GitHub and Project Structure Setup:

- GitHub Repository Creation: Begin by creating a new repository on GitHub named "Machine-Learning-Project" to host your project.
- Initialization and README: Initialize the repository with a README file to provide an initial overview of the project.
- Version Control: Utilize Git commands (git init, git add, git commit, git branch, git remote, git push) to manage version control and track changes in your project.
- .gitignore File: Create a .gitignore file to exclude irrelevant files and directories from being tracked by Git.

Setting up the Environment:

- Python Environment Creation: Use Conda or pip to create a virtual Python environment (conda create or pip install) to manage project dependencies and ensure consistency across environments.
- Activation: Activate the Python environment using conda activate to isolate the project from other Python installations and prevent dependency conflicts.

Project Setup:

- Setup Files: Create setup.py and requirements.txt files to manage project setup and dependencies.
- setup.py: Configure the environment for project creation, including defining metadata and dependencies.
- requirements.txt: List all necessary libraries and their versions for installation using pip install -r requirements.txt.

Project Structure, Logging, and Exception Handling:

- Folder Organization: Organize the project structure into directories and files within the src directory, including subfolders such as components, logs, and pipeline.
- Logging: Implement logging functionality using logger.py to record activities and events during project execution.
- Exception Handling: Create exception.py to handle errors and exceptions gracefully during project execution.

Exploratory Data Analysis (EDA):

- Data Exploration: Analyze the dataset on student performance using various statistical methods and visualizations to gain insights into the data.
- Visualization: Utilize histograms, violin plots, scatter plots, and other visualizations to explore relationships between variables and identify patterns in the data.

Model Training:

- Data Preprocessing: Preprocess the dataset by handling missing values, encoding categorical variables, and scaling numerical features.
- Model Selection: Train and evaluate predictive models using regression algorithms such as Linear Regression, Ridge Regression, Decision Tree, Random Forest, etc.
- Evaluation Metrics: Assess model performance using metrics like R2 score, Mean Absolute Error (MAE), and Root Mean Squared Error (RMSE).

Data Ingestion and Transformation:

- Modularization: Modularize data ingestion and transformation tasks into separate Python files (data_ingestion.py and data_transformation.py) for better organization and maintainability.
- Data Splitting: Split the data into training and testing datasets for model training and evaluation.
- Transformation: Apply transformations such as imputation and scaling to prepare the data for modeling.

Model Evaluation and Hyperparameter Tuning:

- Evaluation: Evaluate models using the prepared datasets and performance metrics to assess their predictive accuracy.
- Hyperparameter Tuning: Fine-tune model hyperparameters using techniques like grid search or randomized search to optimize model performance.

Flask Web Application Development:

- App Development: Develop a Flask web application (app.py) to interact with the trained models and provide predictions.
- Frontend Design: Create HTML templates (home.html and index.html) within the templates folder to define the frontend interface of the application.
- Prediction Pipeline: Implement the prediction logic in predict_pipeline.py to preprocess input data and generate predictions.

Deployment in AWS Cloud Using CI/CD Pipelines:

- Elastic Beanstalk Setup: Configure AWS Elastic Beanstalk to host the Flask application.
- Deployment Configuration: Create a .ebextensions folder and a python.config file to specify deployment settings.
- CI/CD Pipeline: Set up a CI/CD pipeline using AWS CodePipeline to automate the deployment process from GitHub to Elastic Beanstalk.