Airbnb Data Analysis

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

In [66]:
```
df=pd.read_csv('C:/Users/shrut/Downloads/compressed_data.csv/compressed_data.csv',encoding='unicode_escape'
```

```
C:\Users\shrut\AppData\Local\Temp\ipykernel_16016\293713721.py:1: DtypeWarning: Columns (25) have mixed ty
pes. Specify dtype option on import or set low_memory=False.
  df=pd.read_csv('C:/Users/shrut/Downloads/compressed_data.csv/compressed_data.csv',encoding='unicode_esca
pe')
```

In [4]:
```
df
```

Out[4]:

verified	host name	neighbourhood group	neighbourhood	lat	long	country	...	service fee	minimum nights	number of reviews	last review	reviews per month
ɔnfirmed	Madaline	Brooklyn	Kensington	40.64749	-73.97237	United States	...	$193	10.0	9.0	10/19/2021	0.21
verified	Jenna	Manhattan	Midtown	40.75362	-73.98377	United States	...	$28	30.0	45.0	5/21/2022	0.38
NaN	Elise	Manhattan	Harlem	40.80902	-73.94190	United States	...	$124	3.0	0.0	NaN	NaN
ɔnfirmed	Garry	Brooklyn	Clinton Hill	40.68514	-73.95976	United States	...	$74	30.0	270.0	7/5/2019	4.64
verified	Lyndon	Manhattan	East Harlem	40.79851	-73.94399	United States	...	$41	10.0	9.0	11/19/2018	0.10
...	...	...	...	...	...	...	...	...	...	...	...	...
verified	Krik	Brooklyn	Williamsburg	40.70862	-73.94651	United States	...	$169	1.0	0.0	NaN	NaN
ɔnfirmed	Mifan	Manhattan	Morningside Heights	40.80460	-73.96545	United States	...	$167	1.0	1.0	7/6/2015	0.02
ɔnfirmed	Megan	Brooklyn	Park Slope	40.67505	-73.98045	United States	...	$198	3.0	0.0	NaN	NaN
ɔnfirmed	Christopher	Queens	Long Island City	40.74989	-73.93777	United States	...	$109	2.0	5.0	10/11/2015	0.10
ɔnfirmed	Rebecca	Manhattan	Upper West Side	40.76807	-73.98342	United States	...	$206	1.0	0.0	NaN	NaN

In [5]:
```
df.head()
```

Out[5]:

	id	NAME	host id	host_identity_verified	host name	neighbourhood group	neighbourhood	lat	long	coun
0	1001254	Clean & quiet apt home by the park	80014485718	unconfirmed	Madaline	Brooklyn	Kensington	40.64749	-73.97237	Uni Sta
1	1002102	Skylit Midtown Castle	52335172823	verified	Jenna	Manhattan	Midtown	40.75362	-73.98377	Uni Sta
2	1002403	THE VILLAGE OF HARLEM....NEW YORK !	78829239556	NaN	Elise	Manhattan	Harlem	40.80902	-73.94190	Uni Sta
3	1002755	NaN	85098326012	unconfirmed	Garry	Brooklyn	Clinton Hill	40.68514	-73.95976	Uni Sta
4	1003689	Entire Apt: Spacious Studio/Loft by central park	92037596077	verified	Lyndon	Manhattan	East Harlem	40.79851	-73.94399	Uni Sta

5 rows × 26 columns

In [6]:
```
df.columns
```

Out[6]:
```
Index(['id', 'NAME', 'host id', 'host_identity_verified', 'host name',
       'neighbourhood group', 'neighbourhood', 'lat', 'long', 'country',
       'country code', 'instant_bookable', 'cancellation_policy', 'room type',
       'Construction year', 'price', 'service fee', 'minimum nights',
       'number of reviews', 'last review', 'reviews per month',
       'review rate number', 'calculated host listings count',
       'availability 365', 'house_rules', 'license'],
      dtype='object')
```

In [7]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 102599 entries, 0 to 102598
Data columns (total 26 columns):
 #   Column                          Non-Null Count   Dtype  
---  ------                          --------------   -----  
 0   id                              102599 non-null  int64  
 1   NAME                            102349 non-null  object 
 2   host id                         102599 non-null  int64  
 3   host_identity_verified          102310 non-null  object 
 4   host name                       102193 non-null  object 
 5   neighbourhood group             102570 non-null  object 
 6   neighbourhood                   102583 non-null  object 
 7   lat                             102591 non-null  float64
 8   long                            102591 non-null  float64
 9   country                         102067 non-null  object 
 10  country code                    102468 non-null  object 
 11  instant_bookable                102494 non-null  object 
 12  cancellation_policy             102523 non-null  object 
 13  room type                       102599 non-null  object 
 14  Construction year               102385 non-null  float64
 15  price                           102352 non-null  object 
 16  service fee                     102326 non-null  object 
 17  minimum nights                  102190 non-null  float64
 18  number of reviews               102416 non-null  float64
 19  last review                     86706 non-null   object 
 20  reviews per month               86720 non-null   float64
 21  review rate number              102273 non-null  float64
 22  calculated host listings count  102280 non-null  float64
 23  availability 365                102151 non-null  float64
 24  house_rules                     50468 non-null   object 
 25  license                         2 non-null       object 
dtypes: float64(9), int64(2), object(15)
memory usage: 20.4+ MB
```

In [12]:
```
#checking missing values
df.isnull().sum()
```

Out[12]:
```
id                                  0
NAME                              250
host id                             0
host_identity_verified            289
host name                         406
neighbourhood group                29
neighbourhood                      16
lat                                 8
long                                8
country                           532
country code                      131
instant_bookable                  105
cancellation_policy                76
room type                           0
Construction year                 214
price                             247
service fee                       273
minimum nights                    409
number of reviews                 183
last review                     15893
reviews per month               15879
review rate number                326
calculated host listings count    319
availability 365                  448
house_rules                     52131
license                        102597
dtype: int64
```

In [13]:
```
df.drop(['house_rules','license'],axis=1,inplace=True)
```

In [17]:
```
#handling missing values
df['last review'] = df['last review'].astype('datetime64[ns]')
#or df['last review'] = pd.to_datetime64[ns](df['last review'],error='coerce')
```

In [19]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 102599 entries, 0 to 102598
Data columns (total 24 columns):
 #   Column                          Non-Null Count   Dtype
---  ------                          --------------   -----
 0   id                              102599 non-null  int64
 1   NAME                            102349 non-null  object
 2   host id                         102599 non-null  int64
 3   host_identity_verified          102310 non-null  object
 4   host name                       102193 non-null  object
 5   neighbourhood group             102570 non-null  object
 6   neighbourhood                   102583 non-null  object
 7   lat                             102591 non-null  float64
 8   long                            102591 non-null  float64
 9   country                         102067 non-null  object
 10  country code                    102468 non-null  object
 11  instant_bookable                102494 non-null  object
 12  cancellation_policy             102523 non-null  object
 13  room type                       102599 non-null  object
 14  Construction year               102385 non-null  float64
 15  price                           102352 non-null  object
 16  service fee                     102326 non-null  object
 17  minimum nights                  102190 non-null  float64
 18  number of reviews               102416 non-null  float64
 19  last review                     86706 non-null   datetime64[ns]
 20  reviews per month               86720 non-null   float64
 21  review rate number              102273 non-null  float64
 22  calculated host listings count  102280 non-null  float64
 23  availability 365                102151 non-null  float64
dtypes: datetime64[ns](1), float64(9), int64(2), object(12)
memory usage: 18.8+ MB
```

In [22]:
```
df.fillna({'reviews per month':0,'last review':df['last review'].min()},inplace=True)
```

```
C:\Users\shrut\AppData\Local\Temp\ipykernel_16016\1521392419.py:1: DeprecationWarning: In a future versio
n, `df.iloc[:, i] = newvals` will attempt to set the values inplace instead of always setting a new array.
To retain the old behavior, use either `df[df.columns[i]] = newvals` or, if columns are non-unique, `df.is
etitem(i, newvals)`
  df.fillna({'reviews per month':0,'last review':df['last review'].min()},inplace=True)
```

In [23]:
```
df.head()
```

Out[23]:

dentity_verified	host name	neighbourhood group	neighbourhood	lat	long	country	...	Construction year	price	service fee	minimum nights	nun revi
unconfirmed	Madaline	Brooklyn	Kensington	40.64749	-73.97237	United States	...	2020.0	$966	$193	10.0	
verified	Jenna	Manhattan	Midtown	40.75362	-73.98377	United States	...	2007.0	$142	$28	30.0	
NaN	Elise	Manhattan	Harlem	40.80902	-73.94190	United States	...	2005.0	$620	$124	3.0	
unconfirmed	Garry	Brooklyn	Clinton Hill	40.68514	-73.95976	United States	...	2005.0	$368	$74	30.0	2
verified	Lyndon	Manhattan	East Harlem	40.79851	-73.94399	United States	...	2009.0	$204	$41	10.0	

In [24]:
```
df.dropna(inplace=True)
```

In [25]:
```
df.isnull().sum() #0r df.dropna(subset={'NAME'},inplace=true) for perticular column
```

Out[25]:
```
id                                0
NAME                              0
host id                           0
host_identity_verified            0
host name                         0
neighbourhood group               0
neighbourhood                     0
lat                               0
long                              0
country                           0
country code                      0
instant_bookable                  0
cancellation_policy               0
room type                         0
Construction year                 0
price                             0
service fee                       0
minimum nights                    0
number of reviews                 0
last review                       0
reviews per month                 0
review rate number                0
calculated host listings count    0
availability 365                  0
dtype: int64
```

In [26]:
```
df['price'] = df['price'].replace('[\$,]','',regex=True).astype('float')
```

In [27]:
```
df['service fee'] = df['service fee'].replace('[\$,]','',regex=True).astype('float')
```

In [28]:
```
df.head(4)
```

Out[28]:

	id	NAME	host id	host_identity_verified	host name	neighbourhood group	neighbourhood	lat	long	country	..
0	1001254	Clean & quiet apt home by the park	80014485718	unconfirmed	Madaline	Brooklyn	Kensington	40.64749	-73.97237	United States	..
1	1002102	Skylit Midtown Castle	52335172823	verified	Jenna	Manhattan	Midtown	40.75362	-73.98377	United States	..
4	1003689	Entire Apt: Spacious Studio/Loft by central park	92037596077	verified	Lyndon	Manhattan	East Harlem	40.79851	-73.94399	United States	..
5	1004098	Large Cozy 1 BR Apartment In Midtown East	45498551794	verified	Michelle	Manhattan	Murray Hill	40.74767	-73.97500	United States	..

4 rows × 24 columns

In [29]:
```
#removing duplicate values
df.drop_duplicates(inplace=True)
```

In [31]:
```
df.describe()
```

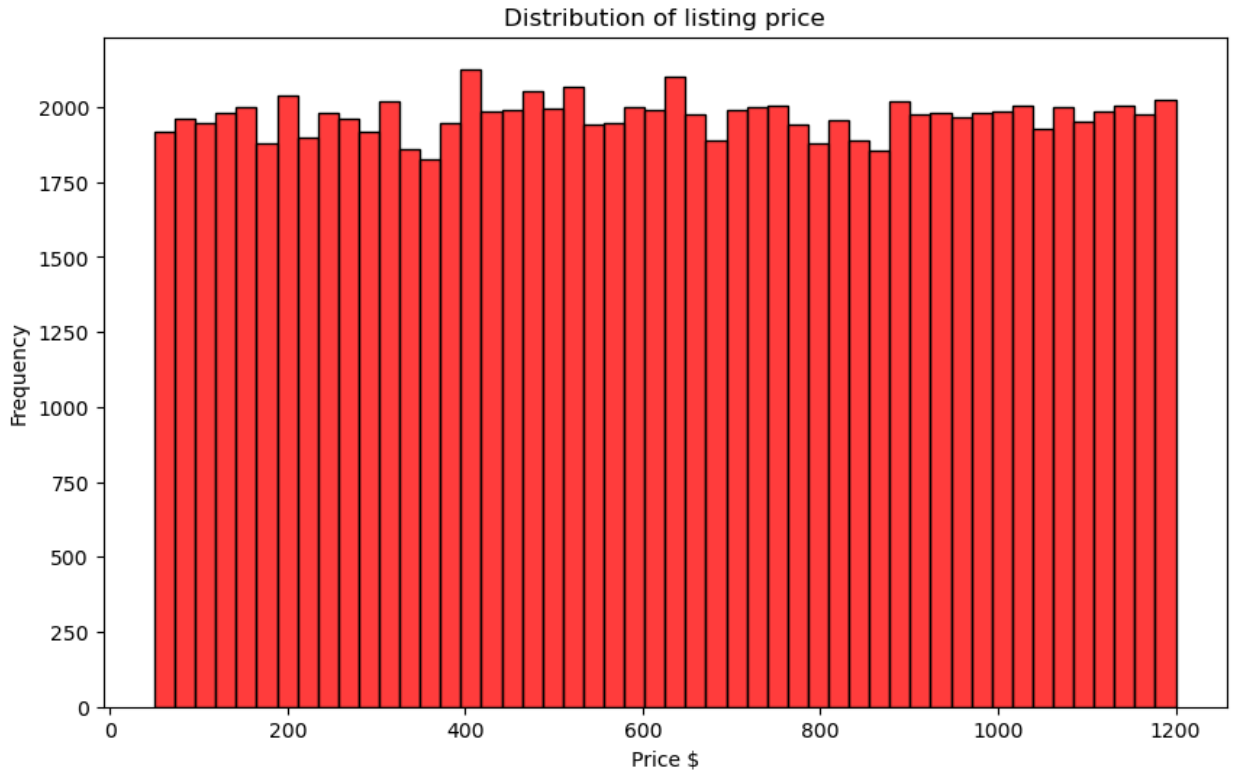
Out[31]:

	id	host id	lat	long	Construction year	price	service fee	minimum nights	number revie
count	9.850900e+04	9.850900e+04	98509.000000	98509.000000	98509.000000	98509.000000	98509.000000	98509.000000	98509.0000
mean	2.927555e+07	4.925024e+10	40.728036	-73.949629	2012.488138	625.530205	125.106843	8.046402	27.3416
std	1.622021e+07	2.854665e+10	0.055831	0.049545	5.761132	331.739043	66.351092	28.434029	49.2162
min	1.001254e+06	1.236005e+08	40.499790	-74.249840	2003.000000	50.000000	10.000000	-1223.000000	0.0000
25%	1.517556e+07	2.455589e+10	40.688720	-73.982570	2008.000000	340.000000	68.000000	2.000000	1.0000
50%	2.932383e+07	4.911063e+10	40.722250	-73.954440	2012.000000	625.000000	125.000000	3.000000	7.0000
75%	4.333016e+07	7.398748e+10	40.762740	-73.932270	2017.000000	913.000000	183.000000	5.000000	30.0000
max	5.735803e+07	9.876313e+10	40.916970	-73.705220	2022.000000	1200.000000	240.000000	5645.000000	1024.0000

DATA VISUALIZATION

What is the distribution of listing prices?

In [45]:
```
plt.figure(figsize=(10,6))
sns.histplot(data=df,x='price',kde=True,bins=50,color='red')
plt.title("Distribution of listing price")
plt.xlabel("Price $")
plt.ylabel("Frequency")
plt.show()
```



The histogram shows a fairly even distribution of listing prices across different price ranges, including no perticular concentration on listings in any specific price range. The KDE line helps visualize this even spread more clearly, confirming that the dataset contains listings with a wide variety of prices.

How are different room types distributed?

In [41]:
```
df['room type']
```

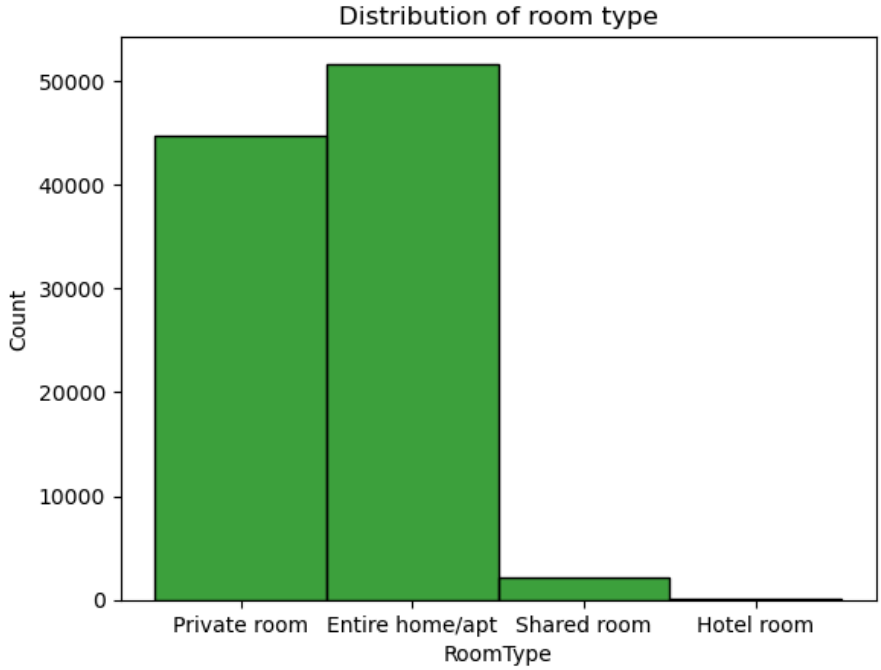
Out[41]:
```
0            Private room
1         Entire home/apt
4         Entire home/apt
5         Entire home/apt
7            Private room
                 ...
102029       Private room
102030       Private room
102031       Private room
102032       Private room
102040       Private room
Name: room type, Length: 98509, dtype: object
```

In [42]:
```
plt.figure(figsize=(8,5))
```

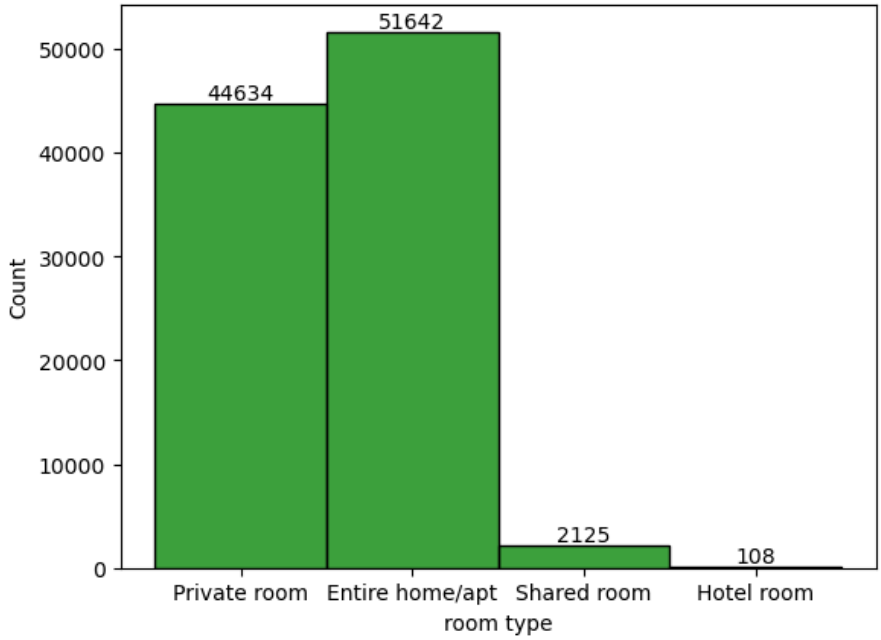
Out[42]:
```
<Figure size 800x500 with 0 Axes>
```

```
<Figure size 800x500 with 0 Axes>
```

In [46]:
```
sns.histplot(data=df,x='room type',bins=50,color='green')
plt.title("Distribution of room type")
plt.xlabel("RoomType")
plt.show()
```



In [47]:
```
ax=sns.histplot(data=df,x='room type',bins=50,color='green')
for bars in ax.containers:
    ax.bar_label(bars)
```

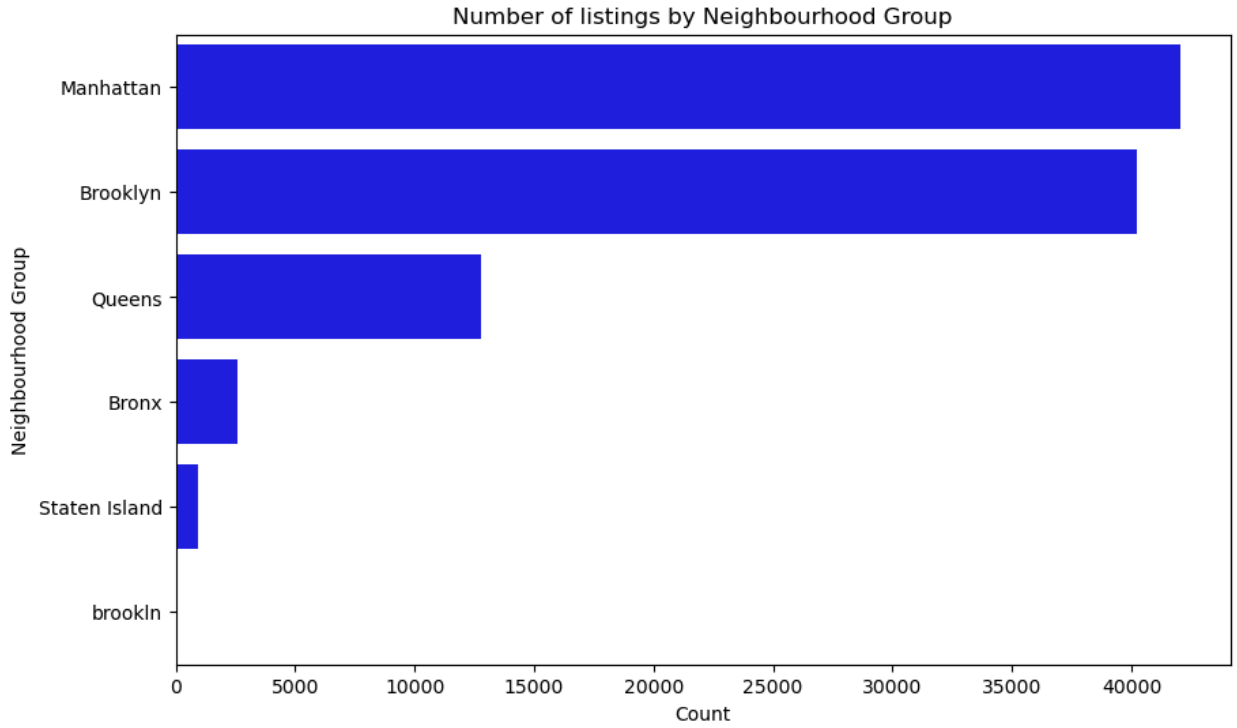


From the histogram of listing counts by room type, it's clear that:

Entire home/apt is by far the most common listing, with just over 50 000 entries. Private rooms are the next most frequent, at around 45 000 listings. Hotel rooms and shared rooms trail well behind (only a few thousand each).

How are the listings distributed across different neighborhoods?

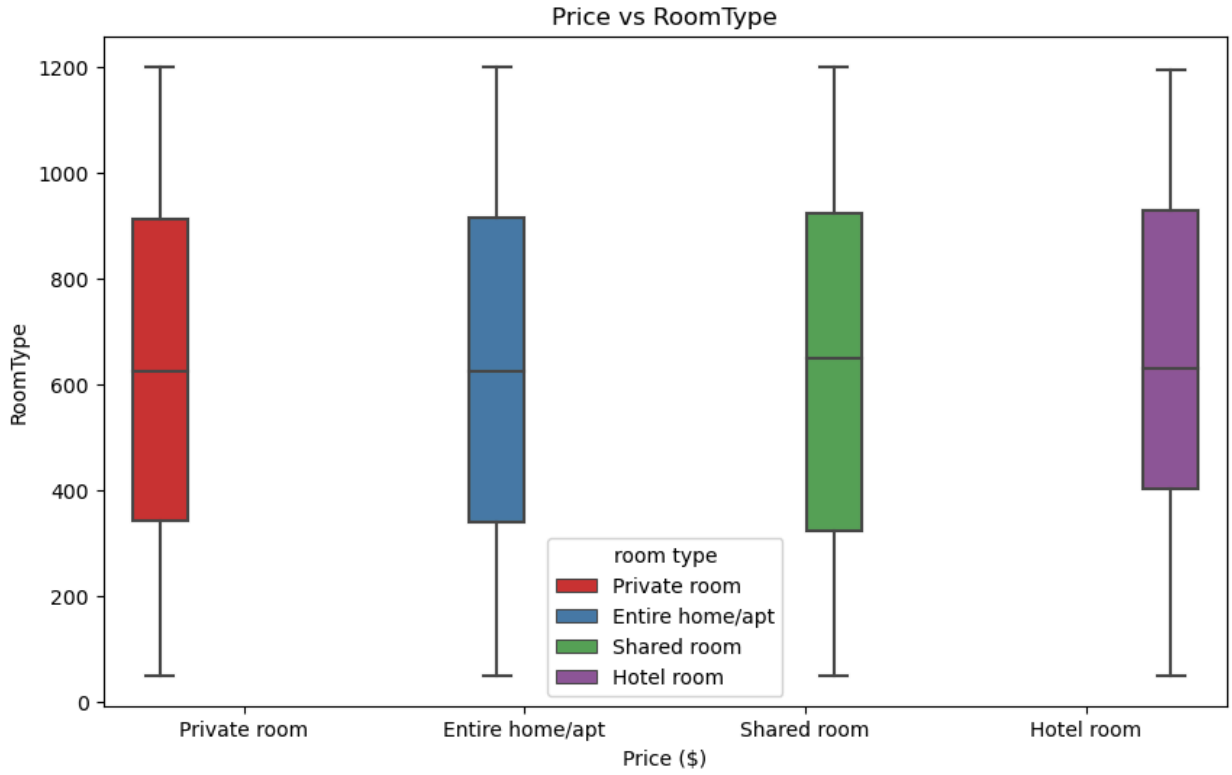
In [51]:
```
plt.figure(figsize=(10,6))
sns.countplot(data=df, y='neighbourhood group', color='blue', order=df['neighbourhood group'].value_counts(
plt.title("Number of listings by Neighbourhood Group")
plt.xlabel("Count")
plt.ylabel("Neighbourhood Group")
plt.show()
```



The vast majority of New York City's Airbnb supply is clustered in Manhattan and Brooklyn, with Queens playing a smaller but still significant role. The Bronx and Staten Island remain under-represented, suggesting either lower host participation or less visitor demand in those areas.

What is the relationship between price and room type?

In [65]:
```
plt.figure(figsize=(10,6))
sns.boxplot(data=df, y='price',x='room type',hue='room type', color='pink',palette='Set1')
plt.title("Price vs RoomType")
plt.xlabel("Price ($)")
plt.ylabel("RoomType")
plt.legend(title='room type')
plt.show()
```



From the boxplot of price by room type, a few clear take-aways emerge:

Entire homes/apartments have the greatest price dispersion: Their interquartile range stretches roughly from $350 upto 900$, and they produce the most extreme high-end outliers (well above $1 200).

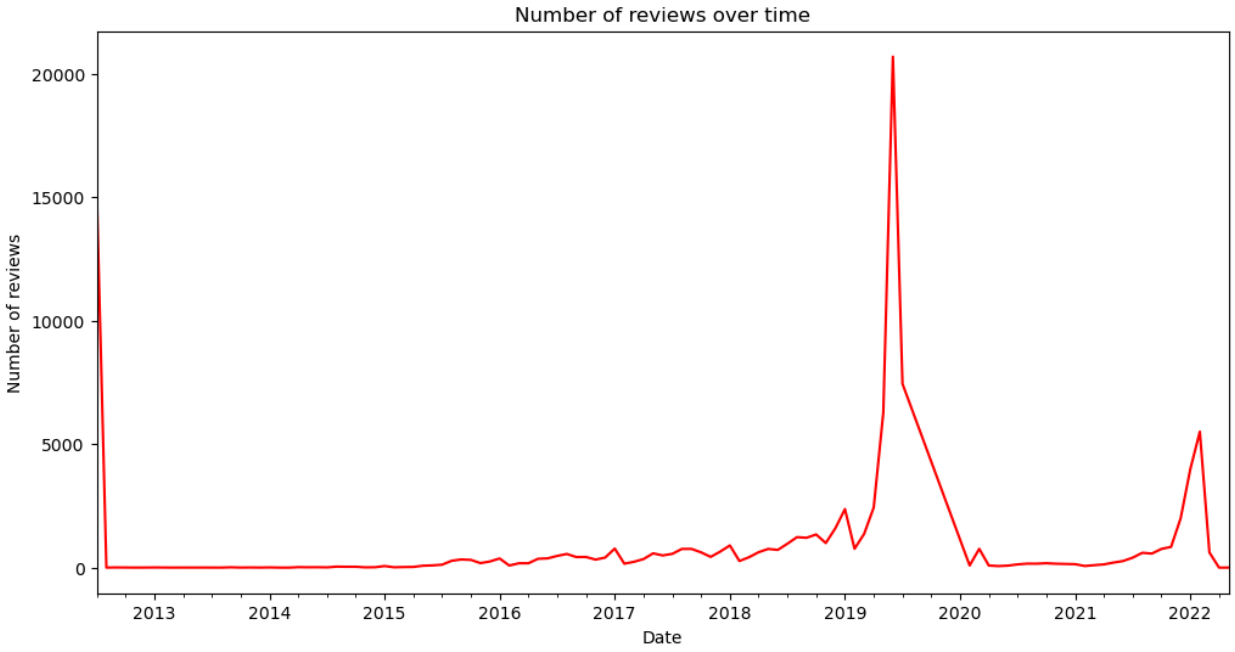
Private rooms sit in the middle: The median price is slightly below that of entire homes, with an IQR of about $350–850$.

Shared rooms and hotel rooms are the most consistent: Both have narrower IQRs (around $400–900$), indicating more predictable pricing. Shared rooms show the lowest minimums (down near $50–100$) and far fewer super-high outliers, making them the cheapest on average.

All categories exhibit some very high-price outliers: Even hotel rooms and private rooms occasionally spike above $1 000, but it's most pronounced for entire homes.

How has the number of reviews change over with time?

In [64]:
```
reviews=df.groupby(df['last review'].dt.to_period('M')).size()
plt.figure(figsize=(12,6))
reviews.plot(kind='line',color='red')
plt.title("Number of reviews over time")
plt.xlabel("Date")
plt.ylabel("Number of reviews")
plt.show()
```



Airbnb reviews grew gradually from 2013 to 2018 as the platform matured. The mid-2019 spike likely reflects a change in data collection or a one-off event, not normal user behavior. The COVID-19 pandemic caused reviews to plummet in 2020. A nascent rebound is visible by early 2022, but it remains well below the anomalous 2019 peak.