Step 1: Data Collection

You will first need to gather the data that will be used to predict student performance. There are several publicly available datasets that you can use:

- **Student Performance Data Set (UCI Machine Learning Repository)**: This dataset includes features like student grades, study time, failures, and other variables. You can find it here.

- **Student Score Prediction Dataset (Kaggle)**: This dataset includes information such as study hours and previous test scores. You can explore it on Kaggle.

Step 2: Data Preprocessing

The next step is to clean the data and handle any missing values.

Steps for Data Preprocessing:

- **Missing Values:** Check for missing or NaN values in the dataset and decide whether to fill them with mean/median/mode or drop rows/columns.
- **Outliers:** Detect and handle any outliers that may affect model performance.
- **Feature Scaling:** Normalize or standardize the features if needed, especially if features have different scales.
- **Encoding Categorical Data:** If there are categorical columns (like "Gender" or "School"), you may need to use one-hot encoding or label encoding.

Example code for handling missing values and encoding categorical data:

```
import pandas as pdf

# Load dataset

data = pd.read_csv("student_data.csv")

# Handle missing values (if any)

data.fillna(data.mean(), inplace=True)


# One-hot encoding for categorical data (if necessary)

data = pd.get_dummies(data, drop_first=True)


# Check for nulls and handle them

data.isnull().sum()
```

Step 3: Exploratory Data Analysis (EDA)

In this step, you will visualize the dataset to find correlations and better understand how the different features impact the target variable (final grades).

Common Visualizations:

- **Correlation Heatmap:** To see how features relate to each other and the target variable.
- **Scatter Plots:** For visualizing the relationship between study hours and final grade.
- **Box Plots:** To detect outliers in the data.

Example code for visualizations:

```
import seaborn as sns

import matplotlib.pyplot as plt


# Correlation Heatmap

sns.heatmap(data.corr(), annot=True, cmap='coolwarm')

plt.title('Correlation Heatmap')

plt.show()


# Scatter plot of Study Hours vs Final Grade

sns.scatterplot(x='Study_Hours', y='Final_Grade', data=data)

plt.title('Study Hours vs Final Grade')

plt.show()
```

Step 4: Model Building

Now that the data is cleaned and you've explored it, you can build a machine learning model. Since you are asked to use **Linear Regression**, here's how you can proceed.

- **Train-Test Split:** Split the data into training and testing sets.
- **Model Training:** Train the linear regression model on the training set.
- **Prediction:** Use the model to make predictions on the test set.

Example code for building and training the Linear Regression model:

```
from sklearn.model_selection import train_test_split

from sklearn.linear_model import LinearRegression

from sklearn.metrics import mean_squared_error, r2_score


# Splitting the data

X = data.drop('Final_Grade', axis=1)

y = data['Final_Grade']

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
```

Step 5: Model Evaluation

Evaluate the performance of your model using various metrics. Since this is a regression task, you can use:

- **R^2 Score:** This tells you how well your model fits the data.
- **Mean Squared Error (MSE):** The average squared difference between the actual and predicted values.

You can also check for residuals to see if the model has any biases.

Step 6: Deployment

Once the model is trained and evaluated, you can create a simple prediction system for student performance.

Prediction Function Example:

python
Copy

```
def predict_student_performance(study_hours, previous_grade, failures, absences):

    # Create a new data point with the given features

    input_data = pd.DataFrame([[study_hours, previous_grade, failures, absences]],

                              columns=['Study_Hours', 'Previous_Grade', 'Failures',
'Absences'])

    # Make prediction

    predicted_grade = model.predict(input_data)

    return predicted_grade[0]
```

Final Report

For your final submission, create a comprehensive report that includes:

- **Introduction:** Overview of the project, objective, and dataset.
- **Data Preprocessing:** Steps taken for cleaning, handling missing values, and preparing the data.
- **Exploratory Data Analysis (EDA):** Visualizations and insights obtained.
- **Model Building:** Explanation of the Linear Regression model, training, and testing.
- **Model Evaluation:** R^2 Score, MSE, and other evaluation metrics.
- **Conclusion:** Summary of findings and potential improvement.