EDA code step by step:

1. **Importing Libraries**:

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import geopandas as gpd
from shapely.geometry import Point
```

   - **pandas**: For data manipulation and analysis.
   - **seaborn**: For statistical data visualization.
   - **matplotlib**: For creating static, animated, and interactive visualizations.
   - **geopandas**: For working with geospatial data.
   - **shapely**: For geometric operations.

7. **Loading Data**:

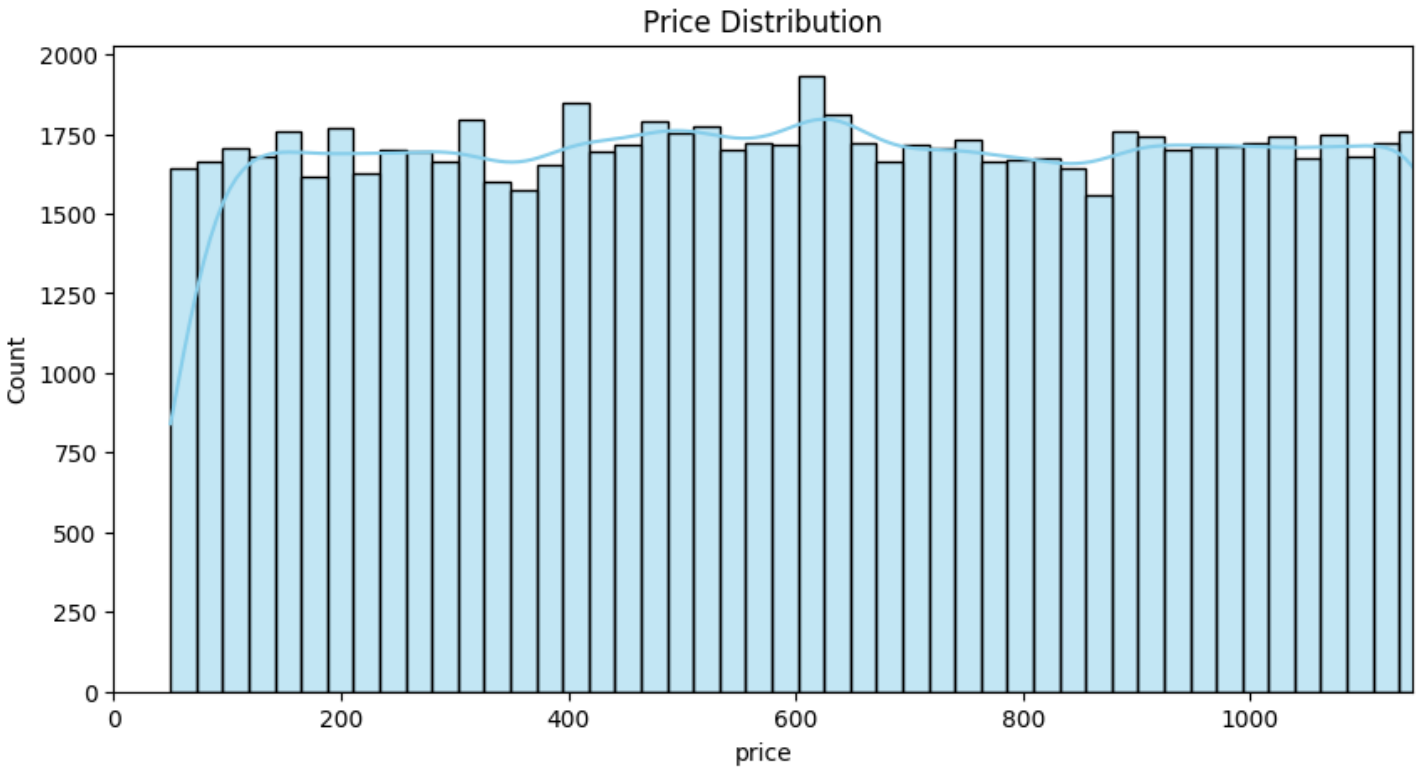
```
data = pd.read_csv("df_cleaneddd.csv")
```

   - Loads the dataset from a CSV file into a pandas DataFrame.

9. **Price Distribution**:

```
plt.figure(figsize=(10, 5))
sns.histplot(data['price'], bins=50, kde=True, color="skyblue")
plt.title('Price Distribution')
plt.xlim(0, data['price'].quantile(0.95))
plt.show()
```

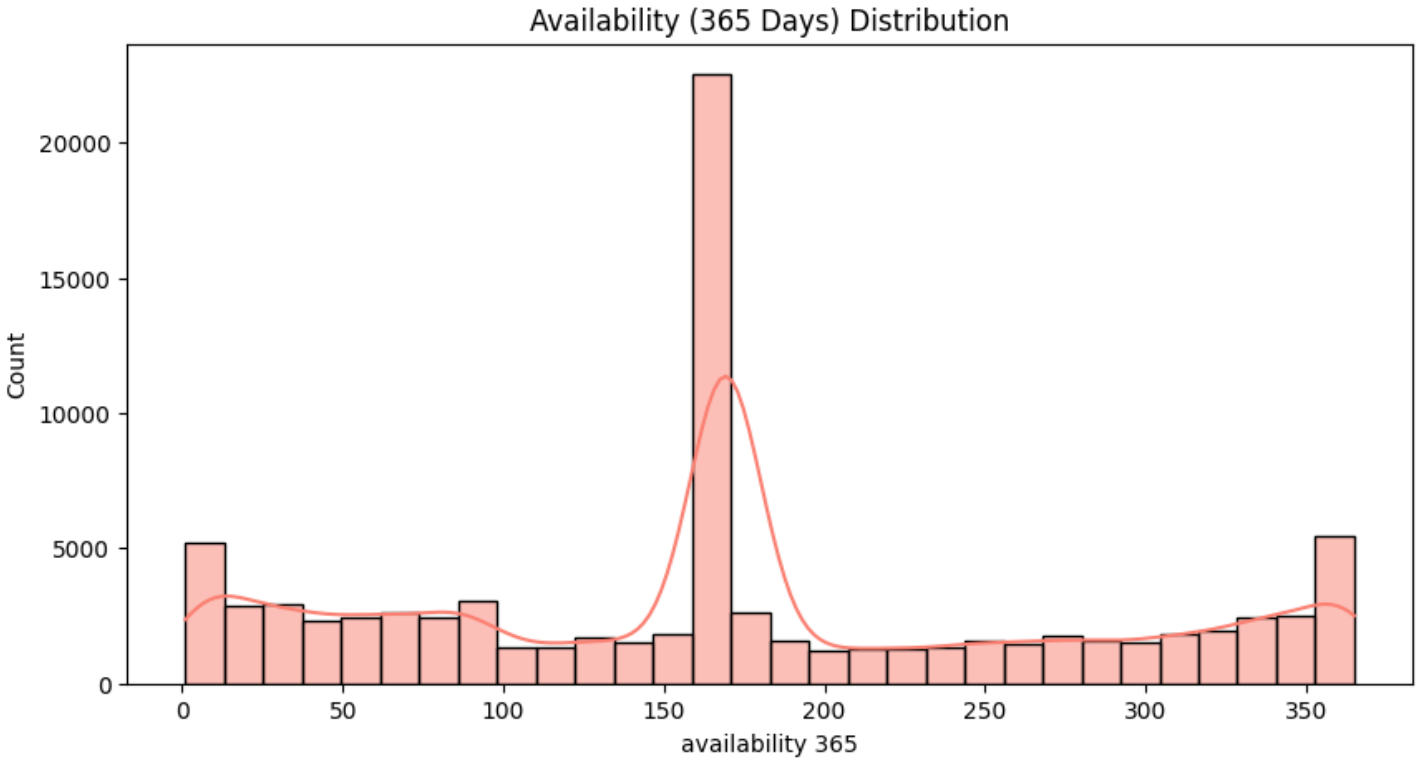
   - Creates a histogram of the `price` column with a kernel density estimate (KDE) overlay.
   - Limits the x-axis to the 95th percentile of the price for better visualization.



15. **Availability Distribution**:

```
plt.figure(figsize=(10, 5))
sns.histplot(data['availability 365'], bins=30, kde=True, color="salmon")
plt.title('Availability (365 Days) Distribution')
plt.show()
```

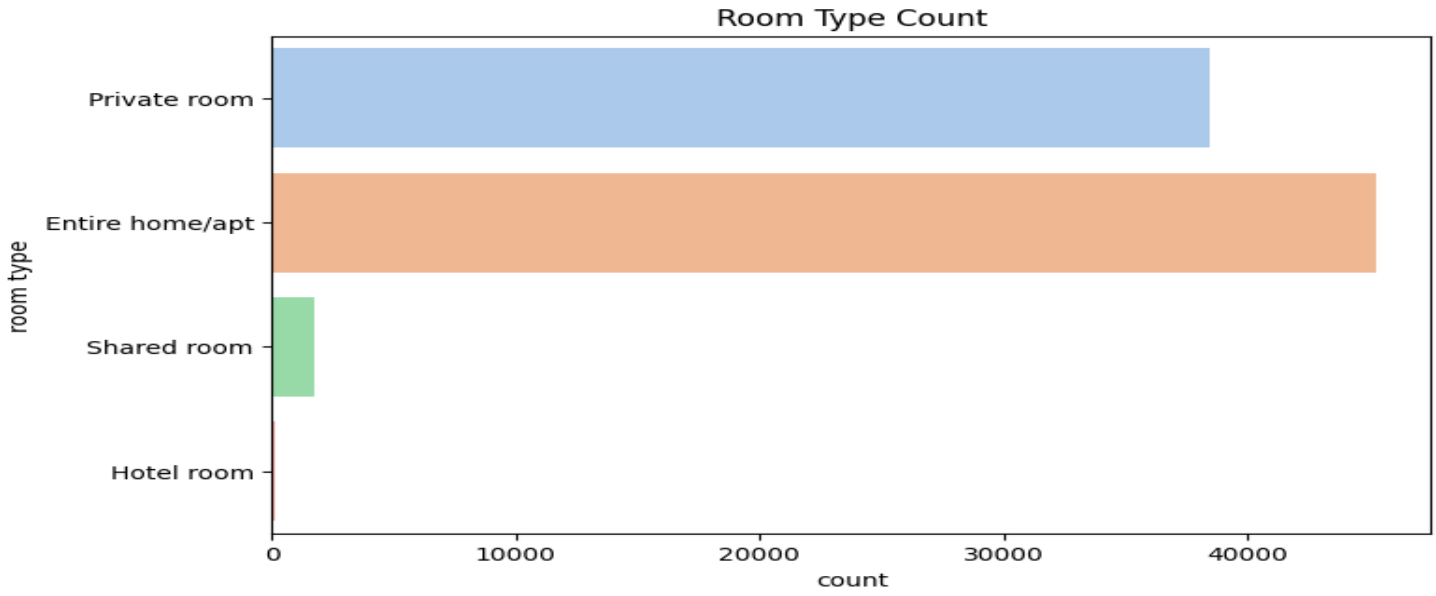
- Plots the distribution of the `availability 365` column.



20. **Room Type Count**:

```
plt.figure(figsize=(8, 5))
sns.countplot(y=data['room type'], palette="pastel")
plt.title('Room Type Count')
plt.show()
```

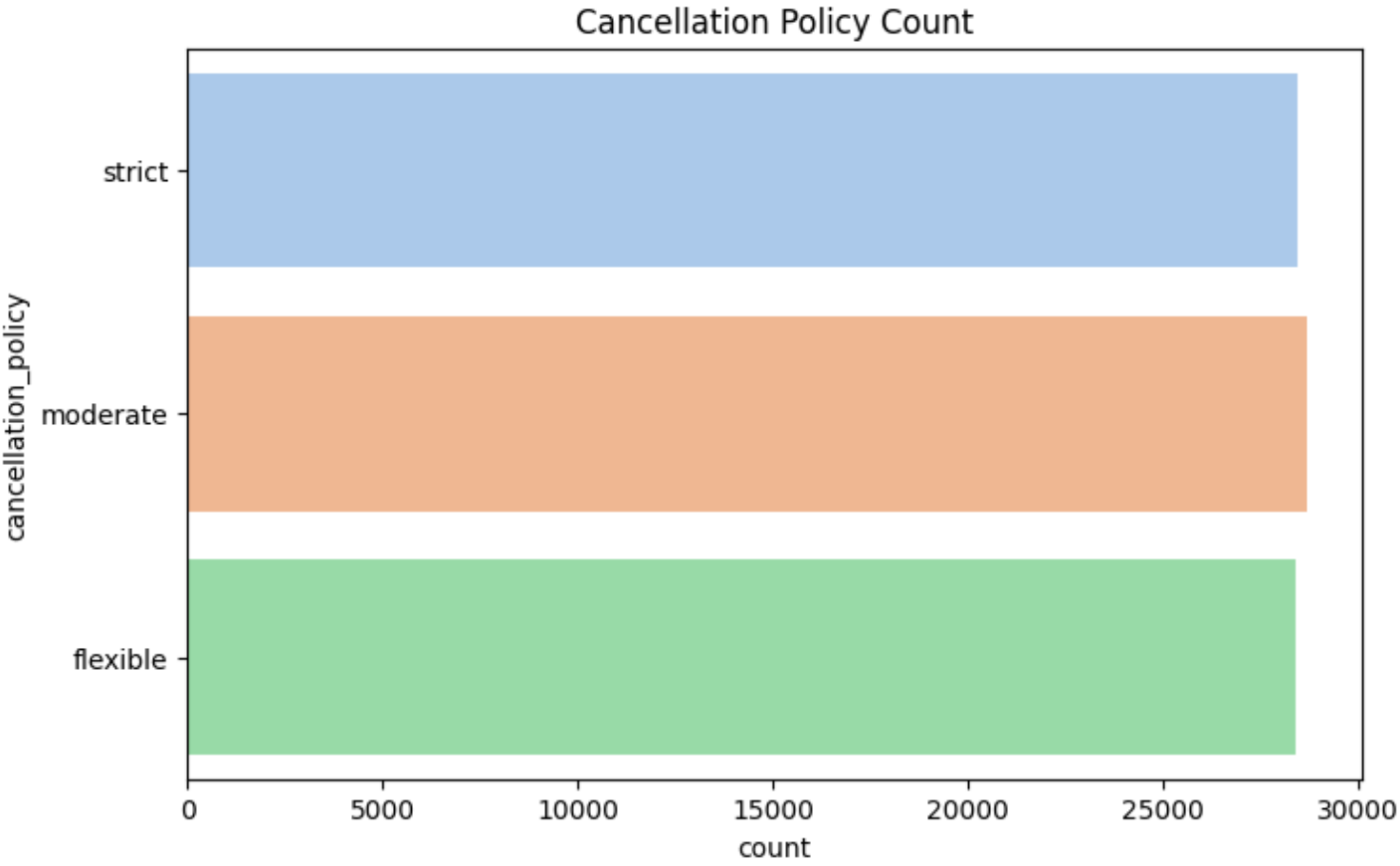
- Creates a count plot for the `room type` column.



25. **Cancellation Policy Count**:

```
plt.figure(figsize=(8, 5))
sns.countplot(y=data['cancellation_policy'], palette="pastel")
plt.title('Cancellation Policy Count')
plt.show()
```

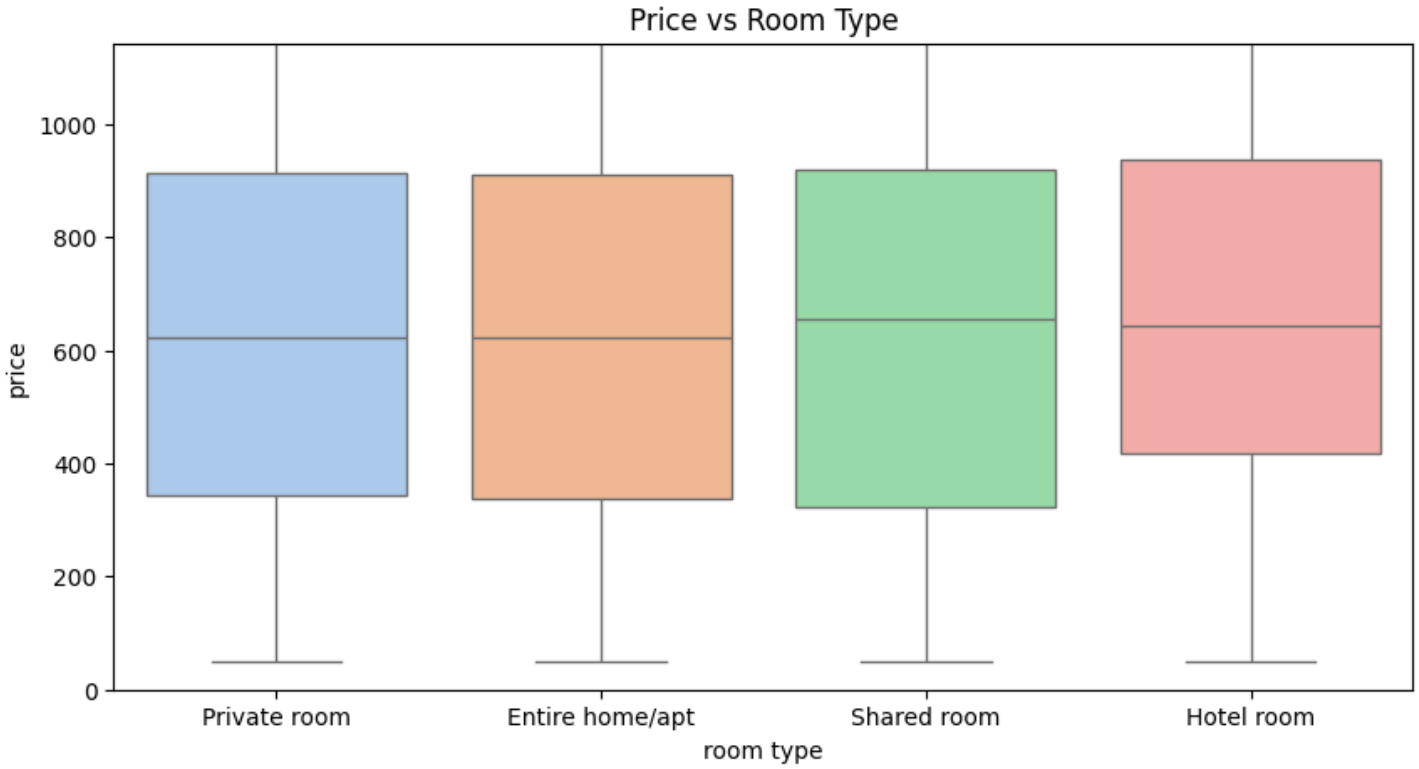
- Creates a count plot for the `cancellation_policy` column.



30. **Price vs Room Type**:

```
plt.figure(figsize=(10, 5))
sns.boxplot(x='room type', y='price', data=data, palette='pastel')
plt.title('Price vs Room Type')
plt.ylim(0, data['price'].quantile(0.95))
plt.show()
```

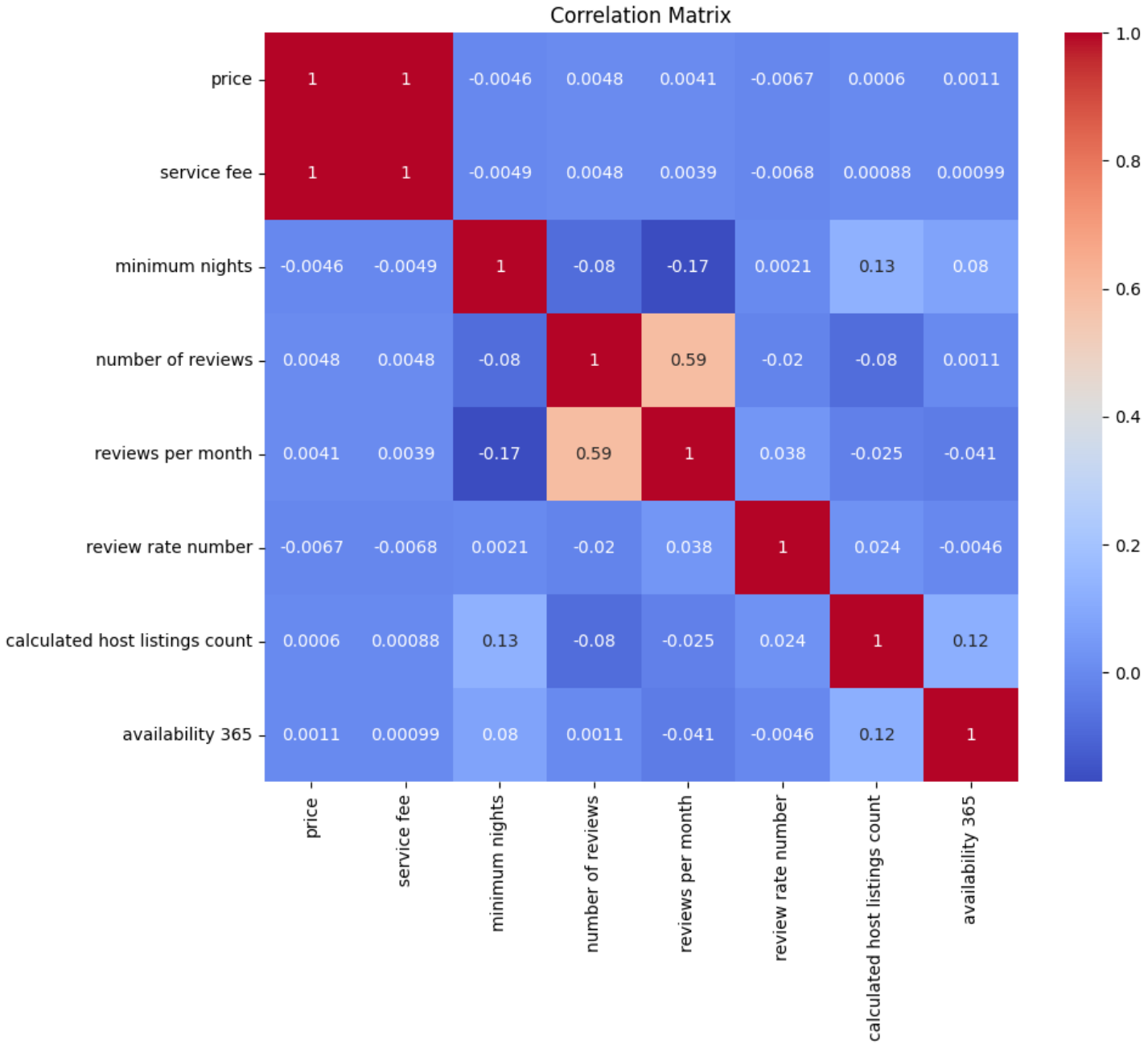
- Creates a box plot to show the distribution of prices across different room types.



36. **Correlation Matrix**:

```
37.  correlation = data[['price', 'service fee', 'minimum nights', 'number of reviews',
38.                      'reviews per month', 'review rate number', 'calculated host
   listings count',
39.                      'availability 365']].corr()
40.  plt.figure(figsize=(10, 8))
41.  sns.heatmap(correlation, annot=True, cmap='coolwarm')
42.  plt.title('Correlation Matrix')
43.  plt.show()
```

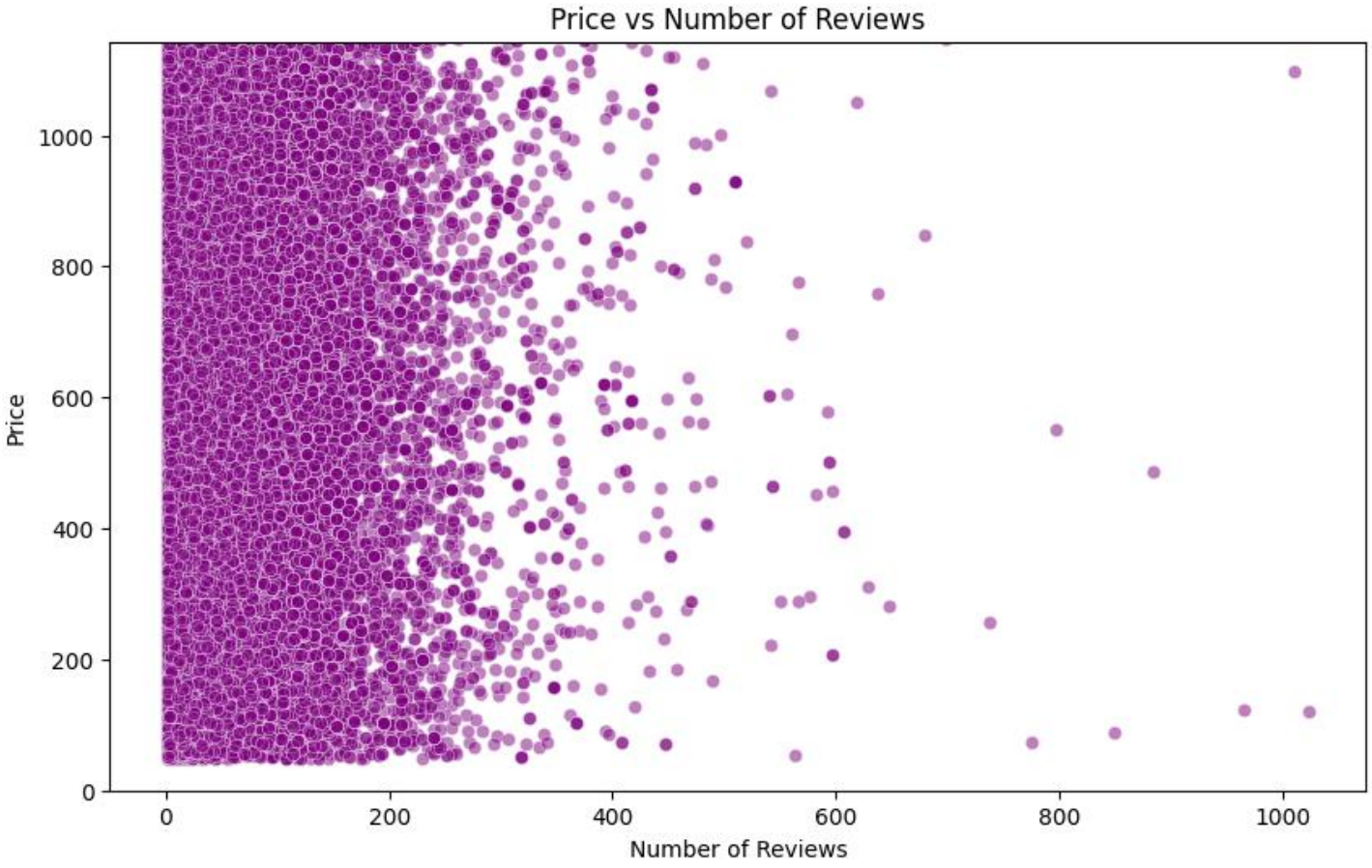
- Computes and visualizes the correlation matrix for selected numerical columns.



44. **Scatter Plot: Price vs Number of Reviews**:

```
plt.figure(figsize=(10, 6))
sns.scatterplot(x='number of reviews', y='price', data=data, color="purple", alpha=0.5)
plt.title('Price vs Number of Reviews')
plt.xlabel('Number of Reviews')
plt.ylabel('Price')
plt.ylim(0, data['price'].quantile(0.95))
plt.show()
```

- Creates a scatter plot to show the relationship between the number of reviews and price.



52. **Convert to GeoDataFrame**:

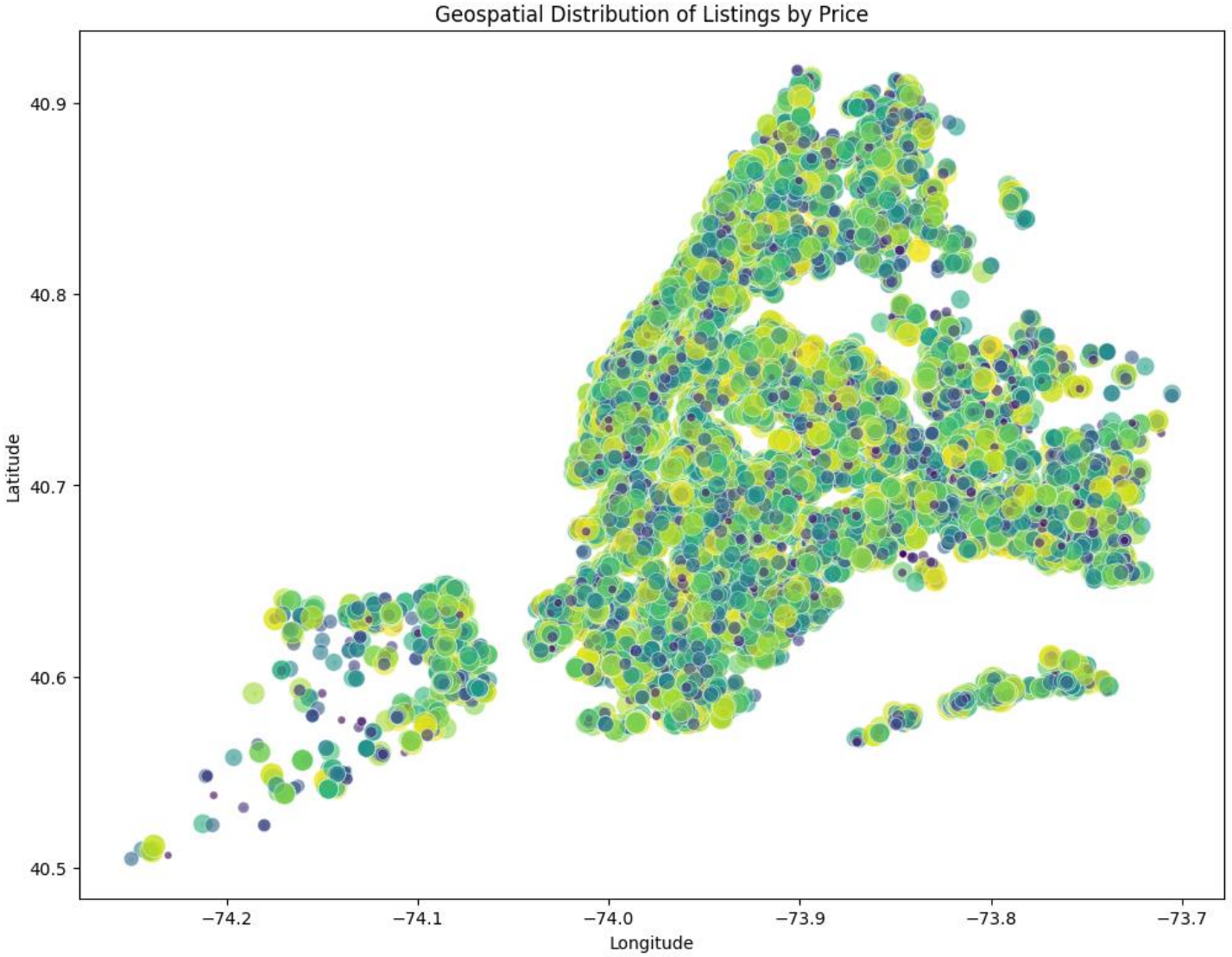
```
53.  gdf = gpd.GeoDataFrame(data, geometry=gpd.points_from_xy(data['long'],
    data['lat']))
```

- Converts the DataFrame to a GeoDataFrame using longitude and latitude for geospatial analysis.

54. **Plot Locations by Price Range**:

```
55.  fig, ax = plt.subplots(figsize=(12, 10))
56.  gdf.plot(ax=ax, markersize=5, color='lightgrey', alpha=0.5)
57.  sns.scatterplot(x='long', y='lat', hue='price', data=data, palette='viridis',
58.                  size='price', sizes=(20, 200), alpha=0.6, legend=False, ax=ax)
59.  plt.title("Geospatial Distribution of Listings by Price")
60.  plt.xlabel("Longitude")
61.  plt.ylabel("Latitude")
62.  plt.show()
```

- Plots the geospatial distribution of listings, with color and size representing the price.



63. **Monthly Average Reviews per Month**:

```
64.  data['last review'] = pd.to_datetime(data['last review'], errors='coerce')
65.  data.set_index('last review').resample('ME')['reviews per
   month'].mean().plot(figsize=(12, 6))
66.  plt.title("Monthly Average Reviews per Month")
67.  plt.xlabel("Date")
68.  plt.ylabel("Average Reviews per Month")
69.  plt.show()
```

- Converts the `last review` column to datetime and plots the monthly average of reviews per month.