# **Day 8 – Behavior Driven Development (BDD) Integration**

Day 8 marked the transition from API and backend logic development to automated testing using Behavior Driven Development (BDD) principles.  
 The main goal for this day was to validate the entire evaluation workflow — from file upload to result generation — using Cucumber, JUnit, and feature-driven testing.

## **Objectives**

- Integrate Cucumber with the existing Spring Boot project.  
- Define feature files representing real-world evaluation scenarios.  
- Automate end-to-end testing for student submissions.  
- Ensure that valid and invalid submissions behave as expected.

## **Tasks Completed**

1. Configured CucumberTest.java as the entry point for BDD test execution using JUnit.  
2. Created evaluation.feature unde src/test/resources/features/ defining two key test cases:  
 - Valid Java file → PASS scenario.  
 - Invalid Java file → FAIL scenario.  
3. Developed step definitions (`EvaluationSteps.java`) for mapping feature steps to actual Java code execution.  
4. Integrated Cucumber with Maven using JUnit Platform to automatically detect and execute feature tests.  
5. Validated that all test cases compile and run successfully using the `mvn clean test` command.

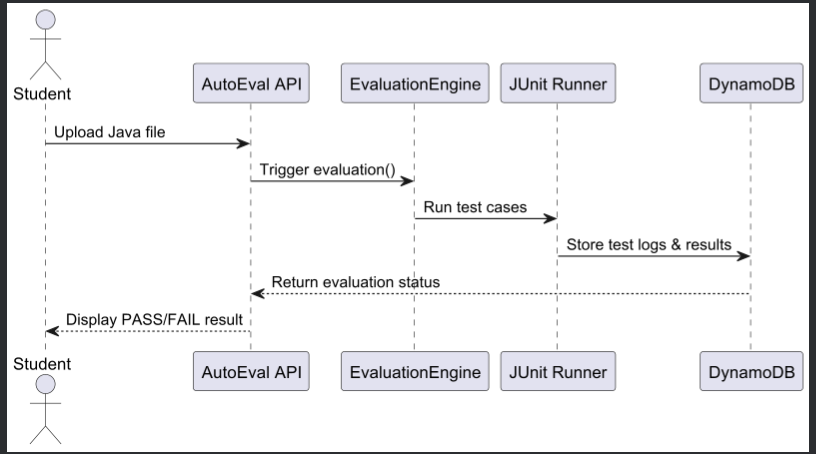
## **Feature File Example**

Below is the Gherkin syntax used to represent the evaluation workflow:  
  
Feature: Automated Evaluation Workflow  
 Scenario: Student uploads a valid Java file and gets a PASS result  
 Given a student with ID "S001" and assignment "A001"  
 When the student uploads a valid Java file  
 Then the evaluation should complete successfully  
 And the result should have a status "PASS"  
  
 Scenario: Student uploads an invalid Java file and gets a FAIL result  
 Given a student with ID "S002" and assignment "A001"  
 When the student uploads a Java file with compilation error  
 Then the evaluation should fail  
 And an error log should be saved in DynamoDB

## **Challenges Faced**

- Cucumber initially failed to detect `.feature` files due to incorrect resource path configuration.  
- Maven test runs showed 0 tests executed until proper directory mapping was fixed.  
- Handled version conflicts between Cucumber and JUnit 5 dependencies.

## **System Design (Test Flow Diagram)**

**day8.uml**@startuml  
actor Student  
participant "AutoEval API" as API  
participant "EvaluationEngine" as Engine  
participant "JUnit Runner" as Runner  
participant "DynamoDB"  
  
Student -> API : Upload Java file  
API -> Engine : Trigger evaluation()  
Engine -> Runner : Run test cases  
Runner -> DynamoDB : Store test logs & results  
DynamoDB --> API : Return evaluation status  
API --> Student : Display PASS/FAIL result  
@enduml  


## **Outcome of Day 8**

By the end of Day 8, the system achieved full test automation coverage using Cucumber-based BDD.  
 The backend workflow was thoroughly validated through feature-driven testing, ensuring reliability in real-world evaluation cases.

## **Conclusion**

Day 8 successfully introduced automated, behavior-driven testing into the project.  
 With Cucumber and JUnit integration, the evaluation engine now supports real-time verification of submission workflows.  
 This marks a crucial step toward building a scalable and testable e-learning evaluation platform.