```
import pandas as pd
import numpy as np
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 r2_score

# Set a random seed for reproducibility
np.random.seed(42)

# Create the dataset
data = {
    'ID': range(1, 501),  # Unique ID for each entry
    'flat': np.random.randint(10, 100, 500),  # Random number of flats
    'houses': np.random.randint(1, 50, 500),  # Random number of houses
    'purchases': np.random.randint(100, 1000, 500)  # Random number of purchases
}

# Convert to DataFrame
df = pd.DataFrame(data)

# Display the first few rows of the dataset
print("Sample of the dataset:")
print(df.head())

# Define independent variable (feature) and target variable
```

```
X = df[['flat']]  # Independent variable (only flat)
y = df['purchases']  # Target variable

# Split the data into training and testing sets (70:30 ratio)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Print the shapes of the training and testing sets
print("\nTraining set shape:", X_train.shape, y_train.shape)
print("Testing set shape:", X_test.shape, y_test.shape)

# Build a simple linear regression model
model = LinearRegression()
model.fit(X_train, y_train)

# Print the model coefficients
print("\nModel coefficients:")
print("Intercept:", model.intercept_)
print("Coefficient (flat):", model.coef_[0])

# Predict purchases on the testing set
y_pred = model.predict(X_test)

# Evaluate the model using R^2 score
r2 = r2_score(y_test, y_pred)
print("\nModel R^2 score on testing set:", r2)

# Plotting scatter plot for flat vs purchases
```

```
plt.figure(figsize=(8, 6))

sns.scatterplot(x=X_test['flat'], y=y_test, color='blue', label='Actual Purchases')

sns.lineplot(x=X_test['flat'], y=y_pred, color='red', label='Predicted Purchases')

plt.title('Simple Linear Regression: Flat vs Purchases')

plt.xlabel('Number of Flats')

plt.ylabel('Purchases')

plt.legend()

plt.show()


# Plotting predicted vs actual purchases

plt.figure(figsize=(6, 6))

sns.scatterplot(x=y_test, y=y_pred, color='purple')

plt.title('Predicted vs Actual Purchases')

plt.xlabel('Actual Purchases')

plt.ylabel('Predicted Purchases')

plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], color='black', linestyle='--')  # Diagonal line

plt.show()
```