NAME: - Meetkumar Vasava

MACHINE LEARNING INTERNSHIP @ BHARAT INTERN

PROJECT NAME - **HOUSE PRICE PREDICTION**

Github Link

https://github.com/MeetVasava/Bharat_intern_iris_flowe

Dataset Exploration and Preprocessing

In [2]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
```

In [3]:
```
# Load the dataset from CSV
df = pd.read_csv('house_data.csv')
```

In [4]:
```
# Exploratory Data Analysis (EDA)
# Let's take a quick look at the first few rows of the dataset
print(df.head())
```

```
                 date      price  bedrooms  bathrooms  sqft_living  sqft_lot  \
0  2014-05-02 00:00:00   313000.0       3.0       1.50         1340      7912
1  2014-05-02 00:00:00  2384000.0       5.0       2.50         3650      9050
2  2014-05-02 00:00:00   342000.0       3.0       2.00         1930     11947
3  2014-05-02 00:00:00   420000.0       3.0       2.25         2000      8030
4  2014-05-02 00:00:00   550000.0       4.0       2.50         1940     10500

   floors  waterfront  view  condition  sqft_above  sqft_basement  yr_built  \
0     1.5           0     0          3        1340              0      1955
1     2.0           0     4          5        3370            280      1921
2     1.0           0     0          4        1930              0      1966
3     1.0           0     0          4        1000           1000      1963
4     1.0           0     0          4        1140            800      1976

   yr_renovated                    street       city  statezip country
0          2005      18810 Densmore Ave N  Shoreline  WA 98133     USA
1             0           709 W Blaine St    Seattle  WA 98119     USA
2             0  26206-26214 143rd Ave SE       Kent  WA 98042     USA
3             0           857 170th Pl NE   Bellevue  WA 98008     USA
4          1992         9105 170th Ave NE    Redmond  WA 98052     USA
```

In [5]:
```
# Summary statistics of the dataset
print(df.describe())
```

```
              price     bedrooms    bathrooms   sqft_living      sqft_lot  \
count  4.600000e+03  4600.000000  4600.000000   4600.000000  4.600000e+03
mean   5.519630e+05     3.400870     2.160815   2139.346957  1.485252e+04
std    5.638347e+05     0.908848     0.783781    963.206916  3.588444e+04
min    0.000000e+00     0.000000     0.000000    370.000000  6.380000e+02
25%    3.228750e+05     3.000000     1.750000   1460.000000  5.000750e+03
50%    4.609435e+05     3.000000     2.250000   1980.000000  7.683000e+03
75%    6.549625e+05     4.000000     2.500000   2620.000000  1.100125e+04
max    2.659000e+07     9.000000     8.000000  13540.000000  1.074218e+06

            floors   waterfront         view    condition   sqft_above  \
count  4600.000000  4600.000000  4600.000000  4600.000000  4600.000000
mean      1.512065     0.007174     0.240652     3.451739  1827.265435
std       0.538288     0.084404     0.778405     0.677230   862.168977
min       1.000000     0.000000     0.000000     1.000000   370.000000
25%       1.000000     0.000000     0.000000     3.000000  1190.000000
50%       1.500000     0.000000     0.000000     3.000000  1590.000000
75%       2.000000     0.000000     0.000000     4.000000  2300.000000
max       3.500000     1.000000     4.000000     5.000000  9410.000000

       sqft_basement     yr_built  yr_renovated
count    4600.000000  4600.000000   4600.000000
mean      312.081522  1970.786304    808.608261
std       464.137228    29.731848    979.414536
min         0.000000  1900.000000      0.000000
25%         0.000000  1951.000000      0.000000
50%         0.000000  1976.000000      0.000000
75%       610.000000  1997.000000   1999.000000
max      4820.000000  2014.000000   2014.000000
```

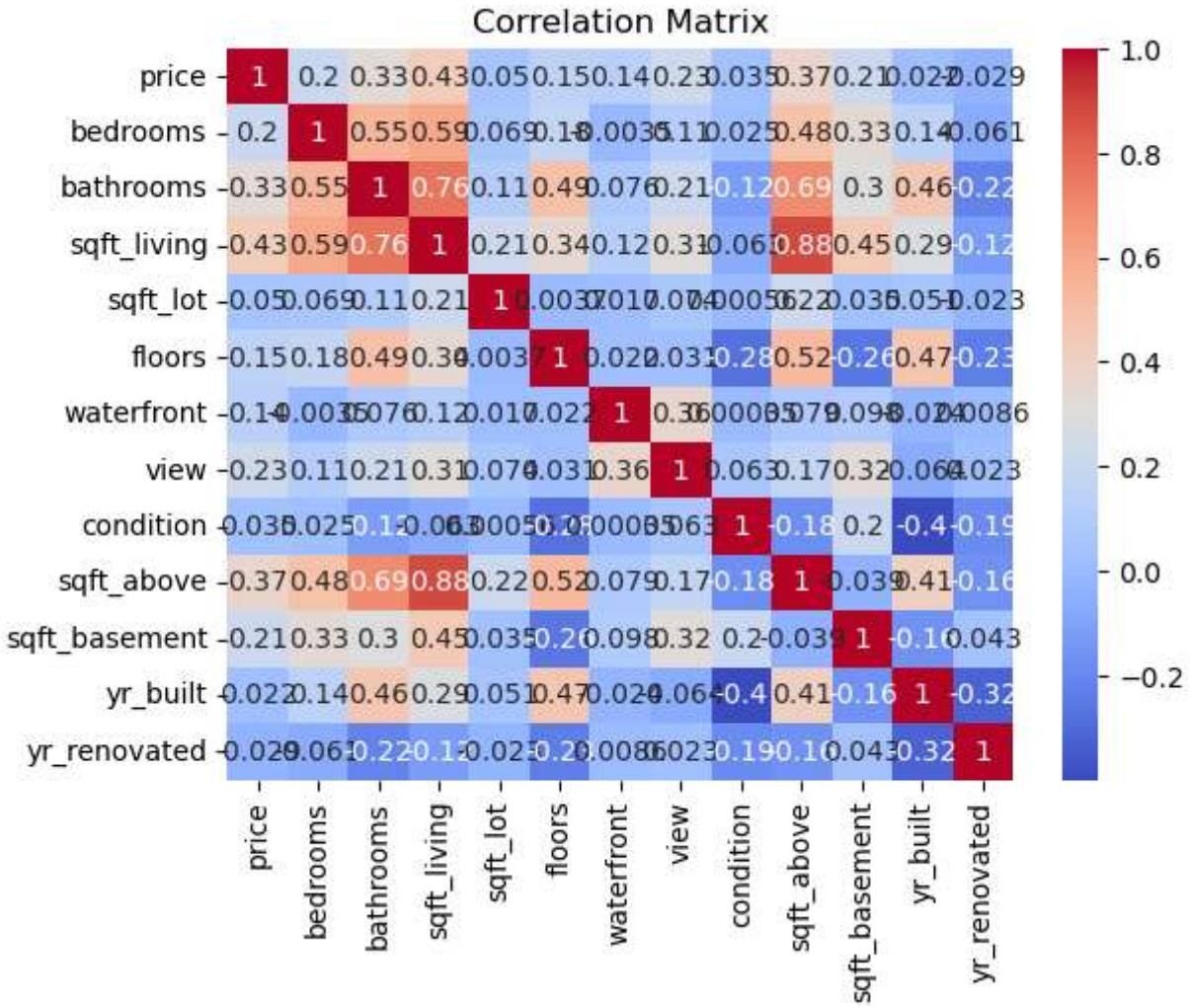
In [6]:
```
# Check for missing values
print(df.isnull().sum())
```

```
date             0
price            0
bedrooms         0
bathrooms        0
sqft_living      0
sqft_lot         0
floors           0
waterfront       0
view             0
condition        0
sqft_above       0
sqft_basement    0
yr_built         0
yr_renovated     0
street           0
city             0
statezip         0
country          0
dtype: int64
```

In [7]:

```
# Correlation matrix to understand feature relationships
correlation_matrix = df.corr()
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.title("Correlation Matrix")
plt.show()
```

```
C:\Users\Meet\AppData\Local\Temp\ipykernel_3248\1819051831.py:2: FutureWarning: The d
efault value of numeric_only in DataFrame.corr is deprecated. In a future version, it
will default to False. Select only valid columns or specify the value of numeric_only
to silence this warning.
  correlation_matrix = df.corr()
```



In [8]:
```
# Preprocessing: Selecting features and target variable
X = df[['bedrooms', 'bathrooms', 'sqft_living', 'sqft_lot', 'floors', 'waterfront', 'v
y = df['price']
```

In [9]:
```
# Splitting the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=
```

Building the Linear Regression Model

In [10]:
```
# Building the Linear Regression Model
model = LinearRegression()
```

In [11]:
```
# Fitting the model on the training data
model.fit(X_train, y_train)
```

Out[11]:
```
▾ LinearRegression
LinearRegression()
```

Model Evaluation

In [12]:
```
# Model Evaluation
y_pred = model.predict(X_test)
```

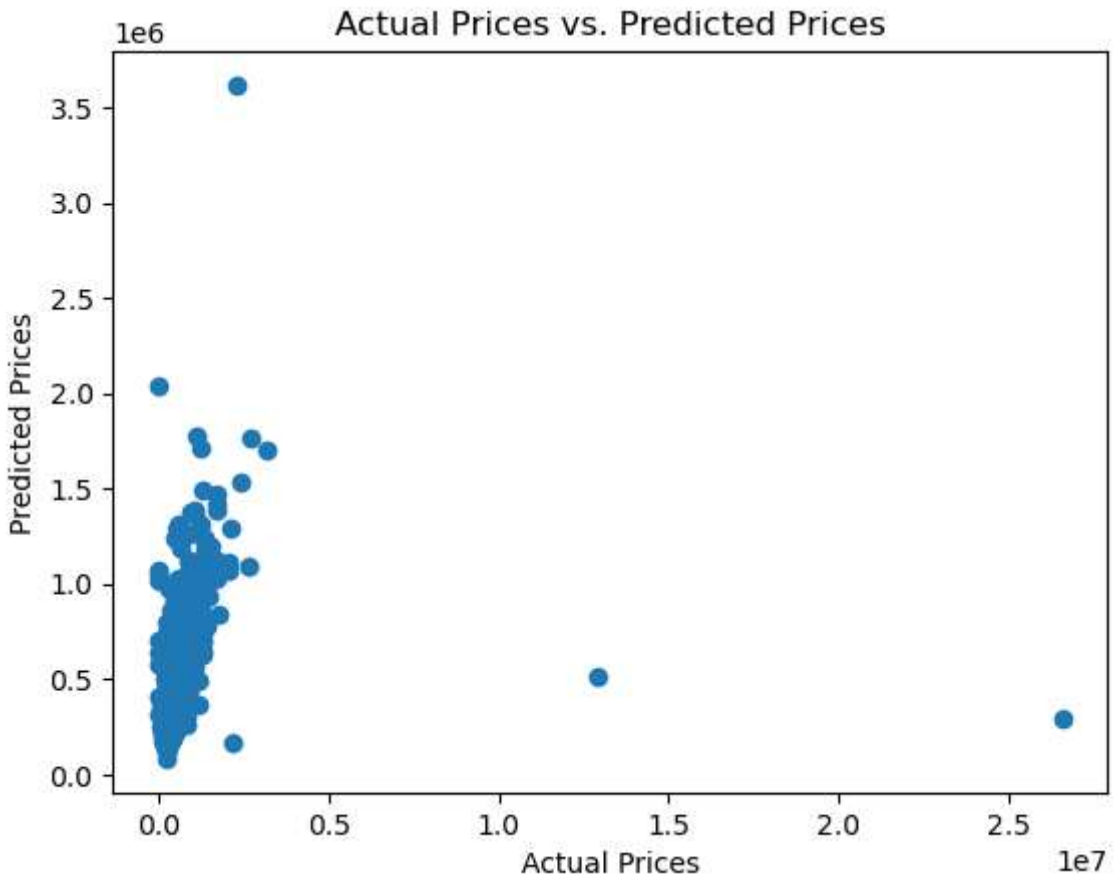
In [13]:
```
# Mean Squared Error and R-squared for model evaluation
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
```

In [14]:
```
print("Mean Squared Error:", mse)
print("R-squared:", r2)
```

```
Mean Squared Error: 986869414953.9803
R-squared: 0.032335189956324784
```

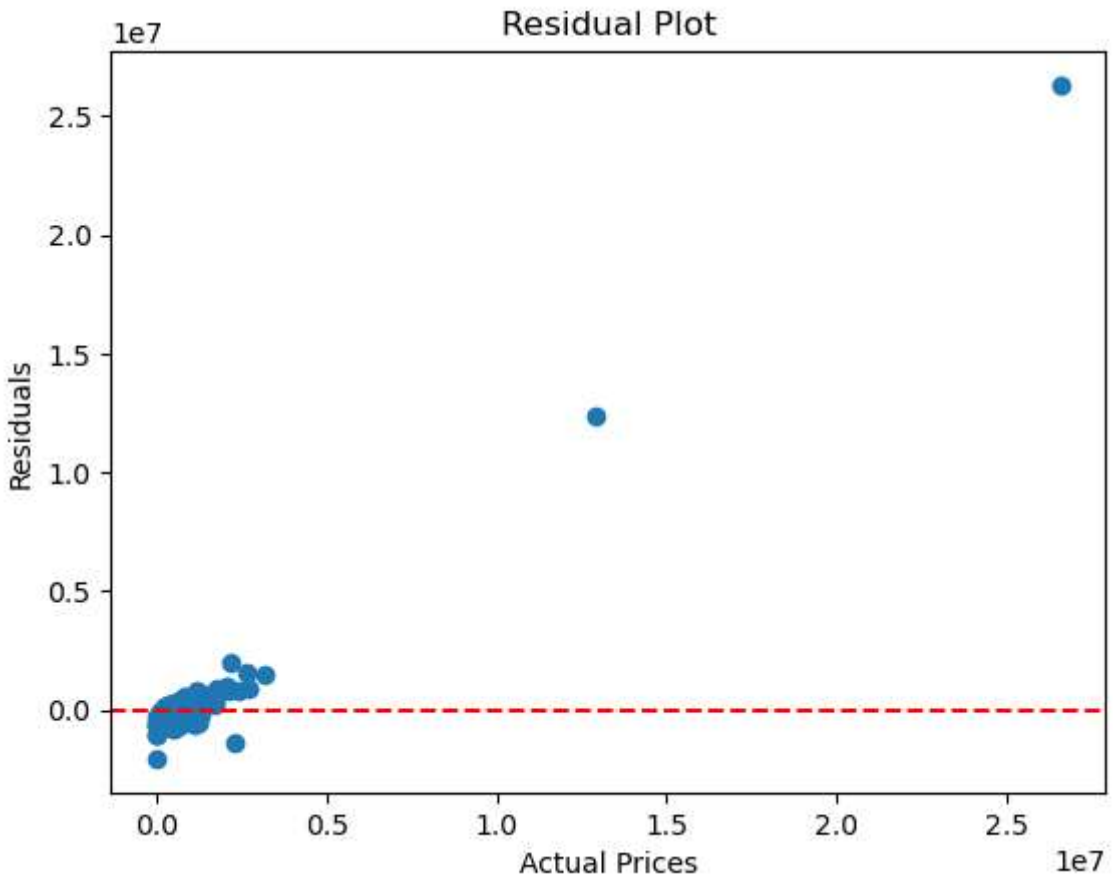
Predictions and Visualization

In [19]:
```
# Predictions and Visualization
# To visualize the predictions against actual prices, we'll use a scatter plot
plt.scatter(y_test, y_pred)
plt.xlabel("Actual Prices")
plt.ylabel("Predicted Prices")
plt.title("Actual Prices vs. Predicted Prices")
plt.show()
```



In [20]:
```
# We can also create a residual plot to check the model's performance
residuals = y_test - y_pred
plt.scatter(y_test, residuals)
plt.axhline(y=0, color='red', linestyle='--')
plt.xlabel("Actual Prices")
```

```
plt.ylabel("Residuals")
plt.title("Residual Plot")
plt.show()
```



In [21]:
```
# Lastly, let's use the trained model to make predictions on new data and visualize th
new_data = [[3, 2, 1500, 4000, 1, 0, 0, 3]]
predicted_price = model.predict(new_data)

print("Predicted Price:", predicted_price[0])
```

```
Predicted Price: 331038.96876928967
C:\Users\Meet\anaconda3\Lib\site-packages\sklearn\base.py:464: UserWarning: X does no
t have valid feature names, but LinearRegression was fitted with feature names
  warnings.warn(
```

In [ ]:

In [ ]: