UE21CS352B - Object Oriented Analysis & Design using Java

Mini Project Report

“Online Coding Evaluation Platform”

Submitted by:

Abhiraj	**PES1UG21CS022**
Advay Agrawal	**PES1UG21CS049**
Akshay Anand	**PES1UG21CS064**
Monisha S	**PES2UG22CS809**

6th Semester A Section
Team Number : A10

Prof. Mahitha G
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

Table of Contents

1. Introduction:

Online coding evaluation platforms have become indispensable tools in the modern tech industry for assessing and honing programming skills. These platforms offer a virtual environment where developers can solve coding challenges, participate in coding competitions, and undergo technical interviews remotely. They provide a scalable and efficient way for companies to evaluate candidates' coding proficiency, identify top talent, and streamline the hiring process.

The existing solutions are:

- HackerRank
- LeetCode
- Codewars

While existing platforms offer valuable resources for practicing coding and technical interviews, developing a new online coding evaluation platform can offer several learning opportunities.

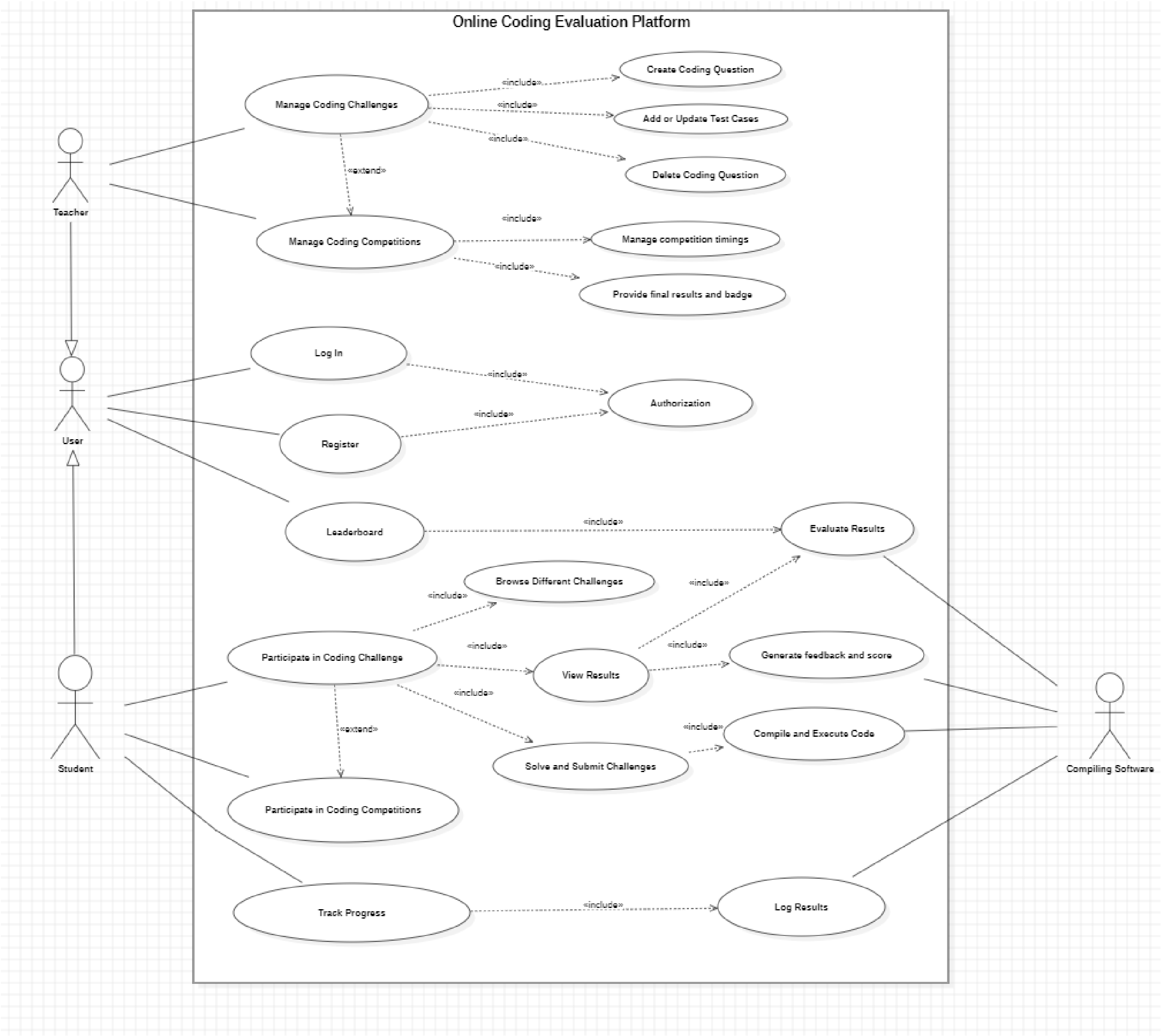
2. Problem Statement:

The problem is to create a platform for students to be able to view different coding challenges, which are competitive questions, and allow them to solve them in the programming language of their choice. The students must be allowed to register to this platform and login successfully on further attempts. They must be able to view the challenges and competitions, which are a collection of challenges. They must be able to solve, submit and receive results for their efforts.

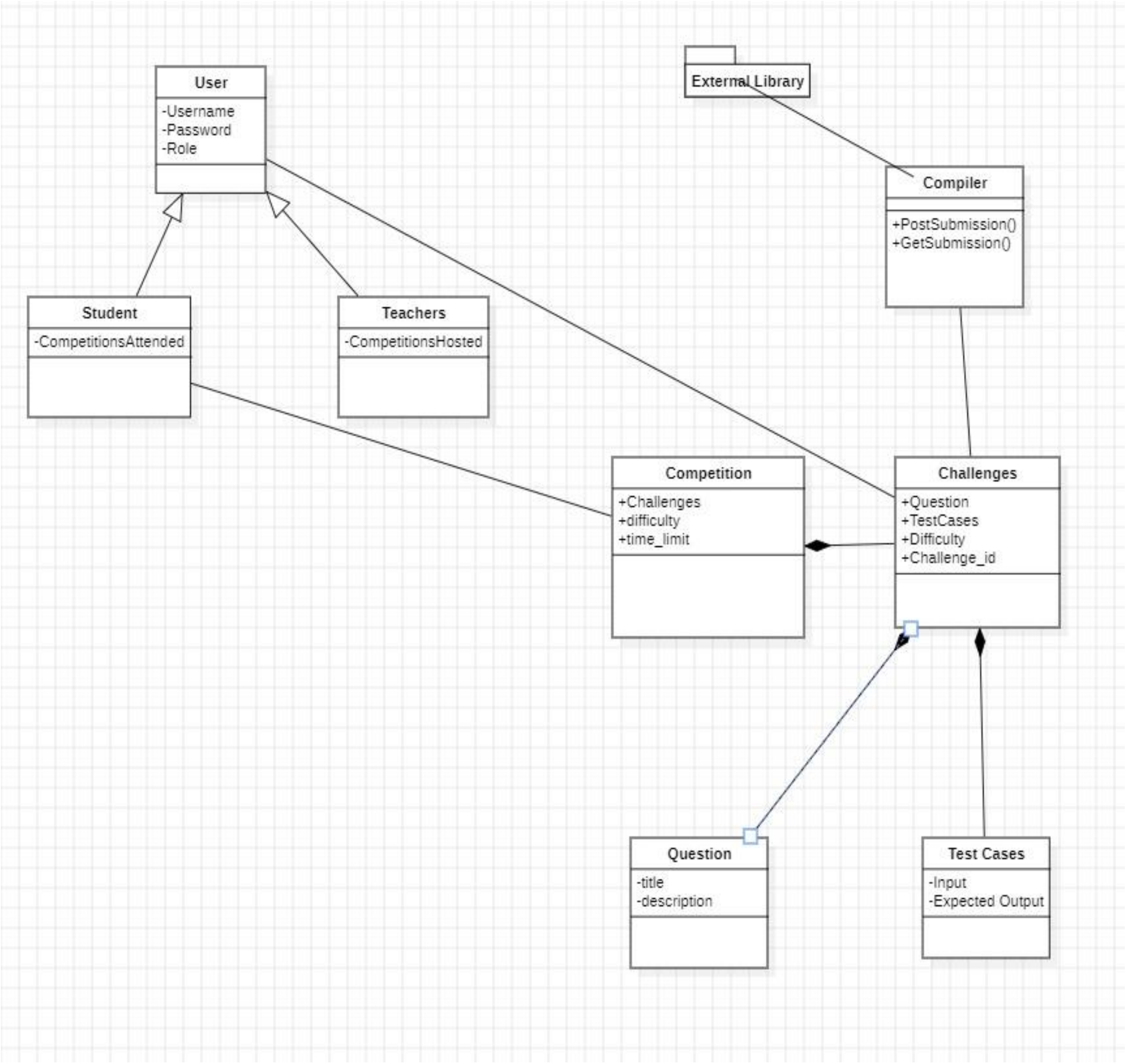
The teachers must also be able to register and login into the platform. But in addition to viewing the challenges and competitions, they must also be able to create the challenges and competitions, which the students can take part in. This will ease the teacher’s job of creating and evaluating assessments.

3. Models

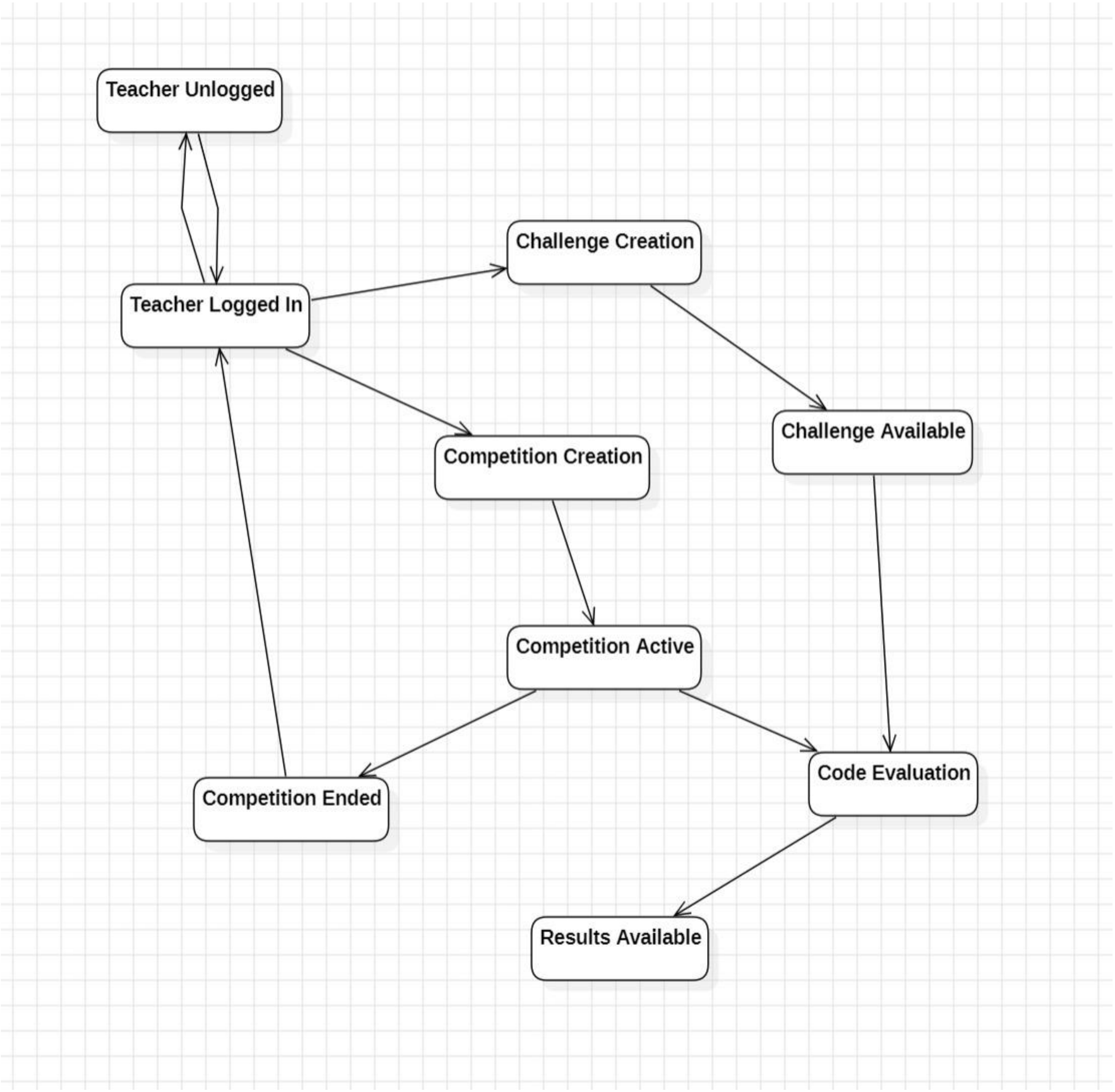
3.1 Use Case Diagram

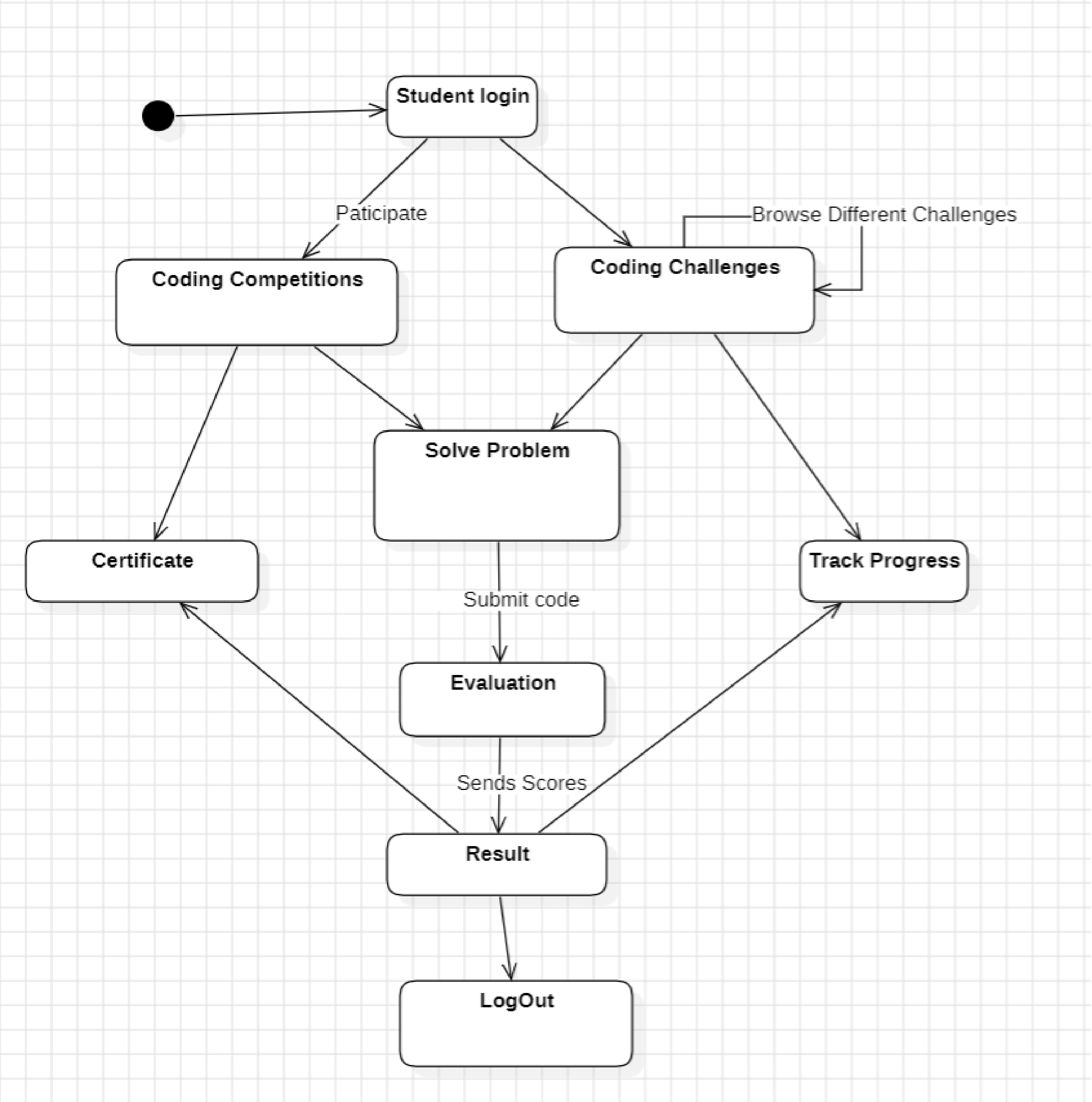


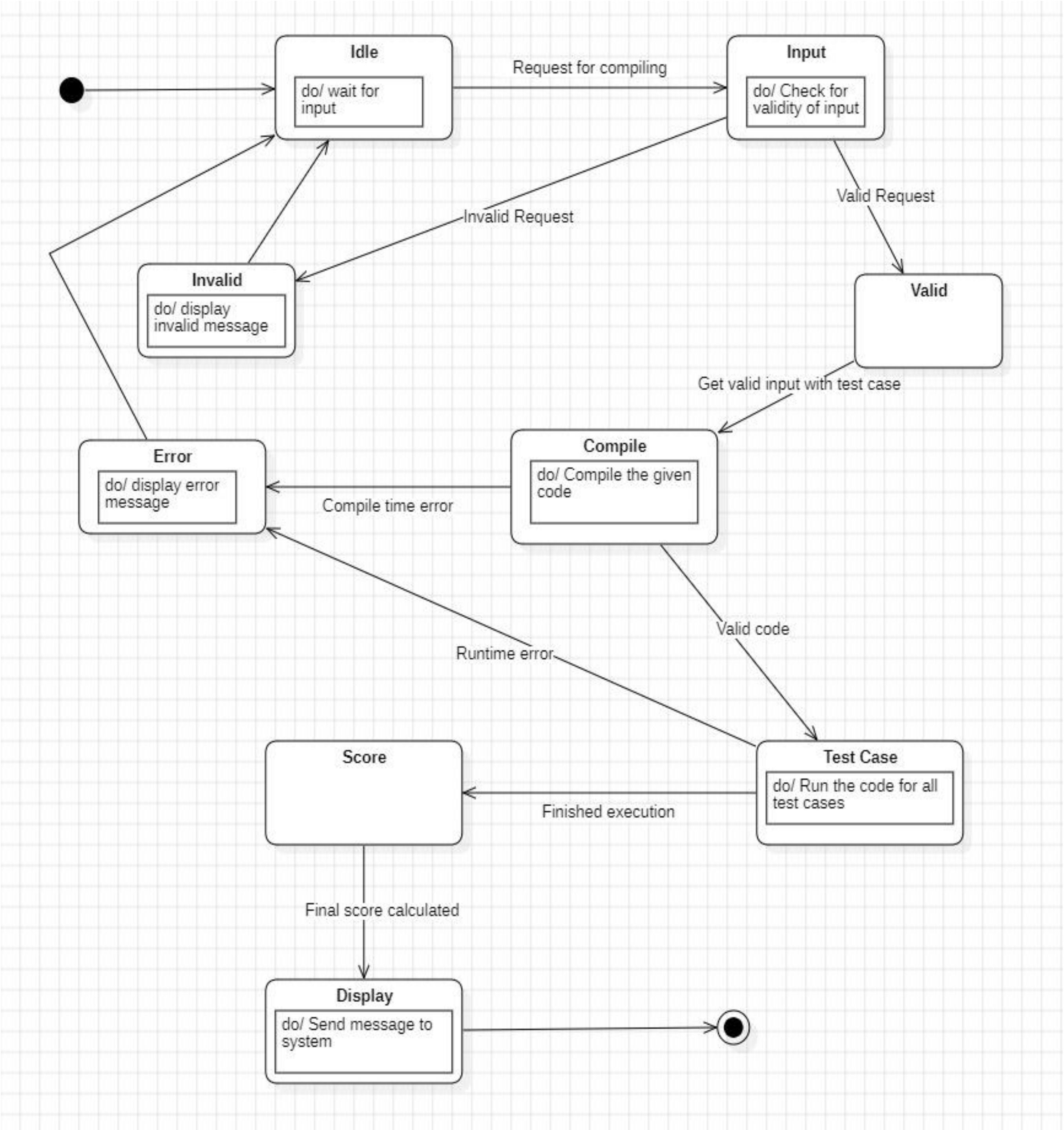
3.2Class Diagram



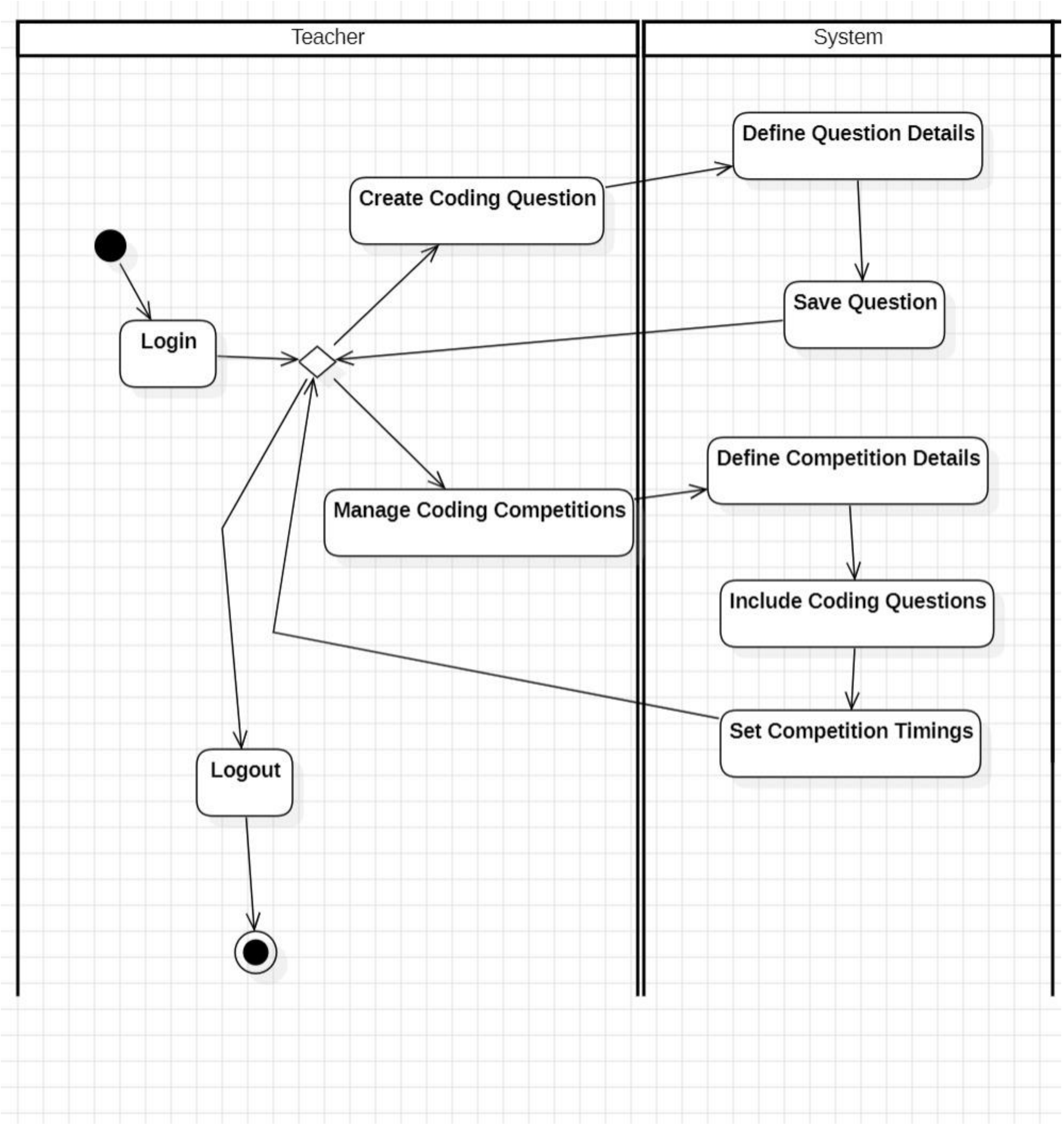
3.3 State Diagram

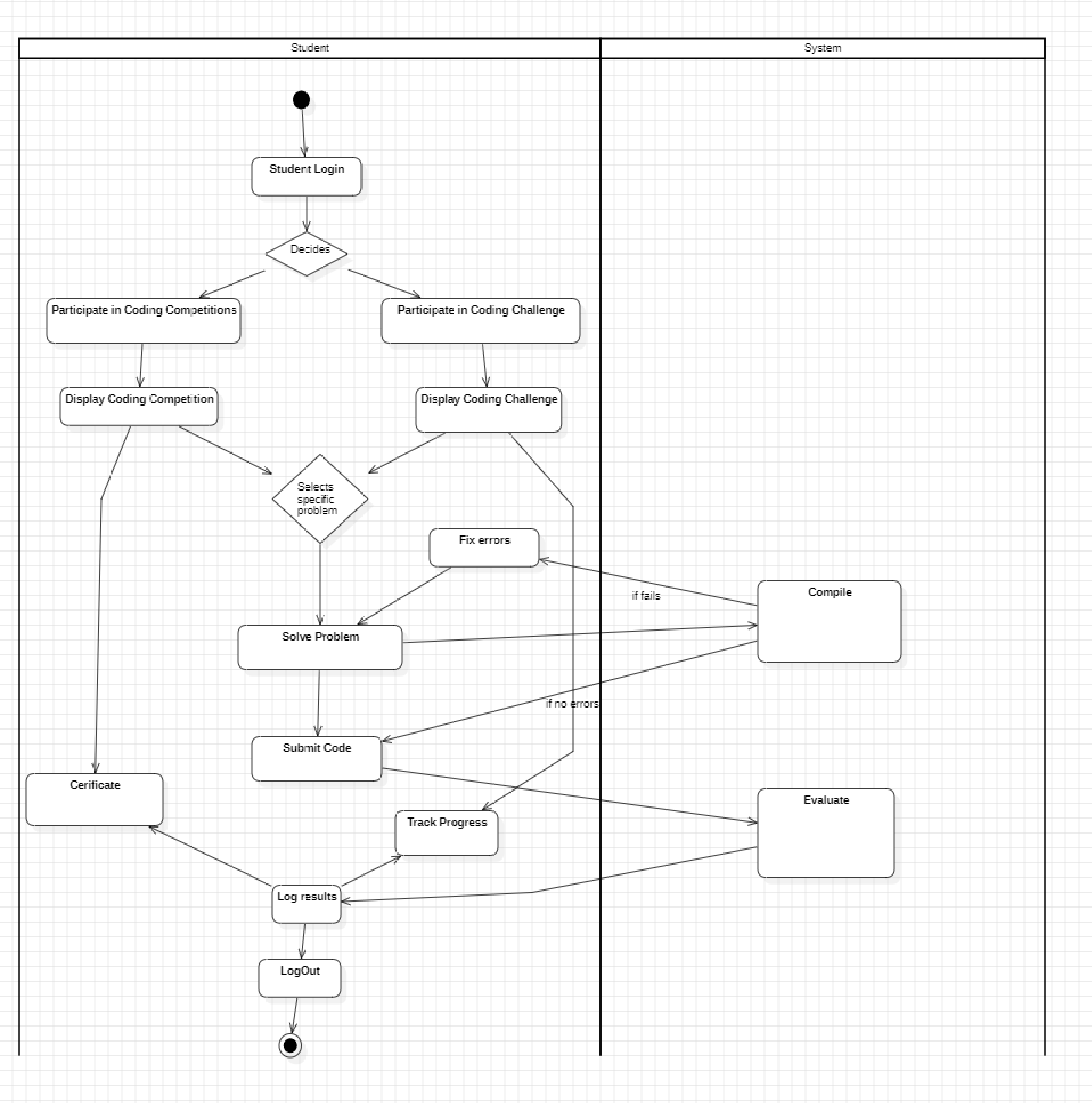


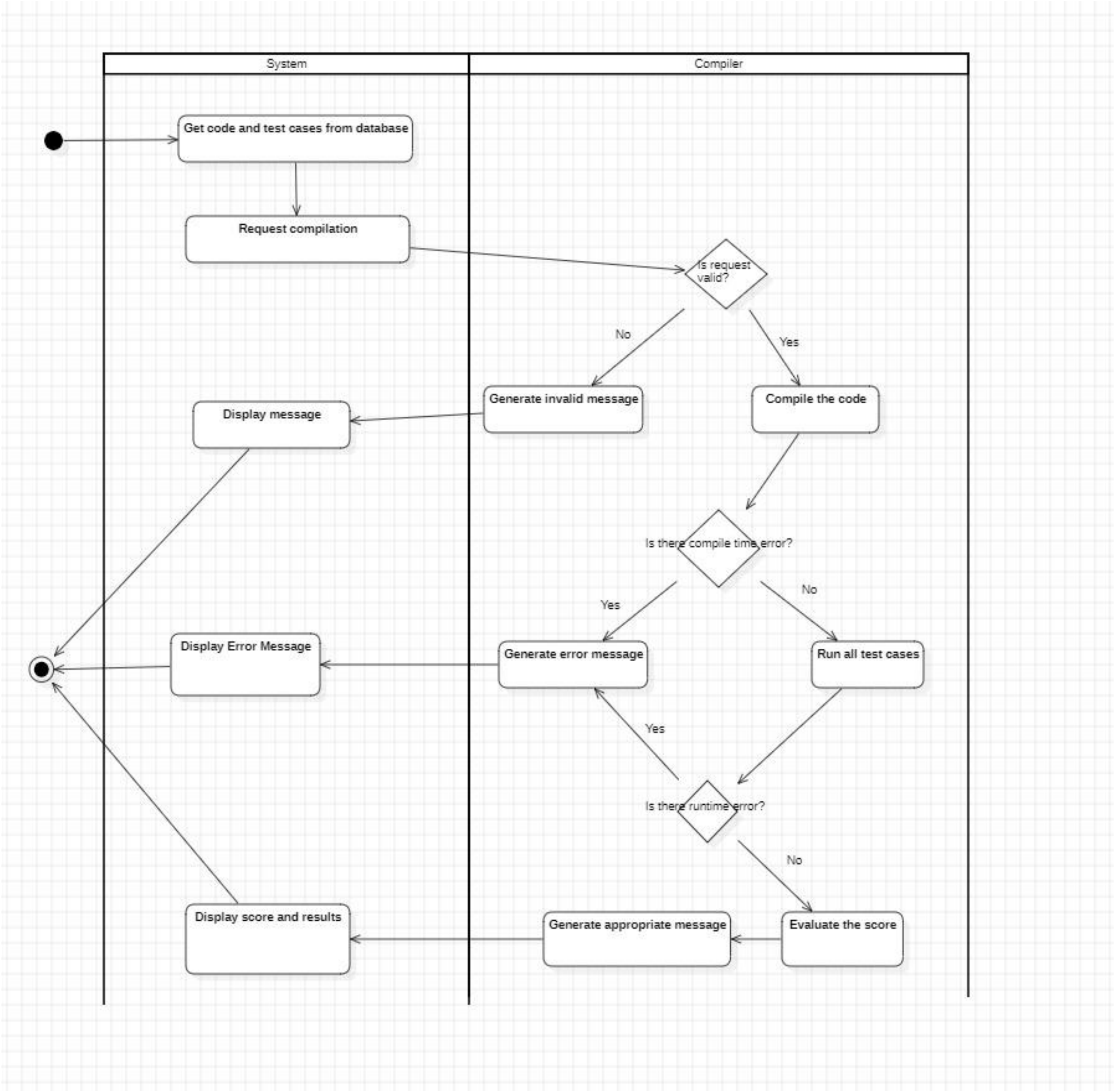

Student login
Paticipate
Browse Different Challenges
Coding Competitions
Coding Challenges
Solve Problem
Certificate
Track Progress
Submit code
Evaluation
Sends Scores
Result
LogOut



3.4Activity Diagram


Teacher
System
Define Question Details
Create Coding Question
Save Question
Login
Define Competition Details
Manage Coding Competitions
Include Coding Questions
Set Competition Timings
Logout


Student
System
Student Login
Decides
Participate in Coding Competitions
Participate in Coding Challenge
Display Coding Competition
Display Coding Challenge
Selects specific problem
Fix errors
Compile
if fails
Solve Problem
if no errors
Submit Code
Cerificate
Evaluate
Track Progress
Log results
LogOut


System
Compiler
Get code and test cases from database
Request compilation
Is request valid?
No
Yes
Generate invalid message
Compile the code
Display message
Is there compile time error?
Yes
No
Display Error Message
Generate error message
Run all test cases
Yes
Is there runtime error?
No
Display score and results
Generate appropriate message
Evaluate the score

4. Architecture Overview:

We made a Spring Boot application which follows MVC architecture for this project. There was a separate Model and Controller for every Spring entity in the application. We used MySQL database for the application and used Hibernate as the ORM to connect to MySQL and run CRUD operations.

Thymeleaf templates were used for the front-end to make a good user interface. The interacted with the controllers in the Spring Boot application.

To compile and execute the code submitted, we used an external service called Judge0 which could give compile, execute and run the given standard input for the code. We communicated with this software through API calls done in Spring Boot application controller.

We also used various design principles such as open-closed principle in our user, student and teacher class. Dependency inversion was used in multiple places and we made sure to ensure single responsibility.

Design patterns such as singleton and decorator were also used in our application. Some more details about said patterns and principles are given.

- **Model-View-Controller (MVC) Architecture:**

MVC is a widely-used architectural pattern that separates an application into three interconnected components: Model, View, and Controller. The Model represents the application's data and business logic, the View displays the user interface, and the Controller handles user input and communicates between the Model and View.

In our CMS, MVC architecture is applied to decouple the presentation layer (View) from the business logic (Model) and user input handling (Controller). This separation of concerns improves maintainability, facilitates code reuse, and allows for easier testing and scalability.

- **Single Responsibility Principle (SRP):**

SRP is a design principle that states that a class or module should have only one reason to change. It advocates for each component to have a single responsibility or purpose, making the system easier to understand, maintain, and extend.

In our CMS, SRP is applied by designing components with clear and focused responsibilities. For example, the content management service is responsible for handling content creation, editing, and deletion, while the authentication service handles user authentication and access control. This adherence to SRP improves code readability, reduces complexity, and enhances maintainability.

- **Open Closed Principle:**

The Open-Closed Principle (OCP) is a fundamental concept in software design that emphasizes the importance of creating classes and modules that are open for extension but closed for modification. In other words, once a class is defined and implemented, it should not need to be modified to add new functionality; instead, new features should be added through inheritance, composition, or other extension mechanisms. This principle encourages developers to design software components in a way that allows for easy extension and modification without requiring changes to existing code, promoting code reusability, maintainability, and scalability.

- **Dependency Inversion:**

Dependency Inversion Principle (DIP) is another key principle in object-oriented design, advocating for the decoupling of high-level modules from low-level implementation details by introducing abstractions and relying on interfaces rather than concrete implementations. According to DIP, high-level modules should not depend on specific implementations of low-level modules; instead, both should depend on abstractions. This inversion of dependencies reduces coupling between modules, making the system more flexible, resilient to changes, and easier to test and maintain.

- **Singleton:**

Singleton is a design pattern used to ensure that a class has only one instance and provides a global point of access to that instance. This pattern is particularly useful in scenarios where a single, shared resource needs to be accessed from multiple parts of the system. By restricting instantiation to a single object and providing a global access point, the Singleton pattern promotes efficient resource management, avoids unnecessary duplication of resources, and simplifies coordination between different parts of the system. However, it's essential to use Singletons judiciously, as they can introduce tight coupling and global state, which may lead to maintenance challenges and hinder testability in some cases.

5. GitHub Link to the codebase:

https://github.com/AkshayAnand2931/OnlineCodingEvaluationPlatform

6. Individual Contributions of Team Members:

- Abhiraj (PES1UG21CS022) :- Worked on connecting spring to database and created tables for competition and challenges. Created the service and controller for CRUD operations for the same.

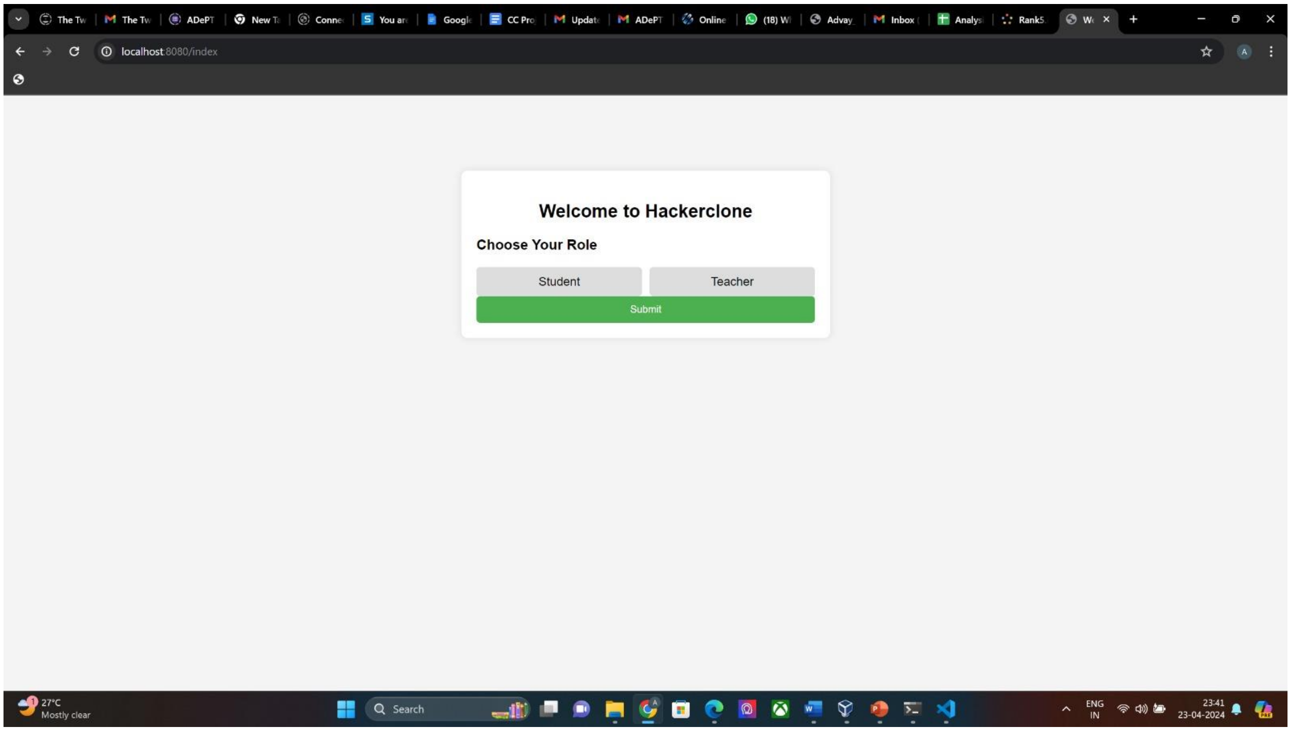
- Advay Agrawal (PES1UG21CS049) :- Worked on creating and connecting teacher tables to the database. Created the create challenge and competition functionality. Connected the CRUD operations of challenges and competition with teacher, to help teacher create and set questions.

- Akshay Anand (PES1UG21CS64) :- Worked on getting code compilation to work. Linked Spring project to external Judge0 library to send and receive code and output through API calls.

- Monisha S (PES2UG22CS809) :- Worked on created and connecting the user tables to the database. Helped get user authentication working and ensure that role based access occurs between student and teachers.

7. Screenshots:

Index Page

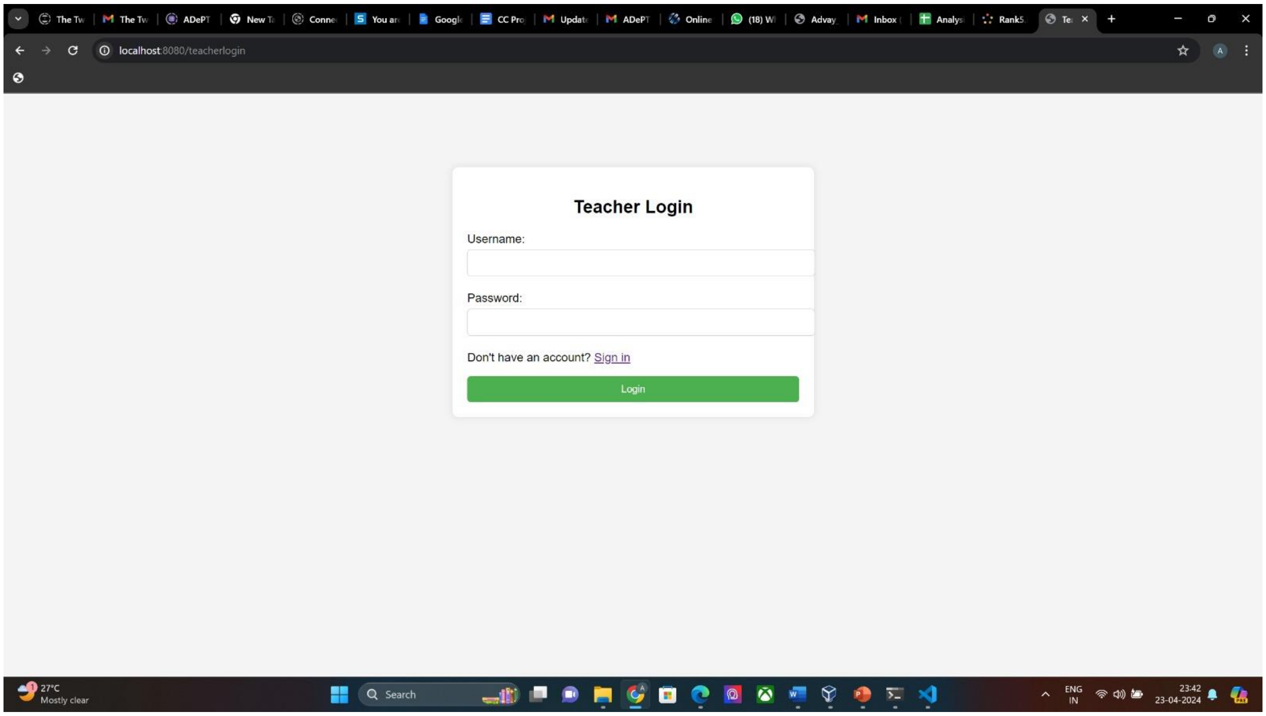


localhost:8080/index

Welcome to Hackerclone

Choose Your Role

Login Page



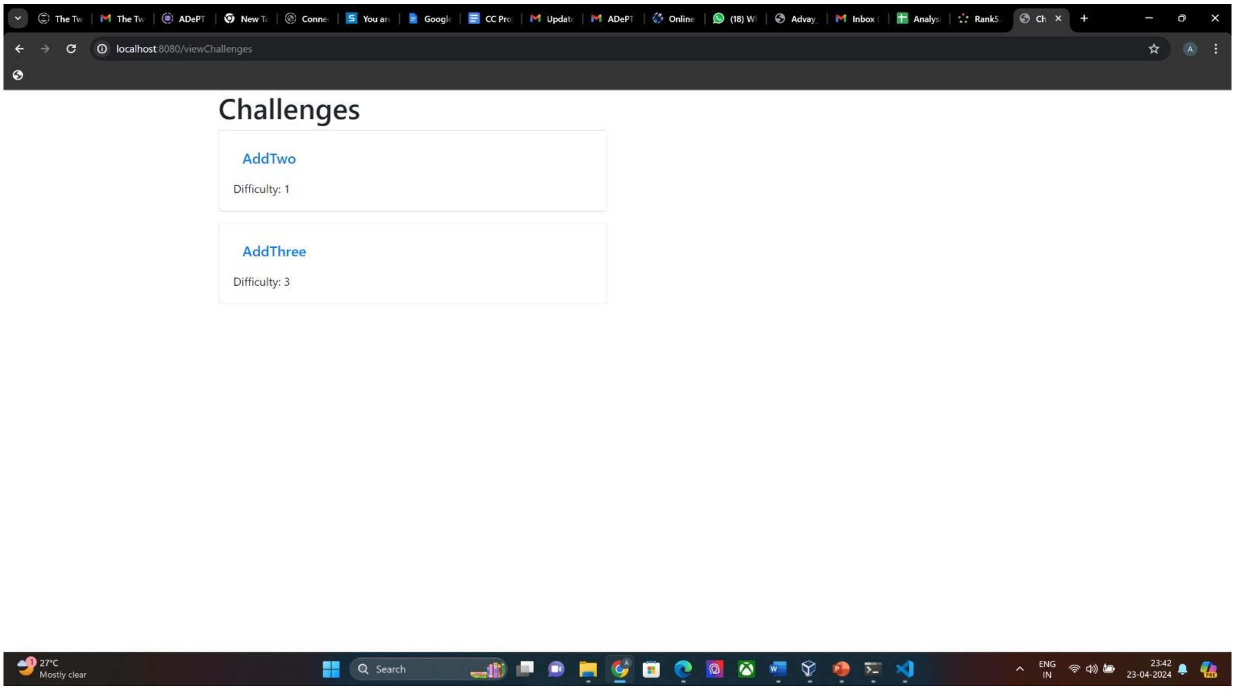
Teacher Login

Username:

Password:

Don't have an account? Sign in

View Challenges Page



Challenges

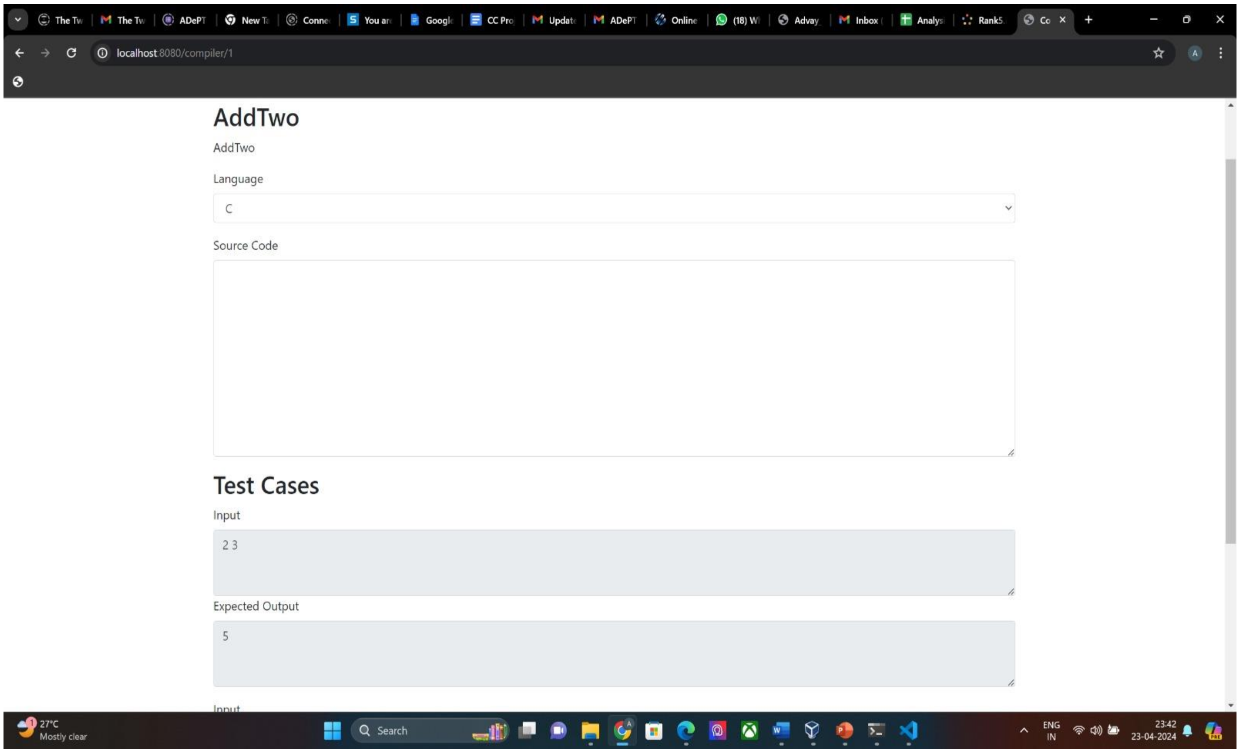
AddTwo

Difficulty: 1

AddThree

Difficulty: 3

Solving challenges page



localhost:8080/compiler/1

AddTwo

AddTwo

Language: C

Source Code

Test Cases

Input

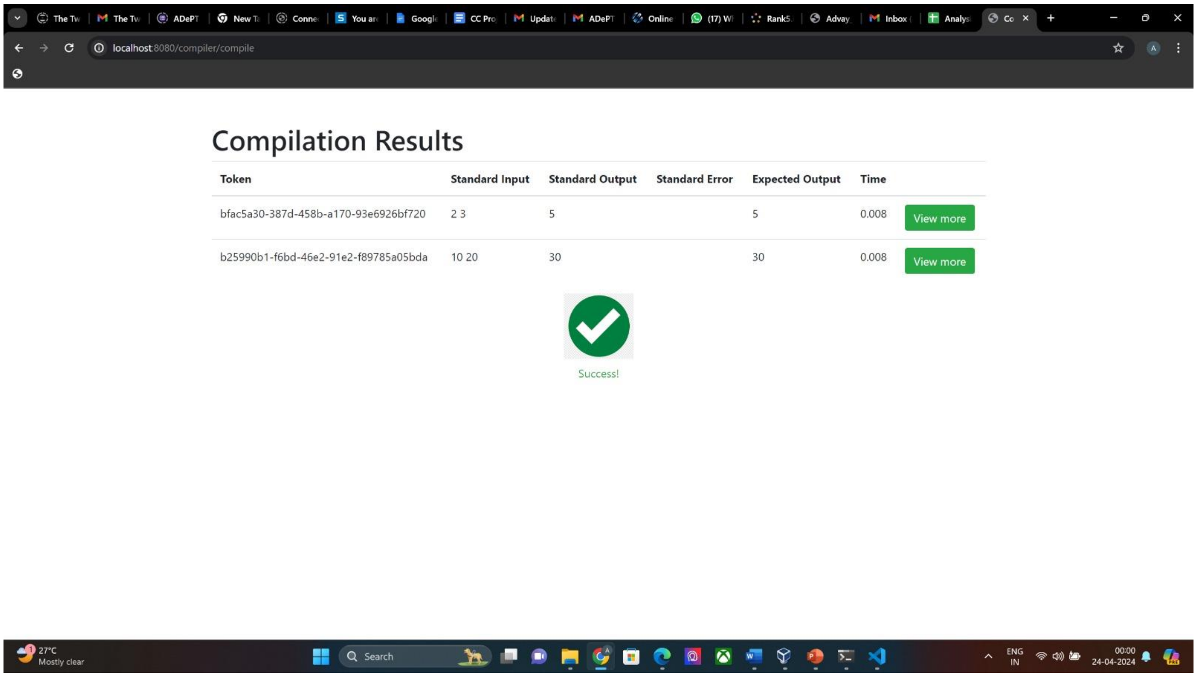
2 3

Expected Output

5

Input

Results Submission:



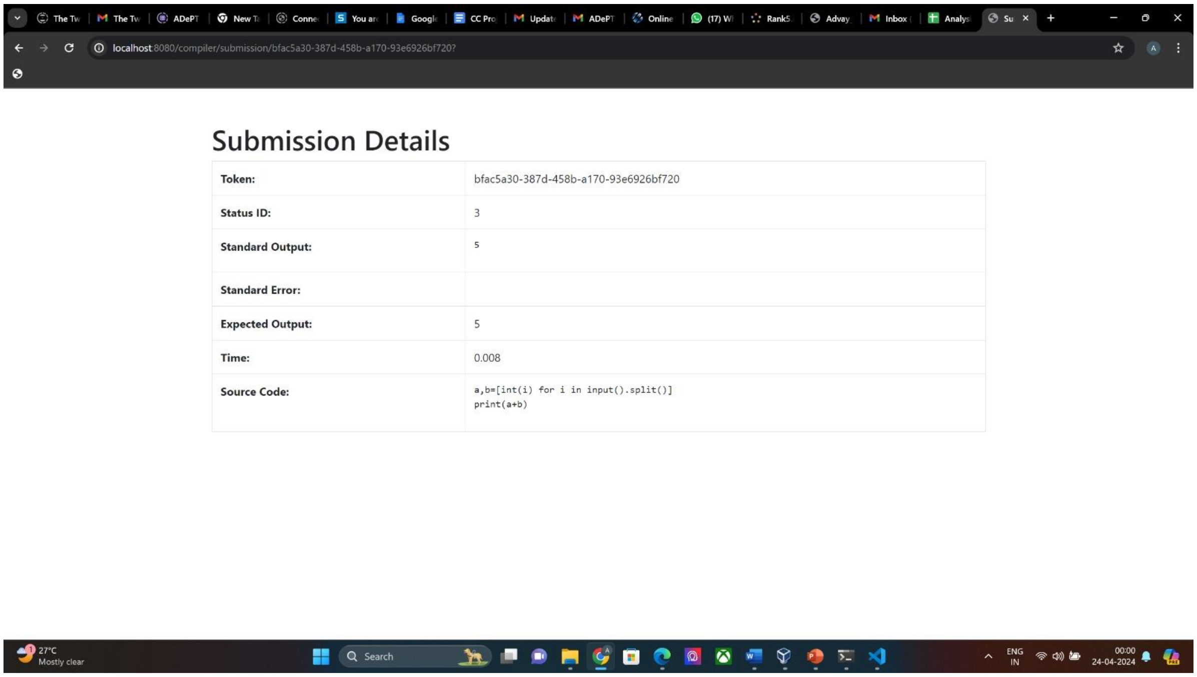
localhost:8080/compiler/compile

Compilation Results

Token	Standard Input	Standard Output	Standard Error	Expected Output	Time	
bfac5a30-387d-458b-a170-93e6926bf720	2 3	5		5	0.008	View more
b25990b1-f6bd-46e2-91e2-f89785a05bda	10 20	30		30	0.008	View more

Success!

Submission Details:



localhost:8080/compiler/submission/bfac5a30-387d-458b-a170-93e6926bf720?

Submission Details

Token:	bfac5a30-387d-458b-a170-93e6926bf720
Status ID:	3
Standard Output:	5
Standard Error:	
Expected Output:	5
Time:	0.008

Source Code:	`a,b=[int(i) for i in input().split()]`
	`print(a+b)`