Importing Modules

In [2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

Reading the csv File

df contains:

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

In [3]:
```
df = pd.read_csv("Airbnb_Open_Data_new.csv")
```

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

Out[4]:

	id	NAME	host id	host_identity_verified	host name	neighbourhood group	nei
0	1001254	Clean & quiet apt home by the park	80014485718	unconfirmed	Madaline	Brooklyn	
1	1002102	Skylit Midtown Castle	52335172823	verified	Jenna	Manhattan	
2	1002403	THE VILLAGE OF HARLEM....NEW YORK !	78829239556	NaN	Elise	Manhattan	
3	1002755	NaN	85098326012	unconfirmed	Garry	Brooklyn	
4	1003689	Entire Apt: Spacious Studio/Loft by central park	92037596077	verified	Lyndon	Manhattan	
...	...	...	...	...	...	...	
102594	6092437	Spare room in Williamsburg	12312296767	verified	Krik	Brooklyn	
102595	6092990	Best Location near Columbia U	77864383453	unconfirmed	Mifan	Manhattan	
102596	6093542	Comfy, bright room in Brooklyn	69050334417	unconfirmed	Megan	Brooklyn	
102597	6094094	Big Studio-One Stop from Midtown	11160591270	unconfirmed	Christopher	Queens	Lo
102598	6094647	585 sf Luxury Studio	68170633372	unconfirmed	Rebecca	Manhattan	

102599 rows × 24 columns

Cleaning and Preprocessing the dataset

1. Checking datatypes of each columns in the dataset

In [5]:
```
print(df.dtypes)
```

```
id                                 int64
NAME                              object
host id                            int64
host_identity_verified            object
host name                         object
neighbourhood group               object
neighbourhood                     object
lat                              float64
long                             float64
country                           object
country code                      object
instant_bookable                  object
cancellation_policy               object
room type                         object
Construction year                float64
price                             object
service fee                       object
minimum nights                   float64
number of reviews                float64
last review                       object
reviews per month                float64
review rate number               float64
calculated host listings count   float64
availability 365                 float64
dtype: object
```

2. Filtering out and displaying only the country with name 'United States'

In [6]:
```
df = df[df['country'] == 'United States']
```

In [7]:
```
df
```

Out[7]:

	id	NAME	host id	host_identity_verified	host name	neighbourhood group	nei
0	1001254	Clean & quiet apt home by the park	80014485718	unconfirmed	Madaline	Brooklyn	
1	1002102	Skylit Midtown Castle	52335172823	verified	Jenna	Manhattan	
2	1002403	THE VILLAGE OF HARLEM....NEW YORK !	78829239556	NaN	Elise	Manhattan	
3	1002755	NaN	85098326012	unconfirmed	Garry	Brooklyn	
4	1003689	Entire Apt: Spacious Studio/Loft by central park	92037596077	verified	Lyndon	Manhattan	
...	...	...	...	...	...	...	
102594	6092437	Spare room in Williamsburg	12312296767	verified	Krik	Brooklyn	
102595	6092990	Best Location near Columbia U	77864383453	unconfirmed	Mifan	Manhattan	
102596	6093542	Comfy, bright room in Brooklyn	69050334417	unconfirmed	Megan	Brooklyn	
102597	6094094	Big Studio-One Stop from Midtown	11160591270	unconfirmed	Christopher	Queens	Lo
102598	6094647	585 sf Luxury Studio	68170633372	unconfirmed	Rebecca	Manhattan	

102067 rows × 24 columns

3. Removing $ sign from 'price' and 'service fee' columns

In [8]:
```
df = df.astype({'price': str, 'service fee': str})
```

In [9]:
```
#print(df.dtypes)
```

In [10]:
```
df['price'] = df['price'].str.replace('$', '')
```

In [11]:
```
df['service fee'] = df['service fee'].str.replace('$', '')
```

In [12]:
```
# changing the value to numeric in 'price' and 'service price'
```

In [13]:
```
df['price'] = pd.to_numeric(df['price'], errors='coerce')
df['service fee'] = pd.to_numeric(df['service fee'], errors='coerce')
```

In [14]:
```
df
```

Out[14]:

	id	NAME	host id	host_identity_verified	host name	neighbourhood group	nei
0	1001254	Clean & quiet apt home by the park	80014485718	unconfirmed	Madaline	Brooklyn	
1	1002102	Skylit Midtown Castle	52335172823	verified	Jenna	Manhattan	
2	1002403	THE VILLAGE OF HARLEM....NEW YORK !	78829239556	NaN	Elise	Manhattan	
3	1002755	NaN	85098326012	unconfirmed	Garry	Brooklyn	
4	1003689	Entire Apt: Spacious Studio/Loft by central park	92037596077	verified	Lyndon	Manhattan	
...	...	...	...	...	...	...	
102594	6092437	Spare room in Williamsburg	12312296767	verified	Krik	Brooklyn	
102595	6092990	Best Location near Columbia U	77864383453	unconfirmed	Mifan	Manhattan	
102596	6093542	Comfy, bright room in Brooklyn	69050334417	unconfirmed	Megan	Brooklyn	
102597	6094094	Big Studio-One Stop from Midtown	11160591270	unconfirmed	Christopher	Queens	Lo
102598	6094647	585 sf Luxury Studio	68170633372	unconfirmed	Rebecca	Manhattan	

102067 rows × 24 columns

4. Changing the datatype of construction year to numeric

In [15]:
```
unique_years = df['Construction year'].unique()
```

In [16]:
```
unique_years
```

Out[16]:
```
array([2020., 2007., 2005., 2009., 2013., 2015., 2004., 2008., 2010.,
       2019., 2018., 2006., 2016., 2017., 2021., 2003., 2011., 2012.,
       2022., 2014.,   nan])
```

In [17]:
```
df['Construction year'] = df['Construction year'].fillna(0000)
```

In [18]:
```
df
```

Out[18]:

	id	NAME	host id	host_identity_verified	host name	neighbourhood group	nei
0	1001254	Clean & quiet apt home by the park	80014485718	unconfirmed	Madaline	Brooklyn	
1	1002102	Skylit Midtown Castle	52335172823	verified	Jenna	Manhattan	
2	1002403	THE VILLAGE OF HARLEM....NEW YORK !	78829239556	NaN	Elise	Manhattan	
3	1002755	NaN	85098326012	unconfirmed	Garry	Brooklyn	
4	1003689	Entire Apt: Spacious Studio/Loft by central park	92037596077	verified	Lyndon	Manhattan	
...	...	...	...	...	...	...	
102594	6092437	Spare room in Williamsburg	12312296767	verified	Krik	Brooklyn	
102595	6092990	Best Location near Columbia U	77864383453	unconfirmed	Mifan	Manhattan	
102596	6093542	Comfy, bright room in Brooklyn	69050334417	unconfirmed	Megan	Brooklyn	
102597	6094094	Big Studio-One Stop from Midtown	11160591270	unconfirmed	Christopher	Queens	Lo
102598	6094647	585 sf Luxury Studio	68170633372	unconfirmed	Rebecca	Manhattan	

102067 rows × 24 columns

In [19]:
```
df = df.astype({'Construction year': int})
```

In [20]:
```
print(df.dtypes)
```

```
id                                 int64
NAME                              object
host id                            int64
host_identity_verified            object
host name                         object
neighbourhood group               object
neighbourhood                     object
lat                              float64
long                             float64
country                           object
country code                      object
instant_bookable                  object
cancellation_policy               object
room type                         object
Construction year                  int32
price                            float64
service fee                      float64
minimum nights                   float64
number of reviews                float64
last review                       object
reviews per month                float64
review rate number               float64
calculated host listings count   float64
availability 365                 float64
dtype: object
```

In [ ]:
```

```

5. Removing blank values from NAME, price

In [21]:
```
df.dropna(subset=['price', 'NAME', 'neighbourhood group', 'neighbourhood'], inplace=Tr
```

In [22]:
```
df
```

Out[22]:

	id	NAME	host id	host_identity_verified	host name	neighbourhood group	nei
0	1001254	Clean & quiet apt home by the park	80014485718	unconfirmed	Madaline	Brooklyn	
1	1002102	Skylit Midtown Castle	52335172823	verified	Jenna	Manhattan	
2	1002403	THE VILLAGE OF HARLEM....NEW YORK !	78829239556	NaN	Elise	Manhattan	
4	1003689	Entire Apt: Spacious Studio/Loft by central park	92037596077	verified	Lyndon	Manhattan	
5	1004098	Large Cozy 1 BR Apartment In Midtown East	45498551794	verified	Michelle	Manhattan	
...	...	...	...	...	...	...	...
102592	6091333	3BR/1 Ba in TriBeCa w/ outdoor deck	53266862889	unconfirmed	Nick	Manhattan	
102594	6092437	Spare room in Williamsburg	12312296767	verified	Krik	Brooklyn	
102595	6092990	Best Location near Columbia U	77864383453	unconfirmed	Mifan	Manhattan	
102596	6093542	Comfy, bright room in Brooklyn	69050334417	unconfirmed	Megan	Brooklyn	
102597	6094094	Big Studio-One Stop from Midtown	11160591270	unconfirmed	Christopher	Queens	Lo

83771 rows × 24 columns

6. Replacing nan values to unconfirmed in host_identity_verified

In [23]:
```
df['host_identity_verified'] = df['host_identity_verified'].fillna('unconfirmed')
```

7. Replacing nan values to Not Available in host name

In [24]:
```
df['host name'] = df['host name'].fillna('Not Available')
```

8. Replacing nan values to US in country code, and United States in country

In [25]:
```
df['country code'] = df['country code'].fillna('US')
df['country'] = df['country'].fillna('United States')
```

In [26]:
```
df
```

Out[26]:

	id	NAME	host id	host_identity_verified	host name	neighbourhood group	nei
0	1001254	Clean & quiet apt home by the park	80014485718	unconfirmed	Madaline	Brooklyn	
1	1002102	Skylit Midtown Castle	52335172823	verified	Jenna	Manhattan	
2	1002403	THE VILLAGE OF HARLEM....NEW YORK !	78829239556	unconfirmed	Elise	Manhattan	
4	1003689	Entire Apt: Spacious Studio/Loft by central park	92037596077	verified	Lyndon	Manhattan	
5	1004098	Large Cozy 1 BR Apartment In Midtown East	45498551794	verified	Michelle	Manhattan	
...	...	...	...	...	...	...	
102592	6091333	3BR/1 Ba in TriBeCa w/ outdoor deck	53266862889	unconfirmed	Nick	Manhattan	
102594	6092437	Spare room in Williamsburg	12312296767	verified	Krik	Brooklyn	
102595	6092990	Best Location near Columbia U	77864383453	unconfirmed	Mifan	Manhattan	
102596	6093542	Comfy, bright room in Brooklyn	69050334417	unconfirmed	Megan	Brooklyn	
102597	6094094	Big Studio-One Stop from Midtown	11160591270	unconfirmed	Christopher	Queens	Lo

83771 rows × 24 columns

9. Replacing nan values to 0 in service fee, 0 in minimum nights, 0 in number of reviews, 0 in reviews per month, 0 in review rate number, 0 in calculated host listings count, 0 in availability 365

In [27]:
```
df['service fee'] = df['service fee'].fillna(0)
df['minimum nights'] = df['minimum nights'].fillna(0)
df['number of reviews'] = df['number of reviews'].fillna(0)
df['reviews per month'] = df['reviews per month'].fillna(0)
df['review rate number'] = df['review rate number'].fillna(0)
df['calculated host listings count'] = df['calculated host listings count'].fillna(0)
df['availability 365'] = df['availability 365'].fillna(0)
```

In [28]:
```
df['instant_bookable'] = df['instant_bookable'].replace('TRUE', 'True')
df['instant_bookable'] = df['instant_bookable'].replace('FALSE', 'False')
df['instant_bookable'] = df['instant_bookable'].fillna('False')
```

In [29]:
```
#Changing the datatype
```

In [30]:
```
df = df.astype({'availability 365': int, 'calculated host listings count':int, 'review
```

In [31]:
```
print(df.dtypes)
```

```
id                                int64
NAME                             object
host id                           int64
host_identity_verified           object
host name                        object
neighbourhood group              object
neighbourhood                    object
lat                             float64
long                            float64
country                          object
country code                     object
instant_bookable                 object
cancellation_policy              object
room type                        object
Construction year                 int32
price                           float64
service fee                     float64
minimum nights                    int32
number of reviews                 int32
last review                      object
reviews per month               float64
review rate number                int32
calculated host listings count    int32
availability 365                  int32
dtype: object
```

10. Last review date is not required for the analysis, so I am deleting the column

In [32]:
```
del df['last review']
```

In [33]:
```
df
```

Out[33]:

	id	NAME	host id	host_identity_verified	host name	neighbourhood group	nei
0	1001254	Clean & quiet apt home by the park	80014485718	unconfirmed	Madaline	Brooklyn	
1	1002102	Skylit Midtown Castle	52335172823	verified	Jenna	Manhattan	
2	1002403	THE VILLAGE OF HARLEM....NEW YORK !	78829239556	unconfirmed	Elise	Manhattan	
4	1003689	Entire Apt: Spacious Studio/Loft by central park	92037596077	verified	Lyndon	Manhattan	
5	1004098	Large Cozy 1 BR Apartment In Midtown East	45498551794	verified	Michelle	Manhattan	
...	...	...	...	...	...	...	...
102592	6091333	3BR/1 Ba in TriBeCa w/ outdoor deck	53266862889	unconfirmed	Nick	Manhattan	
102594	6092437	Spare room in Williamsburg	12312296767	verified	Krik	Brooklyn	
102595	6092990	Best Location near Columbia U	77864383453	unconfirmed	Mifan	Manhattan	
102596	6093542	Comfy, bright room in Brooklyn	69050334417	unconfirmed	Megan	Brooklyn	
102597	6094094	Big Studio-One Stop from Midtown	11160591270	unconfirmed	Christopher	Queens	Lo

83771 rows × 23 columns

Exploratory Data Analysis (EDA)

In [34]:
```
descriptive_stats = df[['price', 'minimum nights', 'number of reviews', 'reviews per m
print(descriptive_stats)
```

```
              price  minimum nights  number of reviews  reviews per month  \
count  83771.000000    83771.000000       83771.000000       83771.000000
mean     524.727853        8.059889          27.289718           1.156603
std      273.454617       29.539345          49.261114           1.676523
min       50.000000     -365.000000           0.000000           0.000000
25%      288.000000        1.000000           1.000000           0.090000
50%      524.000000        3.000000           7.000000           0.480000
75%      759.000000        5.000000          30.000000           1.700000
max      999.000000     5645.000000        1024.000000          90.000000

       availability 365
count      83771.000000
mean         140.553640
std          135.085082
min          -10.000000
25%            3.000000
50%           95.000000
75%          268.000000
max          426.000000
```

In [35]:
```
correlation_matrix = df[['price', 'minimum nights', 'number of reviews', 'reviews per
print(correlation_matrix)
```

```
                     price  minimum nights  number of reviews  \
price              1.000000       -0.004564           0.004466
minimum nights    -0.004564        1.000000          -0.050142
number of reviews  0.004466       -0.050142           1.000000
reviews per month  0.000613       -0.093123           0.618985
availability 365  -0.004766        0.060358           0.096945

                   reviews per month  availability 365
price                       0.000613         -0.004766
minimum nights             -0.093123          0.060358
number of reviews           0.618985          0.096945
reviews per month           1.000000          0.071433
availability 365            0.071433          1.000000
```

1. Count of hotels according to rating

In [36]:
```
star_ratings_count = df['review rate number'].value_counts().sort_index()
print("Number of hotel by star rating:")
for rating, count in star_ratings_count.items():
    print(f"{rating} star rating: {count} hotels")
```

```
Number of hotel by star rating:
0 star rating: 264 hotels
1 star rating: 7429 hotels
2 star rating: 18839 hotels
3 star rating: 18921 hotels
4 star rating: 19086 hotels
5 star rating: 19232 hotels
```

2.Top 10 count of hotels rated 5 star by most people

In [37]:
```
five_star_hotels = df[df['review rate number'] == 5]
top_five_star_hotels = five_star_hotels.sort_values(by = 'number of reviews', ascendir
print("Top 5-star rated hotels by number of reviews:")
print(top_five_star_hotels[['id', 'NAME', 'number of reviews', 'neighbourhood']].head(
```

```
Top 5-star rated hotels by number of reviews:
             id                                                NAME  \
53823   30727779                                         City Queen
13496    8455177                           Room Near JFK Twin Beds
49915   28569391  Private Room in Spacious Quiet Apt., Elevator ...
54327   31006138                          Private room mins from JFK
49738   28471633                    My Little Guest Room in Flushing
77568   43842142  Cozy Room Family Home LGA Airport NO CLEANING FEE
90226   50833156  Cozy Room Family Home LGA Airport NO CLEANING FEE
20404   12270465  Cozy Room Family Home LGA Airport NO CLEANING FEE
100491  56502515  Cozy Room Family Home LGA Airport NO CLEANING FEE
54268   30973552  Victorian Private Brownstone Apartment & Backyard

        number of reviews  neighbourhood
53823                 583           SoHo
13496                 576        Jamaica
49915                 567         Harlem
54327                 563        Jamaica
49738                 550       Flushing
77568                 510  East Elmhurst
90226                 510  East Elmhurst
20404                 510  East Elmhurst
100491                510  East Elmhurst
54268                 491   Clinton Hill
```

3. From 2018 to 2022 yearwise highest and lowest priced room

In [38]:
```
filtered_df = df[(df['Construction year'] >= 2018) & (df['Construction year'] <= 2022)
max_priced_rooms = filtered_df.groupby('Construction year')['price'].max().sort_index(
min_priced_rooms = filtered_df.groupby('Construction year')['price'].min().sort_index(

print("Maximum priced room yearwise: ", max_priced_rooms)
print("Minimum priced room yearwise: ", min_priced_rooms)
```

```
Maximum priced room yearwise:  Construction year
2022    999.0
2021    999.0
2020    999.0
2019    999.0
2018    999.0
Name: price, dtype: float64
Minimum priced room yearwise:  Construction year
2022    50.0
2021    50.0
2020    50.0
2019    50.0
2018    50.0
Name: price, dtype: float64
```

4. Total number of listings that each host has across all their properties

In [39]:
```
host_listings_count = df.groupby('host id')['calculated host listings count'].sum()
```

In [40]:
```
#host_listings_count
```

In [41]:
```
host_listings_count.head()
```

Out[41]:
```
host id
123600518    1
124039648    1
124472619    2
130349612    1
130593431    1
Name: calculated host listings count, dtype: int32
```

5. Average pricing of different room types and average number of review

In [42]:
```
average_price_by_room_type = df.groupby('room type')[['price', 'number of reviews']].m
average_price_by_room_type
```

Out[42]:

room type	price	number of reviews
Entire home/apt	524.941396	27.257117
Hotel room	576.431579	84.031579
Private room	524.296387	27.523009
Shared room	525.925155	20.068655

6. Booking Policy and Cancellation Policy

In [43]:
```
instant_bookable_counts = df['instant_bookable'].value_counts()
calcellation_policy_counts = df['cancellation_policy'].value_counts()
```

In [44]:
```
print(instant_bookable_counts)
print(calcellation_policy_counts)
```

```
instant_bookable
False    42008
True     41741
False       22
Name: count, dtype: int64
cancellation_policy
moderate    28067
strict      27879
flexible    27825
Name: count, dtype: int64
```

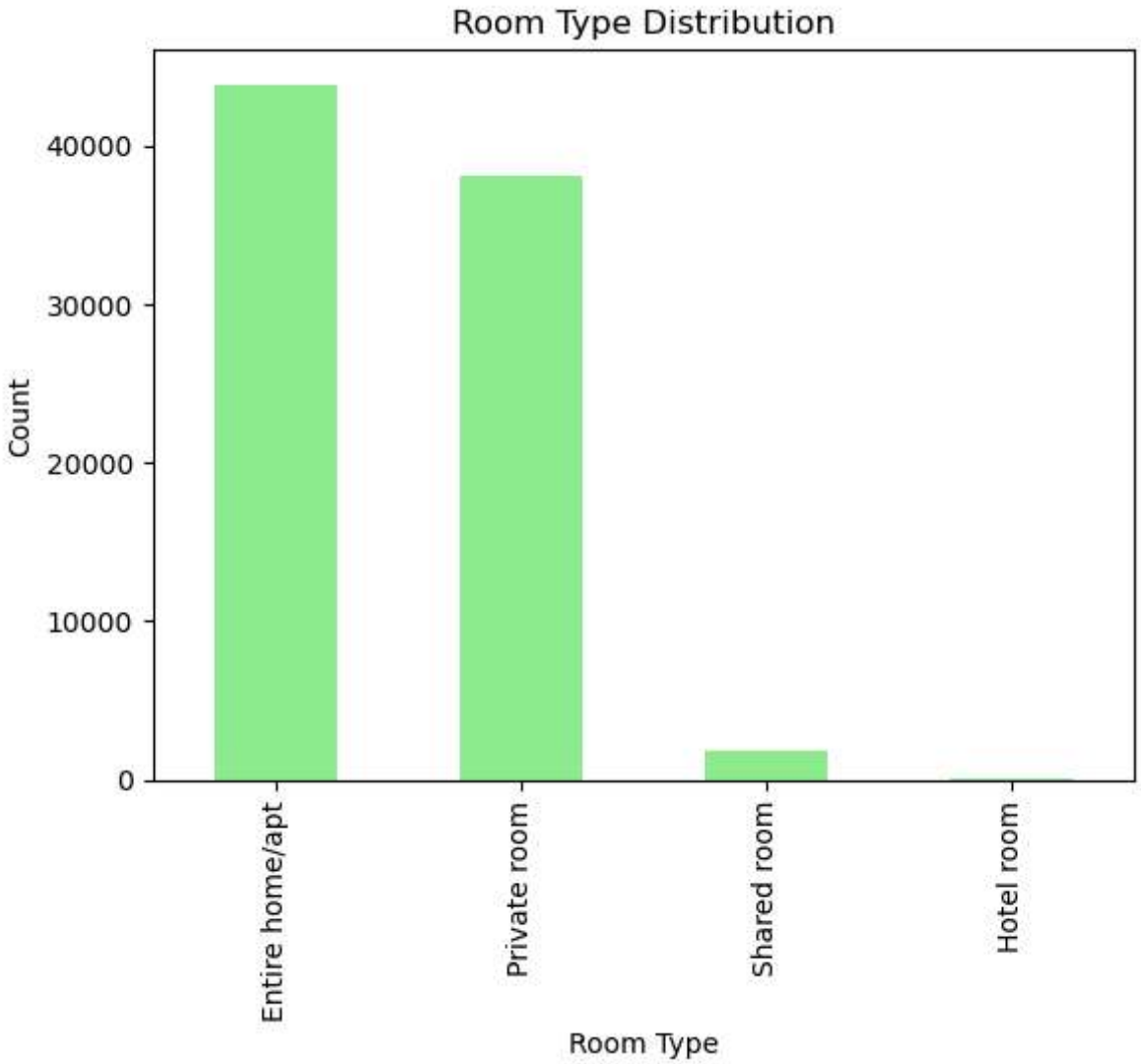
Visualization of the analysis of the dataset

1. Price Distribution

In [45]:
```
plt.hist(df['price'], bins=30, color='lightyellow', edgecolor='black')
plt.title('Price Distribution')
plt.xlabel('Price')
plt.ylabel('Frequency')
plt.show()
```

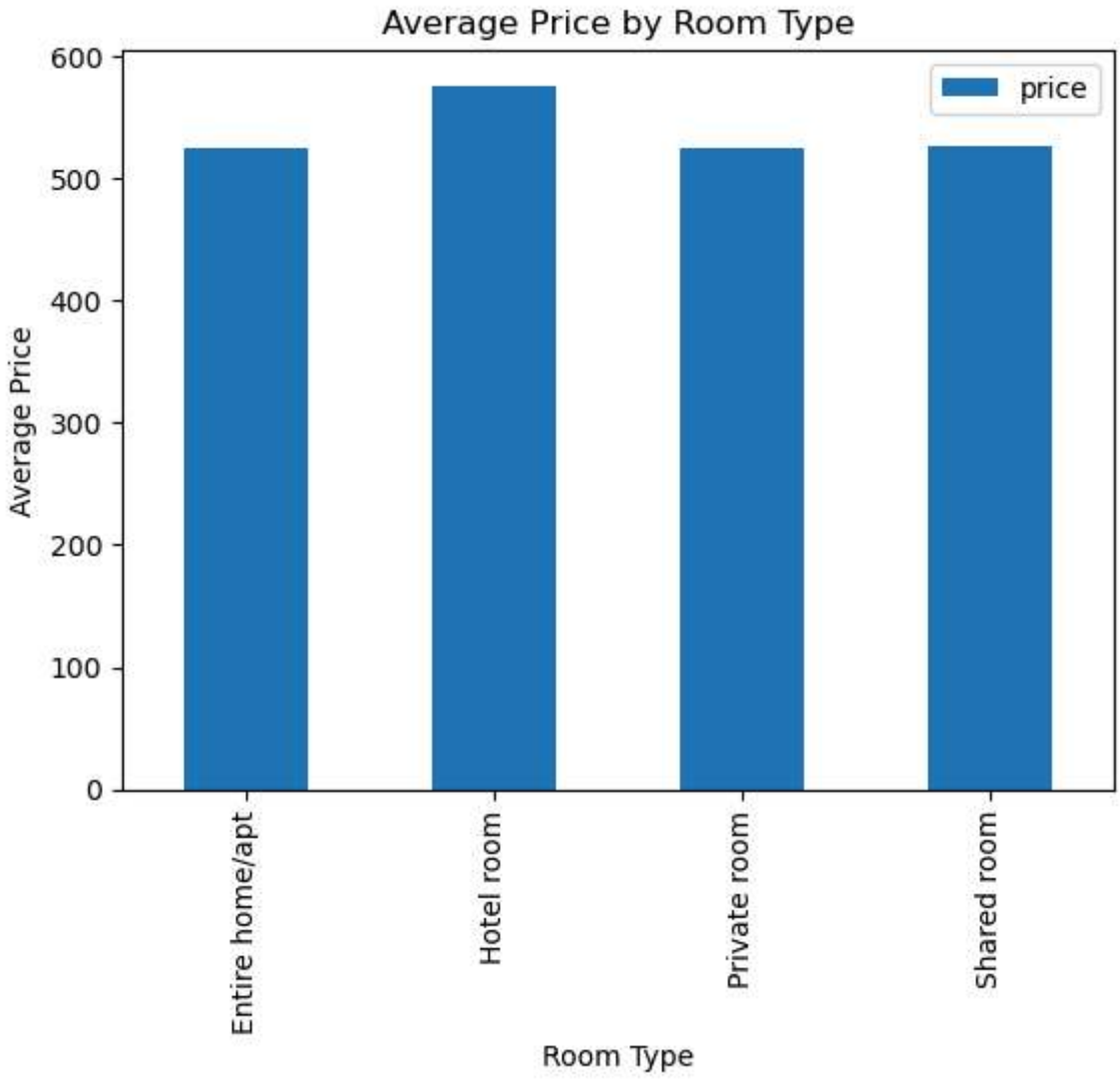
2.Room Type Distribution

In [46]:
```
room_type_counts = df['room type'].value_counts()
room_type_counts.plot(kind='bar', color='lightgreen')
plt.title('Room Type Distribution')
plt.xlabel('Room Type')
plt.ylabel('Count')
plt.show()
```



3. Average Price by Room Type

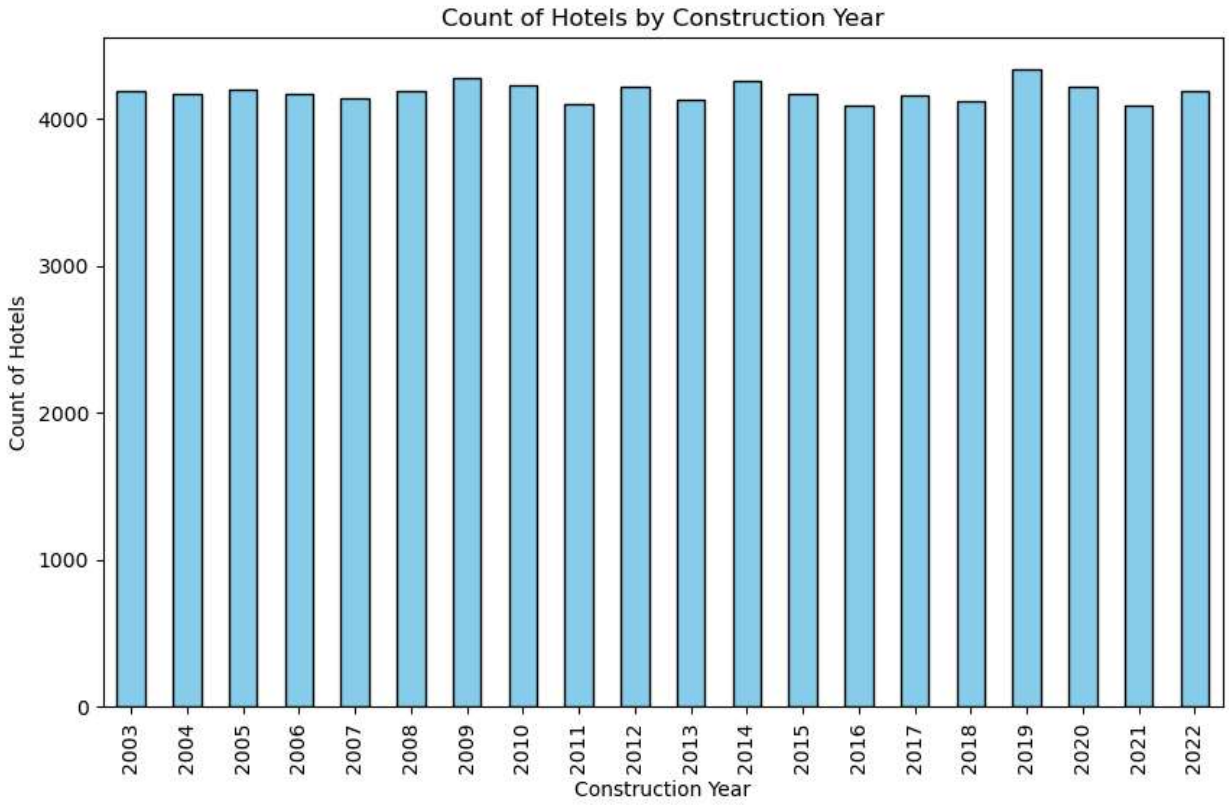
In [47]:
```
avg_price_by_room_type = df.groupby('room type')[['price']].mean()
avg_price_by_room_type.plot(kind = 'bar')
plt.title('Average Price by Room Type')
plt.xlabel('Room Type')
plt.ylabel('Average Price')
plt.show()
```



4. Count of Hotels by Construction Year

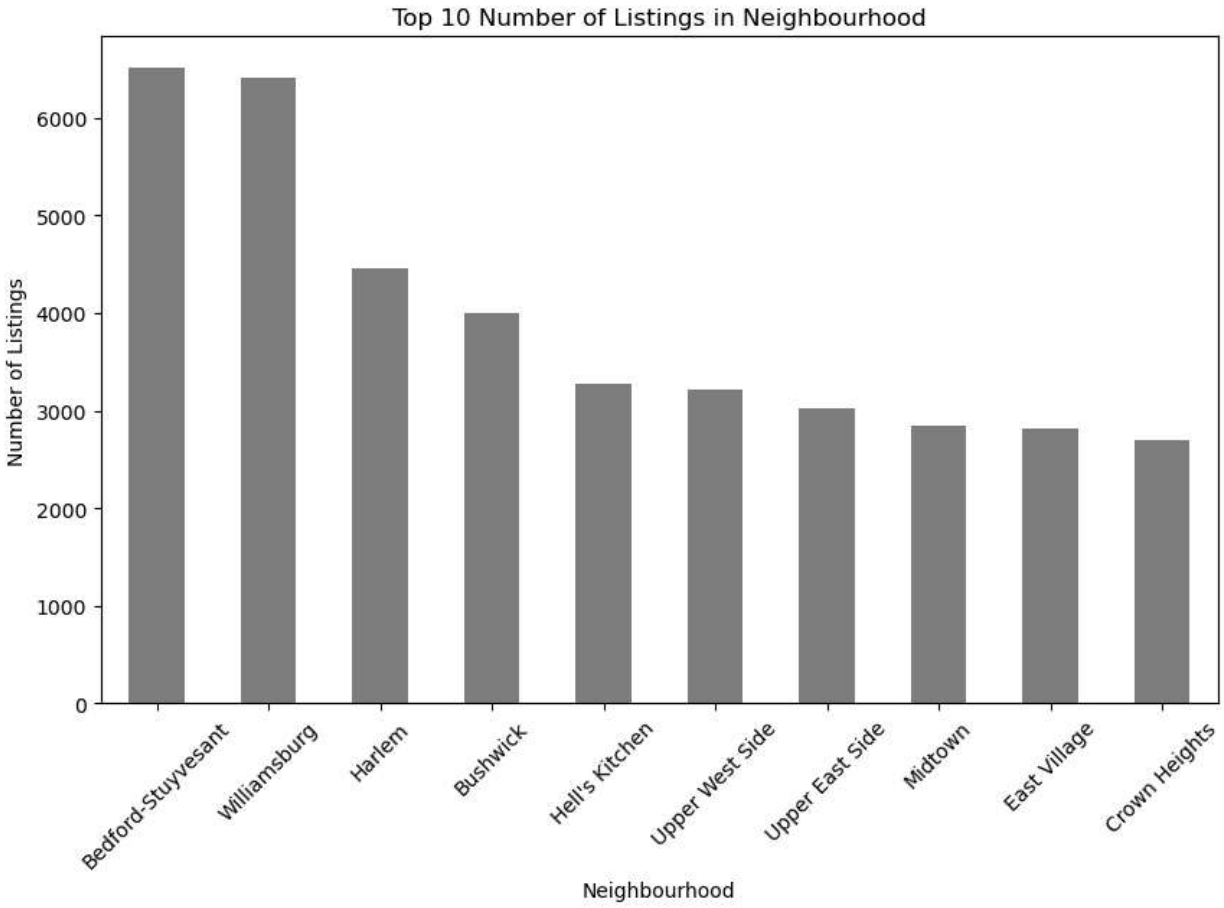
In [48]:
```
filtered_df = df[df['Construction year']!= 0]
construction_year_counts = filtered_df['Construction year'].value_counts().sort_index(

plt.figure(figsize=(10, 6))
construction_year_counts.plot(kind='bar', color='skyblue', edgecolor='black')
plt.title('Count of Hotels by Construction Year')
plt.xlabel('Construction Year')
plt.ylabel('Count of Hotels')
plt.show()
```



5. Top 10 Number of Listings in Neighbourhood

In [49]:
```
neighbourhood_counts = df['neighbourhood'].value_counts()
top_10_neighbourhood = neighbourhood_counts.head(10)
top_10_neighbourhood.plot(kind = 'bar', figsize=(10, 6), color='grey')
plt.title('Top 10 Number of Listings in Neighbourhood')
plt.xlabel('Neighbourhood')
plt.ylabel('Number of Listings')
plt.xticks(rotation=45)
plt.show()
```



6. Number of Reviews vs. Price

In [50]:
```
plt.scatter(df['number of reviews'], df['price'], color='orange', alpha=0.5)
plt.title('Number of Reviews vs. Price')
plt.xlabel('Number of Reviews')
plt.ylabel('Price')
plt.show()
```

Exporting the cleaned dataset as CSV file

In [51]:
```
df.to_csv('cleaned_dataset.csv')
```

In [ ]:

In [ ]:

In [ ]: