Implement simple linear regression for the data set ‘student_score.csv’

I. **Evaluate the model by finding**
   a. **Mean Absolute Error**
   b. **Mean Squared Error**
   c. **Root Mean Squared Error**

II. **Plot Linear Graph**

III. **Bar chart for actual and predicted values of the model**

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_absolute_error, mean_squared_error

student = pd.read_csv('student_scores.csv')
X=student.iloc[:,0].values.reshape(-1,1)
y=student.iloc[:,1].values

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

# Initialize the Linear Regression model
model = LinearRegression()

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

# Predict the target values for the test data
y_pred = model.predict(X_test)

# Evaluate the model

# Mean Absolute Error
mae = mean_absolute_error(y_test, y_pred)

# Mean Squared Error
mse = mean_squared_error(y_test, y_pred)
```

```
# Root Mean Squared Error
rmse = np.sqrt(mse)

print(f"Mean Absolute Error: {mae}")
print(f"Mean Squared Error: {mse}")
print(f"Root Mean Squared Error: {rmse}")

# Plot the results
plt.scatter(X, y, color='blue', label='Data points')
plt.plot(X, model.predict(X), color='red', label='Regression Line')
plt.title('Simple Linear Regression')
plt.xlabel('Hours Studied')
plt.ylabel('Dependent Variable (Scores)')
plt.legend()
plt.show()

# Plt the actual and predicted values using bar chart
X_axis = np.arange(len(y_test))
plt.bar(X_axis-0.2, y_test, 0.6, label='Actual')
plt.bar(X_axis+0.2, y_pred, 0.6, label='Predicted')

plt.xlabel("Test Records")
plt.ylabel("Marks")
plt.title("Student Score prediction")
plt.legend()
plt.show()
```