Capstone Project Creation

IBM SkillsBuild Europe Delivery - Data Analytics

Pre-requisite

- Understanding of Python, Power BI or Tableau
- Understanding of Data Cleaning
- Understanding Data Visualization

Level of Exercise: Intermediate

Duration: approximately 3 hours

Data Analytics of Airbnb Data:

Objective:

In this exericise, you will be performing Data Analytics on an Open Dataset dataset coming from Airbnb. Some of the tasks include

- Data Cleaning.
- Data Transformation
- Data Visualization.

Overview of Airbnb Data:

People's main criteria when visiting new places are reasonable accommodation and food. Airbnb (Air-Bed-Breakfast) is an online marketplace created to meet this need of people by renting out their homes for a short term. They offer this facility at a relatively lower price than hotels. Further people worldwide prefer the homely and economical service offered by them. They offer services across various geographical locations

Dataset Source

YOu can get the dataset for this assessment using the following link: https://www.kaggle.com/datasets/arianazmoudeh/airbnbopendata

This dataset contains information such as the neighborhood offering these services, room type, price,avaliabilty, reviews, service fee, cancellation policy and rules to use the house. This analysis will help airbnb in improving its services.

So all the best for your Data Analytics Journey on Airbnb data!!!

Task 1: Data Loading (Python)

1. Read the csv file and load it into a pandas dataframe.
2. Display the first five rows of your dataframe.
3. Display the data types of the columns.

In [31]:
```
#impoer necessary libraries
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
```

In [32]:
```
## Read the csv file
df = pd.read_csv ("C:/Users/agbat/OneDrive/Mystry project DA202.2/IBM AIR BNB PROJECT/archive/Airbnb_Open_Data.
```

```
C:\Users\agbat\AppData\Local\Temp\ipykernel_23788\3510272140.py:2: DtypeWarning: Columns (25) have mixed types.
Specify dtype option on import or set low_memory=False.
  df = pd.read_csv ("C:/Users/agbat/OneDrive/Mystry project DA202.2/IBM AIR BNB PROJECT/archive/Airbnb_Open_Dat
a.csv")
```

In [33]:
```
## Display the first 5 rows
df.head()
```

Out[33]:

	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	...
2	1002403	THE VILLAGE OF HARLEM....NEW YORK !	78829239556	NaN	Elise	Manhattan	Harlem	40.80902	-73.94190	United States	...
3	1002755	NaN	85098326012	unconfirmed	Garry	Brooklyn	Clinton Hill	40.68514	-73.95976	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 rows × 26 columns

In [34]:
```
## Display the data types
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
house_rules                      object
license                          object
dtype: object
```

Task 2a: Data Cleaning (Any Tool)

1. Drop some of the unwanted columns. These include `host id`, `id`, `country` and `country code` from the dataset.
2. State the reason for not including these columns for your Data Analytics.

If using Python for this exercise, please include the code in the cells below. If using any other tool, please include screenshoots before and after the elimination of the columns.

In [35]:
```
#Drop some of the unwanted columns. These include host id, id, country and country code from the dataset.
df.drop (columns= ["host id", "id", "country", "country code"] , inplace= True )
```

In [36]:
```
(df.head())
```

Out[36]:

	NAME	host_identity_verified	host name	neighbourhood group	neighbourhood	lat	long	instant_bookable	cancellation_policy
0	Clean & quiet apt home by the park	unconfirmed	Madaline	Brooklyn	Kensington	40.64749	-73.97237	False	strict
1	Skylit Midtown Castle	verified	Jenna	Manhattan	Midtown	40.75362	-73.98377	False	moderate
2	THE VILLAGE OF HARLEM....NEW YORK !	NaN	Elise	Manhattan	Harlem	40.80902	-73.94190	True	flexible
3	NaN	unconfirmed	Garry	Brooklyn	Clinton Hill	40.68514	-73.95976	True	moderate
4	Entire Apt: Spacious Studio/Loft by central park	verified	Lyndon	Manhattan	East Harlem	40.79851	-73.94399	False	moderate

5 rows × 22 columns

In [ ]:

reason for not including these columns for my Data Analytics.

Host ID and ID: These columns most likely contain the host's and the listing's individual IDs. These kinds of unique IDs are typically abandoned in data analytics because they don't offer useful information for analysis. A distinct 'id' for each row is there, but it has no bearing on patterns or trends. Likewise, 'host id' is unique to a single host and does not contribute to a more comprehensive study.

Country and Country Code: These columns become unnecessary if the dataset is restricted to a single nation or if all of the data originates there. The 'country' and 'country code' columns can be safely removed in this instance, as they neither vary nor offer new information for analysis within a single-country context.

Task 2b: Data Cleaning (Python)

- Check for missing values in the dataframe and display the count in ascending order. **If the values are missing, impute the values as per the datatype of the columns.**
- Check whether there are any duplicate values in the dataframe and, if present, remove them.
- Display the total number of records in the dataframe before and after removing the duplicates.

In [37]:
```
##check for missing values in the data frame and display the count in ascending order
missing_values = df.isnull().sum()
missing_values_sorted = missing_values.sort_values()

print("Missing values count in each column (ascending order):")
print(missing_values_sorted)
```

```
Missing values count in each column (ascending order):
room type                              0
lat                                    8
long                                   8
neighbourhood                         16
neighbourhood group                   29
cancellation_policy                   76
instant_bookable                     105
number of reviews                    183
Construction year                    214
price                                247
NAME                                 250
service fee                          273
host_identity_verified               289
calculated host listings count       319
review rate number                   326
host name                            406
minimum nights                       409
availability 365                     448
reviews per month                  15879
last review                        15893
house_rules                        52131
license                           102597
dtype: int64
```

In [38]:

```
# Impute missing values
for column in df.columns:
    if df[column].dtype == 'float64' or df[column].dtype == 'int64':
        # For numerical columns, replace missing values with the mean
        df[column].fillna(df[column].mean(), inplace=True)
    else:
        # For non-numerical columns, replace missing values with the mode (most frequent value)
        df[column].fillna(df[column].mode()[0], inplace=True)

# Display the dataframe after imputation
print("Dataframe after imputation:")
(df.head())
```

```
Dataframe after imputation:
```

Out[38]:

	NAME	host_identity_verified	host name	neighbourhood group	neighbourhood	lat	long	instant_bookable	cancellation_policy
0	Clean & quiet apt home by the park	unconfirmed	Madaline	Brooklyn	Kensington	40.64749	-73.97237	False	strict
1	Skylit Midtown Castle	verified	Jenna	Manhattan	Midtown	40.75362	-73.98377	False	moderate
2	THE VILLAGE OF HARLEM....NEW YORK !	unconfirmed	Elise	Manhattan	Harlem	40.80902	-73.94190	True	flexible
3	Home away from home	unconfirmed	Garry	Brooklyn	Clinton Hill	40.68514	-73.95976	True	moderate
4	Entire Apt: Spacious Studio/Loft by central park	verified	Lyndon	Manhattan	East Harlem	40.79851	-73.94399	False	moderate

5 rows × 22 columns

In [39]:

```
# Check for duplicate rows
duplicate_rows = df[df.duplicated()]
print("Number of duplicate rows: ", duplicate_rows.shape[0])

# Remove duplicate rows
df = df.drop_duplicates()

# Verify that duplicates are removed
print("Number of duplicate rows after removal: ", df[df.duplicated()].shape[0])
```

```
Number of duplicate rows:  3453
Number of duplicate rows after removal:  0
```

In [40]:

```
## Display the total number of records in the dataframe after removing the duplicates.
print("Total number of records after removing duplicates: ", df.shape[0])
```

```
Total number of records after removing duplicates:  99146
```

Task 3: Data Transformation (Any Tool)

- Rename the column `availability 365` to `days_booked`
- Convert all column names to lowercase and replace the spaces in the column names with an underscore "_".
- Remove the dollar sign and comma from the columns `price` and `service_fee`. If necessary, convert these two columns to the appropriate data type.

If using Python for this exercise, please include the code in the cells below. If using any other tool, please include screenshoots of your work.

In [41]:
```
## Rename the column availability 365 to days_booked.
# Renaming the column
df.rename(columns={'availability 365': 'days_booked'}, inplace=True)

# Display the DataFrame
(df.head())
```

Out[41]:

	NAME	host_identity_verified	host name	neighbourhood group	neighbourhood	lat	long	instant_bookable	cancellation_policy
0	Clean & quiet apt home by the park	unconfirmed	Madaline	Brooklyn	Kensington	40.64749	-73.97237	False	strict
1	Skylit Midtown Castle	verified	Jenna	Manhattan	Midtown	40.75362	-73.98377	False	moderate
2	THE VILLAGE OF HARLEM....NEW YORK !	unconfirmed	Elise	Manhattan	Harlem	40.80902	-73.94190	True	flexible
3	Home away from home	unconfirmed	Garry	Brooklyn	Clinton Hill	40.68514	-73.95976	True	moderate
4	Entire Apt: Spacious Studio/Loft by central park	verified	Lyndon	Manhattan	East Harlem	40.79851	-73.94399	False	moderate

5 rows × 22 columns

In [42]:
```
## Convert all column names to lowercase and replace the spaces with an underscore "_"
df.columns = [col.lower().replace(' ', '_') for col in df.columns]
# Display the DataFrame to verify changes
(df.head())
```

Out[42]:

	name	host_identity_verified	host_name	neighbourhood_group	neighbourhood	lat	long	instant_bookable	cancellati
0	Clean & quiet apt home by the park	unconfirmed	Madaline	Brooklyn	Kensington	40.64749	-73.97237	False	
1	Skylit Midtown Castle	verified	Jenna	Manhattan	Midtown	40.75362	-73.98377	False	
2	THE VILLAGE OF HARLEM....NEW YORK !	unconfirmed	Elise	Manhattan	Harlem	40.80902	-73.94190	True	
3	Home away from home	unconfirmed	Garry	Brooklyn	Clinton Hill	40.68514	-73.95976	True	
4	Entire Apt: Spacious Studio/Loft by central park	verified	Lyndon	Manhattan	East Harlem	40.79851	-73.94399	False	

5 rows × 22 columns

In [43]:
```
## Remove the dollar sign and comma from the columns. If necessary, convert these two columns to the appropriat
df['service_fee'] = df['service_fee'].str.replace('$', '').str.replace(',', '').astype(float)
df['price'] = df['price'].str.replace('$', '').str.replace(',', '').astype(float)
# Display the first few rows of the DataFrame
df.head()
```

Out[43]:

	name	host_identity_verified	host_name	neighbourhood_group	neighbourhood	lat	long	instant_bookable	cancellati
0	Clean & quiet apt home by the park	unconfirmed	Madaline	Brooklyn	Kensington	40.64749	-73.97237	False	
1	Skylit Midtown Castle	verified	Jenna	Manhattan	Midtown	40.75362	-73.98377	False	
2	THE VILLAGE OF HARLEM....NEW YORK !	unconfirmed	Elise	Manhattan	Harlem	40.80902	-73.94190	True	
3	Home away from home	unconfirmed	Garry	Brooklyn	Clinton Hill	40.68514	-73.95976	True	
4	Entire Apt: Spacious Studio/Loft by central park	verified	Lyndon	Manhattan	East Harlem	40.79851	-73.94399	False	

5 rows × 22 columns

Task 4: Exploratory Data Analysis (Any Tool)

- List the count of various room types avaliable in the dataset.
- Which room type has the most strict cancellation policy?
- List the average price per neighborhood group, and highlight the most expensive neighborhood to rent from.

If using Python for this exercise, please include the code in the cells below. If using any other tool, please include screenshoots of your work.

In [44]:
```
#List the count of various room types avaliable in the dataset.
# Group by the 'room_type' column and count the occurrences
room_type_counts = df['room_type'].value_counts()

# Display the counts of each room type
print(room_type_counts)
```

```
room_type
Entire home/apt    51995
Private room       44887
Shared room         2149
Hotel room           115
Name: count, dtype: int64
```

In [79]:
```
## Which room type adheres to more strict cancellation policy

# Filter the data to include only rows with a strict cancellation policy
strict_policy_df = df[df['cancellation_policy'] == 'strict']

# Group by room type and count the occurrences
strict_policy_count = strict_policy_df.groupby('room_type').size()

# Display the results
print(strict_policy_count)
print ('most strict cancellation policy room type is:', 'Entire home/apt with 17239')
```

```
room_type
Entire home/apt    17239
Hotel room            34
Private room       14936
Shared room          718
dtype: int64
most strict cancellation policy room type is: Entire home/apt with 17239
```

In [78]:
```
## List the prices by neighborhood group and also mention which is the most expensive neighborhood group for re
# Group by neighborhood group and calculate average price
average_prices = df.groupby('neighbourhood_group')['price'].mean()

# Display the average prices by neighborhood group
print("Average Prices by Neighborhood Group:")
print(average_prices)


# Sort the average prices in descending order
sorted_average_prices = average_prices.sort_values(ascending=False)

# Display the sorted average prices by neighborhood group
```

```
print("Average Prices by Neighborhood Group (Descending):")
print(sorted_average_prices)

# The most expensive neighborhood group is now the first one in the sorted list
most_expensive = sorted_average_prices.index[0]
print ('The most expensive neighborhood to rent:', 'Queens at an average of $628.668822')
```

```
Average Prices by Neighborhood Group:
neighbourhood_group
Bronx            625.271511
Brooklyn         625.451927
Manhattan        621.641437
Queens           628.668822
Staten Island    625.060870
brookln          580.000000
manhatan         460.000000
Name: price, dtype: float64
Average Prices by Neighborhood Group (Descending):
neighbourhood_group
Queens           628.668822
Brooklyn         625.451927
Bronx            625.271511
Staten Island    625.060870
Manhattan        621.641437
brookln          580.000000
manhatan         460.000000
Name: price, dtype: float64
The most expensive neighborhood to rent: Queens at an average of $628.668822
```

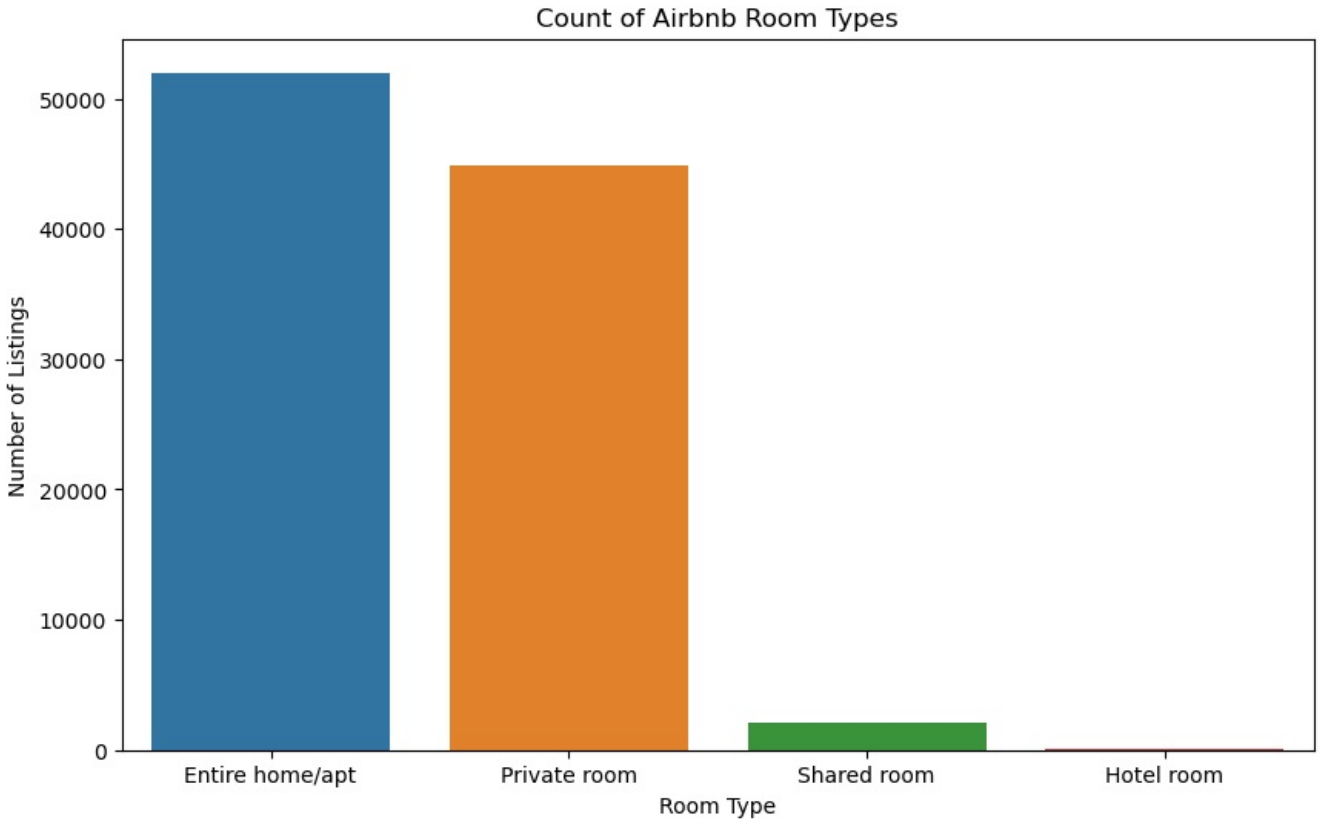
Task 5a: Data Visualization (Any Tool)

- List the count of various room types avaliable with Airnb
- Which room type adheres to more strict cancellation policy
- List the prices by neighborhood group and also mention which is the most expensive neighborhood group for rentals
- List the top 10 neighborhoods in the increasing order of their price with the help of a horizontal bar graph. Which is the cheapest neighborhood.
- List the neighborhoods which offer short term rentals within 10 days. Illustrate with a bar graph
- List the prices with respect to room type using a bar graph and also state your inferences.
- Create a pie chart that shows distribution of booked days for each neighborhood group .Which neighborhood has the highest booking percentage.

If using Python for this exercise, please include the code in the cells below. If using any other tool, please include screenshoots of your work.

In [48]:
```
#List the count of various room types avaliable with Airnb
# Count the number of listings for each room type
room_type_counts = df['room_type'].value_counts()

# Print the counts
print(room_type_counts)

# Visualization
plt.figure(figsize=(10, 6))
sns.barplot(x=room_type_counts.index, y=room_type_counts.values)
plt.title('Count of Airbnb Room Types')
plt.xlabel('Room Type')
plt.ylabel('Number of Listings')
plt.show()
```

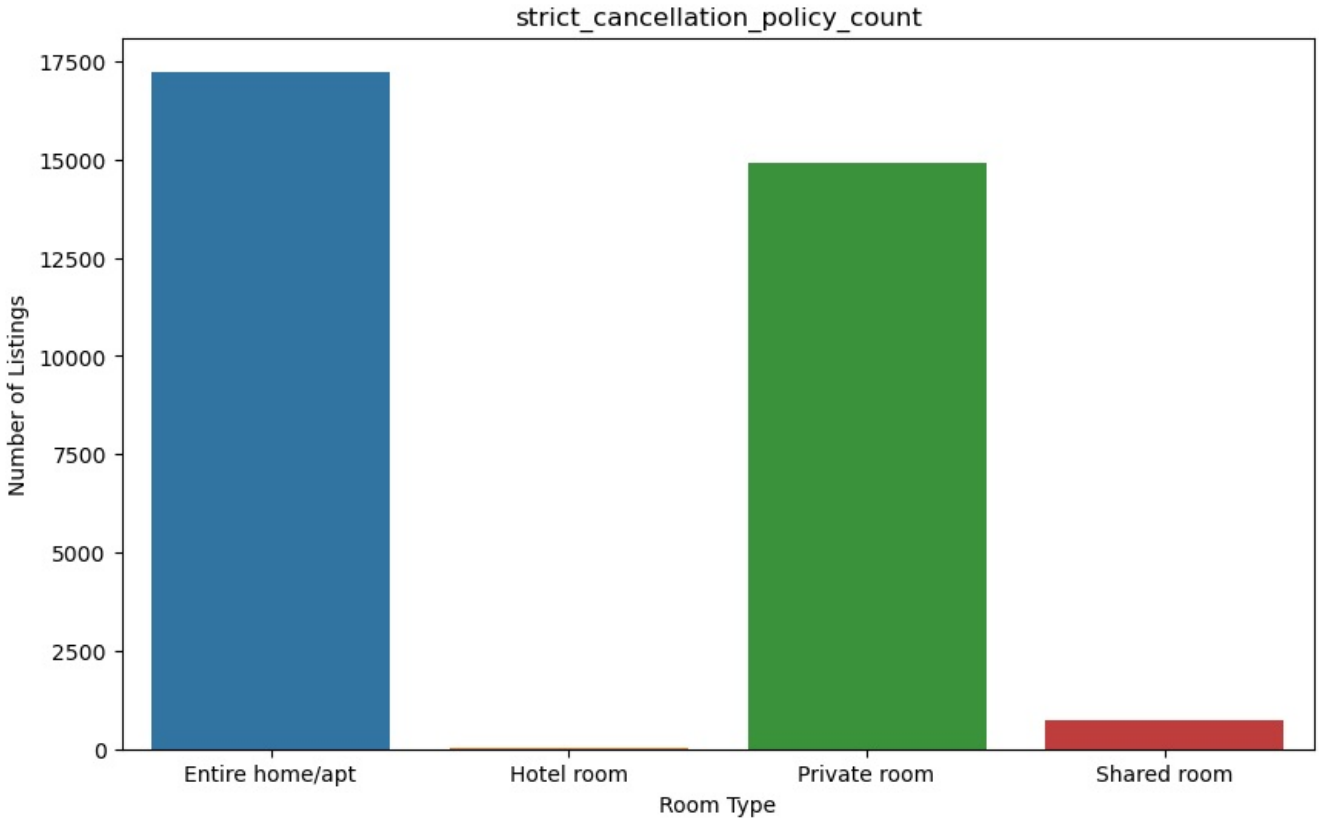
```
room_type
Entire home/apt    51995
Private room       44887
Shared room         2149
Hotel room           115
Name: count, dtype: int64
```



In [49]:

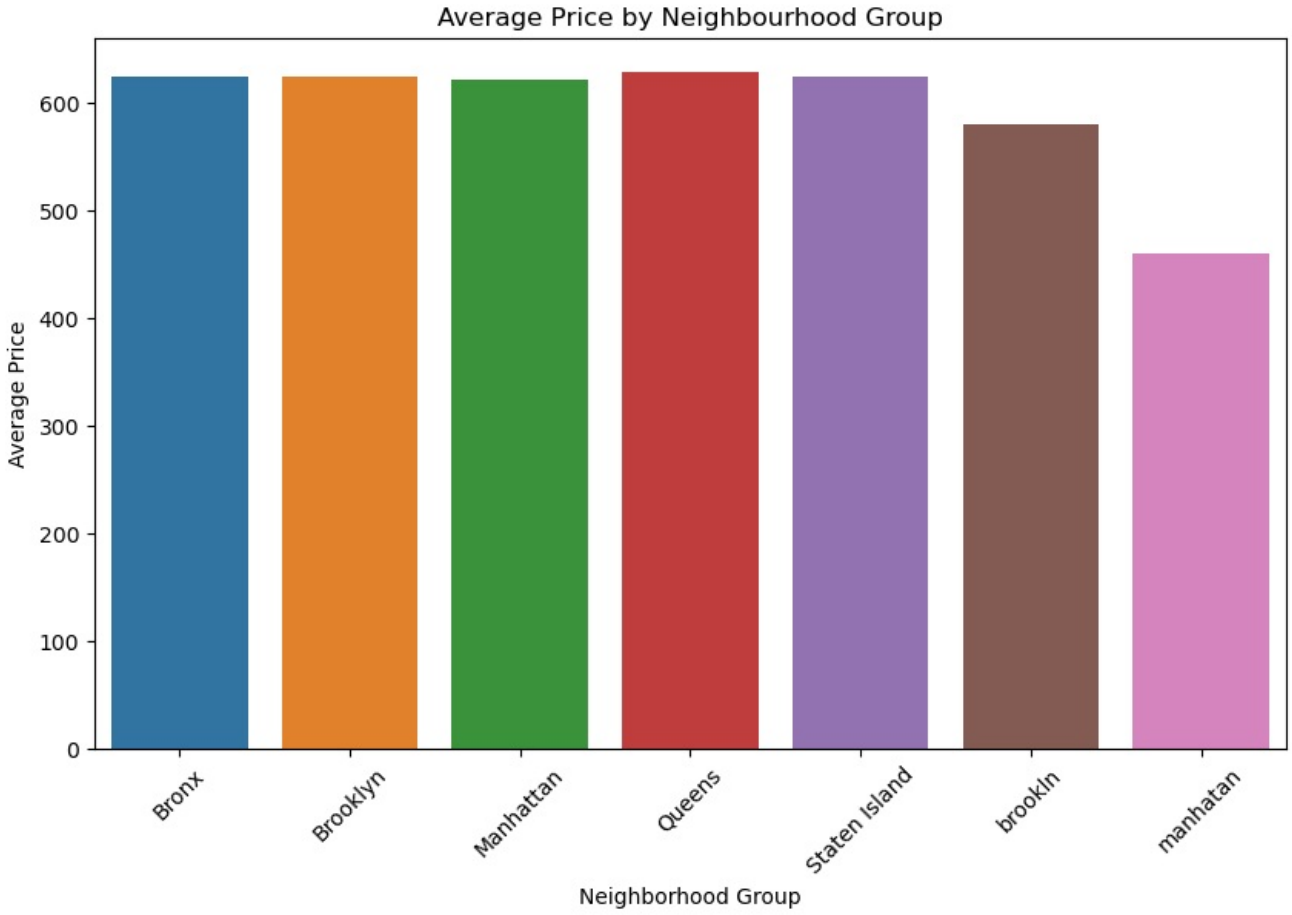
```
#Which room type adheres to more strict cancellation policy
# Filter the data to include only rows with a strict cancellation policy
strict_policy_df = df[df['cancellation_policy'] == 'strict']

# Group by room type and count the occurrences
strict_policy_count = strict_policy_df.groupby('room_type').size()

# Display the results
print(strict_policy_count)

# Visualization
plt.figure(figsize=(10, 6))
sns.barplot(x=strict_policy_count.index, y=strict_policy_count.values)
plt.title('strict_cancellation_policy_count')
plt.xlabel('Room Type')
plt.ylabel('Number of Listings')
plt.show()
```

```
room_type
Entire home/apt    17239
Hotel room            34
Private room       14936
Shared room          718
dtype: int64
```

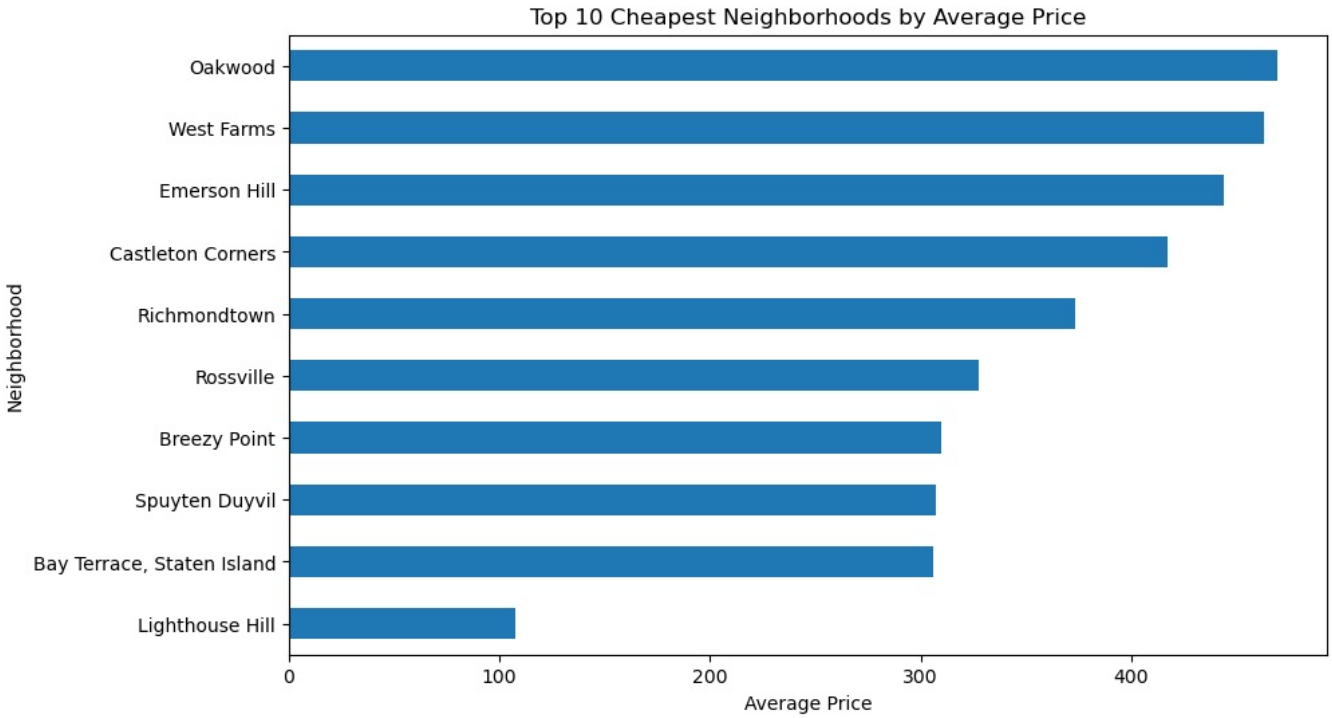


In [50]:
```
#List the prices by neighborhood group and also mention which is the most expensive neighborhood group for rent
grouped_prices = df.groupby('neighbourhood_group')['price'].mean()

# Identify the most expensive neighborhood group
most_expensive = grouped_prices.idxmax()

# Print the most expensive neighborhood group and its average price
print(f"The most expensive neighbourhood group is {most_expensive} with an average price of {grouped_prices[mos

# Visualization
plt.figure(figsize=(10, 6))
sns.barplot(x=grouped_prices.index, y=grouped_prices.values)
plt.title('Average Price by Neighbourhood Group')
plt.xlabel('Neighborhood Group')
plt.ylabel('Average Price')
plt.xticks(rotation=45)
plt.show()
```

```
The most expensive neighbourhood group is Queens with an average price of 628.67
```



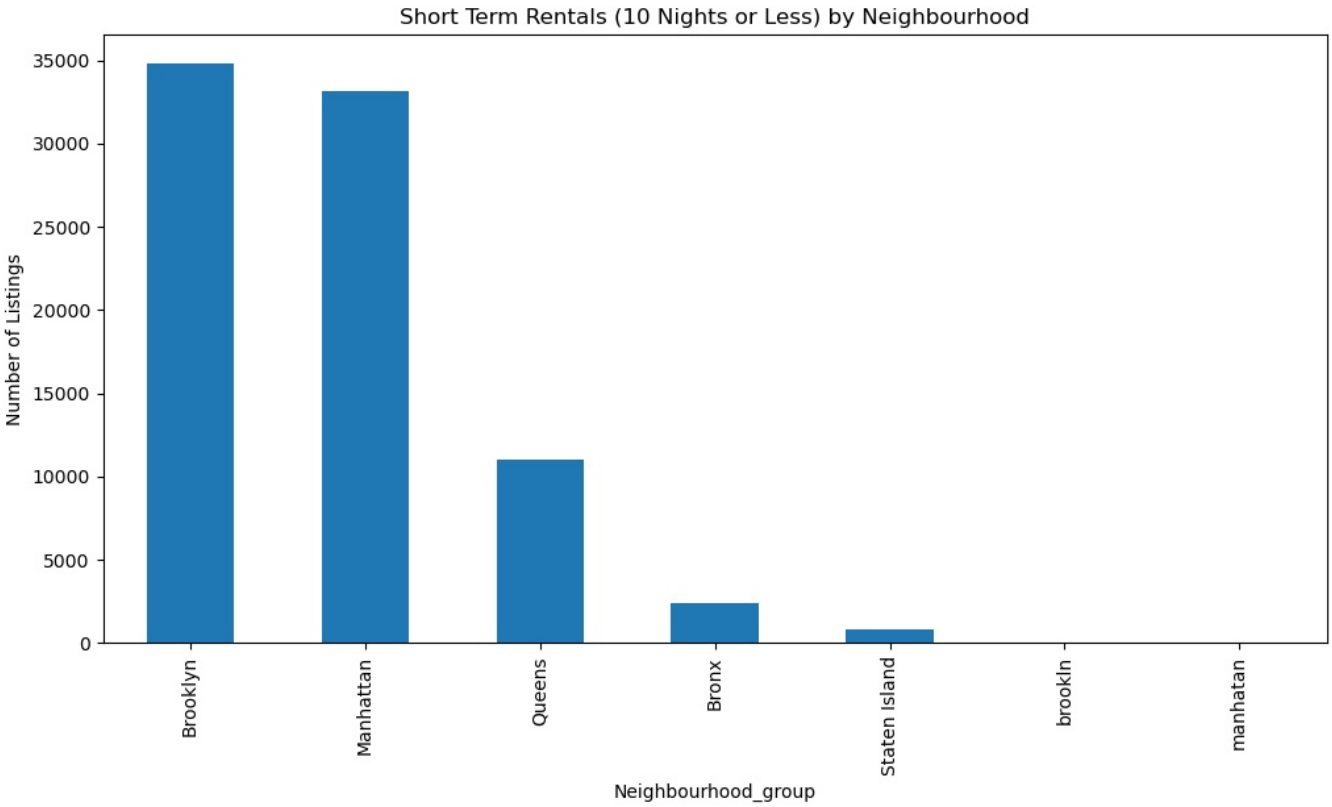
In [76]:
```
#List the top 10 neighborhoods in the increasing order of their price with the help of a horizontal bar graph.
# Aggregate and sort data
neighbourhood_prices = df.groupby('neighbourhood')['price'].mean().sort_values()
top_10_neighborhoods = neighbourhood_prices.head(10)

# Plotting
plt.figure(figsize=(10, 6))
top_10_neighborhoods.plot(kind='barh')
plt.xlabel('Average Price')
plt.ylabel('Neighborhood')
plt.title('Top 10 Cheapest Neighborhoods by Average Price')
plt.show()
print ('Cheapest Neigbourhood:', 'Lighthouse Hill')
```




```
Cheapest Neigbourhood: Lighthouse Hill
```

In [53]:

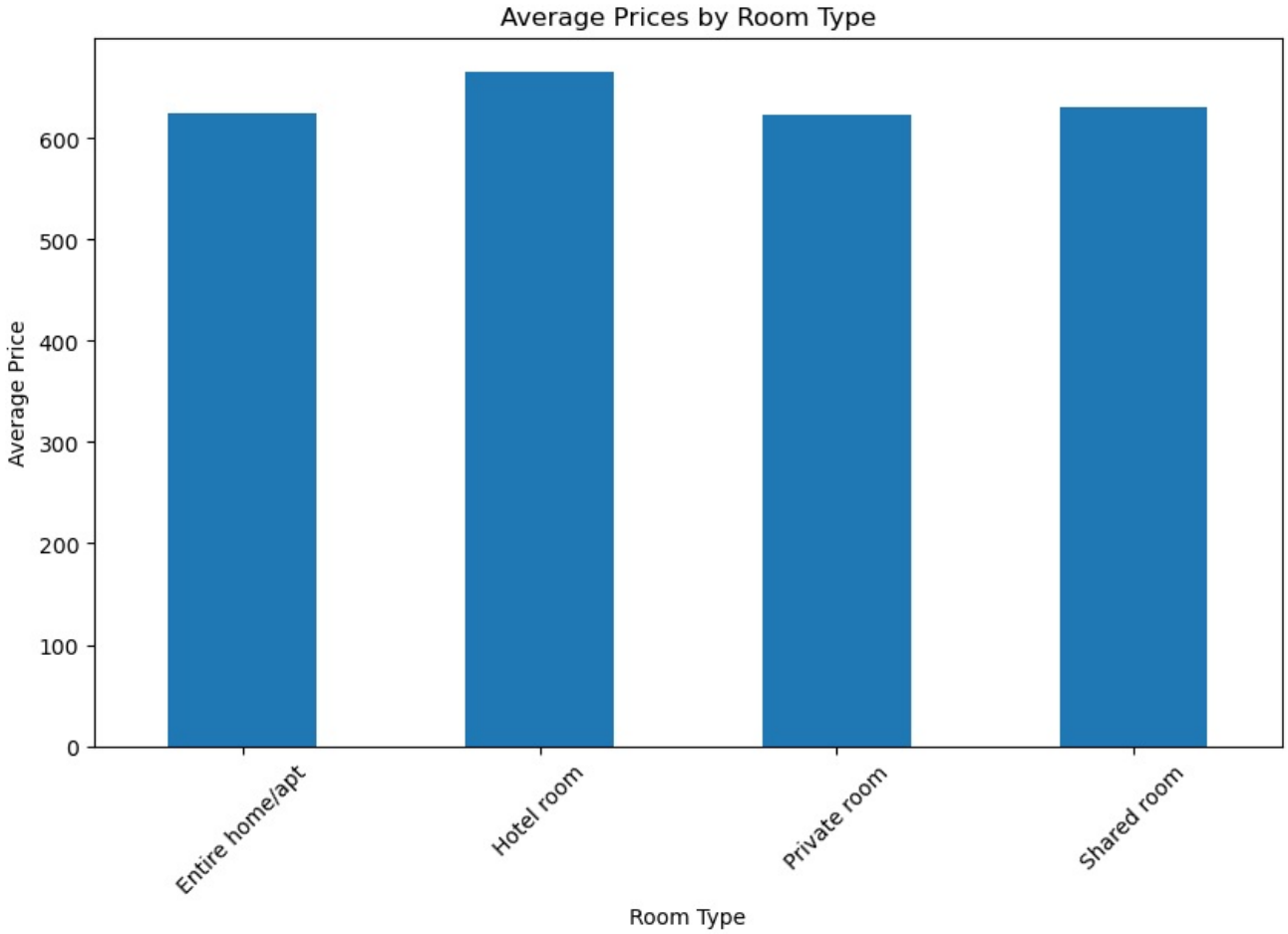
```
#List the neighborhoods which offer short term rentals within 10 days. Illustrate with a bar graph
# Filter the dataset for listings that require 10 or fewer nights
short_term_rentals = df[df['minimum_nights'] <= 10]

# Count the number of listings in each neighborhood
neighbourhood_counts = short_term_rentals['neighbourhood_group'].value_counts()

# Create a bar graph
plt.figure(figsize=(12, 6))
neighbourhood_counts.plot(kind='bar')
plt.title('Short Term Rentals (10 Nights or Less) by Neighbourhood')
plt.xlabel('Neighbourhood_group')
plt.ylabel('Number of Listings')
plt.show()
```



In [75]:

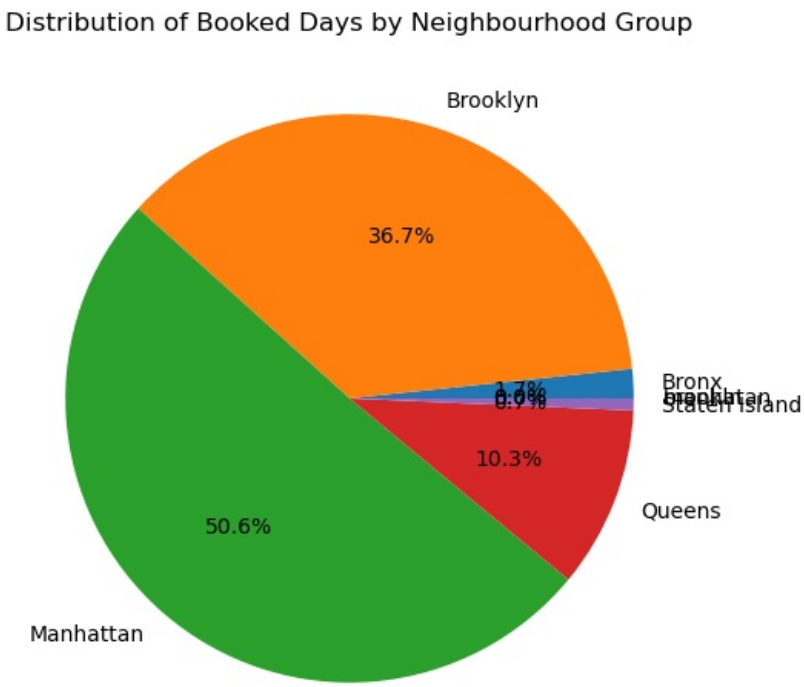
```
#List the prices with respect to room type using a bar graph and also state your inferences

# Group by room type and calculate average price
room_prices = df.groupby('room_type')['price'].mean()

# Plotting
plt.figure(figsize=(10,6))
room_prices.plot(kind='bar')
```

```
plt.title('Average Prices by Room Type')
plt.xlabel('Room Type')
plt.ylabel('Average Price')
plt.xticks(rotation=45)
plt.show()
```



In [ ]:
```
#Inferences: The following conclusions are suggested by the bar graph created using the Airbnb dataset:
The average price of 'hotel rooms' is the highest, which makes sense given that these types of lodging frequent
According to the data, 'shared rooms' have the second-highest average cost. This is uncommon because, because o
The third-highest average price is shown for 'entire homes and apartments'. This is a little surprising, as sha
The average price of 'private rooms' is the lowest, which is consistent with the widely held belief that they a
```

In [80]:
```
#Create a pie chart that shows  distribution of booked days for each neighborhood group .Which neighborhood has
#  Group and Aggregate Data
grouped_data = df.groupby('neighbourhood_group')['minimum_nights'].sum()

# Calculate Percentages
total_booked_days = grouped_data.sum()
percentages = grouped_data / total_booked_days * 100

# Create the Pie Chart
plt.figure(figsize=(10, 6))
plt.pie(percentages, labels=percentages.index, autopct='%1.1f%%')
plt.title('Distribution of Booked Days by Neighbourhood Group')

# Show the plot
plt.show()

# Identify Highest Booking Percentage
highest_booking = percentages.idxmax()
print(f"The neighbourhood with the highest booking percentage is {highest_booking}")
```




```
The neighbourhood with the highest booking percentage is Manhattan
```

Task 5b: Data Visualization (Any Tool)

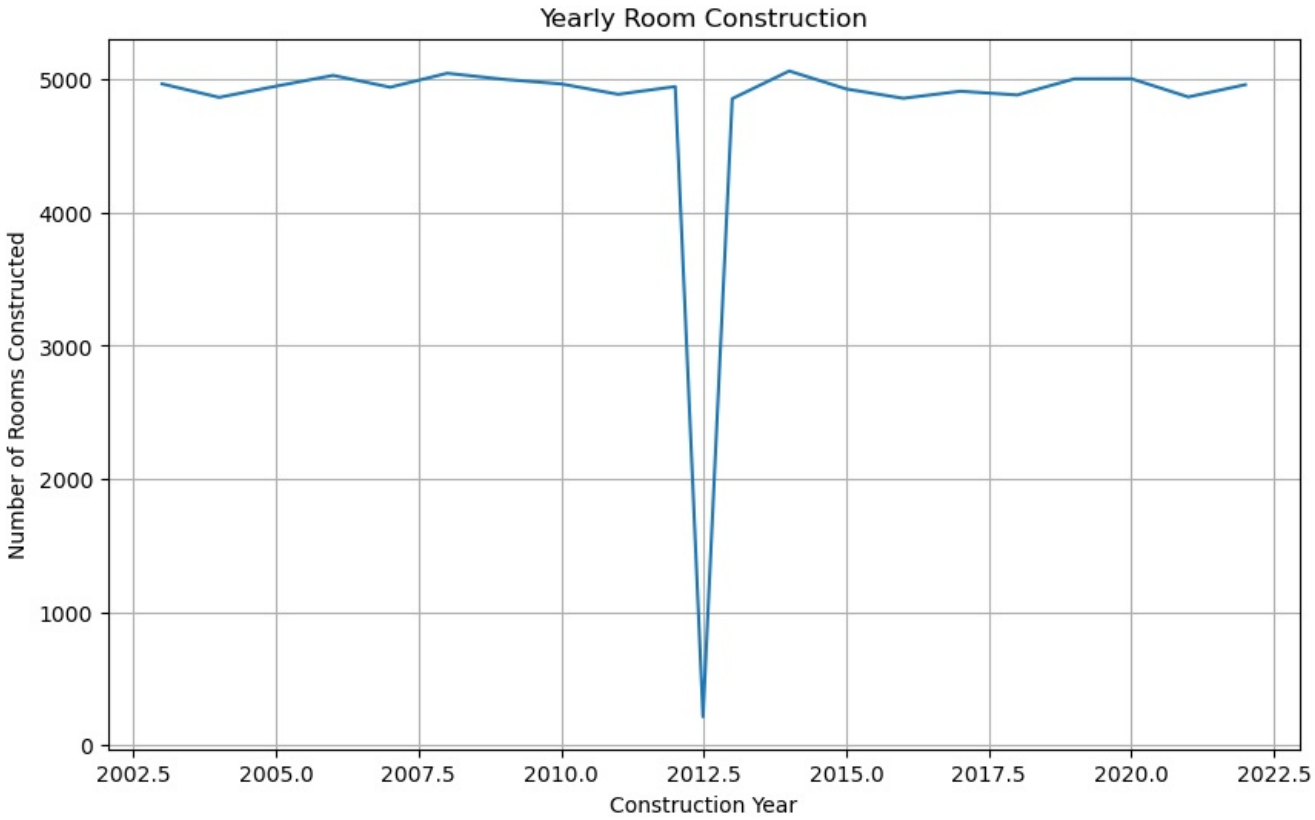
- Does service price and room price have an impact on each other. Illustrate this relationship with a scatter plot and state your inferences
- Using a line graph show in which year the maximum construction of rooms took place.

If using Python for this exercise, please include the code in the cells below. If using any other tool, please include screenshoots of your work.

In [65]:
```
#Does service price and room price have an impact on each other. Illustrate this relationship with a scatter pl
# Scatter plot
plt.figure(figsize=(10, 6))
sns.scatterplot(x='service_fee', y='price', data=df)

# Adding titles and labels
plt.title('Relationship between Service Price and Room Price')
plt.xlabel('Service Price')
plt.ylabel('Room Price')

# Show plot
plt.show()
print ('Inferences :', 'It is nevertheless evident that the two variables have a positive association. The pric
```

Inferences : It is nevertheless evident that the two variables have a positive association. The price of the ac commodation tends to rise in tandem with the service charge, meaning that rooms that are more expensive typical ly have larger service costs. This lends more credence to the theory that service costs represent a portion of the room charge, which is a standard pricing tactic in the hospitality sector

In [81]:
```
#Using a line graph show in which year the maximum construction of rooms took place.

# Aggregate data
yearly_data = df.groupby('construction_year')['room_type'].count()

# Plotting

plt.figure(figsize=(10, 6))
yearly_data.plot(kind='line')
plt.title('Yearly Room Construction')
plt.xlabel('Construction Year')
plt.ylabel('Number of Rooms Constructed')
plt.grid(True)
plt.show()

# Identifying the year with maximum construction
max_year = yearly_data.idxmax()
print(f"The year with maximum room construction is: {max_year}")
```



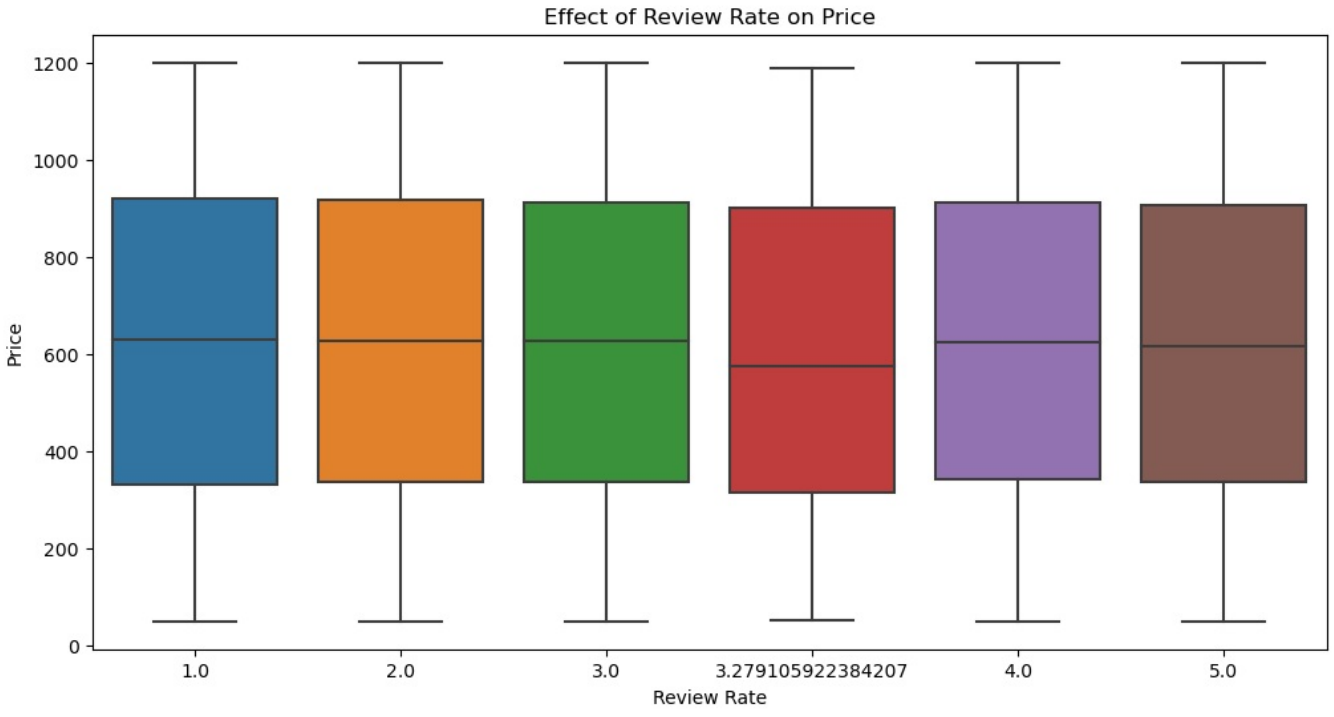

```
The year with maximum room construction is: 2014.0
```

Task 5c: Data Visualization (Any Tool)

- With the help of box plots illustrate the following
  - Effect of Review Rate number on price
  - Effect of host identity verified on price

If using Python for this exercise, please include the code in the cells below. If using any other tool, please include screenshoots of your work.

In [56]:
```
#Effect of Review Rate number on price
plt.figure(figsize=(12, 6))
sns.boxplot(x='review_rate_number', y='price', data=df)
plt.title('Effect of Review Rate on Price')
plt.xlabel('Review Rate')
plt.ylabel('Price')
plt.show()
```



In [57]:
```
#Effect of host identity verified on price
plt.figure(figsize=(12, 6))
```

```
sns.boxplot(x='host_identity_verified', y='price', data=df)
plt.title('Effect of Review Rate on Price')
plt.xlabel('Review Rate')
plt.ylabel('Price')
plt.show()
```