CODE

```
# Import necessary libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.datasets import fetch_california_housing
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

# Load the California housing dataset
housing = fetch_california_housing()
X = pd.DataFrame(housing.data, columns=housing.feature_names)
y = housing.target

# Split the dataset into training and test sets (80% training, 20% testing)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initialize the Linear Regression model
model = LinearRegression()

# Train the model on the training data
model.fit(X_train, y_train)

# Predict house prices on the test set
y_pred = model.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
```

```
print("Mean Squared Error:", mse)
print("R^2 Score:", r2)


# 1. Actual vs Predicted Scatter Plot
plt.figure(figsize=(6, 6))
plt.scatter(y_test, y_pred, alpha=0.5, color='purple')
plt.xlabel("Actual Prices")
plt.ylabel("Predicted Prices")
plt.title("Actual vs Predicted House Prices")
plt.grid(True)
plt.show()

# 2. Histogram of Target Variable
plt.figure(figsize=(8, 5))
plt.hist(y, bins=30, color='skyblue', edgecolor='black')
plt.title("Distribution of House Prices (Target Variable)")
plt.xlabel("Median House Value")
plt.ylabel("Frequency")
plt.grid(True)
plt.show()

# 3. Bar Plot of Feature Coefficients
coefficients = pd.Series(model.coef_, index=X.columns)
plt.figure(figsize=(10, 6))
coefficients.plot(kind='bar', color='teal')
plt.title("Feature Coefficients in Linear Regression Model")
plt.xlabel("Features")
plt.ylabel("Coefficient Value")
plt.grid(axis='y')
```

```
plt.show()

# 4. Residual Plot
residuals = y_test - y_pred
plt.figure(figsize=(8, 5))
plt.scatter(y_pred, residuals, alpha=0.5, color='orange')
plt.axhline(0, color='red', linestyle='--')
plt.xlabel("Predicted Values")
plt.ylabel("Residuals")
plt.title("Residual Plot")
plt.grid(True)
plt.show()

# 5. Correlation Heatmap of Features
data_with_target = X.copy()
data_with_target['Target'] = y

plt.figure(figsize=(10, 8))
sns.heatmap(data_with_target.corr(), annot=True, cmap='coolwarm', fmt=".2f")
plt.title("Feature Correlation Heatmap")
plt.show()

# 6. Line Plot for Predicted vs Actual (Sorted)
sorted_indices = np.argsort(y_test)
sorted_y_test = np.array(y_test)[sorted_indices]
sorted_y_pred = y_pred[sorted_indices]

plt.figure(figsize=(10, 5))
plt.plot(sorted_y_test, label='Actual', color='blue')
plt.plot(sorted_y_pred, label='Predicted', color='green')
plt.title("Actual vs Predicted Prices (Sorted)")
```

```
plt.xlabel("Sample Index")
plt.ylabel("House Price")
plt.legend()
plt.grid(True)
plt.show()
```