In [3]:

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.impute import SimpleImputer  # Import SimpleImputer for hand
import warnings
warnings.filterwarnings("ignore")

# Load the dataset
wine_data = pd.read_csv('Wine_Quality.csv')  # Update with the correct pa

# Encode the 'type' column (assuming it's categorical - 'red' and 'white'
label_encoder = LabelEncoder()
wine_data['type'] = label_encoder.fit_transform(wine_data['type'])  # 0 f

# Separate the features and target variable (quality)
X = wine_data[['type', 'fixed acidity', 'volatile acidity', 'citric acid'
               'chlorides', 'free sulfur dioxide', 'total sulfur dioxide'
               'sulphates', 'alcohol']]  # Features
y = wine_data['quality']  # Target variable

# Handle missing values in the features by imputing them with the mean
imputer = SimpleImputer(strategy='mean')
X_imputed = imputer.fit_transform(X)

# Scale the features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X_imputed)

# Split the dataset into training and testing sets (80% training, 20% tes
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_siz

# Create a linear regression model
model = LinearRegression()

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

# Make predictions using the testing data
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(f"Mean Squared Error: {mse}")
print(f"R^2 Score: {r2}")

# Take user input for prediction
wine_type = input("Enter wine type (red/white): ").strip().lower()
fixed_acidity = float(input("Enter fixed acidity: "))
volatile_acidity = float(input("Enter volatile acidity: "))
citric_acid = float(input("Enter citric acid: "))
residual_sugar = float(input("Enter residual sugar: "))
chlorides = float(input("Enter chlorides: "))
free_sulfur_dioxide = float(input("Enter free sulfur dioxide: "))
```

```python
total_sulfur_dioxide = float(input("Enter total sulfur dioxide: "))
density = float(input("Enter density: "))
pH = float(input("Enter pH: "))
sulphates = float(input("Enter sulphates: "))
alcohol = float(input("Enter alcohol content: "))

# Encode the type (0 for white, 1 for red)
wine_type_encoded = 1 if wine_type == 'red' else 0

# Create a DataFrame with the user's input
new_data = pd.DataFrame({
    'type': [wine_type_encoded],
    'fixed acidity': [fixed_acidity],
    'volatile acidity': [volatile_acidity],
    'citric acid': [citric_acid],
    'residual sugar': [residual_sugar],
    'chlorides': [chlorides],
    'free sulfur dioxide': [free_sulfur_dioxide],
    'total sulfur dioxide': [total_sulfur_dioxide],
    'density': [density],
    'pH': [pH],
    'sulphates': [sulphates],
    'alcohol': [alcohol]
})

# Impute and scale the new input data
new_data_imputed = imputer.transform(new_data)  # Impute missing values
new_data_scaled = scaler.transform(new_data_imputed)

# Predict the wine quality for the given input
predicted_quality = model.predict(new_data_scaled)
print(f"Predicted Wine Quality: {predicted_quality[0]}")
```

```
Mean Squared Error: 0.47142125991978673
R^2 Score: 0.3404845408986842
Predicted Wine Quality: 10.4468036581198
```

In [ ]: