UE21CS352B - Object Oriented Analysis & Design using Java

Mini Project Report

"Computer Based Test Management System"

Submitted by:

Srikrishna R Chitnis	**PES1UG21CS619**
Srinivas Ramakrishnan	**PES1UG21CS624**
Suraj Rao	**PES1UG21CS647**
Suvan Ashwin	**PES1UG21CS651**

6th Semester K Section

Prof. Bhargavi Mokashi
Assistant Professor

January - May 2024

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
FACULTY OF ENGINEERING
PES UNIVERSITY
(Established under Karnataka Act No. 16 of 2013)
100ft Ring Road, Bengaluru – 560 085, Karnataka, India

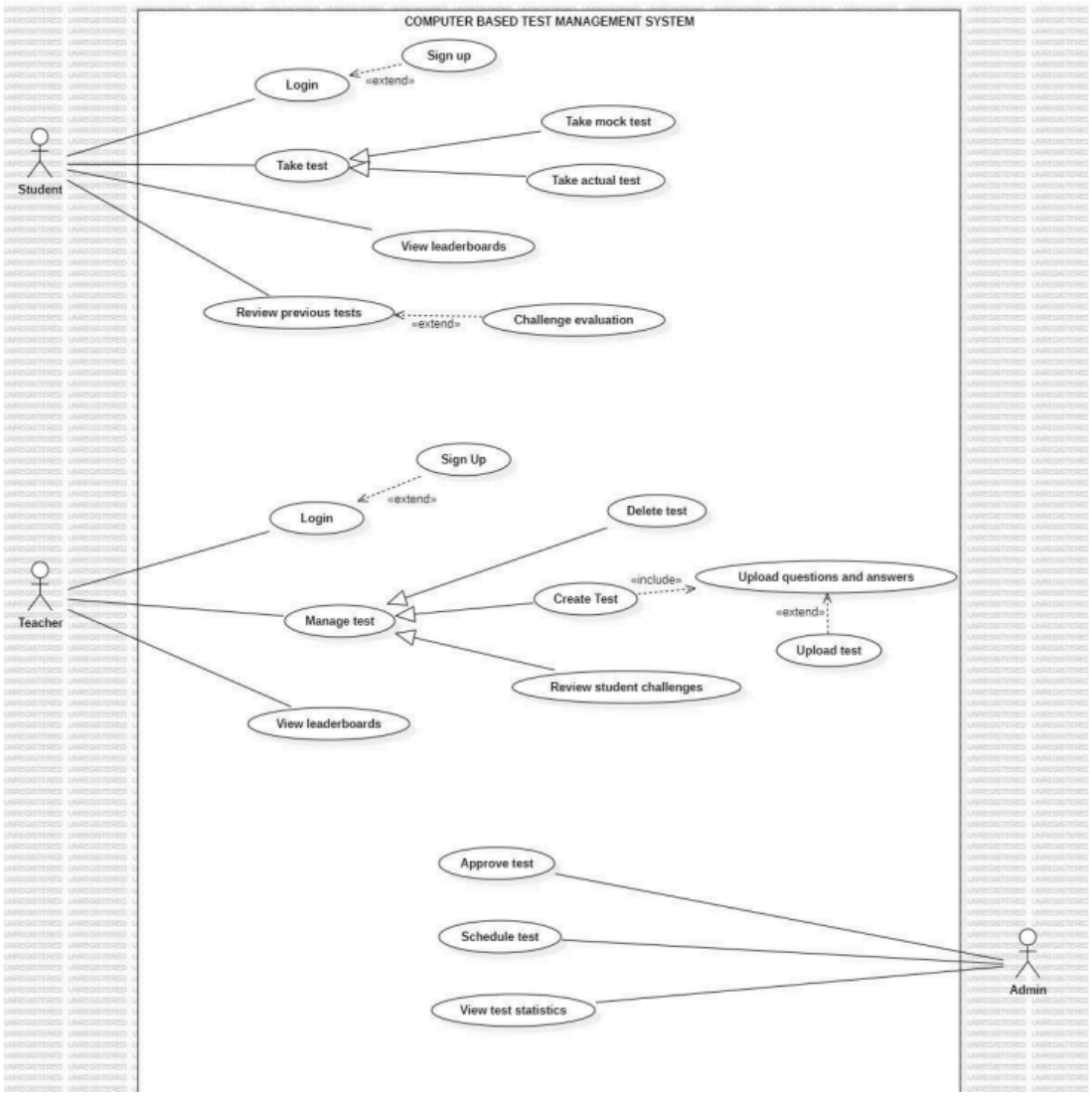
Problem Statement:

Design and develop a comprehensive Computer-Based Test Management System that facilitates the creation, scheduling, and administration of online tests/assessments. The system should cater to the needs of three distinct user roles: Students, Teachers, and Administrators. The primary objectives of the system are:

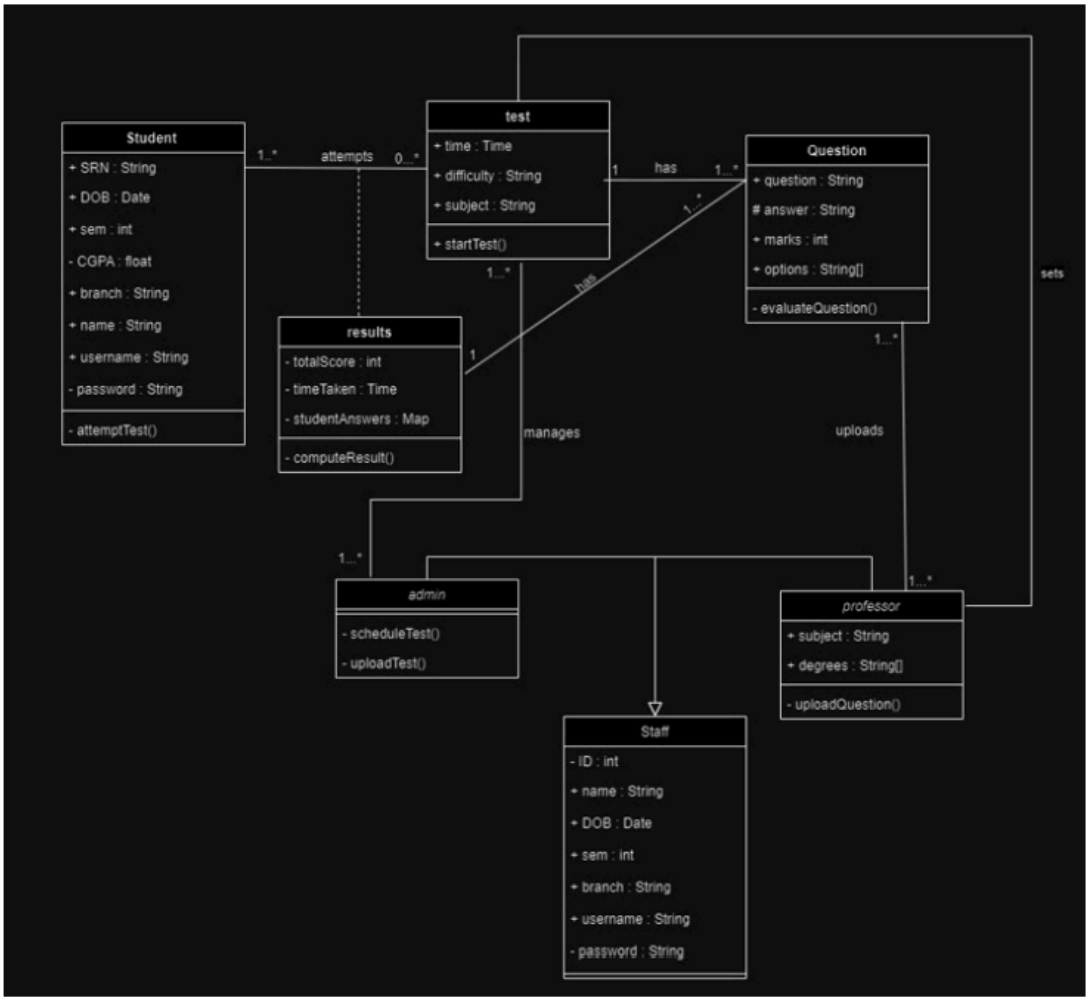
- **Student Module:**
  - **Enable students to view available and previously taken tests.**
  - **Provide a user-friendly interface for taking online tests.**
  - **Allow students to review their previous test submissions and challenge the evaluation if necessary.**
  - **Display a leaderboard to foster healthy competition and motivation among students.**
- **Teacher Module:**
  - **Empower teachers to create, update, and delete test questions and assemble comprehensive tests.**
  - **Facilitate the uploading of test questions.**
  - **Enable teachers to review and respond to student-initiated challenges regarding test evaluations.**
  - **Access the leaderboard for monitoring student performance.**
- **Administrator Module:**
  - **Grant administrators the authority to approve and schedule tests created by teachers.**
  - **Enable administrators to view comprehensive test statistics and analytics for performance monitoring and decision-making.**
  - **Implement a secure login/signup facility for students and teachers to access the system.**

Models:

- USE CASE DIAGRAM :

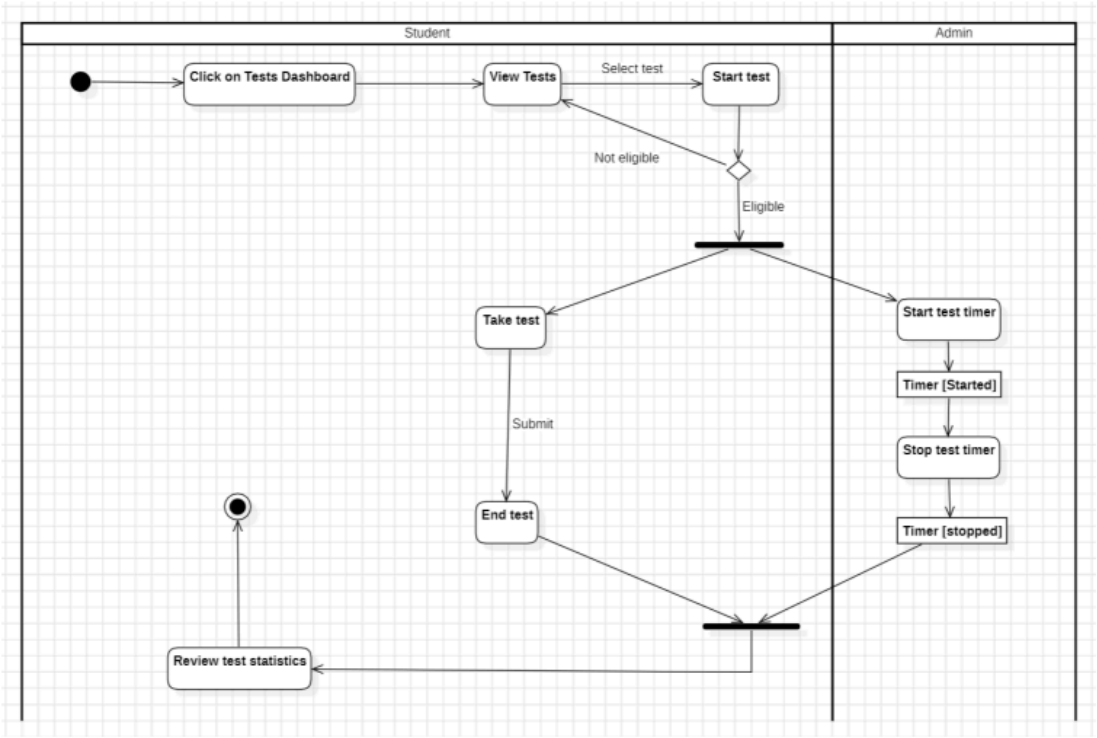


- CLASS DIAGRAM :

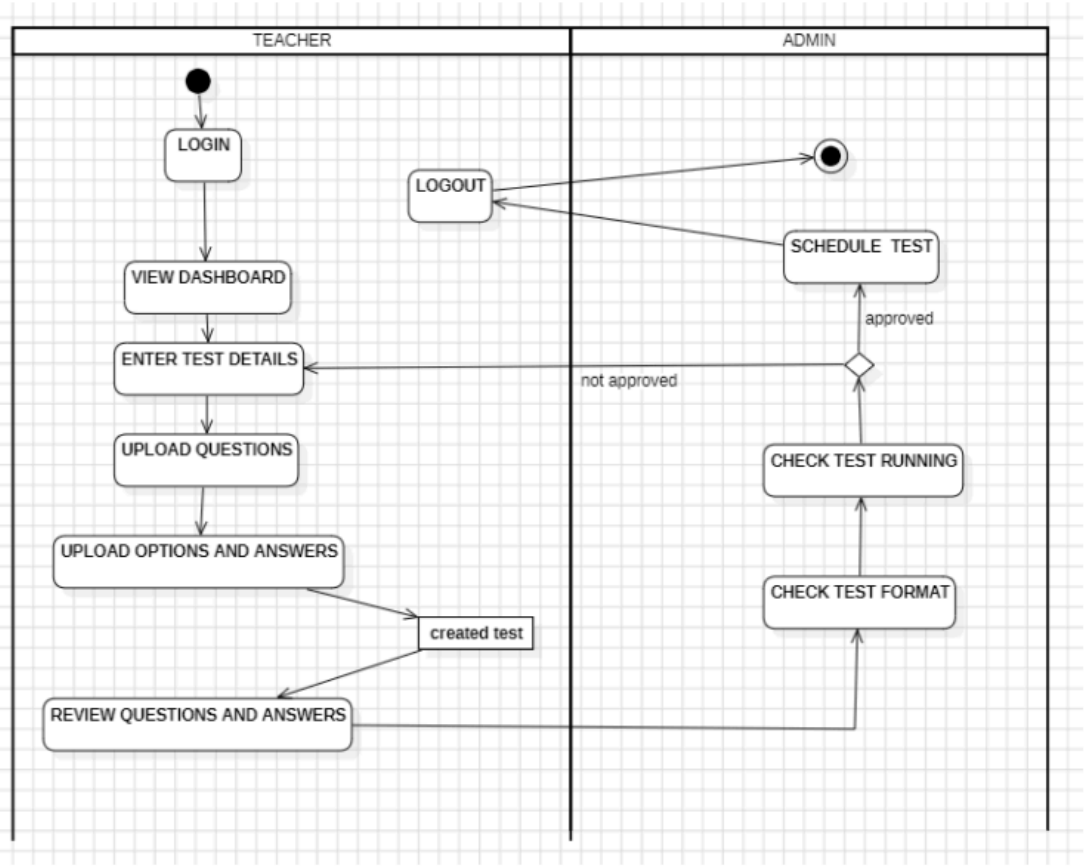


Activity Diagrams:

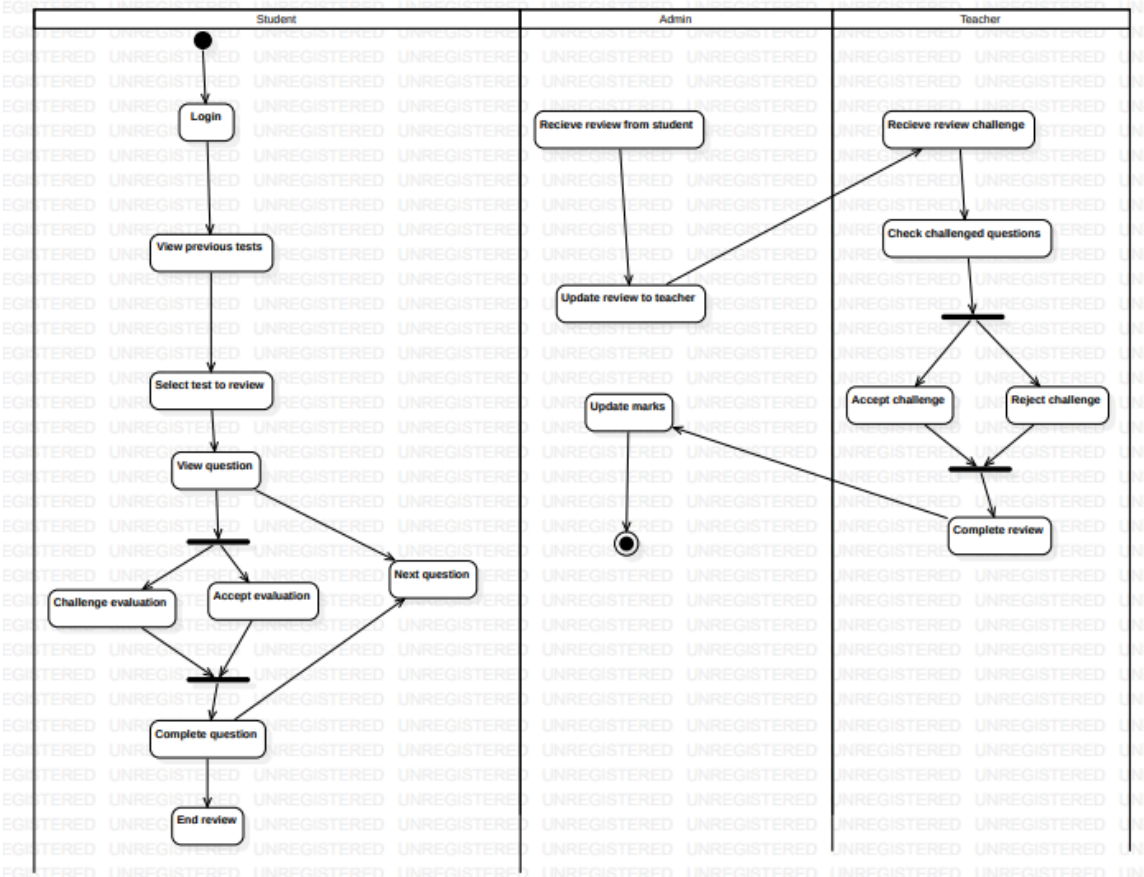
1) Activity Diagram for Taking Test



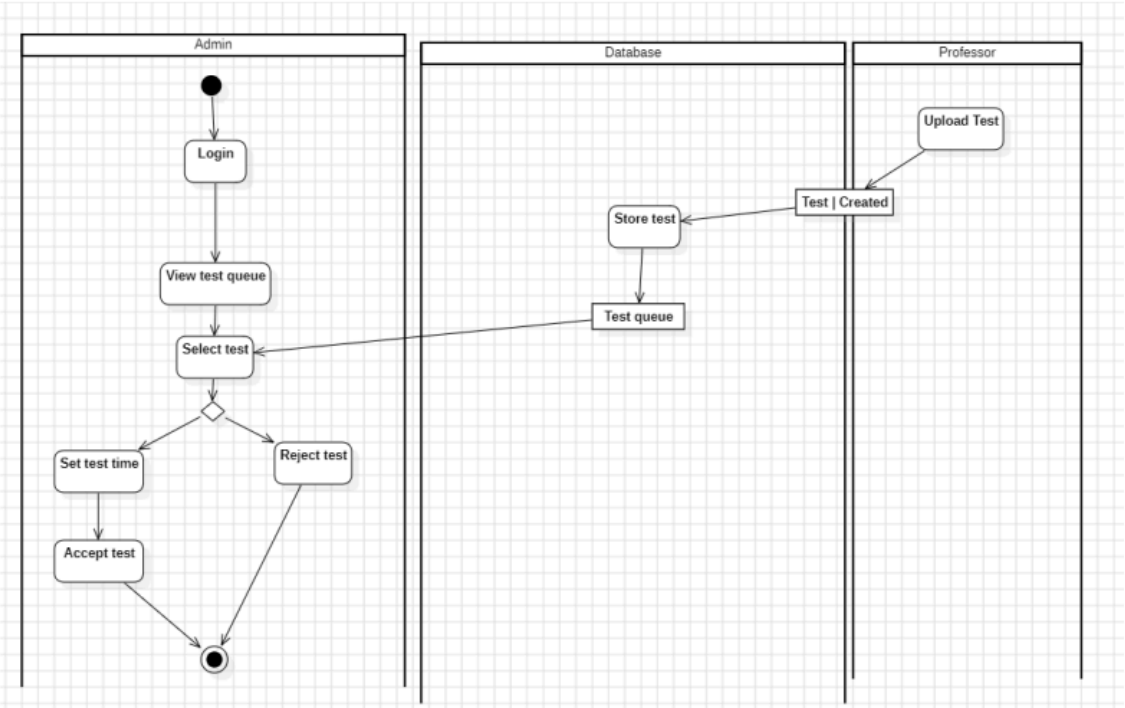
2) Activity Diagram for teacher creating test



3) Activity Diagram for reviewing test

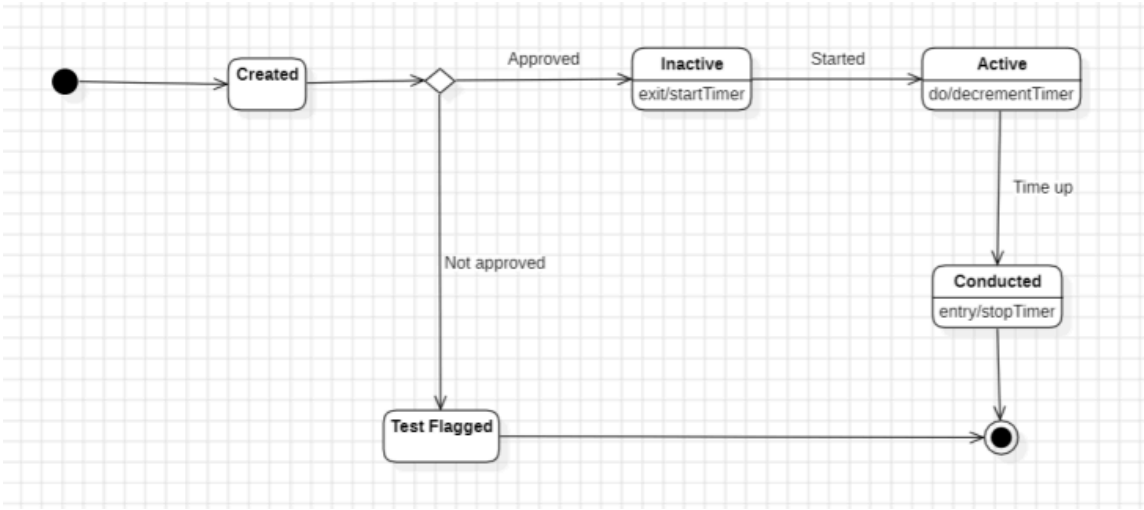


4)Activity Diagram for Admin schedules test

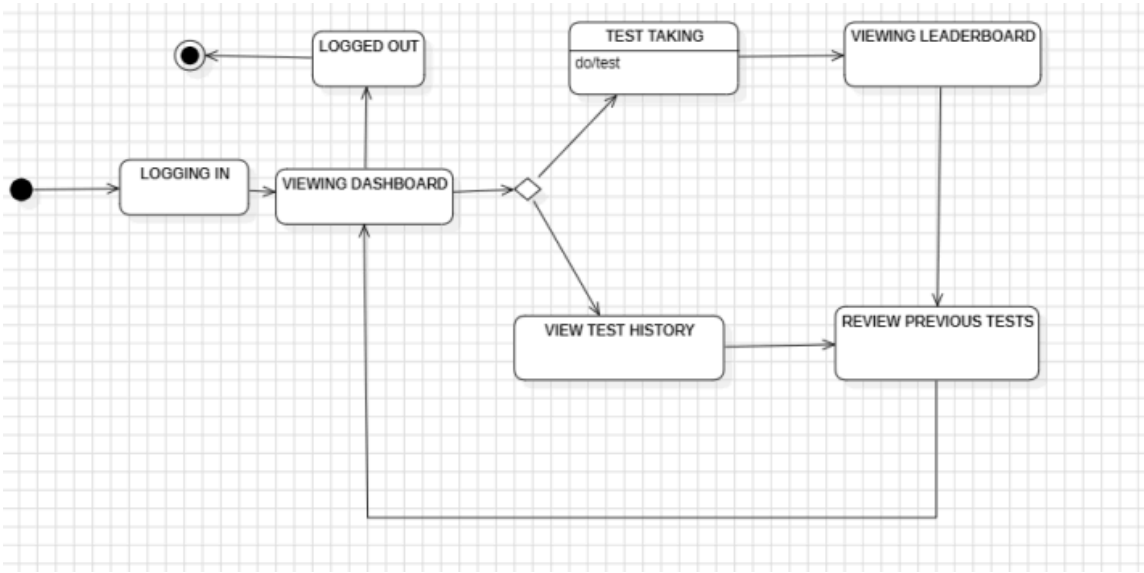


State Diagrams:

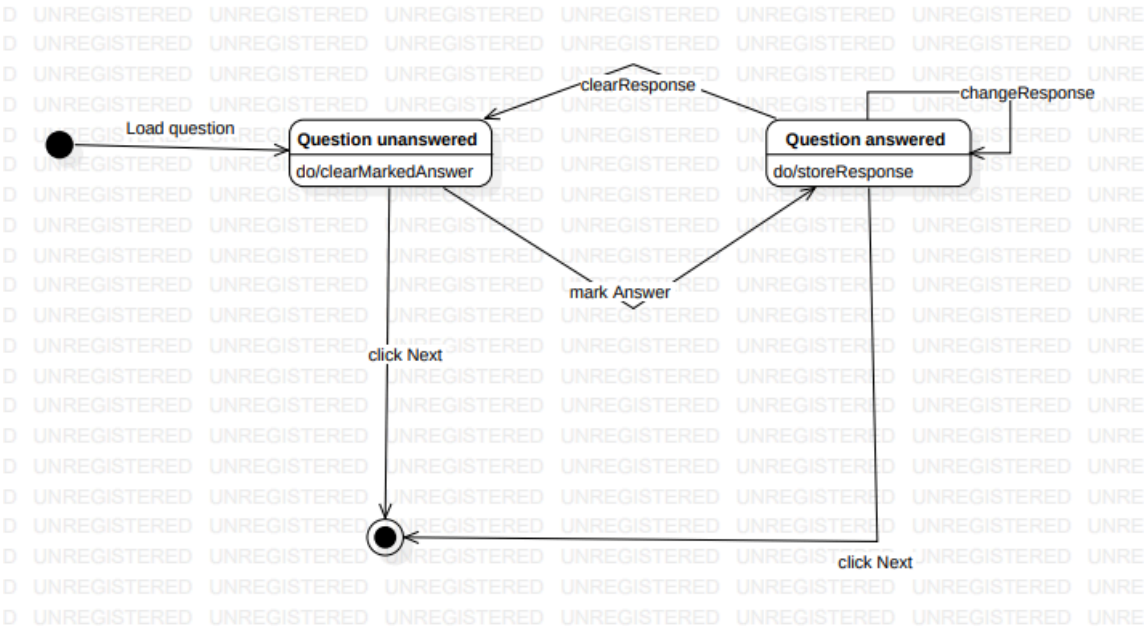
State Diagram for Test Creation:



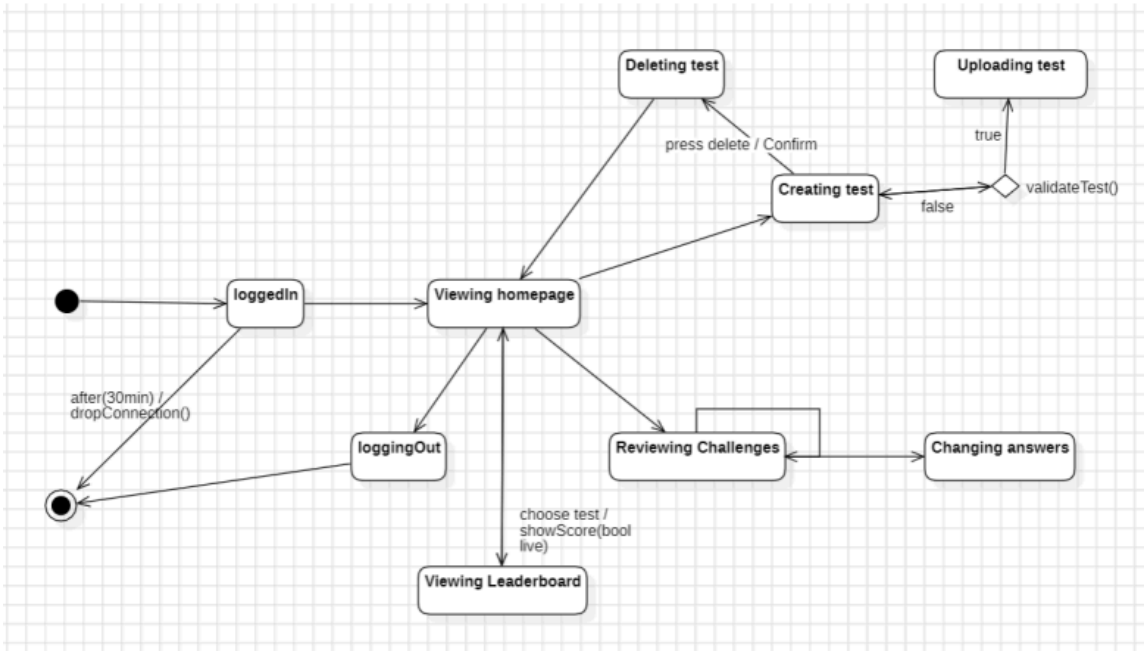
State diagram for Student:



State Diagram for Test taking:



State Diagram for Professor:



Architecture Patterns used:

- **Singleton Pattern (Database Connection):**
  - The Singleton pattern is applied in the database connection module to ensure that only one instance of the database connection is created throughout the application's lifecycle.
  - This ensures efficient resource utilization and prevents multiple connections to the database, which could lead to performance issues.
  - By encapsulating the database connection logic within a Singleton class, we maintain a centralized point of access to the database, enhancing maintainability and scalability.

- **Factory Pattern (Registration Module):**
  - The Factory pattern is utilized in the registration module to facilitate the creation of different types of user objects based on user input.
  - By abstracting the object creation process into a factory class, we promote code flexibility and extensibility.
  - For instance, the registration factory class dynamically creates instances of student or teacher objects based on the user's role selection during the registration process.
  - This decouples the client code from the specific implementations of user objects, allowing for easy addition of new user types in the future.

- **Command Pattern (Admin Module):**
  - The Command pattern is employed in the admin module to encapsulate requests as objects, thereby allowing the parameterization of clients with queues, requests, and operations.
  - Each admin command corresponds to a specific action that an administrator can perform, such as approving tests, scheduling tests, or viewing statistics.
  - By encapsulating these actions within command objects, we promote loose coupling between the invoker (administrator) and the receiver (system components).
  - This enables us to easily extend the system with new admin functionalities without modifying existing code.

Design Principles used:

- **Single Responsibility Principle (SRP):**
  - The SRP is adhered to by designing classes with a single responsibility, focusing on specific use cases or functionalities.
  - For example, separate classes are created for user authentication, test creation, test scheduling, and result generation.
  - This promotes code clarity, maintainability, and reusability, as each class is responsible for a specific task, minimizing the impact of changes and facilitating easier debugging and testing.

- **Open-Closed Principle (OCP):**
    - The OCP is observed by designing abstract classes and interfaces that are open for extension but closed for modification.
    - For instance, the admin module consists of an abstract admin command class that defines the interface for executing admin actions.
    - Concrete command classes extend this abstract class to implement specific admin functionalities.
    - This design allows for the addition of new admin commands without altering existing code, promoting code stability and scalability.

- **Github link:** **https://github.com/SinkableVirus/Online_Test_Management_System**

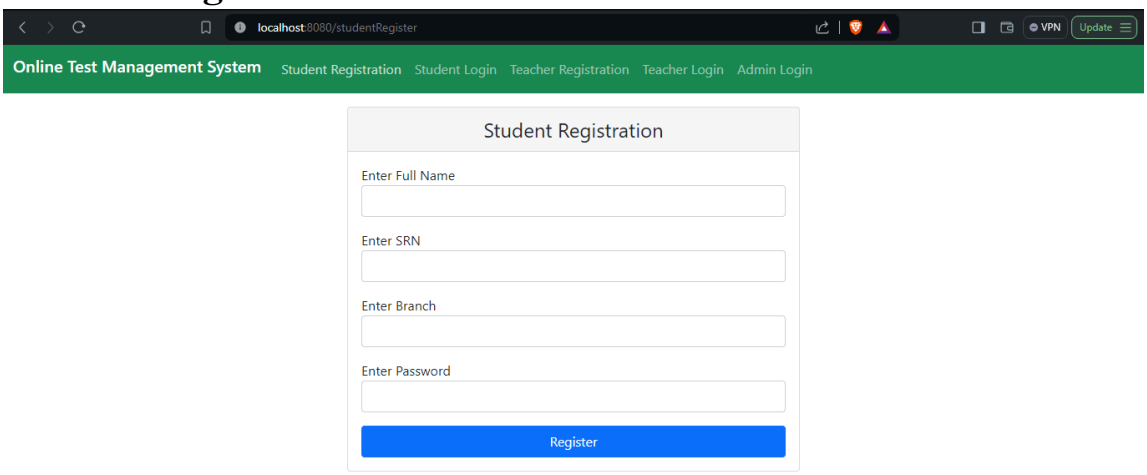
INDIVIDUAL CONTRIBUTIONS :

- Srinivas - Teacher Use case
- Srikrishna - Student Use case
- Suvan - Review Use case and Login In Parts
- Suraj - Admin Use case and leaderboard

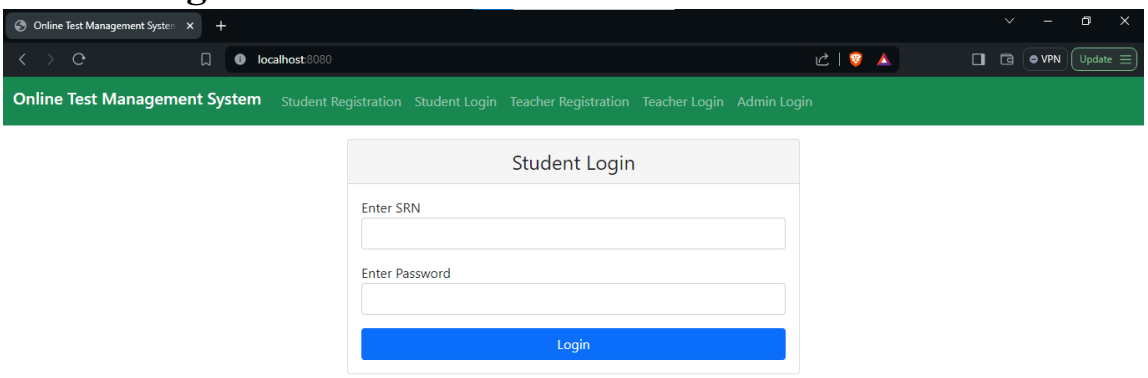
Screenshots :

STUDENT :

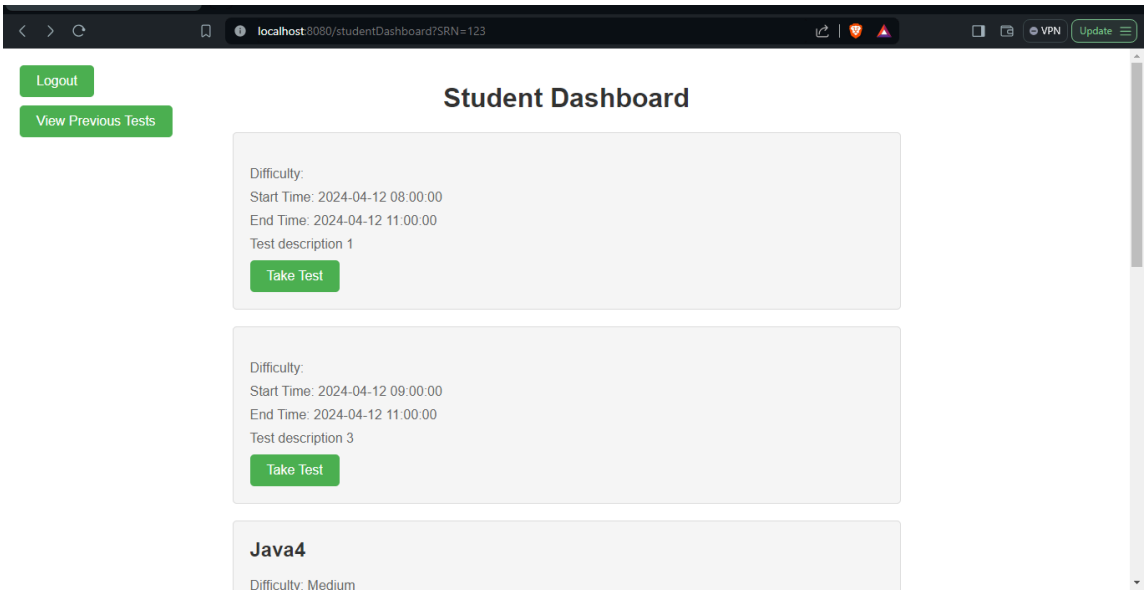
Student Registration :



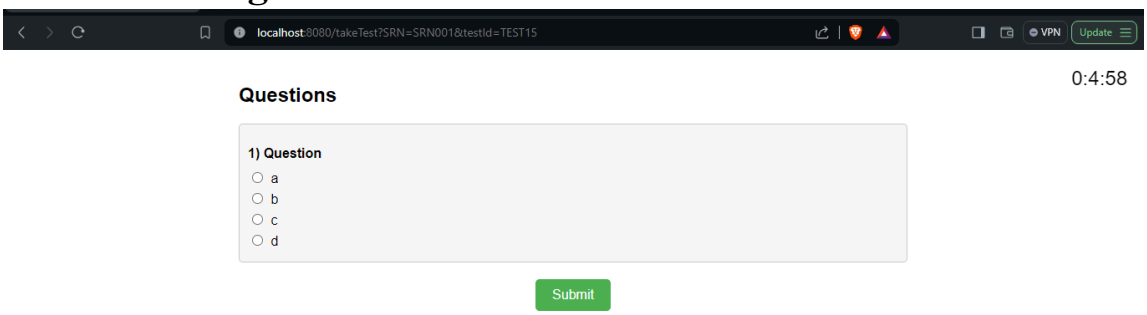
Student login :



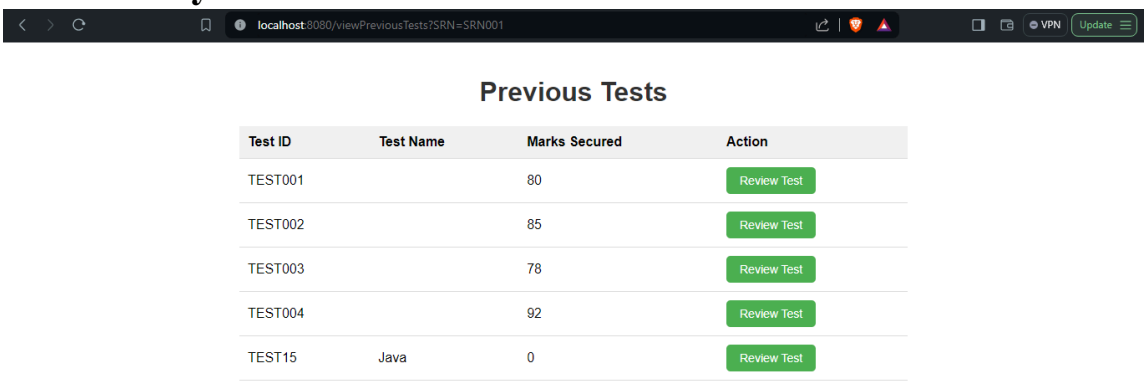
Student dashboard :



Student taking test :



Test history :



REVIEW :
Review Test :

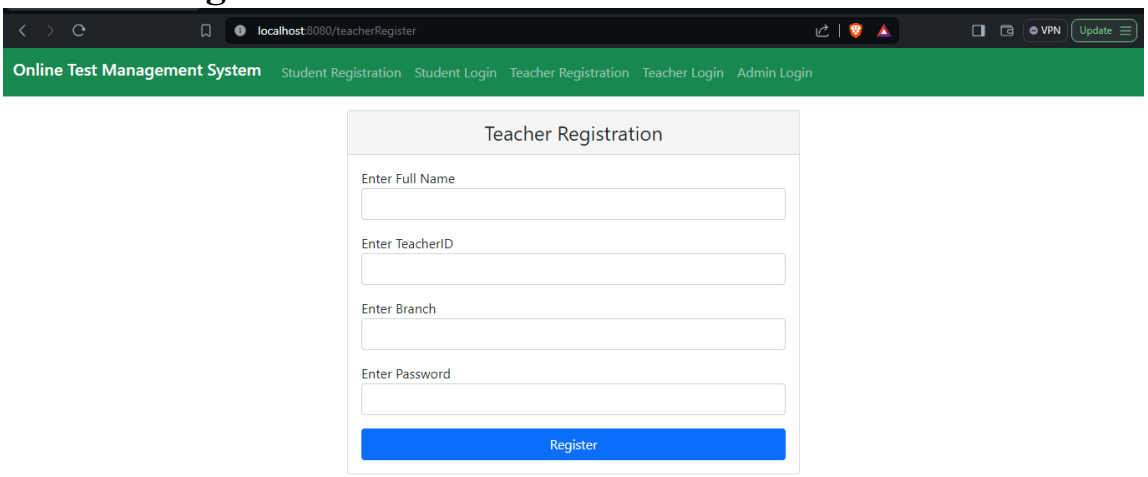
Test Review

Question	Option 1	Option 2	Option 3	Option 4	Marked Answer	Correct Answer	Review Comment
What is the result of "9 + 3 * 2"?	15	18	24	12	a	a	None
Which of the following is a valid way to create an object in Java?	ClassName obj = new ClassName();	ClassName obj = ClassName();	ClassName obj = new ClassName;	None of the above	a	a	None
Which of the following is a Java code editor?	Eclipse	NetBeans	IntelliJ IDEA	All of the above	a	d	scam
What is the purpose of the "static" keyword in Java?	To create a method that can be called without creating an object	To create a variable that is shared among all instances of a class	Both A and B	None of the above	c	c	None
Which of the following is a Java collection interface?	List	Set	Map	All of the above	d	d	None

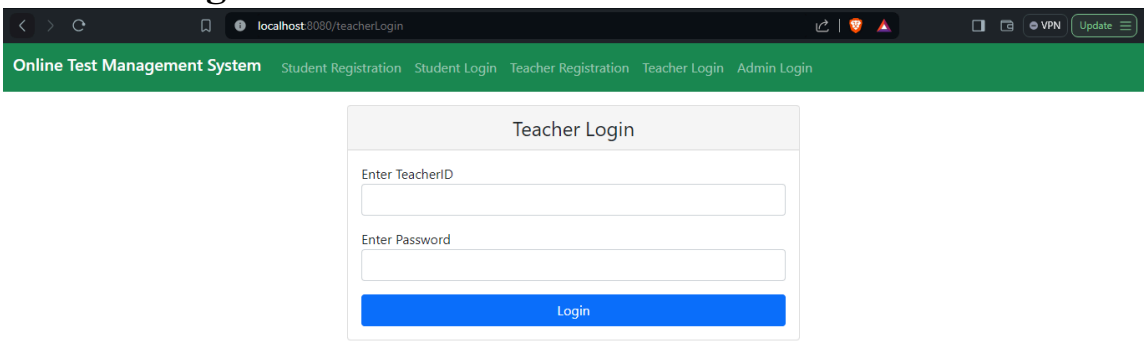
Save Review Comments

TEACHER :

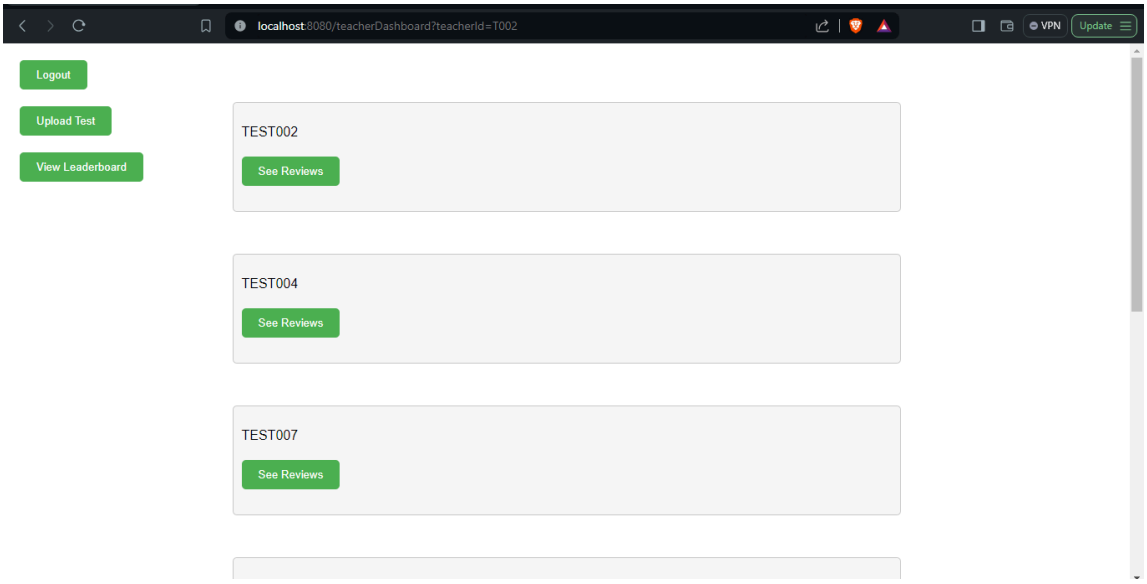
Teacher Registration :



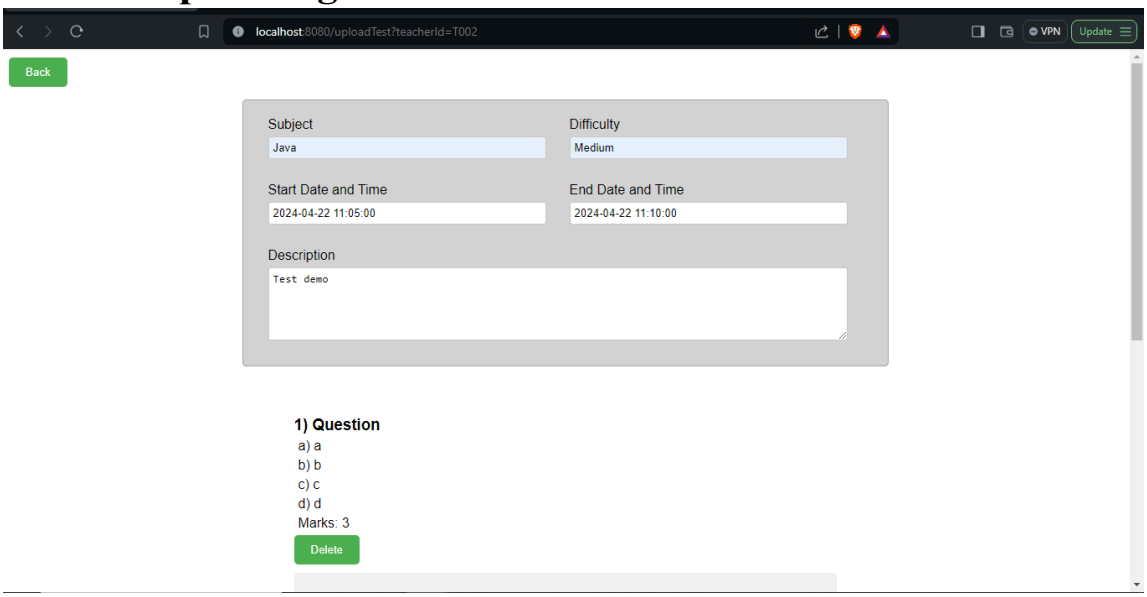
Teacher Login

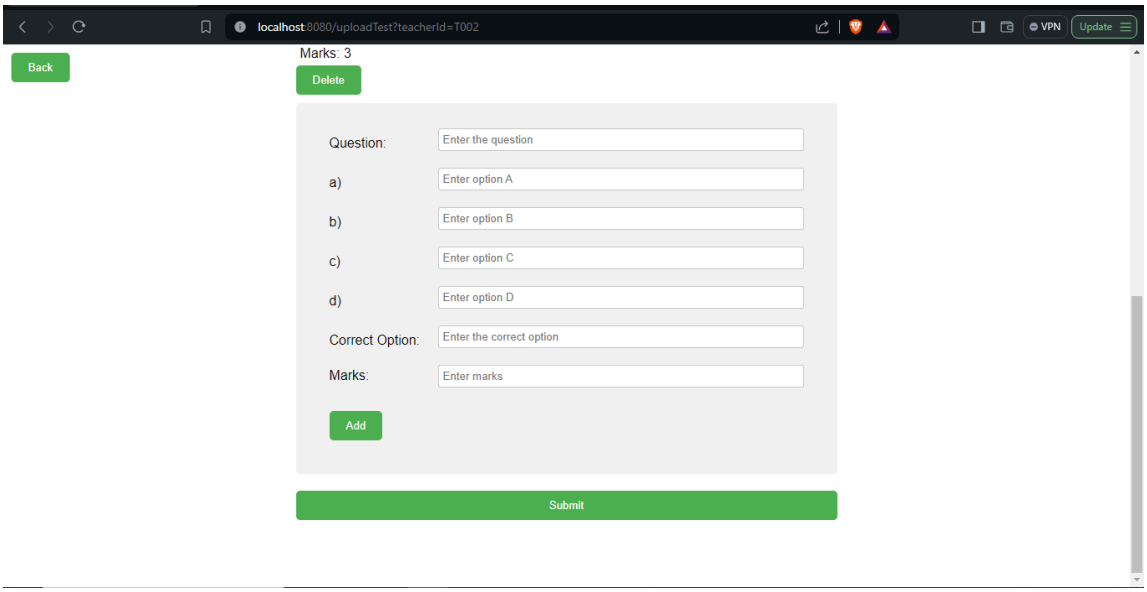


Teacher Dashboard :

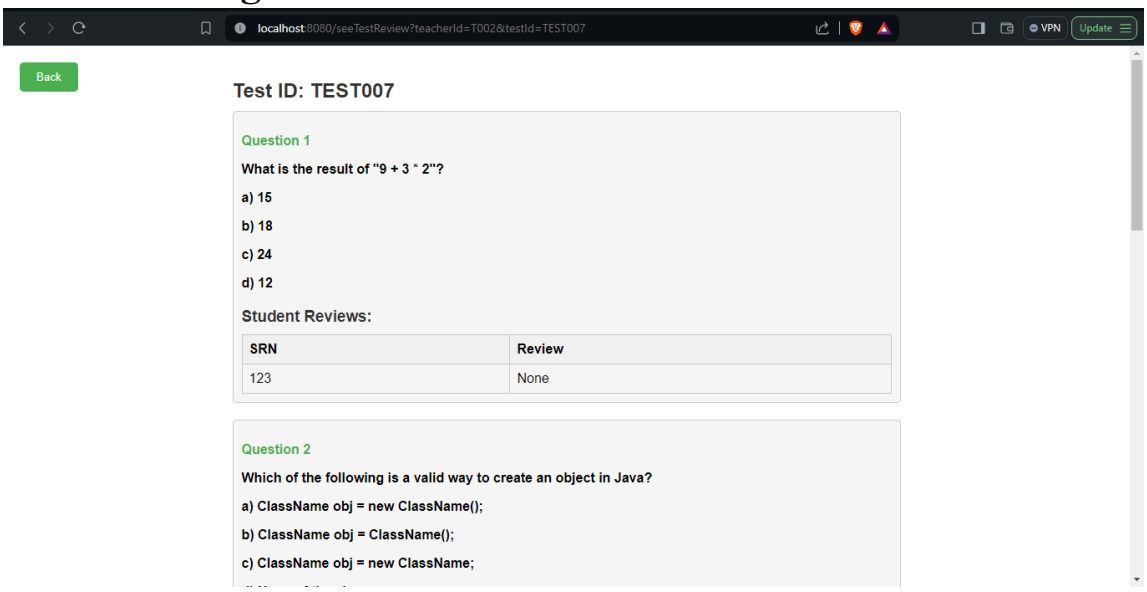


Teacher uploading test :

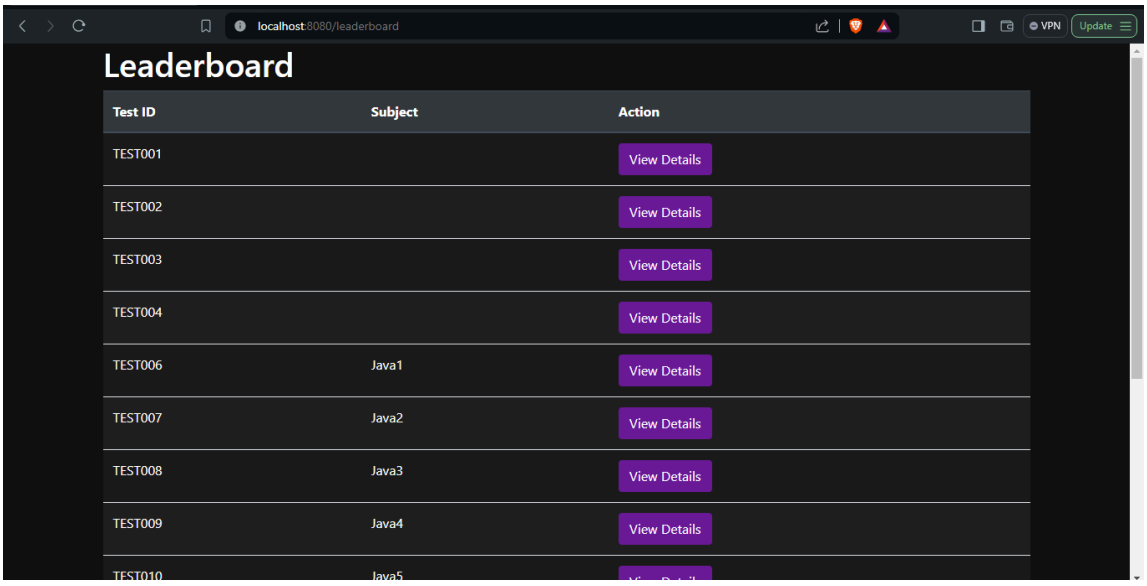




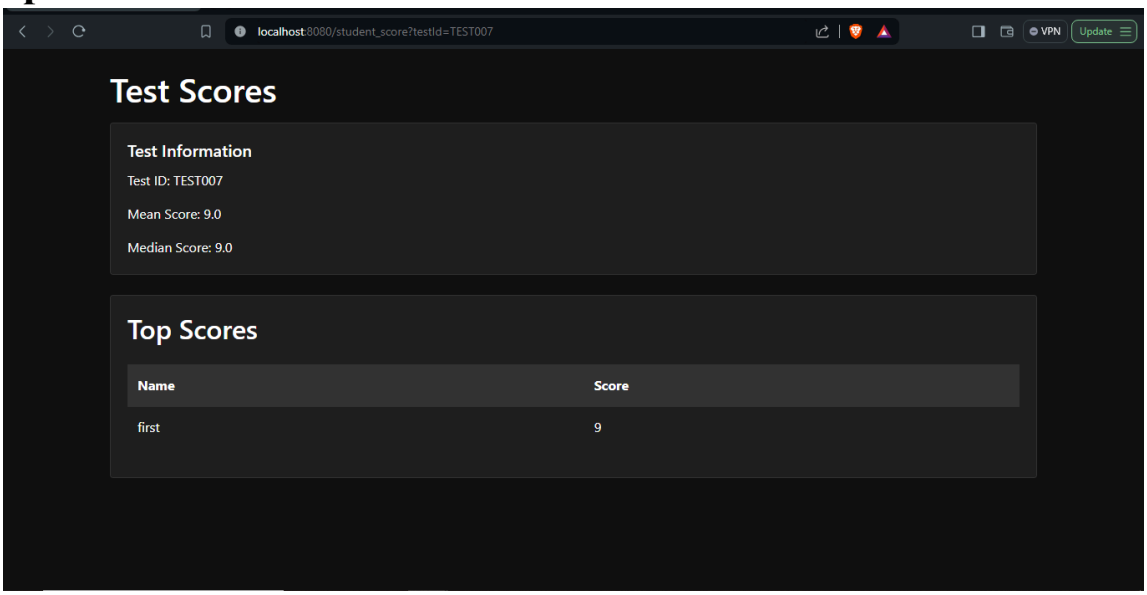
Test reviewing for teacher :



General Leaderboard :

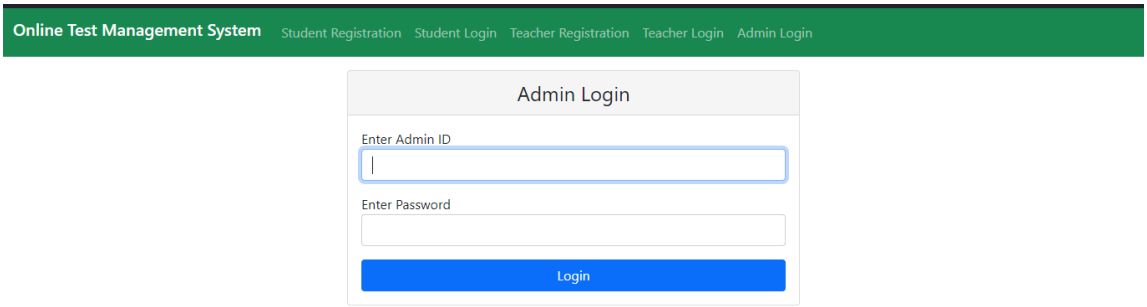


Specific test Leaderboard :

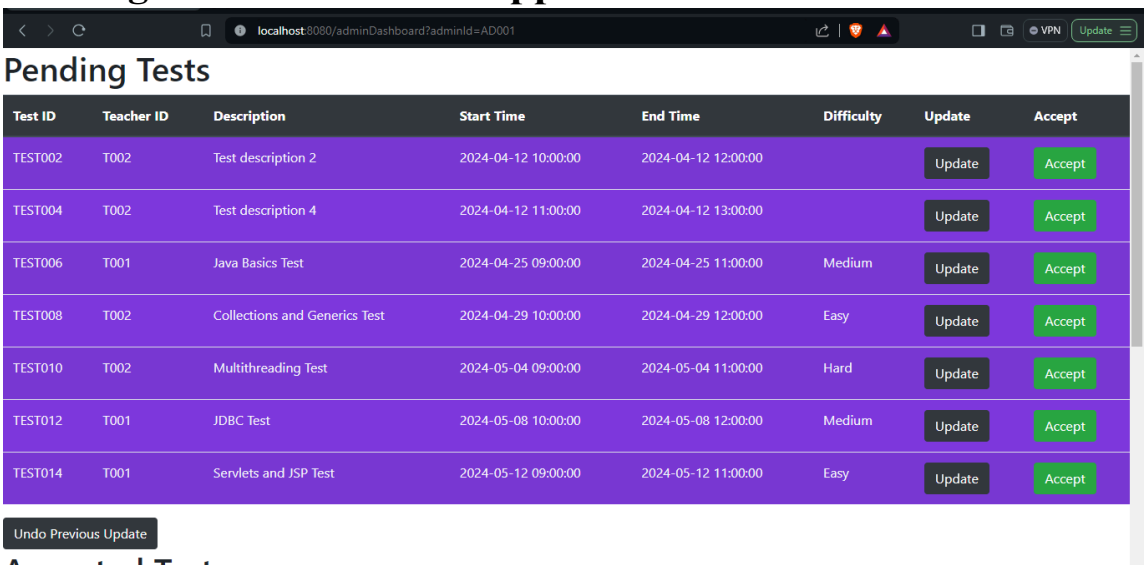


ADMIN :

Admin login :



Pending tests for admin to approve :



Tests accepted by admin :