Implement a linear regression model to predict the prices of houses based on their square footage and the number of bedrooms and bathrooms

Data Collection

```
import pandas as pd

# Load the datasets
train_data = pd.read_csv('train.csv')
test_data = pd.read_csv('test.csv')
```

Load and Preprocess the Data

```
# Select relevant features and target variable
features = ['GrLivArea', 'BedroomAbvGr', 'FullBath']
target = 'SalePrice'

X_train = train_data[features]
y_train = train_data[target]

# Check for missing values in train data
print(X_train.isnull().sum())
X_train = X_train.dropna()
y_train = y_train[X_train.index]  # Ensure target variable aligns wit

# Check for missing values in test data
X_test = test_data[features]
print(X_test.isnull().sum())
X_test = X_test.fillna(X_test.mean())  # Fill missing values with mea
```

```
GrLivArea       0
BedroomAbvGr    0
FullBath        0
dtype: int64
GrLivArea       0
BedroomAbvGr    0
FullBath        0
dtype: int64
```

Train the Linear Regression Model

```
from sklearn.linear_model import LinearRegression

model = LinearRegression()
model.fit(X_train, y_train)
```

```
▾ LinearRegression
LinearRegression()
```

Evaluate the Model (using cross-validation on training data for simplicity)

```
from sklearn.metrics import mean_squared_error, r2_score

# Split the training data into training and validation sets
from sklearn.model_selection import train_test_split

X_train_split, X_val_split, y_train_split, y_val_split = train_test_s

# Train the model on the split training set
model.fit(X_train_split, y_train_split)

# Predict on the validation set
y_val_pred = model.predict(X_val_split)

# Calculate evaluation metrics
mse = mean_squared_error(y_val_split, y_val_pred)
r2 = r2_score(y_val_split, y_val_pred)

print(f'Mean Squared Error: {mse}')
print(f'R-squared Score: {r2}')
```

```
Mean Squared Error: 2806426667.247853
R-squared Score: 0.6341189942328371
```

Make Predictions on the Test Data

```
# Predict the prices for the test set
y_test_pred = model.predict(X_test)

# Create a DataFrame to save predictions along with the input features
predictions = X_test.copy()
predictions['Id'] = test_data['Id']
predictions['PredictedSalePrice'] = y_test_pred

# Display the predictions
print(predictions.head())

# Save predictions
predictions[['Id', 'PredictedSalePrice']].to_csv('house_price_predicti
```

```
   GrLivArea  BedroomAbvGr  FullBath    Id  PredictedSalePrice
0        896             2         1  1461       122173.313104
1       1329             3         1  1462       140561.538683
2       1629             3         2  1463       201783.754896
3       1604             3         2  1464       199183.097221
4       1280             2         2  1465       192133.739106
```