In [1]:
```
"""Question 2:Explain how you can implement ML in a real world application.

Train an SVM regressor on : Bengaluru housing dataset

                  Must include in details:

                        - EDA

                        - Feature engineering """
```

Out[1]:
```
'Question 2:Explain how you can implement ML in a real world application.\n\nTrain an SVM regressor on : Bengaluru housing dataset\n\n                  Must include in details:\n\n                        - EDA\n\n                        - Feature engineering '
```

In [2]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVR
from sklearn.metrics import mean_absolute_error, mean_squared_error

# Load the Bengaluru housing dataset
df = pd.read_csv('Bengaluru_House_Data.csv')
```

In [3]:
```
df
```

Out[3]:

	area_type	availability	location	size	society	total_sqft	bath	balcony	price
0	Super built-up Area	19-Dec	Electronic City Phase II	2 BHK	Coomee	1056	2.0	1.0	39.07
1	Plot Area	Ready To Move	Chikka Tirupathi	4 Bedroom	Theanmp	2600	5.0	3.0	120.00
2	Built-up Area	Ready To Move	Uttarahalli	3 BHK	NaN	1440	2.0	3.0	62.00
3	Super built-up Area	Ready To Move	Lingadheeranahalli	3 BHK	Soiewre	1521	3.0	1.0	95.00
4	Super built-up Area	Ready To Move	Kothanur	2 BHK	NaN	1200	2.0	1.0	51.00
...	...	...	...	...	...	...	...	...	...
13315	Built-up Area	Ready To Move	Whitefield	5 Bedroom	ArsiaEx	3453	4.0	0.0	231.00
13316	Super built-up Area	Ready To Move	Richards Town	4 BHK	NaN	3600	5.0	NaN	400.00
13317	Built-up Area	Ready To Move	Raja Rajeshwari Nagar	2 BHK	Mahla T	1141	2.0	1.0	60.00
13318	Super built-up Area	18-Jun	Padmanabhanagar	4 BHK	SollyCl	4689	4.0	1.0	488.00
13319	Super built-up Area	Ready To Move	Doddathoguru	1 BHK	NaN	550	1.0	1.0	17.00

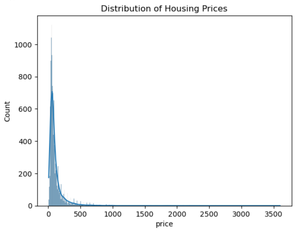
13320 rows × 9 columns

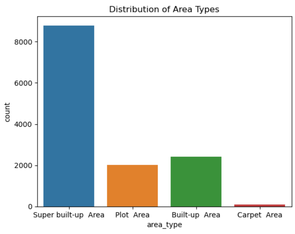
In [4]:
```
df.columns
```

Out[4]:
```
Index(['area_type', 'availability', 'location', 'size', 'society',
       'total_sqft', 'bath', 'balcony', 'price'],
      dtype='object')
```

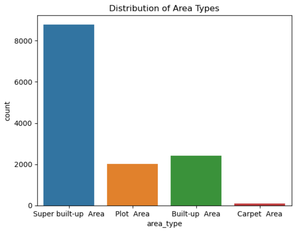
In [5]:
```
# Display basic information about the dataset
print(df.info())

# Display summary statistics
print(df.describe())

# Check for missing values
print(df.isnull().sum())

# Visualize the distribution of the target variable 'price'
sns.histplot(df['price'], kde=True)
plt.title('Distribution of Housing Prices')
plt.show()


# Explore categorical features
sns.countplot(x='area_type', data=df)
plt.title('Distribution of Area Types')
plt.show()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 13320 entries, 0 to 13319
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   area_type     13320 non-null  object
 1   availability  13320 non-null  object
 2   location      13319 non-null  object
 3   size          13304 non-null  object
 4   society       7818 non-null   object
 5   total_sqft    13320 non-null  object
 6   bath          13247 non-null  float64
 7   balcony       12711 non-null  float64
 8   price         13320 non-null  float64
dtypes: float64(3), object(6)
memory usage: 936.7+ KB
None
                bath       balcony         price
count  13247.000000  12711.000000  13320.000000
mean       2.692610      1.584376    112.565627
std        1.341458      0.817263    148.971674
min        1.000000      0.000000      8.000000
25%        2.000000      1.000000     50.000000
50%        2.000000      2.000000     72.000000
75%        3.000000      2.000000    120.000000
max       40.000000      3.000000   3600.000000
area_type          0
availability       0
location           1
size              16
society         5502
total_sqft         0
bath              73
balcony          609
price              0
dtype: int64
```





In [6]:
```
# Explore categorical features
sns.countplot(x='area_type', data=df)
plt.title('Distribution of Area Types')
plt.show()
```



In [7]:
```
df
```

Out[7]:

	area_type	availability	location	size	society	total_sqft	bath	balcony	price
0	Super built-up Area	19-Dec	Electronic City Phase II	2 BHK	Coomee	1056	2.0	1.0	39.07
1	Plot Area	Ready To Move	Chikka Tirupathi	4 Bedroom	Theanmp	2600	5.0	3.0	120.00
2	Built-up Area	Ready To Move	Uttarahalli	3 BHK	NaN	1440	2.0	3.0	62.00
3	Super built-up Area	Ready To Move	Lingadheeranahalli	3 BHK	Soiewre	1521	3.0	1.0	95.00
4	Super built-up Area	Ready To Move	Kothanur	2 BHK	NaN	1200	2.0	1.0	51.00
...	...	...	...	...	...	...	...	...	...
13315	Built-up Area	Ready To Move	Whitefield	5 Bedroom	ArsiaEx	3453	4.0	0.0	231.00
13316	Super built-up Area	Ready To Move	Richards Town	4 BHK	NaN	3600	5.0	NaN	400.00
13317	Built-up Area	Ready To Move	Raja Rajeshwari Nagar	2 BHK	Mahla T	1141	2.0	1.0	60.00
13318	Super built-up Area	18-Jun	Padmanabhanagar	4 BHK	SollyCl	4689	4.0	1.0	488.00
13319	Super built-up Area	Ready To Move	Doddathoguru	1 BHK	NaN	550	1.0	1.0	17.00

13320 rows × 9 columns

In [10]:
```
# Drop irrelevant columns or columns with too many missing values
df = df.drop(['bath'], axis=1)
# Handle missing values in numerical features
df['size'].fillna(df['size'].mode()[0], inplace=True)
```

In [14]:
```
# Convert 'total_sqft' to numerical values
#df['total_sqft'] = df['total_sqft'].apply(lambda x: eval(x) if isinstance(x, str) else x)

# Handle categorical variables using one-hot encoding
df = pd.get_dummies(df, columns=['area_type', 'location'], drop_first=True)
```

In [15]:
```
df
```

Out[15]:

	availability	size	society	total_sqft	balcony	price	area_type_Carpet Area	area_type_Plot Area	area_type_Super built-up Area	location_ Banaswadi	...	location_rr nagar	location_sankeswari	location_sapthagiri Layout	location_sarjapura main road	location_singapura paradise	lo
0	19-Dec	2 BHK	Coomee	1056	1.0	39.07	False	False	True	False	...	False	False	False	False	False	
1	Ready To Move	4 Bedroom	Theanmp	2600	3.0	120.00	False	True	False	False	...	False	False	False	False	False	
2	Ready To Move	3 BHK	NaN	1440	3.0	62.00	False	False	False	False	...	False	False	False	False	False	
3	Ready To Move	3 BHK	Soiewre	1521	1.0	95.00	False	False	True	False	...	False	False	False	False	False	
4	Ready To Move	2 BHK	NaN	1200	1.0	51.00	False	False	True	False	...	False	False	False	False	False	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
13315	Ready To Move	5 Bedroom	ArsiaEx	3453	0.0	231.00	False	False	False	False	...	False	False	False	False	False	
13316	Ready To Move	4 BHK	NaN	3600	NaN	400.00	False	False	True	False	...	False	False	False	False	False	
13317	Ready To Move	2 BHK	Mahla T	1141	1.0	60.00	False	False	False	False	...	False	False	False	False	False	
13318	18-Jun	4 BHK	SollyCl	4689	1.0	488.00	False	False	True	False	...	False	False	False	False	False	
13319	Ready To Move	1 BHK	NaN	550	1.0	17.00	False	False	True	False	...	False	False	False	False	False	

13320 rows × 1313 columns

In [21]:
```
#Train-Test Split and Standardization
#Split the data into features (X) and target variable (y)
X = df.drop('price', axis=1)
y = df['price']

 Split 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)

 Standardize numerical features
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

In [22]:
```
#Train the SVM Regressor
# Train the Support Vector Machine Regressor
svm_regressor = SVR(kernel='linear')
svm_regressor.fit(X_train_scaled, y_train)
```

In [ ]:
```
#Evaluate the Model
 #Make predictions on the test set
predictions = svm_regressor.predict(X_test_scaled)

 Evaluate the model
mae = mean_absolute_error(y_test, predictions)
mse = mean_squared_error(y_test, predictions)
print(f'Mean Absolute Error: {mae}')
print(f'Mean Squared Error: {mse}')
```

In [ ]:

In [ ]: