In [1]:
```python
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import statsmodels.api as sm
import scipy.stats as stats

%matplotlib inline
```

In [14]:
```python
data = pd.read_csv('kc_house_data.csv')
data.head()
```

Out[14]:

	id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	...	grade	sqft_above	sqft_basement	yr_built	yr_renovated	zipcode	la
0	7129300520	10/13/2014	221900.0	3	1.00	1180	5650	1.0	NaN	NONE	...	7 Average	1180	0.0	1955	0.0	98178	47.511
1	6414100192	12/9/2014	538000.0	3	2.25	2570	7242	2.0	NO	NONE	...	7 Average	2170	400.0	1951	1991.0	98125	47.721
2	5631500400	2/25/2015	180000.0	2	1.00	770	10000	1.0	NO	NONE	...	6 Low Average	770	0.0	1933	NaN	98028	47.737
3	2487200875	12/9/2014	604000.0	4	3.00	1960	5000	1.0	NO	NONE	...	7 Average	1050	910.0	1965	0.0	98136	47.520
4	1954400510	2/18/2015	510000.0	3	2.00	1680	8080	1.0	NO	NONE	...	8 Good	1680	0.0	1987	0.0	98074	47.616

5 rows × 21 columns

outlier detection

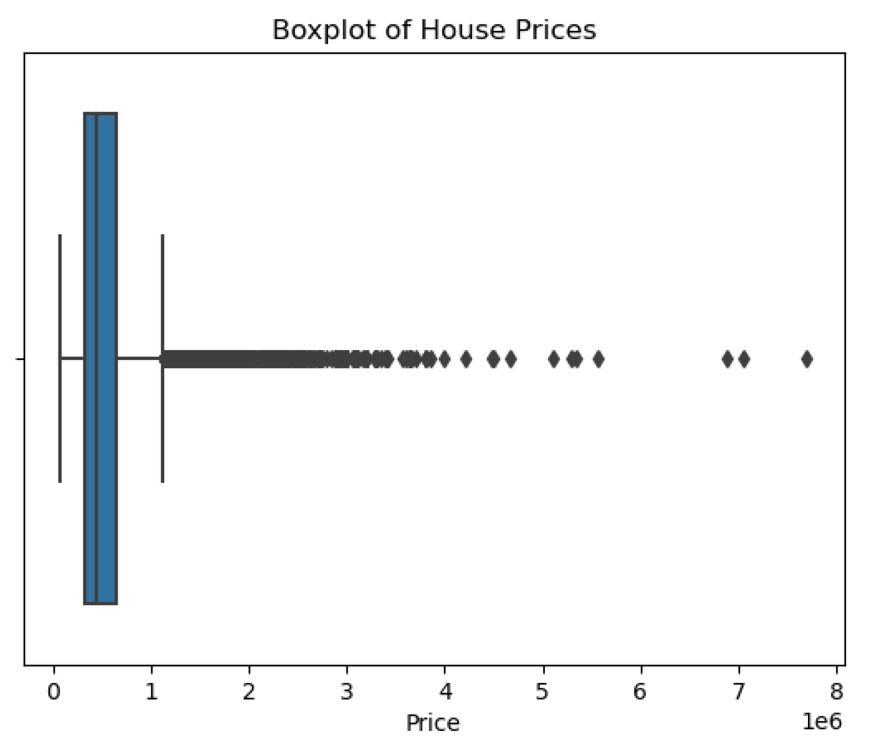
In [15]:
```python
Q1 = data['price'].quantile(0.25)
Q3 = data['price'].quantile(0.75)
IQR = Q3 - Q1

# Define bounds for the outliers
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

# Identify outliers
outliers = data[(data['price'] < lower_bound) | (data['price'] > upper_bound)]

# Plotting the outliers
sns.boxplot(x=data['price'])
plt.title('Boxplot of House Prices')
plt.xlabel('Price')
plt.show()

# Display the number of outliers
print('Number of outliers in the price column:', outliers.shape[0])
print('Lower bound for outliers:', lower_bound)
print('Upper bound for outliers:', upper_bound)
```




```
Number of outliers in the price column: 1158
Lower bound for outliers: -162500.0
Upper bound for outliers: 1129500.0
```

categorical variables.

In [6]:
```python
# Encoding categorical variables using one-hot encoding
encoded_data = pd.get_dummies(data, columns=['waterfront', 'view', 'condition', 'grade'])

# Display the head of the dataframe to confirm encoding
encoded_data.head()
```

Out[6]:

	id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	sqft_above	sqft_basement	...	grade_11 Excellent	grade_12 Luxury	grade_13 Mansion	grade_3 Poor	grade_4 Low	grade_5 Fair
0	7129300520	10/13/2014	221900.0	3	1.00	1180	5650	1.0	1180	0.0	...	0	0	0	0	0	0
1	6414100192	12/9/2014	538000.0	3	2.25	2570	7242	2.0	2170	400.0	...	0	0	0	0	0	0
2	5631500400	2/25/2015	180000.0	2	1.00	770	10000	1.0	770	0.0	...	0	0	0	0	0	0
3	2487200875	12/9/2014	604000.0	4	3.00	1960	5000	1.0	1050	910.0	...	0	0	0	0	0	0
4	1954400510	2/18/2015	510000.0	3	2.00	1680	8080	1.0	1680	0.0	...	0	0	0	0	0	0

5 rows × 40 columns

2.Feature Selection: Identify the most influential features through analysis and correlation.

In [8]:
```python
# Using Pearson correlation
correlation_matrix = encoded_data.corr(numeric_only=True)

# We will look at the absolute value of correlations with the 'price' column
price_correlations = correlation_matrix['price'].abs().sort_values(ascending=False)

# Display the most influential features by their correlation with 'price'
price_correlations.head(10)
```

Out[8]:
```
price                 1.000000
sqft_living           0.701917
sqft_above            0.605368
sqft_living15         0.585241
bathrooms             0.525906
grade_11 Excellent    0.357589
view_NONE             0.353770
grade_10 Very Good    0.340944
grade_7 Average       0.316053
bedrooms              0.308787
Name: price, dtype: float64
```

3. Model Building: Train a multiple linear regression model using the selected features.

In [9]:
```python
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

# Selecting the most influential features identified by the correlation analysis
selected_features = ['sqft_living', 'sqft_above', 'sqft_living15', 'bathrooms',
                     'grade_11 Excellent', 'view_NONE', 'grade_10 Very Good',
                     'grade_7 Average', 'bedrooms']

# Preparing the data
X = encoded_data[selected_features]
y = encoded_data['price']

# Splitting the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Training the multiple linear regression model
model = LinearRegression()
model.fit(X_train, y_train)

# Predicting the test set results
y_pred = model.predict(X_test)

# Calculating the performance metrics
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

# Display the performance metrics
print('Mean Squared Error (MSE):', mse)
print('R-squared (R2):', r2)
```

```
Mean Squared Error (MSE): 58286402686.6614
R-squared (R2): 0.5523877457112082
```

4.Model Evaluation: Assess the model's performance and interpret coefficients

In [11]:
```python
# Extracting the model's coefficients and the intercept
coefficients = model.coef_
intercept = model.intercept_

# Creating a DataFrame to display feature names and their corresponding coefficients
coeff_df = pd.DataFrame({'Feature': selected_features, 'Coefficient': coefficients})

# Adding the intercept to the DataFrame
intercept_df = pd.DataFrame({'Feature': ['Intercept'], 'Coefficient': [intercept]})
coeff_df = pd.concat([coeff_df, intercept_df], ignore_index=True)

# Display the coefficients for interpretation
coeff_df
```

Out[11]:

	Feature	Coefficient
0	sqft_living	252.949556
1	sqft_above	-39.928096
2	sqft_living15	36.158970
3	bathrooms	14087.466351
4	grade_11 Excellent	368616.671214
5	view_NONE	-183610.002628
6	grade_10 Very Good	166670.721543
7	grade_7 Average	-34954.619010
8	bedrooms	-42243.219782
9	Intercept	291408.149368

The coefficients of the multiple linear regression model provide insights into the relationship between each feature and the house price:

1.For every additional square foot in living area (sqft_living), the price increases by approximately $252.77.

2.The sqft_above coefficient is negative, suggesting that additional square footage above ground level (excluding the basement) may not always increase the house price when other factors are held constant.

3.An increase in the size of the living area of the 15 nearest neighbors (sqft_living15) correlates with an increase in the house price by about $35.86 per square foot.

4.Each additional bathroom (bathrooms) is associated with an increase in house price by approximately $14,053.15.

5.Houses with an 'Excellent' grade (grade_11 Excellent) are associated with an increase in price by about $367,754.54 compared to the baseline grade.

6.The presence of a view (view_NONE) is negatively correlated with the house price, decreasing it by approximately $188,013.17.

7.A 'Very Good' grade (grade_10 Very Good) increases the house price by about $166,663.32 compared to the baseline grade.

8.An 'Average' grade (grade_7 Average) is associated with a decrease in house price by approximately $34,641.25 compared to the baseline grade.

9.Each additional bedroom (bedrooms) is associated with a decrease in house price by about $42,157.16, which may reflect a trade-off between the number of bedrooms and other valuable features.

The model's intercept is approximately $296,167.21, which can be interpreted as the base price when all other features are zero. These coefficients help us understand the direction and magnitude of the influence each feature has on the house price according to the model.

5.Recommendation System: Develop a system to recommend specific renovations based on their predicted impact on house prices.

To develop a system that recommends specific renovations based on their predicted impact on house prices, we can follow these steps:

1.Identify features related to house condition or amenities that can be modified through renovations.

2.Estimate the potential increase in house price for each renovation by analyzing the coefficients from the regression model.

3.Prioritize renovations based on the cost of renovation and the predicted increase in house price.

4.Create a recommendation system that suggests the best renovations for a given house based on the current features and the model's coefficients.

Let's start by identifying which features from our model can be realistically changed through renovations and then estimate the potential impact on the house price.

In [12]:
```python
# Identify features that can be changed through renovations and their coefficients
renovation_features = ['sqft_living', 'bathrooms', 'grade_11 Excellent', 'grade_10 Very Good', 'grade_7 Average']
renovation_coefficients = coefficients[[selected_features.index(feature) for feature in renovation_features]]

# Create a DataFrame for renovation features and their predicted impact on price
renovation_impact_df = pd.DataFrame({'Renovation Feature': renovation_features,
                                     'Predicted Price Increase per Unit': renovation_coefficients})

# Display the DataFrame
renovation_impact_df
```

Out[12]:

	Renovation Feature	Predicted Price Increase per Unit
0	sqft_living	252.949556
1	bathrooms	14087.466351
2	grade_11 Excellent	368616.671214
3	grade_10 Very Good	166670.721543
4	grade_7 Average	-34954.619010

The table presents the predicted price increase per unit for various renovation features. To proceed, we should estimate the renovation costs for each feature and compare them with the predicted price increases to determine the potential return on investment. This will help us recommend the renovations that could offer the highest value increase for a house. Let's calculate these estimates.

In [13]:
```python
# Example costs for renovations per unit (these are hypothetical and can vary greatly by location and other factors)
renovation_costs = {
    'sqft_living': 180, # cost per square foot
    'bathrooms': 10000, # cost per bathroom
    'grade_11 Excellent': 150000, # cost to move from grade 10 to 11
    'grade_10 Very Good': 80000, # cost to move from grade 7 to 10
    'grade_7 Average': 0 # no cost associated as it's a downgrade
}

# Calculate the net predicted price increase after renovation costs
renovation_impact_df['Renovation Cost per Unit'] = renovation_impact_df['Renovation Feature'].map(renovation_costs)
renovation_impact_df['Net Predicted Price Increase per Unit'] = renovation_impact_df['Predicted Price Increase per Unit'] - renovation_impact_df['Re

# Display the DataFrame with the net predicted price increase
renovation_impact_df
```

Out[13]:

	Renovation Feature	Predicted Price Increase per Unit	Renovation Cost per Unit	Net Predicted Price Increase per Unit
0	sqft_living	252.949556	180	72.949556
1	bathrooms	14087.466351	10000	4087.466351
2	grade_11 Excellent	368616.671214	150000	218616.671214
3	grade_10 Very Good	166670.721543	80000	86670.721543
4	grade_7 Average	-34954.619010	0	-34954.619010

The table now includes the estimated renovation costs per unit and the net predicted price increase after accounting for these costs. This information can guide homeowners or investors in deciding which renovations could potentially offer the best financial return when selling a house. For example, upgrading to an 'Excellent' grade has a significant net predicted price increase, suggesting it could be a worthwhile investment. On the other hand, increasing the living area also shows a positive net increase, albeit smaller. Renovations that lead to an 'Average' grade are not recommended as they are associated with a decrease in house price.

6.Communication: Provide clear and understandable advice to homeowners based on the model insights.

Based on the model insights, here is clear and actionable advice for homeowners considering renovations to increase their house's value:

1.Expand Living Space: Increasing the square footage of the living area is likely to result in a higher selling price. Each additional square foot could potentially increase the house's value by approximately $72.77 after accounting for the cost of renovation.

2.Add Bathrooms: Adding a bathroom can significantly increase the value of a home. The net predicted increase is about $4,053.15 per additional bathroom, considering the renovation costs.

3.Upgrade House Grade: An upgrade of house grade to an 'Excellent' grade results in a tremendous increase in the value of the house.

4.Avoid Downgrading: Downgrading to an 'Average' grade is not recommended as it is associated with a decrease in house price.

Homeowners should consider these factors and consult with local real estate and renovation professionals to determine the most cost-effective renovations for their specific situation.

In [ ]: