# E-Learning Platform with Auto Evaluation

Title: Day 1 – Requirements for E-Learning Platform with Auto Evaluation

Date: 13 Sept 2025

Name: Abdul Samad

## Problem Statement

Current e-learning platforms often require manual evaluation, which is time-consuming for instructors and delays feedback for students. The E-Learning Platform with Auto Evaluation will automate the evaluation process for programming assignments, providing instant feedback and freeing up instructors to focus on more complex tasks. The platform will include features for user registration, course enrollment, assignment submission, and automated grading.

### Background

As a student, I want to submit my programming assignments and receive immediate feedback on my code so I can learn from my mistakes quickly and efficiently. As an instructor, I want to create and manage courses and assignments, and have my students' submissions graded automatically so I can dedicate more time to teaching and student support. This project will use Java Full Stack technologies, with MySQL to store user and course metadata.

### Functional Requirements

#### Text User Interface (TUI)

* All options (Student Registration, Student Login, Enroll in Course, Submit Assignment, View Grades, Exit) must be available in a menu-driven TUI.
* Users will interact with the system by selecting numbered options from the console.

#### Student Registration

* **Purpose:** Allow a new student to create an account.
* **Fields:** Full name, unique email, password (entered twice), createdAt timestamp (system-generated).
* **Validation:**
  + Name must not be empty.
  + Email must be unique and valid.
  + Password must meet security rules (length, complexity).
  + Confirm password must match.
* **System Action:** Store student information in a MySQL students table with the password hashed using bcrypt.
* **Acceptance:**
  + Success → "Thank you, Abdul Samad — your account has been created successfully."
  + Failure → an error message (e.g., "Email already exists.").

#### Student Login

* **Purpose:** Authenticate an existing student.
* **Fields:** Email, password.
* **Validation:**
  + Email must exist.
  + Password must match the bcrypt hash.
* **System Action:**
  + On success → Greet the student and show the main menu.
* **Acceptance:**
  + Success → "Login successful. Hello, Abdul Samad."
  + Failure → "Invalid email or password."

#### Enroll in Course

* **Purpose:** Allow a student to enroll in an available course.
* **Fields:** Course ID.
* **Validation:**
  + Course ID must be valid and available.
* **System Action:** Insert an enrollment record into a MySQL enrollments table, linking the student to the course.
* **Acceptance:**
  + Success → "Successfully enrolled in the course."
  + Failure → an error message (e.g., "Course not found.").

#### Submit Assignment

* **Purpose:** Allow students to submit a programming assignment.
* **Fields:** Assignment ID, File Path to the code.
* **Validation:**
  + The student must be enrolled in the course associated with the assignment.
  + The assignment must be open for submission.
* **System Action:**
  + Save the submitted code file.
  + Trigger the auto-evaluation engine to grade the submission.
* **Acceptance:**
  + Success → "Assignment submitted successfully. Your submission is being evaluated."

#### View Grades

* **Purpose:** Allow a student to view their grades for submitted assignments.
* **System Action:** Query the MySQL grades table to retrieve and display all grades for the logged-in student.
* **Acceptance:**
  + The console displays a list of grades, including the assignment name, score, and feedback.
  + If no grades are found → "No grades found for your account."

#### Exit Application

* **Purpose:** Provide a clean way to close the application.
* **System Action:** Close DB connections and terminate the program.
* **Acceptance:**
  + Success → "Goodbye, Abdul Samad."

### Non-Functional Requirements

* **Usability:** The system must use a simple, menu-driven TUI with clear prompts and error messages.
* **Security:** Passwords must be stored using bcrypt. Login failures should be generic to prevent user enumeration.
* **Reliability:** Every operation should confirm success or failure. The system must handle file I/O errors gracefully.
* **Performance:** All menu responses and MySQL queries should complete in less than 1 second.
* **Maintainability:** Use modular code (DAOs, services, models) and externalize database configurations.
* **Portability:** The application must be able to run on any OS with Java installed.

### Conclusion

On Day 1, I gathered and documented the requirements for the E-Learning Platform with Auto Evaluation. I defined the functional and non-functional requirements, including validations and acceptance criteria.