```
import pandas as pd

# Load the data dictionary and the Airbnb dataset
data_dict = pd.read_excel('/content/Data-Dictionary.xlsx')
airbnb_data = pd.read_csv('/content/AirBNB.csv')

# Check the first few rows of the data to understand its structure
airbnb_data.head()
```

```
<ipython-input-63-569b13bcc151>:5: DtypeWarning: Columns (5) have mixed types. Spe
  airbnb_data = pd.read_csv('/content/AirBNB.csv')
```

	id	room_type	accommodates	bathrooms	cancellation_policy	cleaning_fee
0	6901257	Entire home/apt	3.0	1.0	strict	True
1	6304928	Entire home/apt	7.0	1.0	strict	True
2	7919400	Entire home/apt	5.0	1.0	moderate	True

```
data_dict.head()
```

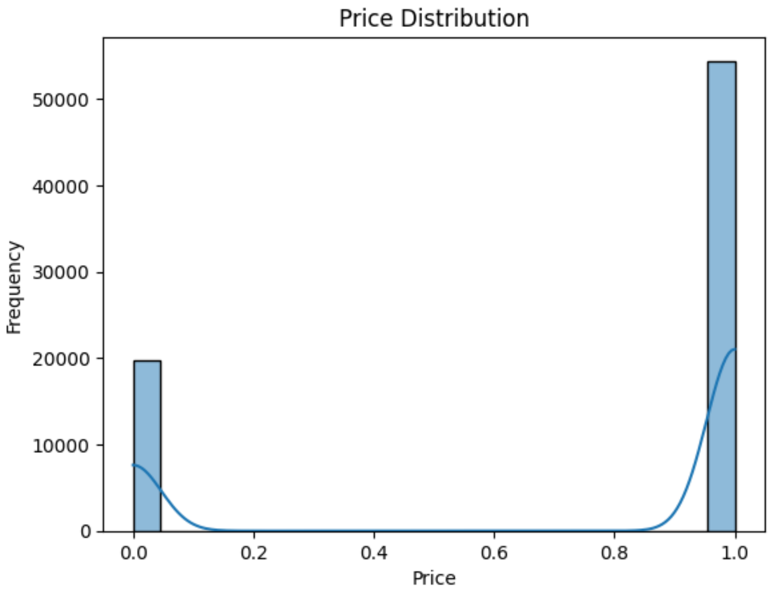
	id	Property ID
0	room_type	Type of Room in the property
1	accommodates	How many adults can this property accomodates
2	bathrooms	Number of bathrooms in the property
3	cancellation_policy	Cancellation policy of the property
4	cleaning_fee	This denotes whether propoerty cleaning fee is...

```
# Checking for missing values
airbnb_data.isnull().sum()

# Handling missing values for numeric columns only
for column in airbnb_data.select_dtypes(include=['number']).columns:
  airbnb_data[column].fillna(airbnb_data[column].mean(), inplace=True)
# Handling missing values for non-numeric columns using the mode
for column in airbnb_data.select_dtypes(exclude=['number']).columns:
  airbnb_data[column].fillna(airbnb_data[column].mode()[0], inplace=True)
# Convert categorical features using one-hot encoding (example)
airbnb_data = pd.get_dummies(airbnb_data, drop_first=True)
```

```
<ipython-input-68-64450501b2e8>:6: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform the operation
```

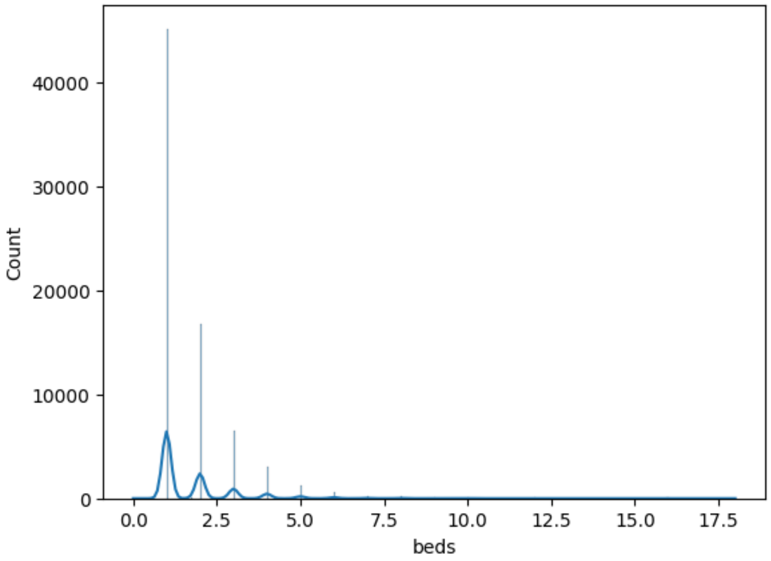
```
  airbnb_data[column].fillna(airbnb_data[column].mean(), inplace=True)
<ipython-input-68-64450501b2e8>:9: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform the operation


  airbnb_data[column].fillna(airbnb_data[column].mode()[0], inplace=True)
```

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# ... (your previous code for loading and cleaning data) ...

# Convert categorical features using one-hot encoding, EXCLUDING 'price'
categorical_cols = [col for col in airbnb_data.select_dtypes(include=['object']).columns if col != 'price']  # Exclude 'price' from categorical columns
airbnb_data = pd.get_dummies(airbnb_data, columns=categorical_cols, drop_first=True)

# Visualizing price distribution
sns.histplot(airbnb_data['cleaning_fee'], kde=True)
plt.title("Price Distribution")
plt.xlabel("Price")
plt.ylabel("Frequency")
plt.show()
```



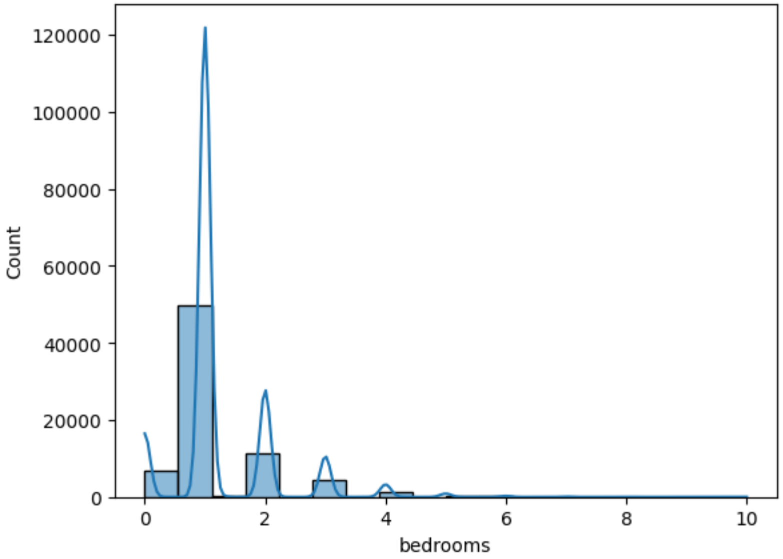
```
sns.histplot(airbnb_data['beds'], kde=True)
```

```
<Axes: xlabel='beds', ylabel='Count'>
```



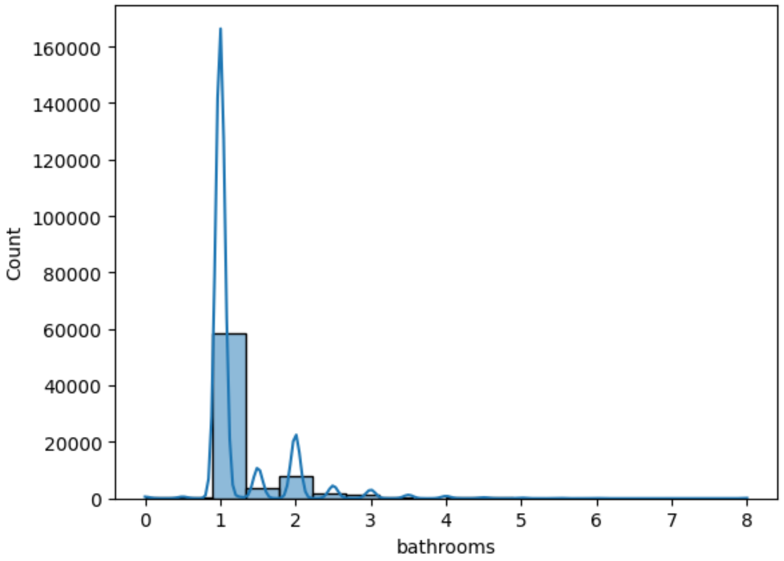
```
sns.histplot(airbnb_data['bedrooms'], kde=True)
```

```
<Axes: xlabel='bedrooms', ylabel='Count'>
```

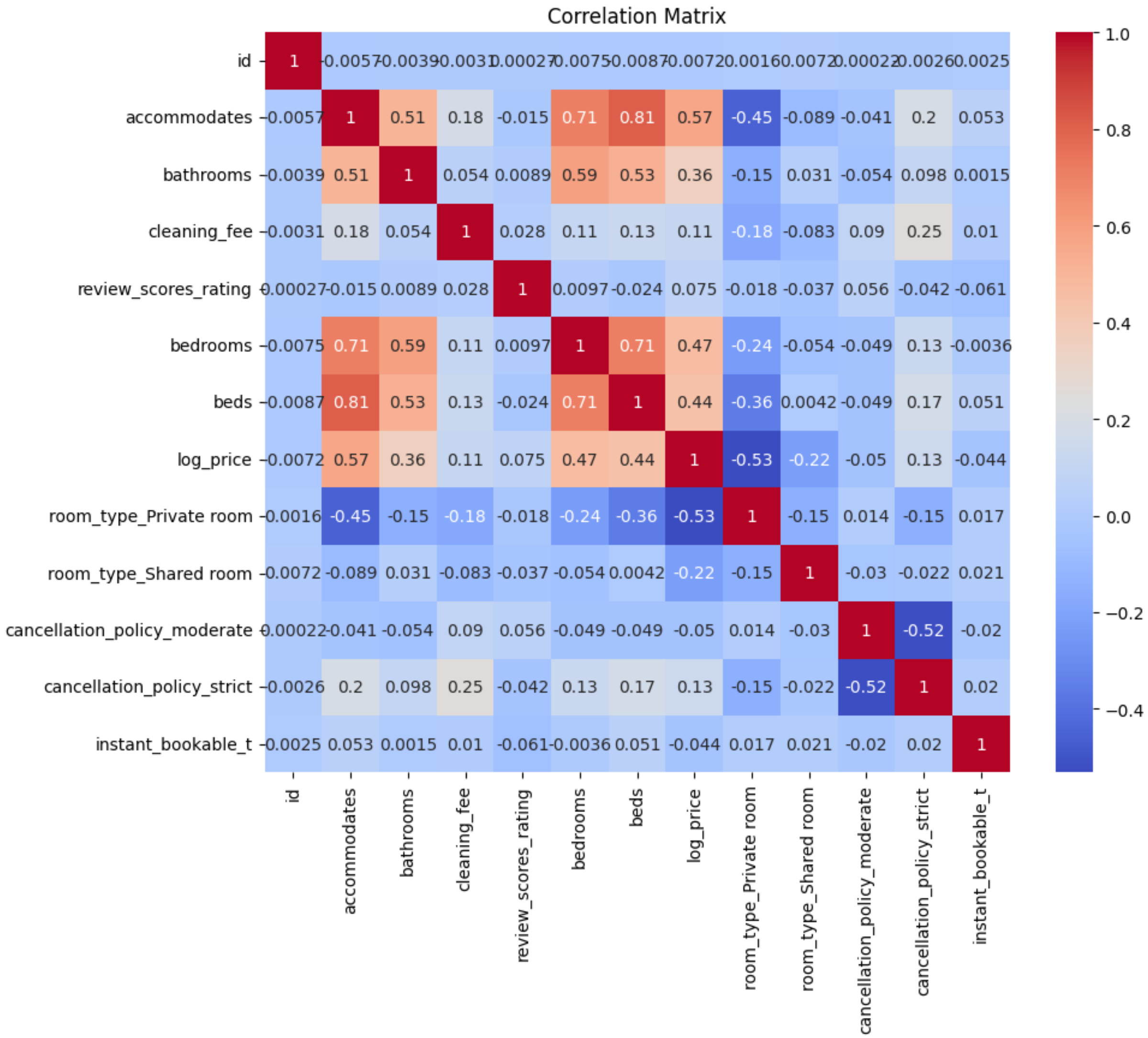


```
sns.histplot(airbnb_data['bathrooms'], kde=True)
```

```
<Axes: xlabel='bathrooms', ylabel='Count'>
```



```
# Check correlation matrix
correlation_matrix = airbnb_data.corr()
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap="coolwarm")
plt.title("Correlation Matrix")
plt.show()
```



Correlation Matrix

	id	accommodates	bathrooms	cleaning_fee	review_scores_rating	bedrooms	beds	log_price	room_type_Private room	room_type_Shared room	cancellation_policy_moderate	cancellation_policy_strict	instant_bookable_t
id	1	-0.0057	-0.0039	-0.0031	0.00027	-0.0075	-0.0087	-0.0072	-0.0016	-0.0072	0.00022	-0.0026	-0.0025
accommodates	-0.0057	1	0.51	0.18	-0.015	0.71	0.81	0.57	-0.45	-0.089	-0.041	0.2	0.053
bathrooms	-0.0039	0.51	1	0.054	0.0089	0.59	0.53	0.36	-0.15	0.031	-0.054	0.098	0.0015
cleaning_fee	-0.0031	0.18	0.054	1	0.028	0.11	0.13	0.11	-0.18	-0.083	0.09	0.25	0.01
review_scores_rating	0.00027	-0.015	0.0089	0.028	1	0.0097	-0.024	0.075	-0.018	-0.037	0.056	-0.042	-0.061
bedrooms	-0.0075	0.71	0.59	0.11	0.0097	1	0.71	0.47	-0.24	-0.054	-0.049	0.13	-0.0036
beds	-0.0087	0.81	0.53	0.13	-0.024	0.71	1	0.44	-0.36	0.0042	-0.049	0.17	0.051
log_price	-0.0072	0.57	0.36	0.11	0.075	0.47	0.44	1	-0.53	-0.22	-0.05	0.13	-0.044
room_type_Private room	-0.0016	-0.45	-0.15	-0.18	-0.018	-0.24	-0.36	-0.53	1	-0.15	0.014	-0.15	0.017
room_type_Shared room	-0.0072	-0.089	0.031	-0.083	-0.037	-0.054	0.0042	-0.22	-0.15	1	-0.03	-0.022	0.021
cancellation_policy_moderate	0.00022	-0.041	-0.054	0.09	0.056	-0.049	-0.049	-0.05	0.014	-0.03	1	-0.52	-0.02
cancellation_policy_strict	-0.0026	0.2	0.098	0.25	-0.042	0.13	0.17	0.13	-0.15	-0.022	-0.52	1	0.02
instant_bookable_t	-0.0025	0.053	0.0015	0.01	-0.061	-0.0036	0.051	-0.044	0.017	0.021	-0.02	0.02	1

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

# Choose a single feature (e.g., 'bedrooms') for simple linear regression
X_simple = airbnb_data[['bedrooms']]  # Use 'bedrooms' as an example feature
y = airbnb_data['log_price']

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X_simple, y, test_size=0.2, random_state=42)
```

```
# Build the model
simple_model = LinearRegression()
simple_model.fit(X_train, y_train)

# Predictions and evaluation
y_pred_simple = simple_model.predict(X_test)
print("Simple Linear Regression - Mean Squared Error:", mean_squared_error(y_test, y_pred_simple))
print("Simple Linear Regression - R^2 Score:", r2_score(y_test, y_pred_simple))
```

```
Simple Linear Regression - Mean Squared Error: 0.40321575404146726
Simple Linear Regression - R^2 Score: 0.21511499582843563
```

```
# Use multiple features for multiple linear regression
X_multiple = airbnb_data.drop(columns=['bedrooms'])  # Use all features except 'price'

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X_multiple, y, test_size=0.2, random_state=42)

# Build the model
multiple_model = LinearRegression()
multiple_model.fit(X_train, y_train)

# Predictions and evaluation
y_pred_multiple = multiple_model.predict(X_test)
print("Multiple Linear Regression - Mean Squared Error:", mean_squared_error(y_test, y_pred_multiple))
print("Multiple Linear Regression - R^2 Score:", r2_score(y_test, y_pred_multiple))
```

```
Multiple Linear Regression - Mean Squared Error: 4.160917971391156e-30
Multiple Linear Regression - R^2 Score: 1.0
```