# **Day 2 – System Design and Architecture**

The main objective of Day 2 was to design the system architecture and define the flow of the E-Learning Auto Evaluation Platform.  
 We analyzed how each module interacts from file uploads to evaluation, logging, and database storage. The focus was to prepare  
 a modular, scalable, and maintainable architecture.

## **Activities Completed**

1. Designed the high-level architecture for the Auto Evaluation Platform.  
2. Created the package structure for different layers — Controller, Service, Repository, Model.  
3. Defined how data flows from user input (file upload) to the database (DynamoDB).  
4. Established service-level responsibilities such as evaluation, logging, and result generation.  
5. Defined exception handling and logging standards.

## **System Architecture**

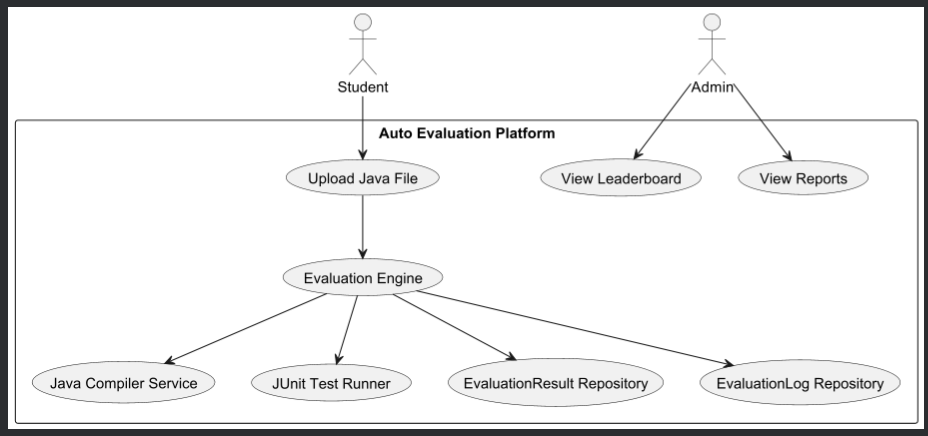
The system follows a layered architecture:  
- **Controller Layer:** Handles REST API requests (e.g., upload, fetch reports).  
- **Service Layer**: Implements business logic (compilation, testing, evaluation).  
- **Repository Layer**: Communicates with AWS DynamoDB for data storage and retrieval.  
- **Model Layer:** Defines data structures such as Submission, EvaluationResult, and EvaluationLog.

## 

## 

## **Architecture Diagram (PlantUML Code)**

**day2.uml**@startuml  
actor Student  
actor Admin  
  
rectangle "Auto Evaluation Platform" {  
 Student --> (Upload Java File)  
 (Upload Java File) --> (Evaluation Engine)  
 (Evaluation Engine) --> (Java Compiler Service)  
 (Evaluation Engine) --> (JUnit Test Runner)  
 (Evaluation Engine) --> (EvaluationResult Repository)  
 (Evaluation Engine) --> (EvaluationLog Repository)  
 Admin --> (View Leaderboard)  
 Admin --> (View Reports)  
}  
@enduml



## 

## 

## 

## **Key Components**

- **EvaluationEngineImpl.java** – Responsible for controlling the compilation, testing, and result generation process.  
- **JavaCompilerService.java** – Handles dynamic compilation of submitted Java files.  
- **JUnitTestRunnerService.java**– Executes test cases and logs results.  
- **EvaluationResultRepository.java** – Manages CRUD operations for evaluation results in DynamoDB.  
- **EvaluationLogRepository.java** – Stores test logs for debugging and transparency.

## **Outcome of Day 2**

By the end of Day 2, the complete system design and architecture were finalized. The modular approach ensures scalability and simplifies  
 future maintenance. Clear separation between layers enables independent testing and easier debugging.

## **Conclusion**

Day 2 successfully established the technical foundation for the Auto Evaluation Platform. With a clear architecture and defined  
 component responsibilities, the project is ready for the development of core functionalities starting from Day 3.