In [1]:
```
#Loading the required Libraries

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

In [2]:
```
#The two types of dataset which are required are:

#1] Listing Dataset
#2] Reviews Dataset
```

In [3]:
```
#Loading the Listings dataset
listings = pd.read_csv(r"C:\Users\sanji\Downloads\Listings.csv\Listings
```

```
C:\Users\sanji\AppData\Local\Temp\ipykernel_14928\3171851516.py:2: DtypeWa
rning: Columns (5,13) have mixed types. Specify dtype option on import or
set low_memory=False.
  listings = pd.read_csv(r"C:\Users\sanji\Downloads\Listings.csv\Listings.
csv",encoding = ' latin-1')
```

In [4]:
```
#Loading the Reviews dataset
reviews = pd.read_csv(r"C:\Users\sanji\Downloads\Reviews.csv\Reviews.cs
```

In [5]:
```
# Displaying the first and last rows of the dataset
```

In [6]:
```
listings.head(5)
```

Out[6]:

	listing_id	name	host_id	host_since	host_location	host_response_time	host_res
0	281420	Beautiful Flat in le Village Montmartre, Paris	1466919	2011-12-03	Paris, Ile-de-France, France	NaN	
1	3705183	39 mÃ Â² Paris (Sacre CÃ â ur)	10328771	2013-11-29	Paris, Ile-de-France, France	NaN	
2	4082273	Lovely apartment with Terrace, 60m2	19252768	2014-07-31	Paris, Ile-de-France, France	NaN	
3	4797344	Cosy studio (close to Eiffel tower)	10668311	2013-12-17	Paris, Ile-de-France, France	NaN	
4	4823489	Close to Eiffel Tower - Beautiful flat : 2 rooms	24837558	2014-12-14	Paris, Ile-de-France, France	NaN	

5 rows × 33 columns

In [7]:
```
reviews.head(5)
```

Out[7]:

	listing_id	review_id	date	reviewer_id
0	11798	330265172	2018-09-30	11863072
1	15383	330103585	2018-09-30	39147453
2	16455	329985788	2018-09-30	1125378
3	17919	330016899	2018-09-30	172717984
4	26827	329995638	2018-09-30	17542859

In [8]:
```
listings.tail(5)
```

Out[8]:

	listing_id	name	host_id	host_since	host_location	host_response_time	ho
279707	38338635	Appartement T2 neuf prÃ□Â¨s du tram T3a Porte ...	31161181	2015-04-13	Paris, Ile-de-France, France	NaN	
279708	38538692	Cozy Studio in Montmartre	10294858	2013-11-27	Paris, Ile-de-France, France	NaN	
279709	38683356	Nice and cosy mini-appartement in Paris	2238502	2012-04-27	Paris, Ile-de-France, France	NaN	
279710	39659000	Charming apartment near Rue Saint Maur / Oberk...	38633695	2015-07-16	Paris, Ile-de-France, France	NaN	
279711	40219504	Cosy apartment with view on Canal St Martin	6955618	2013-06-17	Paris, Ile-de-France, France	NaN	

5 rows × 33 columns

In [9]:
```
reviews.tail(5)
```

Out[9]:

	listing_id	review_id	date	reviewer_id
5373138	47779342	726766332	2021-01-25	283094516
5373139	47823964	727963021	2021-01-31	76411977
5373140	47896175	728548625	2021-02-02	71370946
5373141	47900451	727399287	2021-01-29	109011160
5373142	47998038	730320626	2021-02-11	276790978

In [10]:
```
# Check for missing values in listings

print(listings.isnull().sum())
```

```
listing_id                        0
name                            173
host_id                           0
host_since                      165
host_location                   840
host_response_time           128782
host_response_rate           128782
host_acceptance_rate         113087
host_is_superhost               165
host_total_listings_count       165
host_has_profile_pic            165
host_identity_verified          165
neighbourhood                     0
district                     242700
city                              0
latitude                          0
longitude                         0
property_type                     0
room_type                         0
accommodates                      0
bedrooms                      29435
amenities                         0
price                             0
minimum_nights                    0
maximum_nights                    0
review_scores_rating          91405
review_scores_accuracy        91713
review_scores_cleanliness     91665
review_scores_checkin         91771
review_scores_communication   91687
review_scores_location        91775
review_scores_value           91785
instant_bookable                  0
dtype: int64
```

In [11]:
```
# Checking for missing values in reviews

print(reviews.isnull().sum())
```

```
listing_id     0
review_id      0
date           0
reviewer_id    0
dtype: int64
```

In [12]:
```
# Handling missing values (example: filling or dropping)

listings.fillna({'price': listings['price'].median()}, inplace=True)
reviews.dropna(inplace=True)
```

In [13]:
```
# Converting columns to appropriate data types (example: price to numer

listings['price'] = listings['price'].replace('[\$,]', '', regex=True).
listings['price']
```

Out[13]:
```
0          53.0
1         120.0
2          89.0
3          58.0
4          60.0
          ...
279707    120.0
279708     60.0
279709     50.0
279710    105.0
279711     70.0
Name: price, Length: 279712, dtype: float64
```

In [14]:
```
#Desrcriptive statsitics
```

In [15]:
```
listings.describe()
```

Out[15]:

	listing_id	host_id	host_response_rate	host_acceptance_rate	host_total_listi
count	2.797120e+05	2.797120e+05	150930.000000	166625.000000	279!
mean	2.638196e+07	1.081658e+08	0.865939	0.827168	
std	1.442576e+07	1.108570e+08	0.283744	0.289202	:
min	2.577000e+03	1.822000e+03	0.000000	0.000000	
25%	1.384462e+07	1.720656e+07	0.900000	0.780000	
50%	2.767098e+07	5.826911e+07	1.000000	0.980000	
75%	3.978485e+07	1.832853e+08	1.000000	1.000000	
max	4.834353e+07	3.901874e+08	1.000000	1.000000	7:

In [16]:
```
reviews.describe()
```

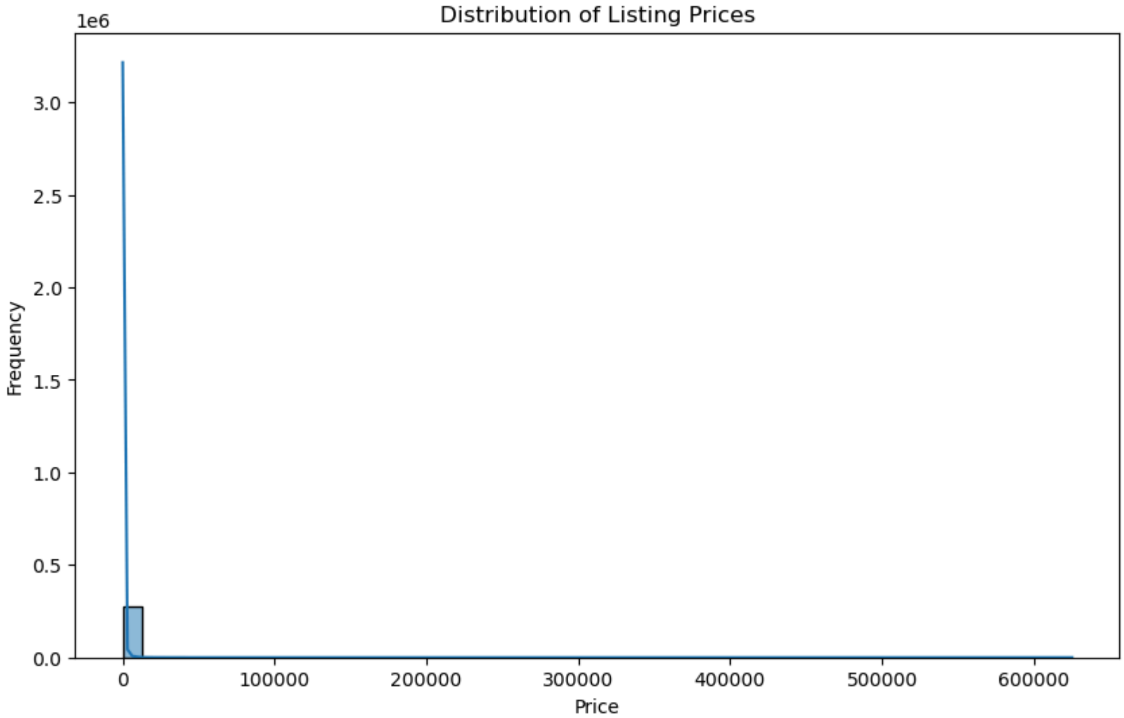
Out[16]:

	listing_id	review_id	reviewer_id
count	5.373143e+06	5.373143e+06	5.373143e+06
mean	1.602989e+07	3.486753e+08	9.808133e+07
std	1.198676e+07	2.061019e+08	9.080596e+07
min	2.577000e+03	2.820000e+02	1.000000e+00
25%	5.332708e+06	1.666435e+08	2.390206e+07
50%	1.450814e+07	3.425727e+08	6.697814e+07
75%	2.414496e+07	5.334045e+08	1.528936e+08
max	4.826387e+07	7.356237e+08	3.903385e+08

In [17]:
```
#Visualisation
```

Scenario 1: Distribution of listing prices

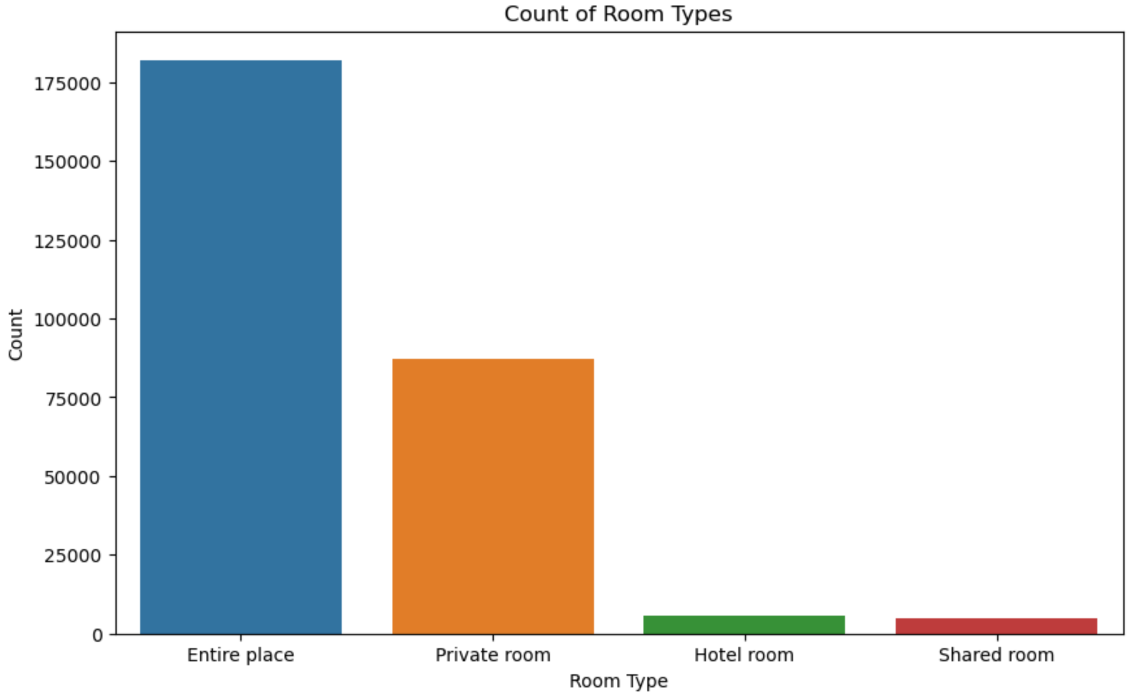
In [18]:
```python
plt.figure(figsize=(10, 6))
sns.histplot(listings['price'], bins=50, kde=True)
plt.title('Distribution of Listing Prices')
plt.xlabel('Price')
plt.ylabel('Frequency')
plt.show()

```



Scenario 2: Room type Analysis

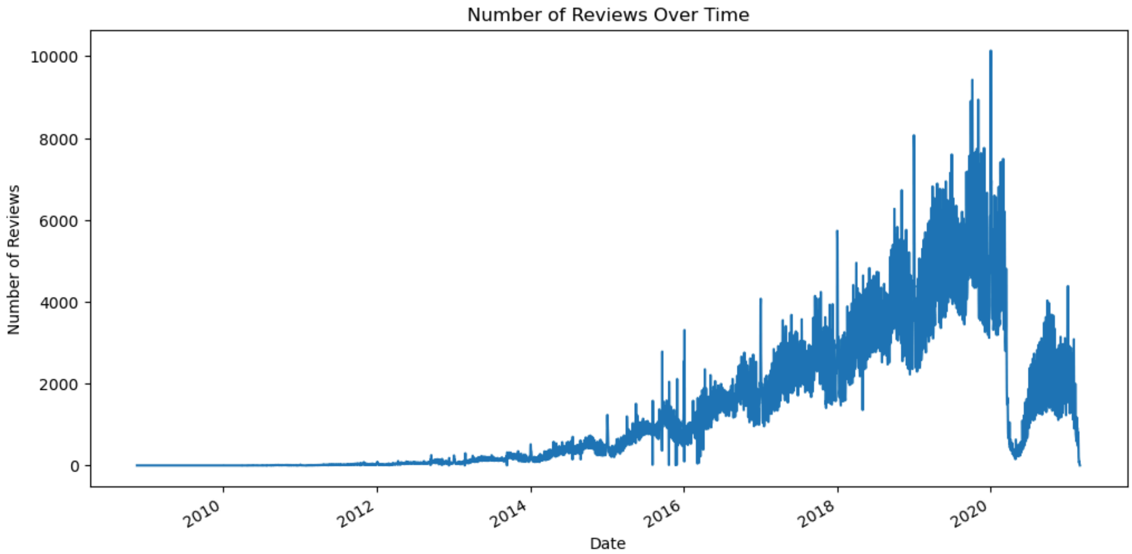
In [19]:

```
plt.figure(figsize=(10, 6))
sns.countplot(data=listings, x='room_type', order=listings['room_type']
plt.title('Count of Room Types')
plt.xlabel('Room Type')
plt.ylabel('Count')
plt.show()

```



Scenario 3 : How many number of reviews do we get over time?

In [20]:
```
reviews['date'] = pd.to_datetime(reviews['date'])
plt.figure(figsize=(12, 6))
reviews['date'].value_counts().sort_index().plot()
plt.title('Number of Reviews Over Time')
plt.xlabel('Date')
plt.ylabel('Number of Reviews')
plt.show()

```

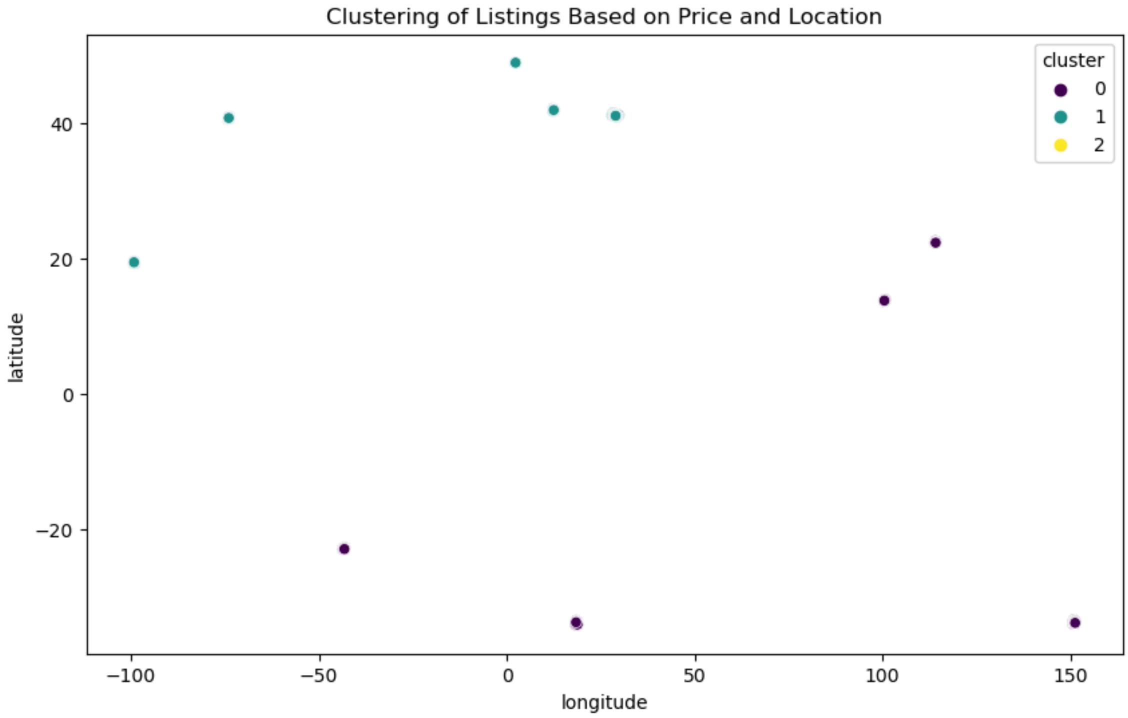


Scenario 4 : What is the clustering of listings based on prices and locations?

In [21]:

```
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler

features = listings[['price', 'latitude', 'longitude']]
scaler = StandardScaler()
scaled_features = scaler.fit_transform(features)

kmeans = KMeans(n_clusters=3)
kmeans.fit(scaled_features)

listings['cluster'] = kmeans.labels_

plt.figure(figsize=(10, 6))
sns.scatterplot(data=listings, x='longitude', y='latitude', hue='cluste
plt.title('Clustering of Listings Based on Price and Location')
plt.show()
```

```
C:\Users\sanji\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:141
2: FutureWarning: The default value of `n_init` will change from 10 to 'au
to' in 1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
```



Scenario 5: What are the most common types of Airbnb listings?

In [22]:
```python
property_counts = listings['property_type'].value_counts()
print(property_counts)
```

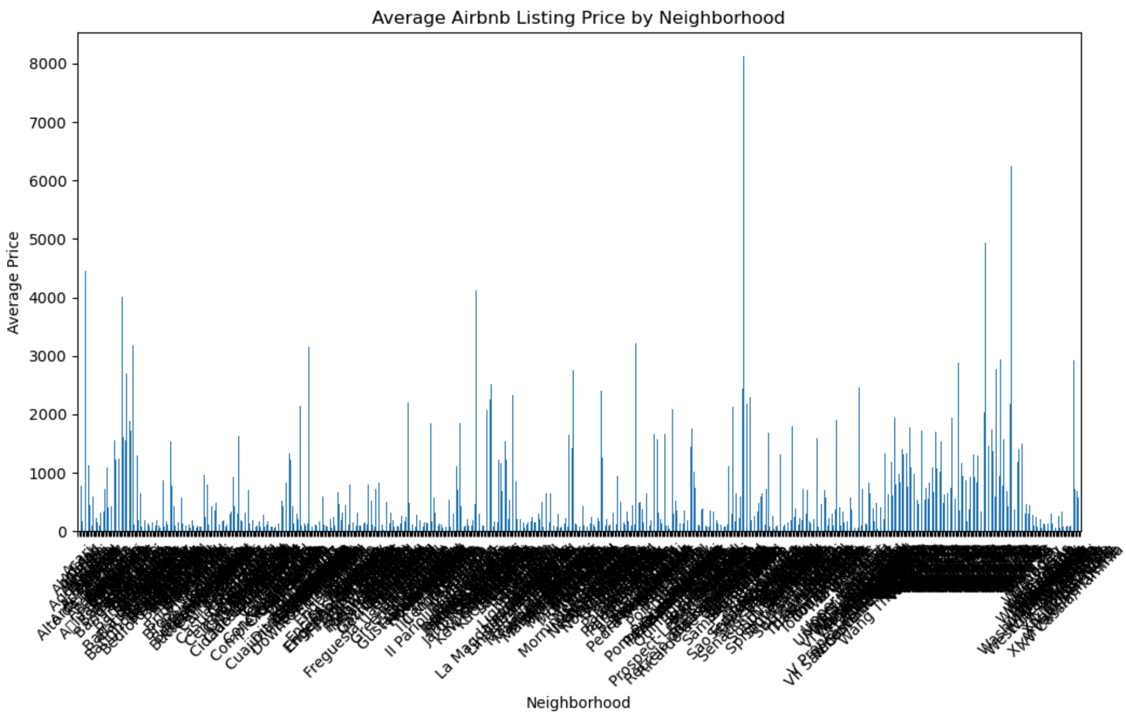
```
Entire apartment                138989
Private room in apartment        47322
Private room in house            13292
Entire house                     13273
Entire condominium               11250
                                 ...
Shared room in floor                 1
Shared room in parking space         1
Shared room in tent                  1
Train                                1
Tipi                                 1
Name: property_type, Length: 144, dtype: int64
```

Scenario 6: How do listing prices vary across different neighborhoods or regions?

In [23]:

```
# Grouping by neighborhood and calculating average price
neighborhood_prices = listings.groupby('neighbourhood')['price'].mean(

# Plotting
neighborhood_prices.plot(kind='bar', figsize=(12, 6))
plt.xlabel('Neighborhood')
plt.ylabel('Average Price')
plt.title('Average Airbnb Listing Price by Neighborhood')
plt.xticks(rotation=45)
plt.show()

```

```
C:\Users\sanji\anaconda3\Lib\site-packages\IPython\core\pylabtools.py:152:
UserWarning: Glyph 129 (\x81) missing from current font.
  fig.canvas.print_figure(bytes_io, **kw)
```

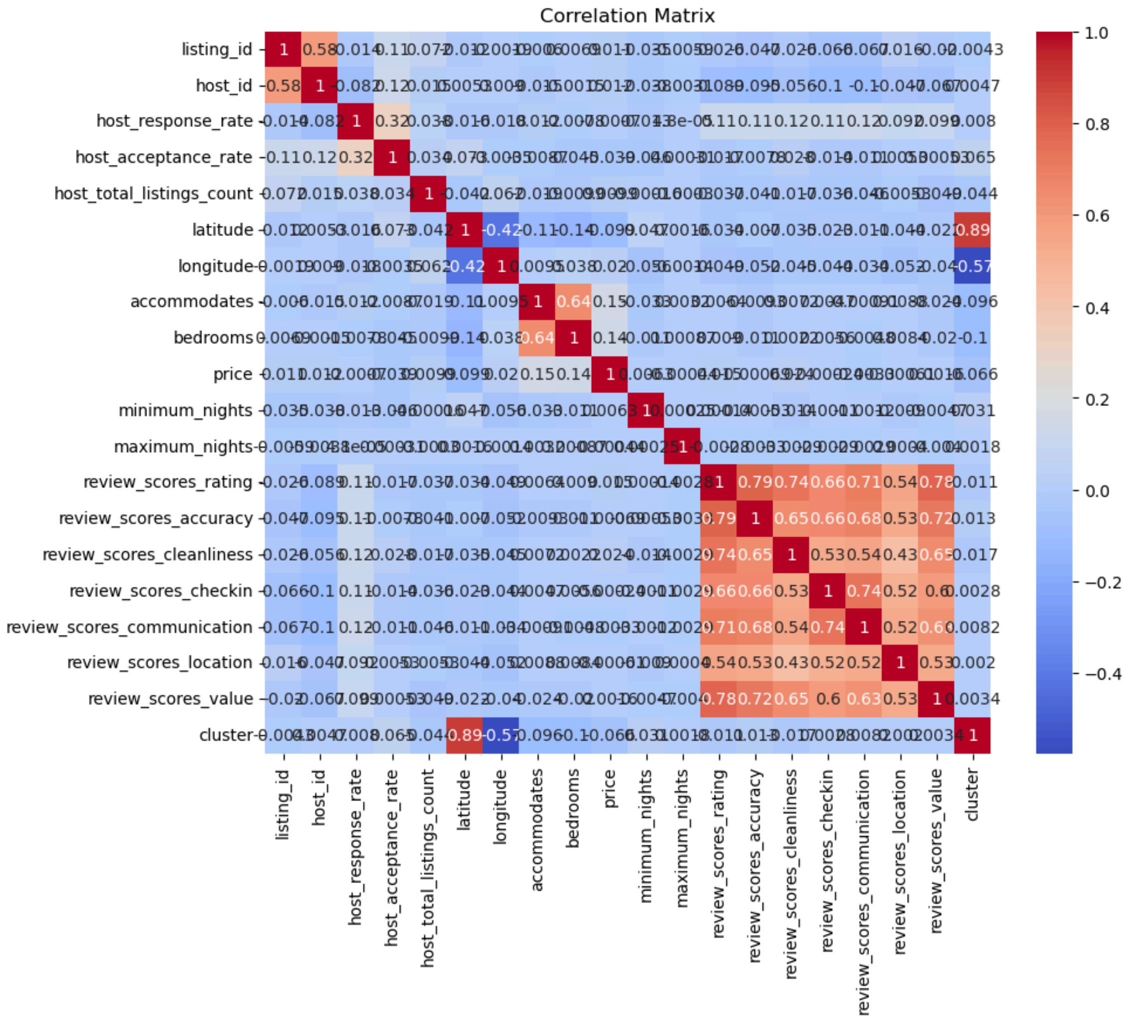


Scenario 7: What are the key factors influencing listing prices?

In [24]:

```
# Correlation matrix
correlation_matrix = listings.corr()

# Plotting heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show()

```

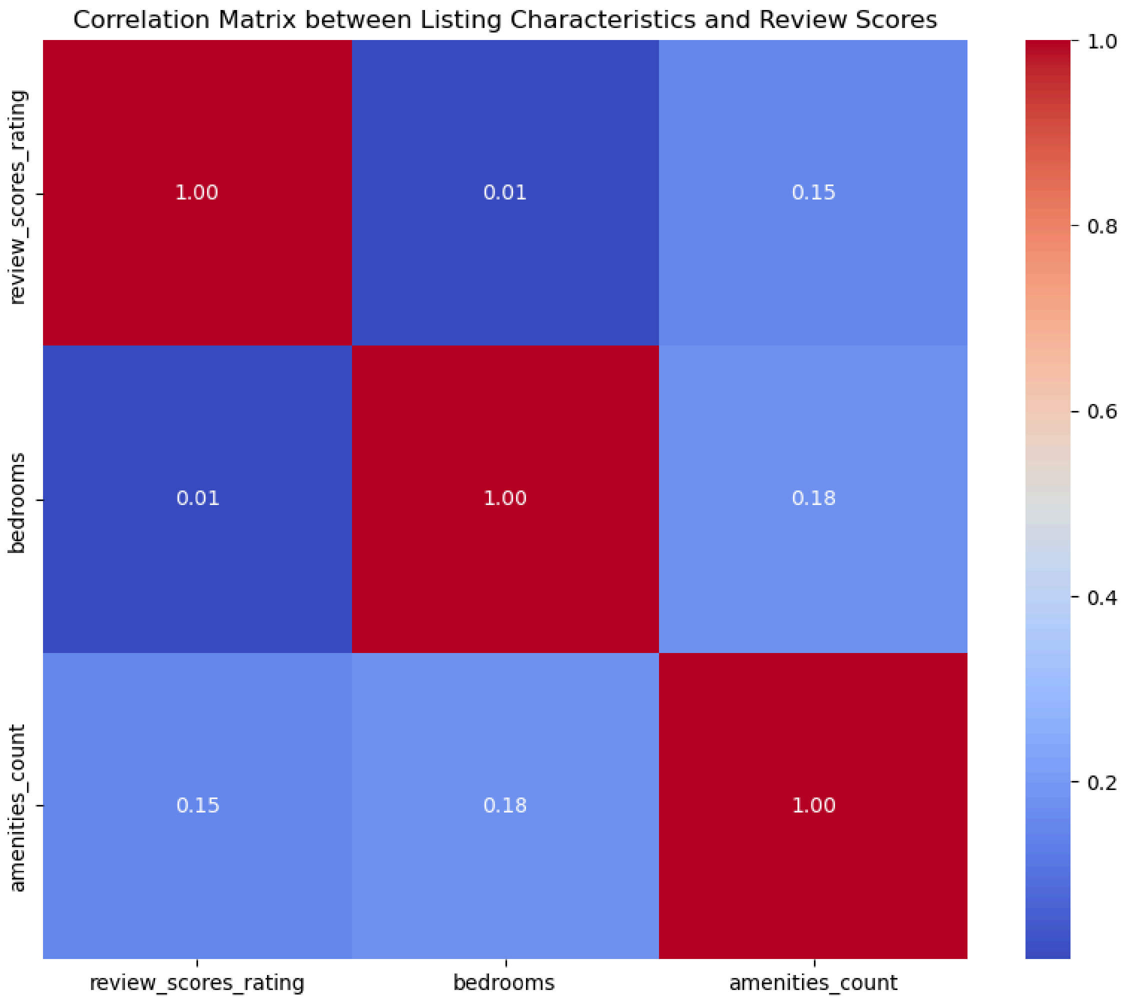
```
C:\Users\sanji\AppData\Local\Temp\ipykernel_14928\598132496.py:2: FutureWa
rning: The default value of numeric_only in DataFrame.corr is deprecated.
In a future version, it will default to False. Select only valid columns o
r specify the value of numeric_only to silence this warning.
  correlation_matrix = listings.corr()
```



Scenario 8:How do listing characteristics (e.g., number of bedrooms, amenities) correlate with review scores?

In [25]:

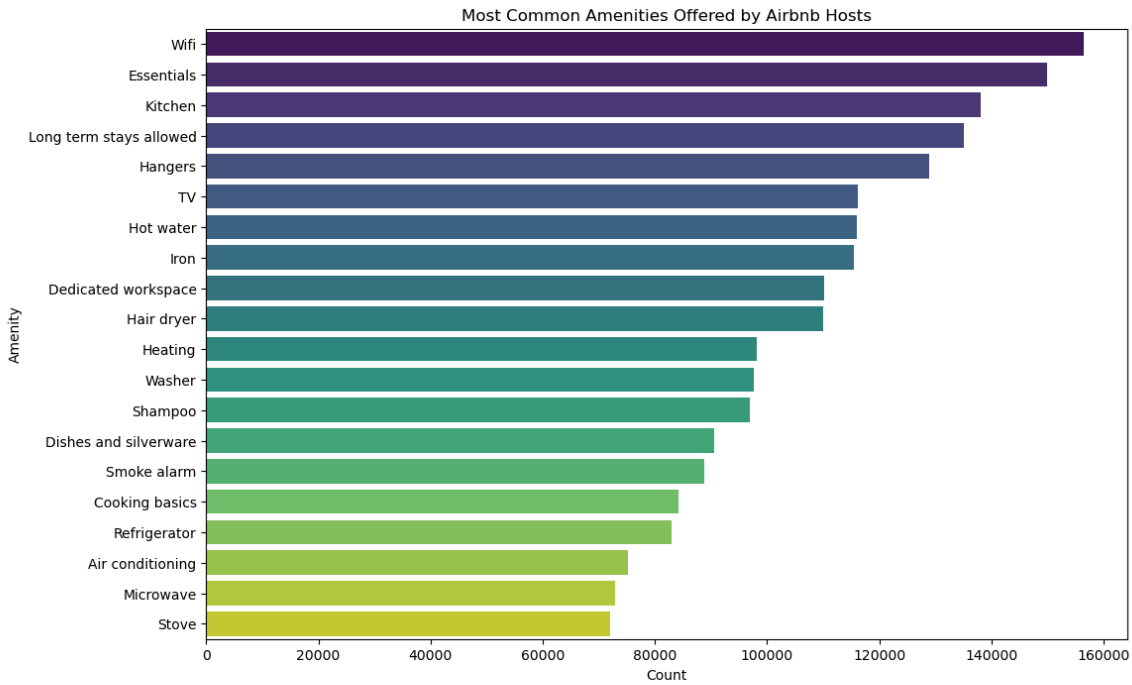
```
# Ensure relevant columns are present and without missing values
columns_of_interest = ['review_scores_rating', 'bedrooms', 'amenities']
listings = listings[columns_of_interest].dropna()

# Convert amenities to a numerical feature (e.g., count the number of a
listings['amenities'] = listings['amenities'].str.strip('{}').str.repla
listings['amenities_count'] = listings['amenities'].apply(lambda x: ler

# Select relevant numerical columns for correlation analysis
numerical_features = ['review_scores_rating', 'bedrooms', 'amenities_cc

# Calculate correlation matrix
correlation_matrix = listings[numerical_features].astype(float).corr()

# Plot heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Matrix between Listing Characteristics and Revie
plt.show()

```



Scenario 9:What are the most common amenities offered by Airbnb hosts?

In [26]:

```
# Assume the amenities column is a string with comma-separated values
# For example: "{Wifi, Kitchen, Heating}"
listings['amenities'] = listings['amenities'].str.strip('{}').str.repla

# Split amenities into a list
listings['amenities'] = listings['amenities'].apply(lambda x: x.split(

# Flatten the list of amenities and count the occurrences
amenities_list = [amenity.strip() for sublist in listings['amenities']
amenities_counter = Counter(amenities_list)

# Convert to DataFrame for easier plotting
amenities_df = pd.DataFrame(amenities_counter.items(), columns=['amenit
amenities_df = amenities_df.sort_values(by='count', ascending=False)

# Plot the most common amenities
plt.figure(figsize=(12, 8))
sns.barplot(data=amenities_df.head(20), x='count', y='amenity', palette
plt.xlabel('Count')
plt.ylabel('Amenity')
plt.title('Most Common Amenities Offered by Airbnb Hosts')
plt.show()

```



In [27]:

```

```

In [ ]:

```


```

In [ ]:

```

```

In [ ]:

```
1
```