In [16]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import sklearn
import micropip
await micropip.install('seaborn')
import seaborn as sns
from sklearn.feature_selection import SelectKBest, f_regression
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.neighbors import KNeighborsRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
from statsmodels.stats.outliers_influence import variance_inflation_factor
import warnings
warnings.filterwarnings('ignore')
```

In [17]:
```
df = pd.read_csv("USA_Housing.csv")
df.head()
```

Out[17]:

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price	
0	79545.458574	5.682861	7.009188	4.09	23086.800503	1.059034e+06	20 nL N
1	79248.642455	6.002900	6.730821	3.09	40173.072174	1.505891e+06	18 Vi (
2	61287.067179	5.865890	8.512727	5.13	36882.159400	1.058988e+06	S nDa W
3	63345.240046	7.188236	5.586729	3.26	34310.242831	1.260617e+06	US
4	59982.197226	5.040555	7.839388	4.23	26354.109472	6.309435e+05	I

In [32]:
```
print(" Shape of Dataset:", df.shape)
print("\nData Types:\n")
print(df.dtypes)
```

```
 Shape of Dataset: (5000, 7)

Data Types:

Avg. Area Income               float64
Avg. Area House Age            float64
Avg. Area Number of Rooms      float64
Avg. Area Number of Bedrooms   float64
Area Population                float64
Price                          float64
Address                         object
dtype: object
```

In [33]:
```
df.describe()
```

Out[33]:

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	
count	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000	5.00
mean	68583.108984	5.977222	6.987792	3.981330	36163.516039	1.23
std	10657.991214	0.991456	1.005833	1.234137	9925.650114	3.53
min	17796.631190	2.644304	3.236194	2.000000	172.610686	1.59
25%	61480.562388	5.322283	6.299250	3.140000	29403.928702	9.97
50%	68804.286404	5.970429	7.002902	4.050000	36199.406689	1.23
75%	75783.338666	6.650808	7.665871	4.490000	42861.290769	1.47
max	107701.748378	9.519088	10.759588	6.500000	69621.713378	2.46

In [34]:
```
df.isnull().sum()
```

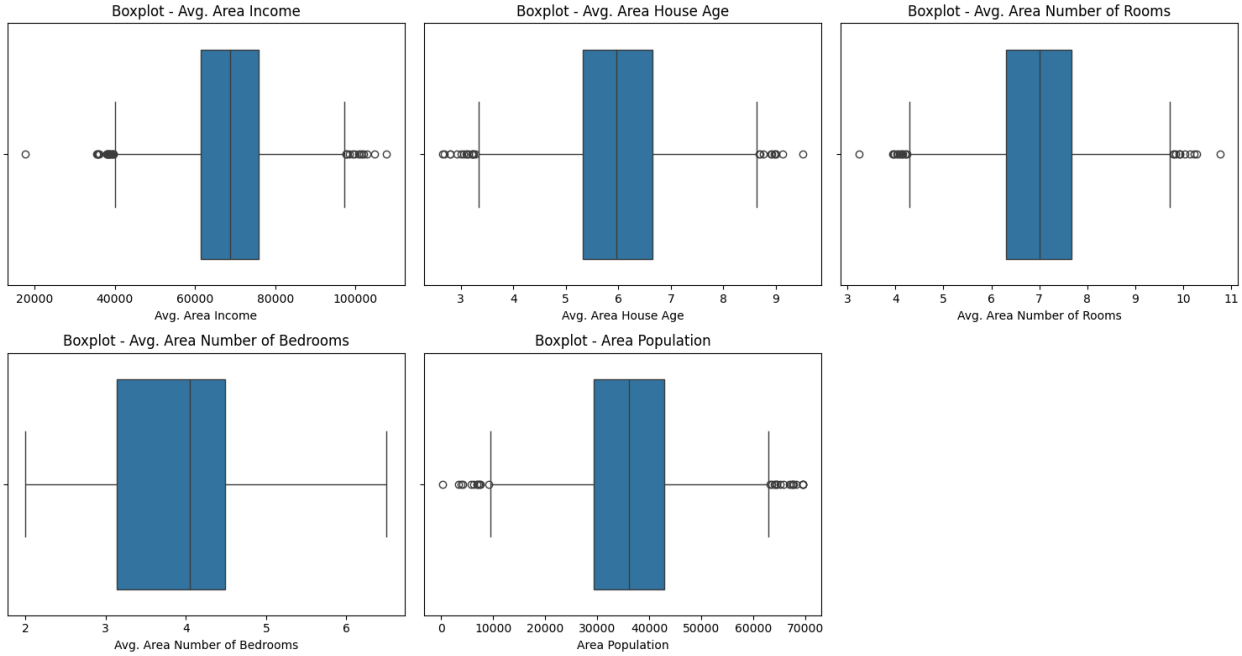
Out[34]:
```
Avg. Area Income               0
Avg. Area House Age            0
Avg. Area Number of Rooms      0
Avg. Area Number of Bedrooms   0
Area Population                0
Price                          0
Address                        0
dtype: int64
```

In [35]:
```
print("Number  of duplicate rows:", df.duplicated().sum())
```

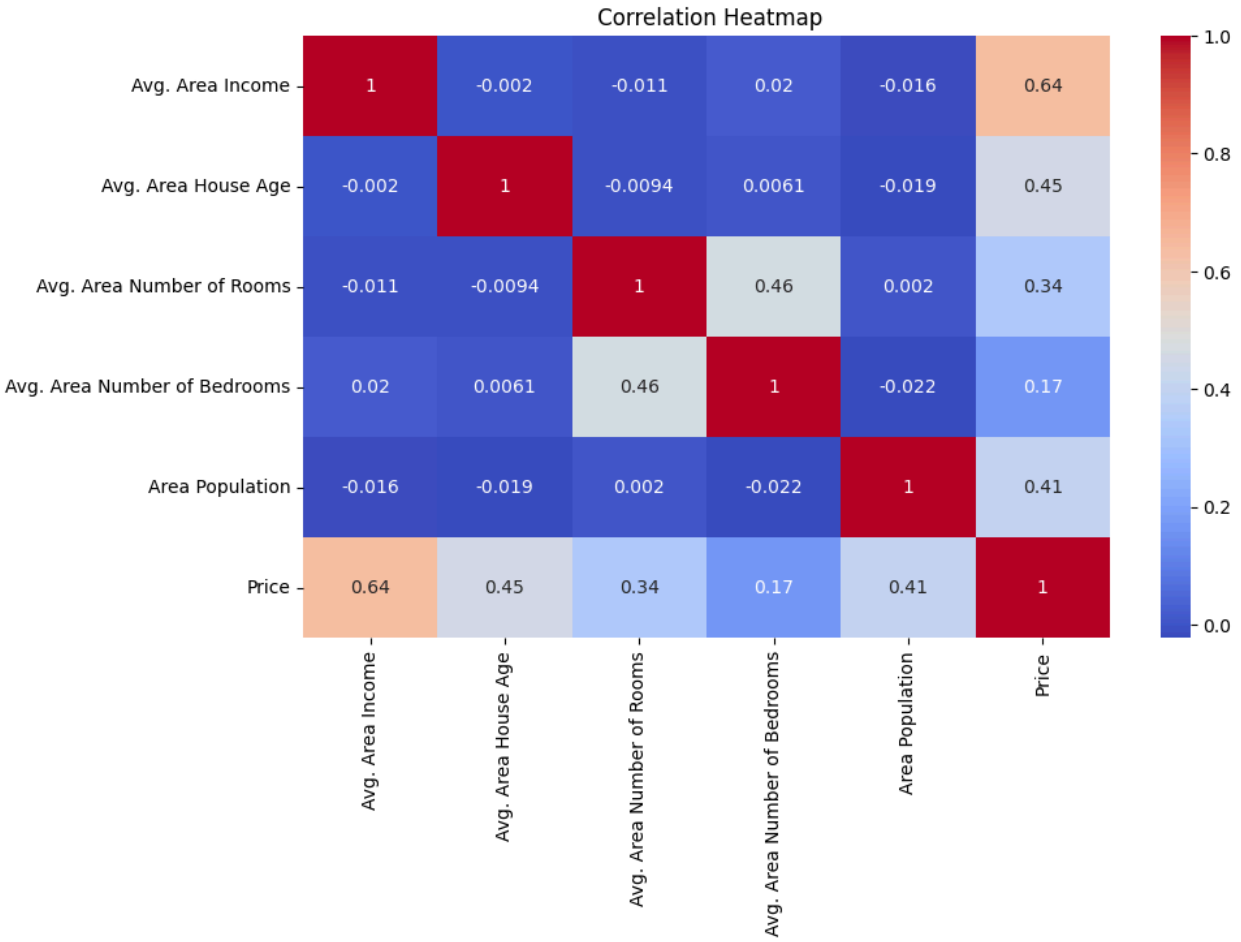
```
Number  of duplicate rows: 0
```

In [36]:
```
numerical_cols = df.select_dtypes(include=np.number).columns.drop('Price', err
plt.figure(figsize=(15, 8))
for i, col in enumerate(numerical_cols, 1):
    plt.subplot(2, 3, i)
    sns.boxplot(x=df[col])
    plt.title(f"Boxplot - {col}")
```

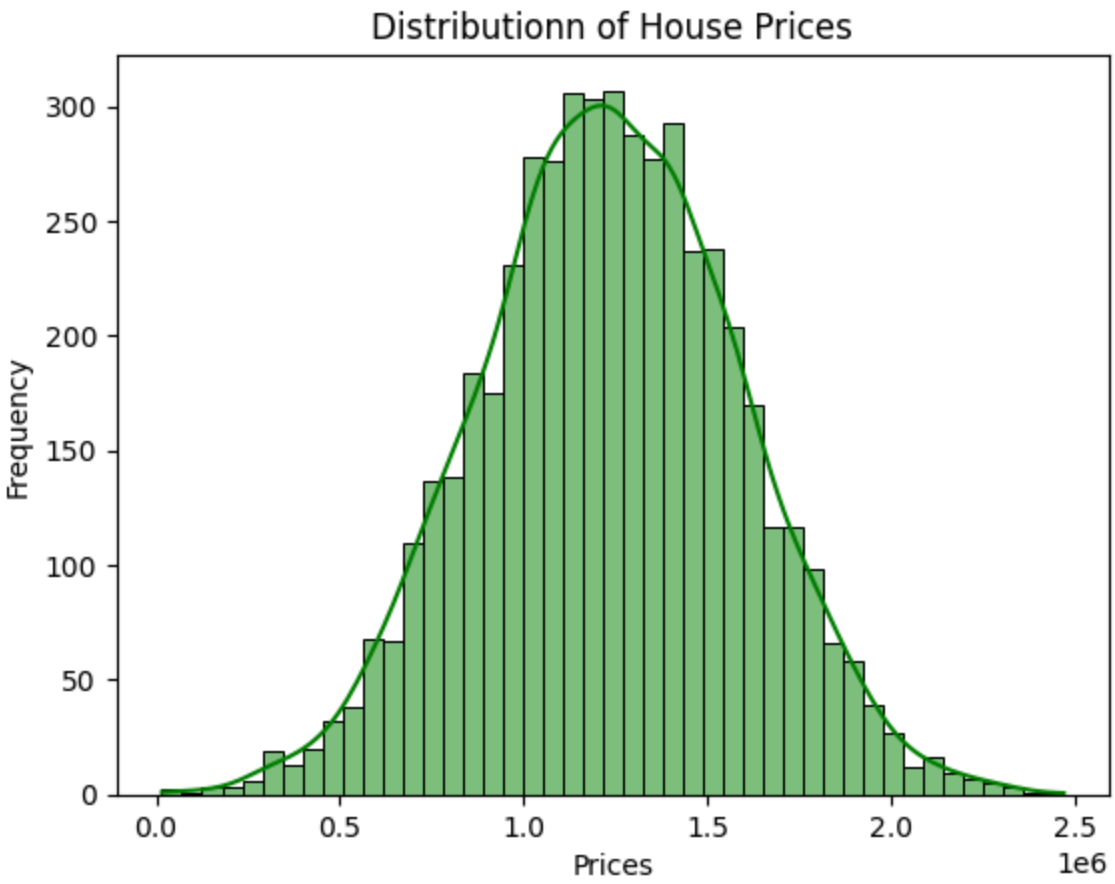
```
plt.tight_layout()
plt.show()
```



In [42]:
```
corr_matrix = df.drop(columns=['Address']).corr()
plt.figure(figsize=(10, 6))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')
plt.title(" Correlation Heatmap")
plt.show()
```



In [43]:
```
sns.histplot(df['Price'], kde=True, color='green')
plt.title("Distributionn of House Prices")
plt.xlabel("Prices")
plt.ylabel("Frequency")
plt.show()
```



In [44]:
```
df.select_dtypes(include='object').columns
```

Out[44]:
```
Index(['Address'], dtype='object')
```

In [ ]:
```

```

In [45]:
```
from statsmodels.stats.outliers_influence import variance_inflation_factor

X = df.select_dtypes(include=np.number).drop('Price', axis=1)

vif_data = pd.DataFrame()
vif_data["Feature"] = X.columns
vif_data["VIF"] = [variance_inflation_factor(X.values, i) for i in range(X.sha

vif_data.sort_values(by="VIF", ascending=False)
```

Out[45]:

	Feature	VIF
2	Avg. Area Number of Rooms	45.257291
0	Avg. Area Income	29.650899
1	Avg. Area House Age	27.447775
3	Avg. Area Number of Bedrooms	14.537873
4	Area Population	12.825450

In [50]:

```
from sklearn.feature_selection import SelectKBest, f_regression

X = df.drop(columns=['Price', 'Address'])
y = df['Price']

selector = SelectKBest(score_func=f_regression, k='all')
selector.fit(X, y)

score_df = pd.DataFrame({'Feature': X.columns, 'Score': selector.scores_})
score_df.sort_values(by='Score', ascending=False)
```

Out[50]:

	Feature	Score
0	Avg. Area Income	3462.564948
1	Avg. Area House Age	1287.169756
4	Area Population	1001.408749
2	Avg. Area Number of Rooms	634.632191
3	Avg. Area Number of Bedrooms	150.677576

In [ ]:

In [ ]:

In [ ]:

In [ ]: