Python EDA Project- AirBnB Listing

Importing Libraries

In [2]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Loading Dataset

In [3]:

```
data = pd.read_csv("datasets.csv")
```

Initial Exploration

In [4]:

```
data.head(5)
```

Out[4]:

	id	name	host_id	host_name	neighbourhood_group	neighbourhood
0	1.312228e+06	Rental unit in Brooklyn · ★5.0 · 1 bedroom	7130382	Walter	Brooklyn	Clinton Hill
1	4.527754e+07	Rental unit in New York · ★4.67 · 2 bedrooms ·...	51501835	Jeniffer	Manhattan	Hell's Kitchen
2	9.710000e+17	Rental unit in New York · ★4.17 · 1 bedroom · ...	528871354	Joshua	Manhattan	Chelsea
3	3.857863e+06	Rental unit in New York · ★4.64 · 1 bedroom · ...	19902271	John And Catherine	Manhattan	Washington Heights
4	4.089661e+07	Condo in New York · ★4.91 · Studio · 1 bed · 1...	61391963	Stay With Vibe	Manhattan	Murray Hill

5 rows × 22 columns

In [5]:
```
data.shape
```

Out[5]:
```
(20770, 22)
```

In [6]:
```
data.describe()
```

Out[6]:

	id	host_id	latitude	longitude	price	minimum_r
count	2.077000e+04	2.077000e+04	20763.000000	20763.000000	20736.000000	20763.0
mean	3.033858e+17	1.749049e+08	40.726821	-73.939179	187.714940	28.5
std	3.901221e+17	1.725657e+08	0.060293	0.061403	1023.245124	33.5
min	2.595000e+03	1.678000e+03	40.500314	-74.249840	10.000000	1.0
25%	2.707260e+07	2.041184e+07	40.684159	-73.980755	80.000000	30.0
50%	4.992852e+07	1.086990e+08	40.722890	-73.949597	125.000000	30.0
75%	7.220000e+17	3.143997e+08	40.763106	-73.917475	199.000000	30.0
max	1.050000e+18	5.504035e+08	40.911147	-73.713650	100000.000000	1250.0

In [7]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20770 entries, 0 to 20769
Data columns (total 22 columns):
 #   Column                          Non-Null Count  Dtype  
---  ------                          --------------  -----  
 0   id                              20770 non-null  float64
 1   name                            20770 non-null  object 
 2   host_id                         20770 non-null  int64  
 3   host_name                       20770 non-null  object 
 4   neighbourhood_group             20770 non-null  object 
 5   neighbourhood                   20763 non-null  object 
 6   latitude                        20763 non-null  float64
 7   longitude                       20763 non-null  float64
 8   room_type                       20763 non-null  object 
 9   price                           20736 non-null  float64
 10  minimum_nights                  20763 non-null  float64
 11  number_of_reviews               20763 non-null  float64
 12  last_review                     20763 non-null  object 
 13  reviews_per_month               20763 non-null  float64
 14  calculated_host_listings_count  20763 non-null  float64
 15  availability_365                20763 non-null  float64
 16  number_of_reviews_ltm           20763 non-null  float64
 17  license                         20770 non-null  object 
 18  rating                          20770 non-null  object 
 19  bedrooms                        20770 non-null  object 
 20  beds                            20770 non-null  int64  
 21  baths                           20770 non-null  object 
dtypes: float64(10), int64(2), object(10)
memory usage: 3.5+ MB
```

Data Cleaning

In [9]:

```
data.isnull().sum()
```

```
Out[9]:  id                                 0
         name                               0
         host_id                            0
         host_name                          0
         neighbourhood_group                0
         neighbourhood                      7
         latitude                           7
         longitude                          7
         room_type                          7
         price                             34
         minimum_nights                     7
         number_of_reviews                  7
         last_review                        7
         reviews_per_month                  7
         calculated_host_listings_count     7
         availability_365                   7
         number_of_reviews_ltm              7
         license                            0
         rating                             0
         bedrooms                           0
         beds                               0
         baths                              0
         dtype: int64
```

```
In [11]:  #dropping all null values
          data.dropna(inplace= True)

          data.isnull().sum()
```

```
Out[11]:  id                                0
          name                              0
          host_id                           0
          host_name                         0
          neighbourhood_group               0
          neighbourhood                     0
          latitude                          0
          longitude                         0
          room_type                         0
          price                             0
          minimum_nights                    0
          number_of_reviews                 0
          last_review                       0
          reviews_per_month                 0
          calculated_host_listings_count    0
          availability_365                  0
          number_of_reviews_ltm             0
          license                           0
          rating                            0
          bedrooms                          0
          beds                              0
          baths                             0
          dtype: int64
```

```
In [12]:  #Dealing with duplicates
          data.duplicated().sum()
```

Out[12]:

```
np.int64(12)
```

In [14]:

```
#Deleting all duplicate values
data.drop_duplicates(inplace=True)

data.duplicated().sum()
```

Out[14]:

```
np.int64(0)
```

In [16]:

```
# type casting
# changing data types

data.dtypes

data["id"] = data["id"].astype(object)
data["host_id"] = data["host_id"].astype(object)
```

In [17]:

```
data.dtypes
```

Out[17]:

```
id                                 object
name                               object
host_id                            object
host_name                          object
neighbourhood_group                object
neighbourhood                      object
latitude                          float64
longitude                         float64
room_type                          object
price                             float64
minimum_nights                    float64
number_of_reviews                 float64
last_review                        object
reviews_per_month                 float64
calculated_host_listings_count    float64
availability_365                  float64
number_of_reviews_ltm             float64
license                            object
rating                             object
bedrooms                           object
beds                                int64
baths                              object
dtype: object
```

In [37]:

```
data.columns
```

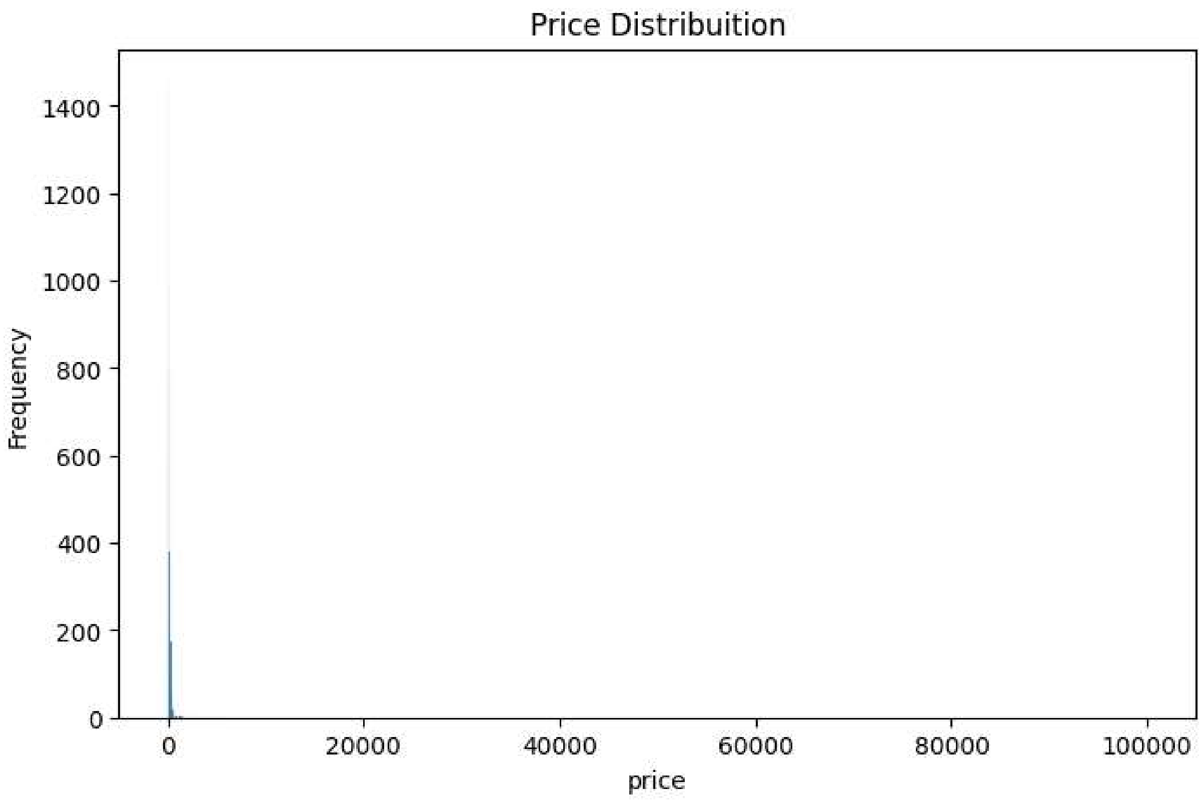
Out[37]:

```
Index(['id', 'name', 'host_id', 'host_name', 'neighbourhood_group',
       'neighbourhood', 'latitude', 'longitude', 'room_type', 'price',
       'minimum_nights', 'number_of_reviews', 'last_review',
       'reviews_per_month', 'calculated_host_listings_count',
       'availability_365', 'number_of_reviews_ltm', 'license', 'rating',
       'bedrooms', 'beds', 'baths'],
      dtype='object')
```

EDA

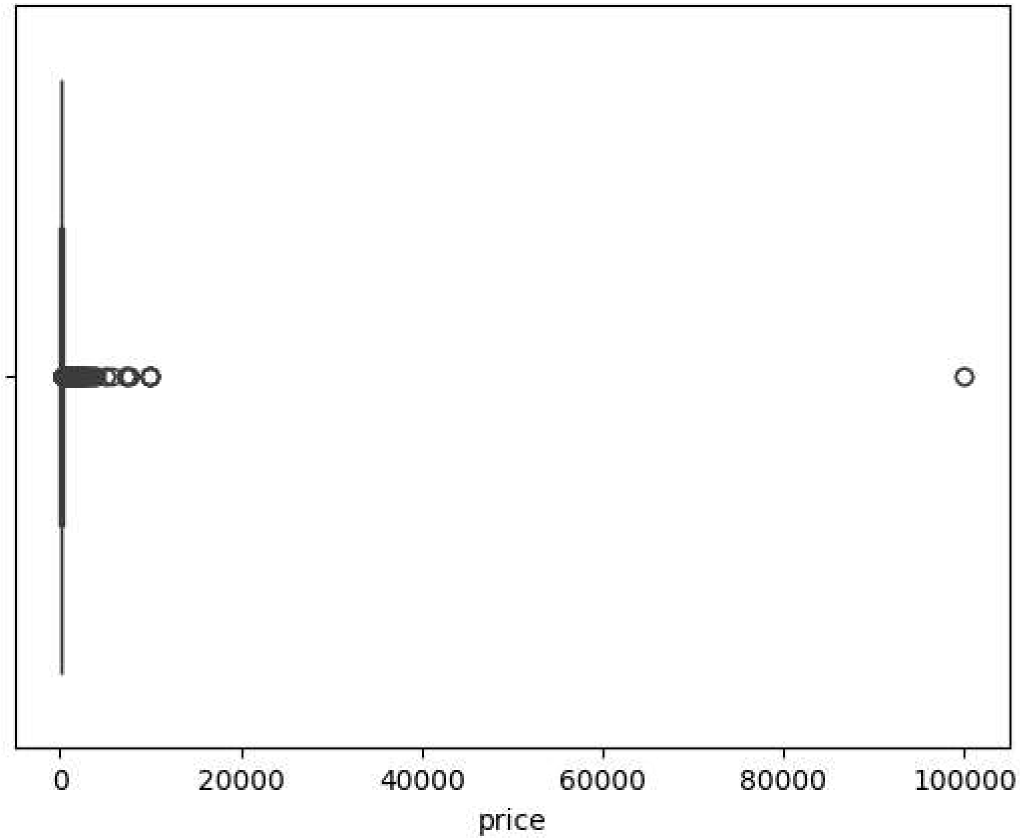
In [27]:

```
#Price distribuion

plt.figure(figsize=(8, 5))
sns.histplot(data=data, x='price')
plt.title('Price Distribuition')
plt.ylabel("Frequency")
plt.show()
```



In [29]:

```
# idenfying outliers in price

sns.boxplot(data=data, x='price')
plt.show()
```

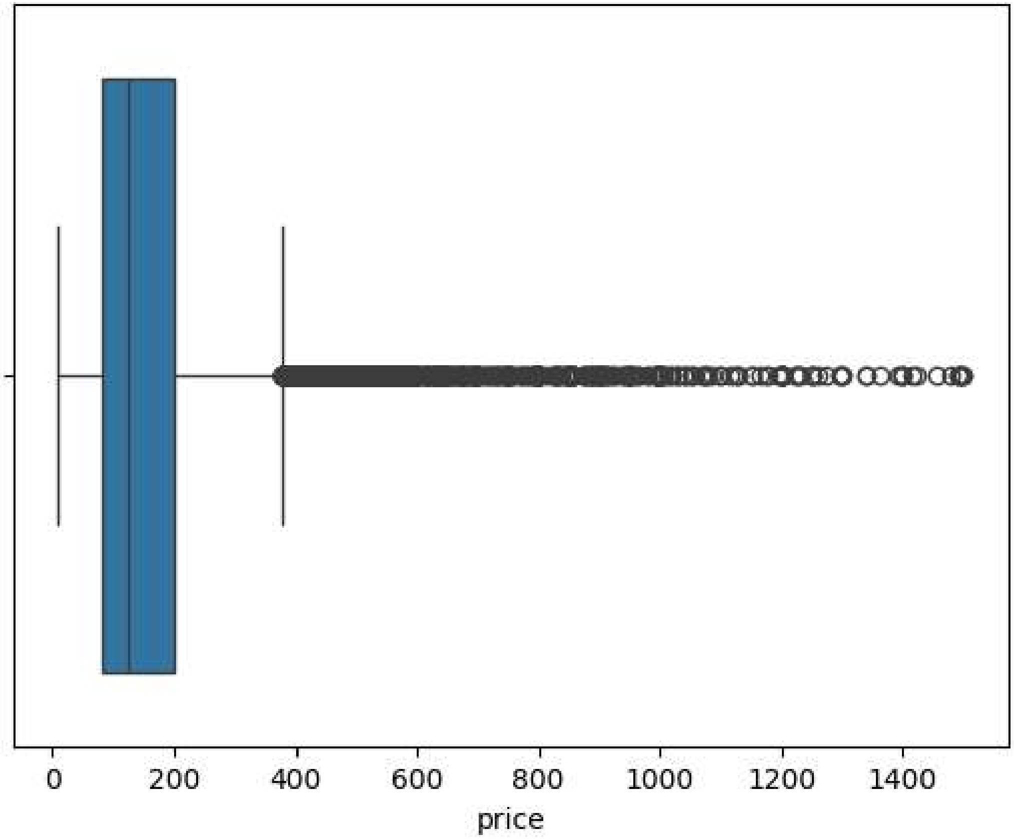


In [30]:

```
df = data[data['price'] < 1500]
```

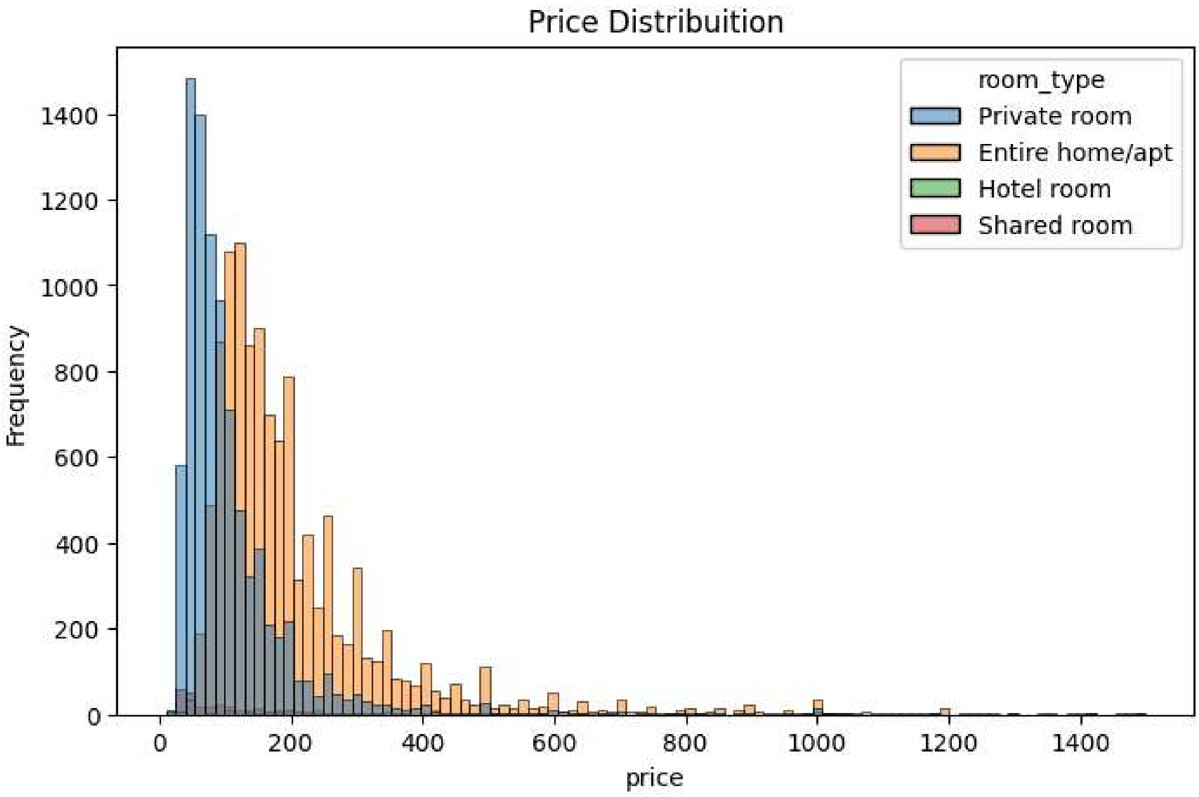
In [33]:

```
sns.boxplot(data=df, x='price')
plt.show()
```



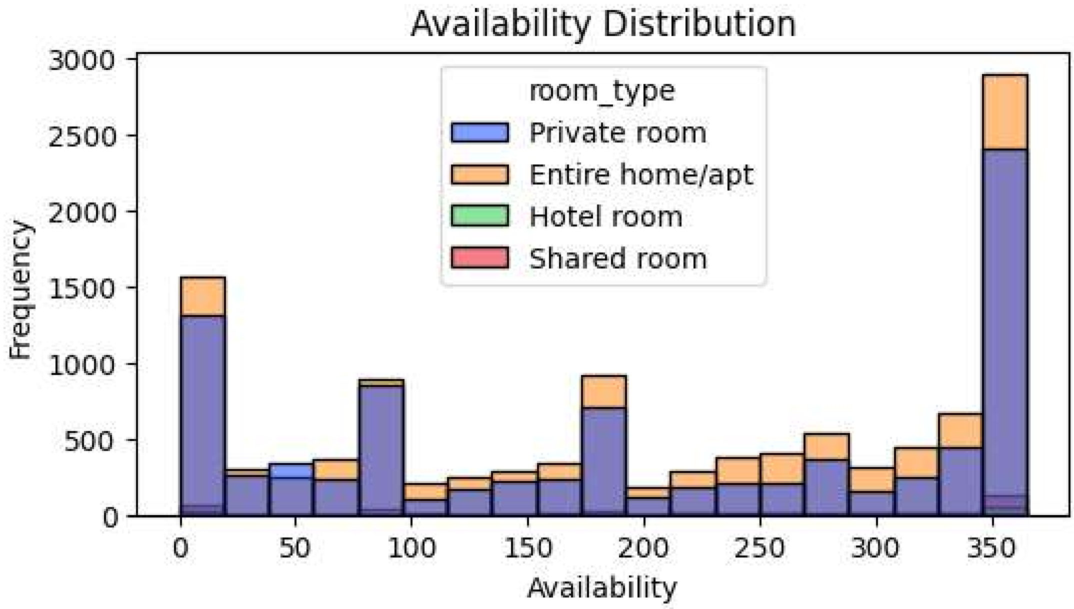
In [35]:

```
#Price distribuion

plt.figure(figsize=(8, 5))
sns.histplot(data=df, x='price', bins=100, hue = "room_type")
plt.title('Price Distribuition')
plt.ylabel("Frequency")
plt.show()
```



In [79]:

```
# Availability distribution
plt.figure(figsize=(6,3))
sns.histplot(x='availability_365', data = df, hue = 'room_type', palette = "bright"
plt.xlabel('Availability')
plt.ylabel('Frequency')
plt.title('Availability Distribution')
plt.show()
```



In [72]:

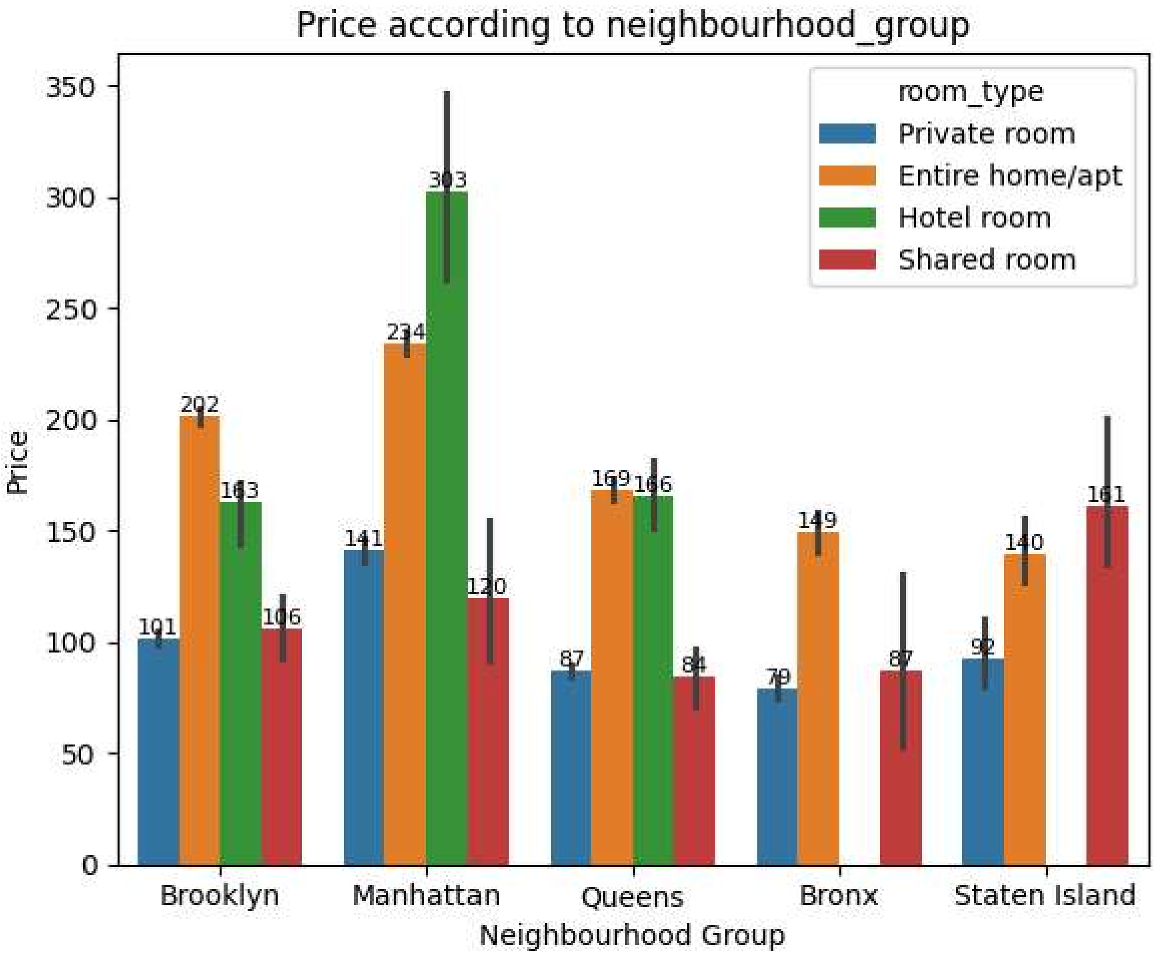
```
#df.groupby(([ 'neighbourhood_group','room_type'])['price'].mean())
df.groupby(['neighbourhood_group', 'room_type'])['price'].mean().sort_values(ascend
```

Out[72]:

```
neighbourhood_group  room_type      
Bronx                Private room        79.075506
Queens               Shared room         84.053333
Bronx                Shared room         87.333333
Queens               Private room        87.404591
Staten Island        Private room        92.389313
Brooklyn             Private room       101.478992
                     Shared room        106.075000
Manhattan            Shared room        120.000000
Staten Island        Entire home/apt    139.852564
Manhattan            Private room       140.857590
Bronx                Entire home/apt    149.043590
Staten Island        Shared room        161.250000
Brooklyn             Hotel room         162.750000
Queens               Hotel room         165.714286
                     Entire home/apt    168.606578
Brooklyn             Entire home/apt    201.698270
Manhattan            Entire home/apt    234.357197
                     Hotel room         302.734694
Name: price, dtype: float64
```

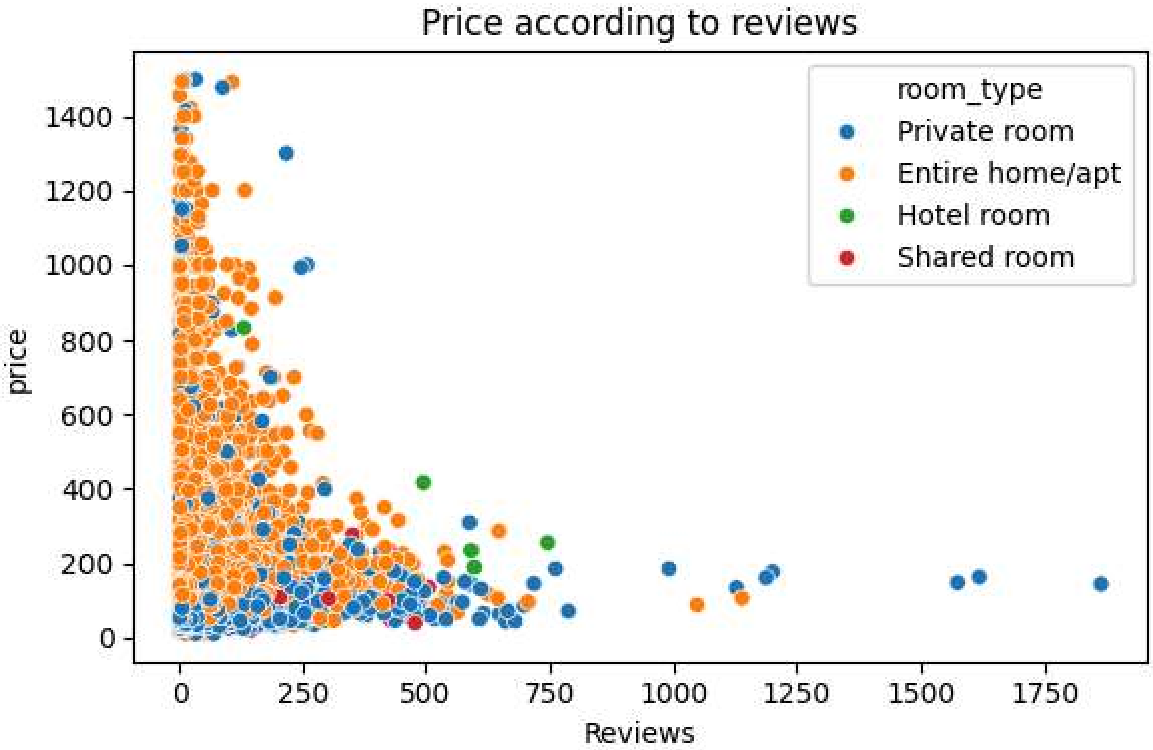
In [75]:

```python
# price dependency on neighbourhood
plt.figure(figsize=(6,5))
ax = sns.barplot(x='neighbourhood_group', y='price', data=df, hue = 'room_type')
plt.xlabel('Neighbourhood Group')
plt.ylabel('Price')
plt.title('Price according to neighbourhood_group')

# Add callout values
for container in ax.containers:
    for bar in container:
        height = bar.get_height()
        if not pd.isna(height):  # In case of NaN bars
            ax.text(
                bar.get_x() + bar.get_width() / 2,
                height,
                f'{height:.0f}',      # format without decimal
                ha='center',
                va='bottom',
                fontsize=8
            )
plt.tight_layout()
plt.show()
```



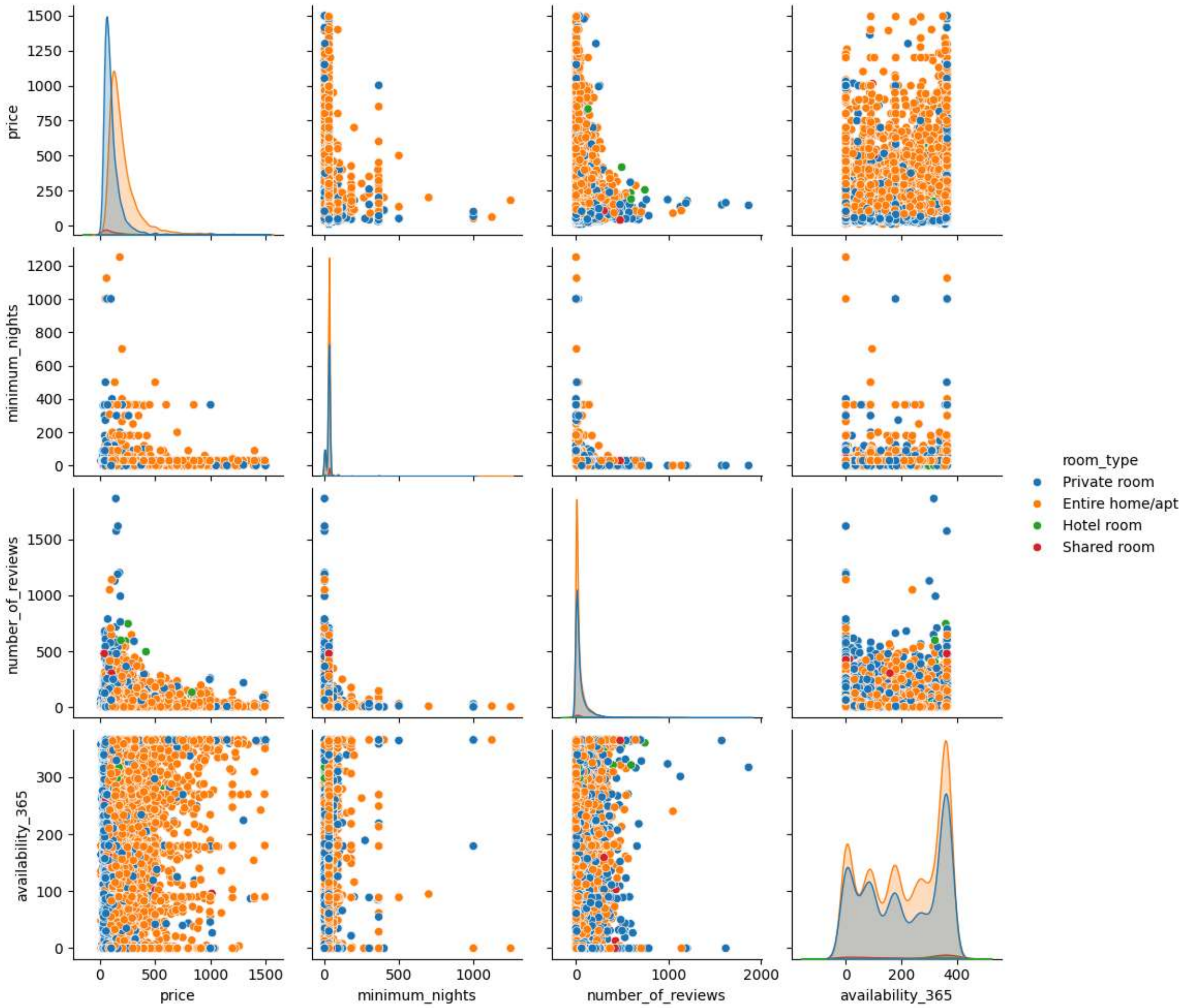
In [80]:

```
# price dependency on reviews

plt.figure(figsize=(6,4))
sns.scatterplot(x='number_of_reviews', y='price', data = df, hue = 'room_type')
plt.xlabel('Reviews')
plt.ylabel('price')
plt.title('Price according to reviews')
plt.tight_layout()
plt.show()
```



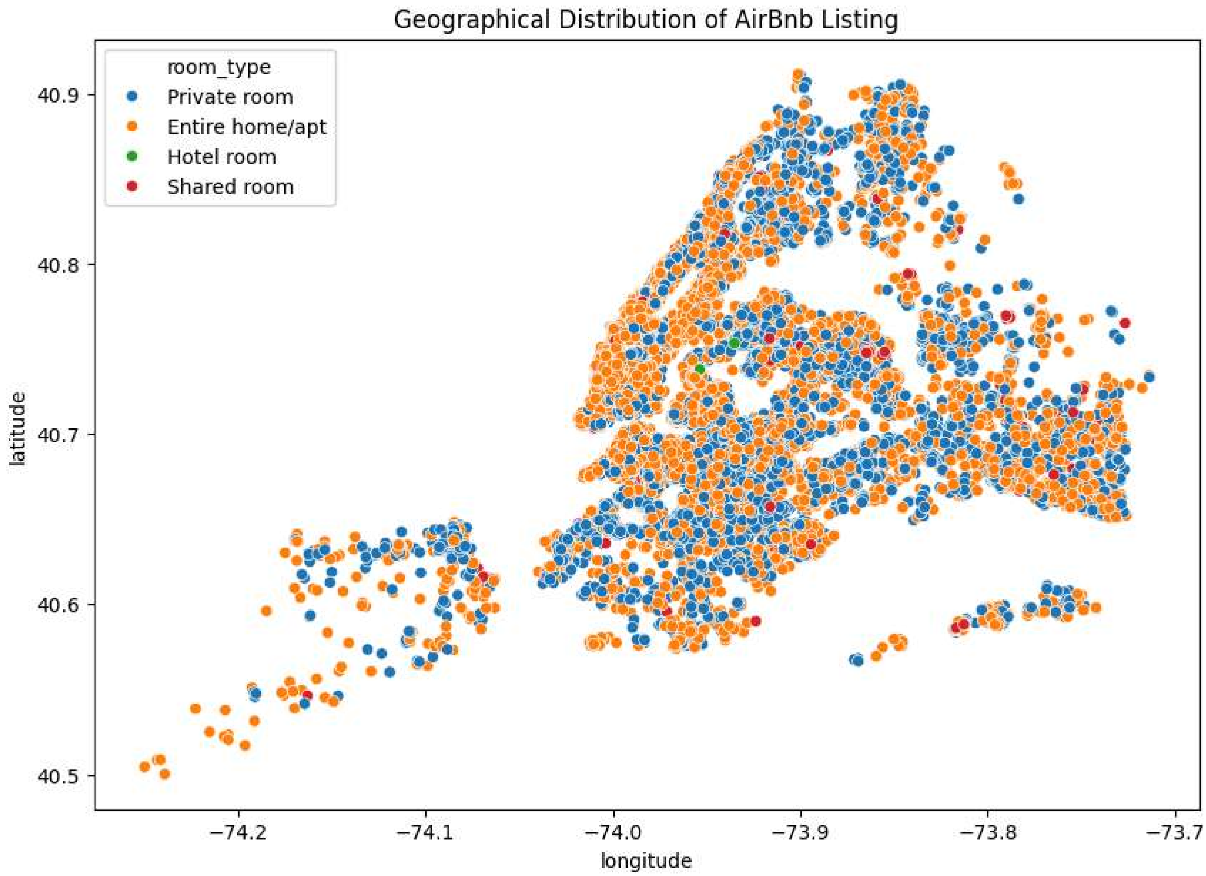
```
In [81]: sns.pairplot(data=df, vars=['price', 'minimum_nights', 'number_of_reviews', 'availa
```

```
Out[81]: <seaborn.axisgrid.PairGrid at 0x22074892660>
```



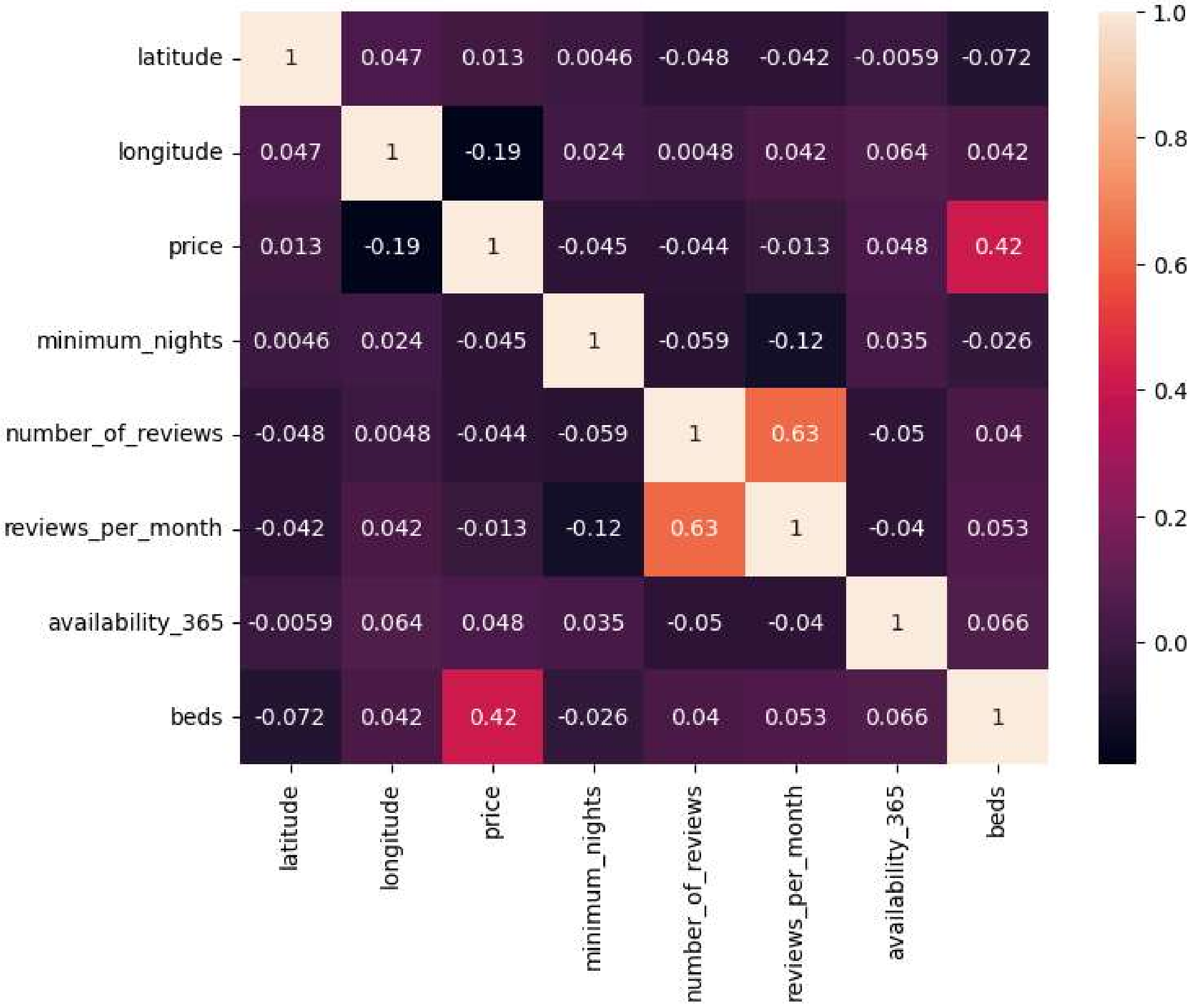
In [82]:

```
#Geographical Distribution of AirBnb Listing
plt.figure(figsize=(10, 7))
sns.scatterplot(data=df, x='longitude', y='latitude', hue='room_type')
plt.title("Geographical Distribution of AirBnb Listing")
plt.show()
```



In [84]:

```
# heat map - correlation of one variable with others for numerical column

corr = df[['latitude', 'longitude', 'price', 'minimum_nights', 'number_of_reviews',
corr

plt.figure(figsize=(8, 6))
sns.heatmap(data=corr, annot=True)
plt.show()
```



Insights from Visualization

Room Type vs Price (by Neighbourhood Group)

Private rooms tend to have lower average prices compared to entire homes/apartments across all neighbourhood groups.

Manhattan shows the highest price levels, especially for entire homes.

Bronx and Staten Island generally have the lowest prices for all room types.

Neighbourhood Group Distribution

Brooklyn and Manhattan dominate the listings in terms of count.

Queens, Bronx, and Staten Island have relatively fewer listings, but can be strategic for budget travelers.

Price Distribution Patterns

There's a positive skew in price distributions—most listings are priced at the lower end, but a few high-priced listings pull up the average.

Outliers exist particularly in Manhattan, suggesting the presence of premium accommodations.