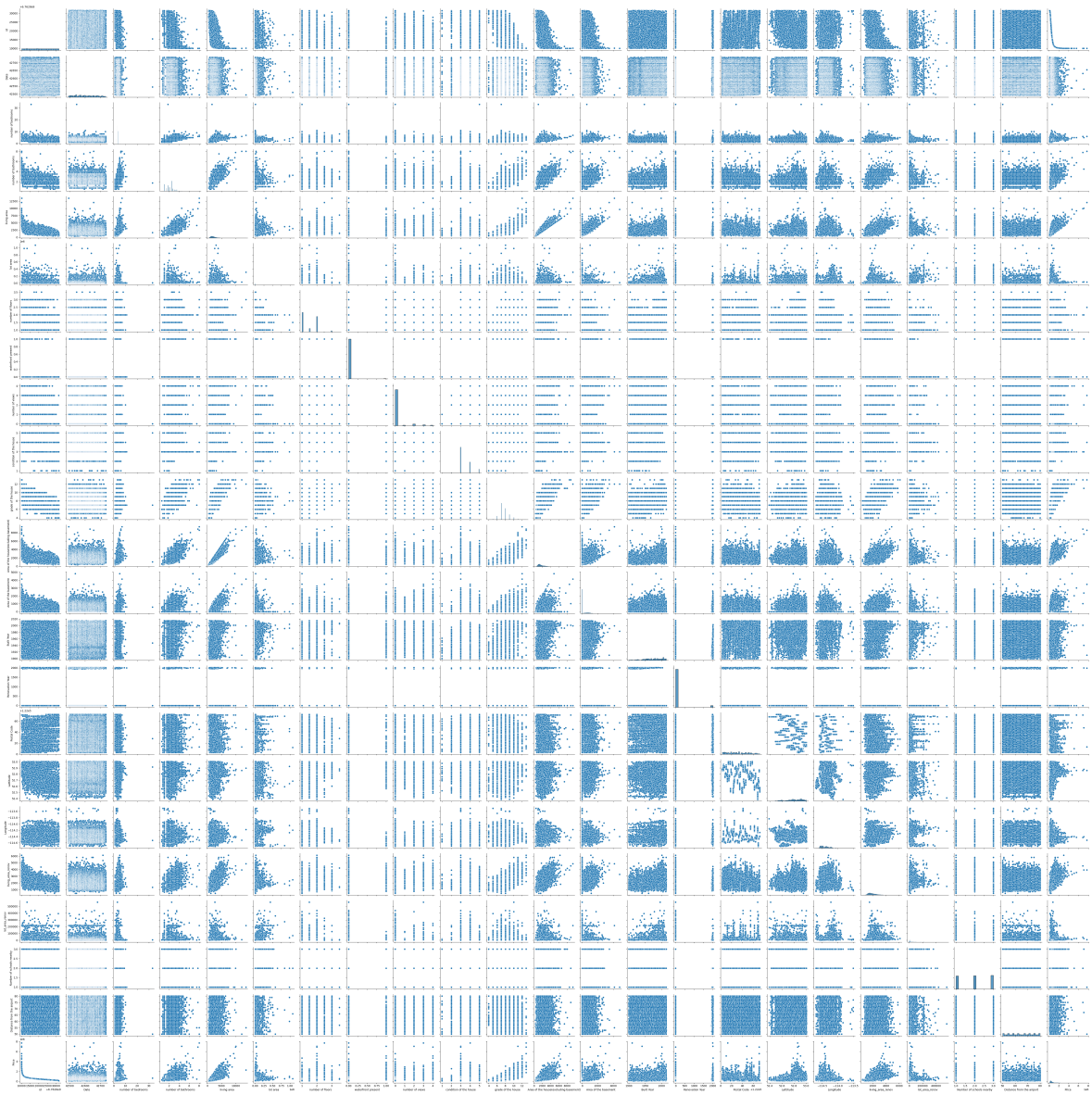
In [ ]:
```
# also bivariate
intervals = [0, 1900, 1950, 2000, df['Built Year'].max()]
labels = ['<1900', '1900-1950', '1950-2000', '2000+']
df['Built Year Interval'] = pd.cut(df['Built Year'], bins=intervals, labels=labe
plt.figure(figsize=(10, 6))  # Adjust the figure size as needed
sns.barplot(x="Built Year Interval", y="condition of the house", data=df)
plt.xlabel("Built Year Interval")
plt.ylabel("Condition of the House")
plt.title("Condition of the House VS Built Year Interval")
plt.show()
```



iii) Multivariate Analysis

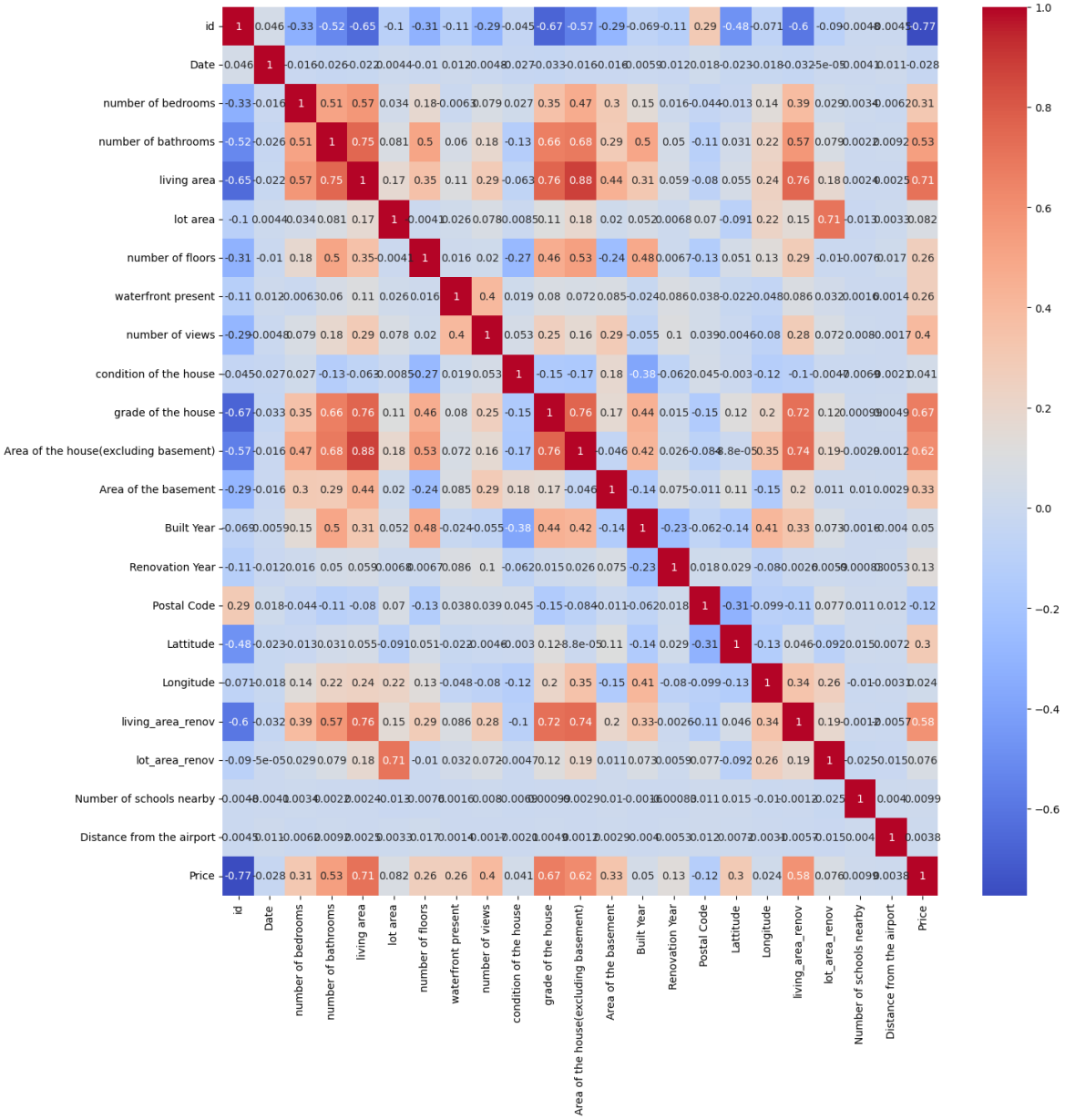
In [ ]:
```
sns.pairplot(df)
```

Out[ ]:
```
<seaborn.axisgrid.PairGrid at 0x19da619b700>
```



In [ ]:
```
plt.figure(figsize=(15,15))
sns.heatmap(df.corr(), annot=True, cmap="coolwarm")
```

Out[ ]: <AxesSubplot: >



Task4 -> Descriptive analysis of data

In [ ]:
```
df.describe()
```

Out[ ]:

	id	Date	number of bedrooms	number of bathrooms	living area	lot
count	1.462000e+04	14620.000000	14620.000000	14620.000000	14620.000000	1.462000e
mean	6.762821e+09	42604.538646	3.379343	2.129583	2098.262996	1.509328e
std	6.237575e+03	67.347991	0.938719	0.769934	928.275721	3.791962e
min	6.762810e+09	42491.000000	1.000000	0.500000	370.000000	5.200000e
25%	6.762815e+09	42546.000000	3.000000	1.750000	1440.000000	5.010750e
50%	6.762821e+09	42600.000000	3.000000	2.250000	1930.000000	7.620000e
75%	6.762826e+09	42662.000000	4.000000	2.500000	2570.000000	1.080000e
max	6.762832e+09	42734.000000	33.000000	8.000000	13540.000000	1.074218e

8 rows × 23 columns

In [ ]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14620 entries, 0 to 14619
Data columns (total 25 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
 23  Price Interval                         14620 non-null  category
 24  Built Year Interval                    14620 non-null  category
dtypes: category(2), float64(4), int64(19)
memory usage: 2.6 MB
```

Task5: Handling Missing Values

In [ ]:
```
df.isnull().sum()
```

Out[ ]:
```
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
Price Interval                          0
Built Year Interval                     0
dtype: int64
```

In [ ]:
```
df.isnull().any()
```

Out[ ]:
```
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
Price Interval                          False
Built Year Interval                     False
dtype: bool
```

No missing values are there => No need to do anything further

Completed - HRISHIKESH G KULKARNI (21BAI1660)

hrishikesh.gkulkarni2021@vitstudent.ac.in

==========X======