## **Project Day 2 – Design**

**Date:** 13-09-2025

**Name**: **Abdul Samad**

**Title: E-Learning Platform with Auto Evaluation.**

**Objective:**Design the core classes and their interactions for the **E-Learning Platform with Auto Evaluation**.

### **1. Overview**

The system allows students to upload assignments, which are then automatically evaluated.  
 Key focus on **Object-Oriented Design** with clear separation of concerns.

### **2. Core Classes**

#### **Student**

* **Attributes:**
  + studentId : String
  + name : String
  + email : String
  + enrolledCourses : List<Course>
* **Methods:**
  + submitAssignment(Assignment assignment)
  + viewGrades()

#### 

#### **Assignment**

* **Attributes:**
  + assignmentId : String
  + courseId : String
  + title : String
  + description : String
  + submissionFilePath : String
  + submissionDate : LocalDateTime
* **Methods:**
  + getDeadline()
  + isLate()

#### **EvaluationEngine**

* **Responsibilities:** Automatically evaluate uploaded assignments.
* **Attributes:**
  + evaluationRules : List<EvaluationRule>
* **Methods:**
  + evaluate(Assignment assignment) : EvaluationResult
  + generateReport(EvaluationResult result)

### **3. Relationships**

* **Student ↔ Assignment:** One-to-many (a student can submit many assignments).
* **Assignment ↔ EvaluationEngine:** EvaluationEngine consumes Assignment objects to generate EvaluationResult.

### **4. Design Diagram (Textual/UML)**

+-------------+ +-------------+ +-----------------+

| Student |-----> | Assignment | -----> | EvaluationEngine|

+-------------+ +-------------+ +-----------------+

* Student creates Assignment.
* EvaluationEngine evaluates Assignment.

+-----------------------------------------------------+

| Student |

+-----------------------------------------------------+

| - studentId : String |

| - name : String |

| - email : String |

| - enrolledCourses : List<Course> (optional) |

+-----------------------------------------------------+

| + submitAssignment(a : Assignment) : void |

| + viewGrades() : List<EvaluationResult> |

+-----------------------------------------------------+

1 submits \*

-------------------

| |

v ^

+-----------------------------------------------------+

| Assignment |

+-----------------------------------------------------+

| - assignmentId : String |

| - title : String |

| - description : String |

| - studentId : String |

| - filePath : String |

| - submissionTime : LocalDateTime |

| - deadline : LocalDateTime |

+-----------------------------------------------------+

| + isLate() : boolean |

| + getFile() : File |

+-----------------------------------------------------+

1 evaluates \*

-------------------

| |

v ^

+-----------------------------------------------------+

| EvaluationEngine |

+-----------------------------------------------------+

| - evaluationRules : List<EvaluationRule> |

+-----------------------------------------------------+

| + evaluate(a : Assignment) : EvaluationResult |

| + generateReport(r : EvaluationResult) : String |

+-----------------------------------------------------+

produces

|

v

+-----------------------------------------------------+

| EvaluationResult |

+-----------------------------------------------------+

| - assignmentId : String |

| - score : int |

| - remarks : String |

+-----------------------------------------------------+

| + getSummary() : String |

+-----------------------------------------------------+

### **Relationships**

* **Student → Assignment:** *One-to-Many* (a student can submit many assignments).
* **Assignment → EvaluationEngine:** *Many-to-One* (engine evaluates each assignment).
* **EvaluationEngine → EvaluationResult:** *One-to-One per evaluation* (produces a single result object).

### **Interaction Flow**

1. **Student** calls submitAssignment(), creating a new **Assignment** record.
2. **EvaluationEngine** picks up the assignment and runs evaluate().
3. An **EvaluationResult** is generated and stored for later retrieval by the student.

**5. Technology Choices**

* **Language:** Java
* **Storage:** MySQL (for user and assignment metadata)
* **Frameworks:** Spring Boot (optional for API layer)