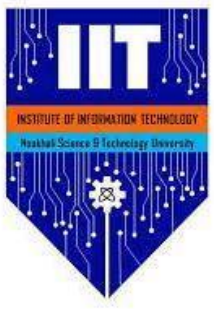
Software Requirements Specification documents for

CodeNestIIT- Elevate your C programming skill



Team Members :

1.Arpita Majumder(BFH2125015F)

2.Majedur Rahman(ASH2125021M)

3.Mohammed Maruf Islam(MUH2125022M)

Institute of Information Technology,

Noakhali Science and Technology University.

Table Of Contents

Introduction:

Welcome to the Software Requirements Specification (SRS) for CodeNestIIT. This document outlines the blueprint for developing CODENESTIIT, a transformative platform tailored for Noakhali Science and Technology University's Institute of Information Technology (IIT). CodeNestIIT aims to revolutionize coding education by providing seamless access to resources and fostering a dynamic learning environment. It defines policies, scope, and references while emphasizing CodeNestIIT advantages and participant requirements. Through meticulous detailing, this document ensures a comprehensive understanding of CodeNestIIT functionalities. It sets the stage for its emergence as a cornerstone in enhancing coding education within Noakhali Science and Technology University's IIT.

Problem Statement:

The Institute of Information Technology (IIT) at Noakhali Science and Technology University (NSTU) faces the challenge of effectively teaching C programming to its students. Traditional methods lack interactivity and engagement, hindering students' learning experiences. Moreover, instructors require efficient tools to assess student progress and facilitate hands-on learning opportunities. To address these challenges, there is a pressing need for a software solution that provides a dynamic platform for interactive learning and practice. This software should encompass a comprehensive repository of programming problems from authoritative sources like "Teach Yourself C" and "C-The Complete Reference," supplemented with additional test cases. It should enable students to engage in self-paced learning, while teachers can organize contests, evaluate student submissions, and deliver personalized feedback. Thus, the development of such a software solution is imperative to enhance the teaching and learning of C programming within the IIT community at NSTU.

Purpose:

The purpose of CodeNestIIT is to revolutionize C programming education for students at Noakhali Science and Technology University's Institute of Information Technology (IIT). By offering an interactive platform with a comprehensive repository of programming problems sourced from authoritative texts like "Teach Yourself C" and "C-The Complete Reference," supplemented with additional test cases, CodeNestIIT aims to make learning C programming easier, more enjoyable, and more effective. It also provides teachers with tools to organize contests, evaluate student work, and deliver personalized feedback, thereby enhancing the overall teaching and learning experience in the field of C programming.

Glossary:

SRS – Software Requirements Specification

API – Application Programming Interface

References:

IEEE. IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications. IEEE Computer Society, 1998.

Overview:

CodeNestIIT transforms C programming education at IIT NSTU. It provides an interactive learning hub with a rich problem repository, instant feedback, and teacher tools. Students enjoy self-paced learning, while instructors offer personalized guidance. Designed to meet dynamic educational demands, CodeNestIIT ensures lasting impact and effectiveness.

Stakeholders:

Student: Student are stakeholders because the quality of education, the relevance of the curriculum, and the support they receive impact their learning experience.

Teacher: The instructors and educators at CodeNestIIT are stakeholders because their role in teaching, curriculum development, and the learning environment directly affects the success of the institute**.**

Developers: The success of CodeNestIIT in the coding and IT domain would be crucial for developers associated with the organization.

System:

The system itself is a stakeholder as it directly influences the interactions and experiences of users (students, teachers, and developers). Its functionality, usability, and reliability impact the effectiveness of teaching and learning processes within CodeNestIIT.

Design and Implementation Constraints:

Language:

Java: Java is a versatile and widely-used programming language renowned for its platform independence and robustness. It is utilized for developing a wide range of applications, including web-based platforms, mobile apps, enterprise software, and more. With its rich libraries, comprehensive documentation, and strong community support, Java enables developers to build scalable, secure, and high-performance solutions. Its object-oriented nature and simplicity make it an ideal choice for projects requiring reliability and maintainability across various domains.

Server Side Technology:

When an application is used, server-side development refers to the processes that happen in the background. Databases, scripting, website architecture, backend logic, APIs, and servers are the main topics covered.

Database Server:

MySQL, an open-source relational database management system (RDBMS), organizes data into tables, facilitating connections between different datasets. It employs SQL for database manipulation, allowing programmers to create, modify, and retrieve data, as well as manage user access. MySQL's versatility and scalability make it a preferred choice for a wide range of applications, from small-scale projects to enterprise-level systems, providing efficient data management solutions for diverse requirements.

Requirements Specification:

Functional Requirements:

Functional requirements are those that serve as examples for the system's internal operation, its description, and an explanation of each subsystem. It comprises of the task that the system should complete, the associated processes, the data that the system should store, and the user interfaces

User Authentication:

FR-1	User Authentication
Description	Students and Teachers of IIT should be able to securely log in to the platform using their unique credentials, which are given before, to access personalized features and content.
Stakeholders	Students , Teachers of IIT , System
Priority	High

Problem Repository:

FR-2	Problem Repository
Description	The system should provide a centralized repository of C programming problems sourced from authoritative texts, allowing users to browse, search, and access relevant learning materials.
Stakeholders	System
Priority	High

Contest Organization:

FR-3	Contest Organization
Description	Teachers should have the ability to organize coding contests within the platform, setting up parameters, inviting participants, and evaluating submissions to foster healthy competition and skill development.
Stakeholders	Teachers, System
Priority	High

Feedback Mechanism:

FR-4	Feedback Mechanism
Description	Teachers should be able to provide personalized feedback on student submissions, track student progress, and identify areas for improvement to support student learning and growth.
Stakeholders	Students , Teachers , System
Priority	High

Interactive Learning Features:

FR-5	Interactive Learning Features
Description	The platform should offer interactive learning features such as coding exercises, hints to engage students and facilitate self-paced learning.
Stakeholders	Teachers , Students, System
Priority	High

Data Requirements:

User data:

DR-1	Storing data of Students and Teachers
Description	The system should maintain user profiles, including authentication credentials, personal information, learning progress, and contest participation history, to provide personalized experiences and track user interactions.
Stakeholders	System
Priority	High

Problem data:

DR-2	Storing data of problems
Description	The system should store a database of programming problems, including problem statements, solutions, test cases, and metadata, to provide a comprehensive learning resource.
Stakeholders	System
Priority	High

Performance Requirements:

Response Time:

FR-1	Fast Response Time
Description	The system should provide fast response times for user interactions, including problem search, submission uploads, and feedback retrieval, to ensure a seamless user experience.
Stakeholders	System, Students, Teachers
Priority	High

Maintenance and Supportability requirements:

Modular Codebase:

MR-1	Modular Codebase
Descriptions	The software should be developed with a modular codebase, allowing for easy maintenance, updates, and enhancements without affecting other system components. This modular structure ensures that developers can efficiently identify and address issues, implement new features, and adapt to changing requirements, promoting long-term maintainability and supportability.
Stakeholders	Developers
Priority	High

Requirements Engineering Process:

Requirements Elicitation Techniques:

Interviews: I conducted one-on-one interviews with both students and teachers to delve deeper into their perspectives on C programming education. These interviews provided valuable insights into their individual experiences, preferences, and challenges.

Surveys: Surveys were distributed among students to gather feedback on various aspects of C programming education. By analyzing the responses, we gained quantitative insights into the prevailing opinions, preferences, and areas for improvement.

Focus groups:
Focus groups were organized, bringing together small cohorts of students to engage in collaborative discussions. These sessions allowed for the exploration of diverse viewpoints, fostering idea generation and in-depth exploration of specific topics related to C programming education.

Observations:
Through direct observation in classroom settings, I gained firsthand insights into the dynamics of C programming education. By observing students' and teachers' interactions, behaviors, and challenges, I was able to uncover implicit requirements and nuances that might not have been evident through other elicitation techniques.

Sample of Requirements Collection:

Requirement Elicitation Techniques	Interviews,Surveys,Focus Groups
Collected From	Students of IIT
Findings	• **Topic-Wise Problem Selection:** Students emphasized the importance of accessing a diverse range of programming problems categorized by topic, facilitating focused practice and skill development. • **Rating-Based Sorting Preference:** Many students expressed a desire for the ability to sort problems based on ratings or difficulty levels, enabling them to choose challenges aligned with their skill level and learning objectives. • **Interest in Hints Feature:** There was notable interest among students in having hints available for challenging problems, recognizing their potential to provide

	guidance and support during problem-solving endeavors. • **Desire for Batch-Wise Ranking:** Students showed keen interest in a feature that ranks participants within their respective batches or cohorts, fostering friendly competition and providing a benchmark for personal progress among peers. • **Request for Messaging Option:** Several students expressed the need for a messaging feature within the platform, allowing for seamless communication with instructors or peers to seek clarification, share insights, and discuss strategies related to problem-solving and learning.

Requirement Elicitation Techniques	Interviews
Collected From	Md Eusha Kadir, Lecturer, IIT, NSTU
Findings	During the interview with the teacher, it was indicated that everything within the current system was satisfactory. However, he emphasized the importance of incorporating a feature to facilitate the arrangement of coding contests within the platform. This addition would enhance the educational experience by providing students with opportunities for practical application and friendly competition.

Requirements Validation:

Prototyping: Prototypes or mockups of the software platform were developed to visually represent the proposed features and functionalities. Stakeholders, including students and teachers, were invited to review these prototypes, providing feedback on the user interface, functionality, and overall user experience. Their input helped validate the requirements and ensure that the final product met their expectations.

Requirements Reviews: Thorough reviews of the documented requirements were conducted with key stakeholders, including students, teachers, and developers. During these reviews, stakeholders provided feedback on the clarity, completeness, and feasibility of the requirements. Any

inconsistencies or gaps in the requirements were identified and addressed, ensuring that the final set of requirements accurately captured the needs and expectations of the stakeholders.

Walkthrough: Walkthrough sessions were organized with stakeholders to systematically review the requirements documentation in detail. During these sessions, stakeholders were guided through the requirements document, and each requirement was discussed thoroughly. Any ambiguities or discrepancies were identified and clarified, ensuring a common understanding of the requirements among all stakeholders. Walkthroughs provided an opportunity for stakeholders to raise questions, provide feedback, and validate the requirements before proceeding with the development phase.

Use Case Diagram:

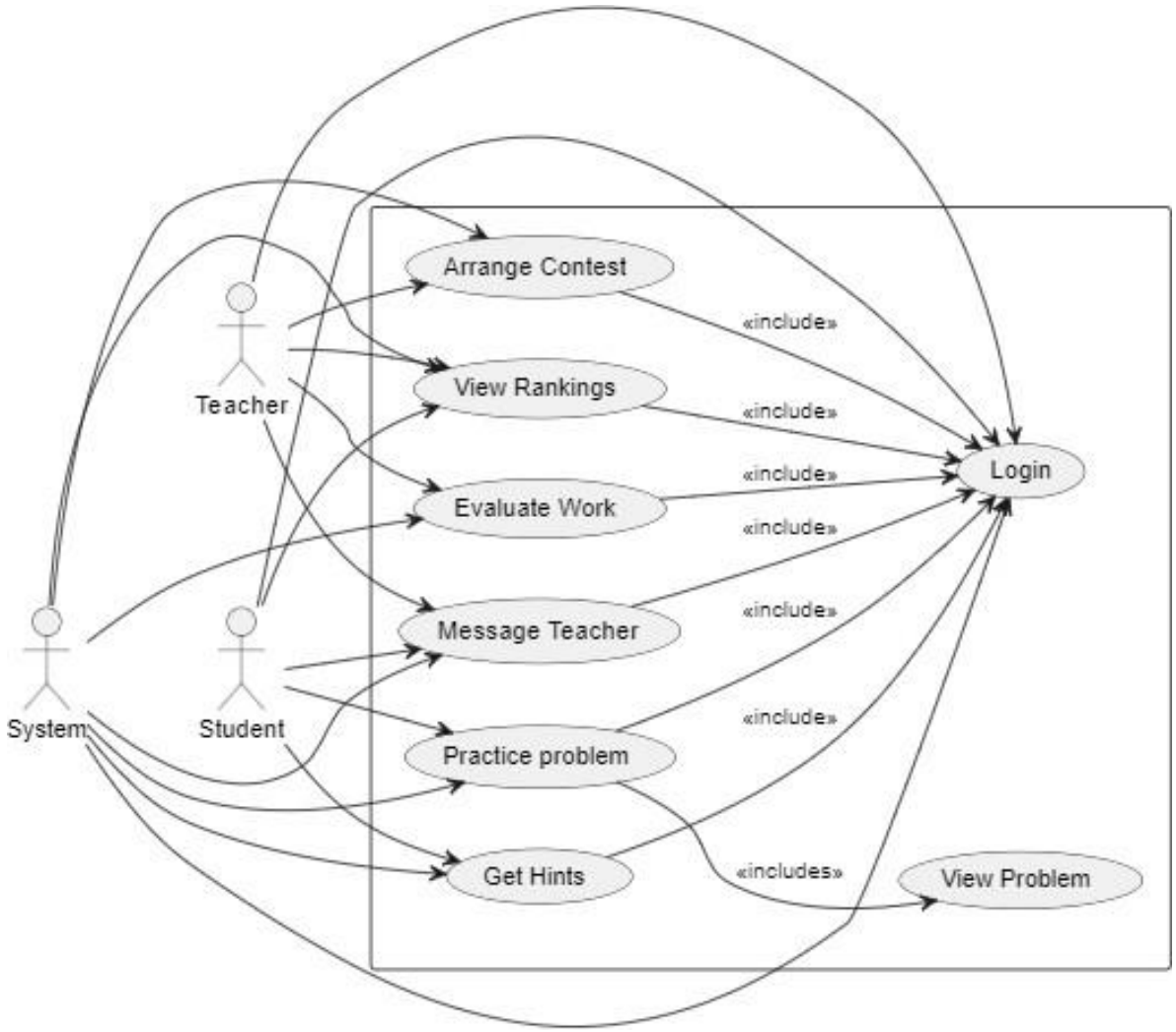


Fig: Use Case Diagram

Use Case Description:

Use Case No.	01
Use Case	Login
Description	Allows users (students, teachers, administrators) to access the platform usingtheir assigned user IDs and passwords. User IDs are provided by the system, eliminating the need for individual registration.
Primary Actors **Secondary Actor**	Student, Teacher System
Preconditions	• Users must have a valid user ID provided by the system. • Users must remember their assigned passwords.
Flows	1. User navigates to the platform's login page. 2. Enters the provided user ID in the designated field. 3. Inputs the associated password. 4. Submits the login form for authentication.
Alternative flows	If the provided user ID or password is incorrect, the system prompts the user tore-enter the correct credentials.

Use Case No.	02
Use Case	View Problem
Description	Enables users, specifically students, to explore and access a curated collection of C programming problems from authoritative sources such as "Teach Yourself C" and "C-The Complete Reference." This feature provides a comprehensive set of exercises to enhance learning.
Primary Actors **Secondary Actors**	Student Teacher
Preconditions	• Student must be logged in with a valid user ID. • A list of available problems must be accessible to the student.
Flows	1. The user, whether a student or a teacher, logs into the platform. 2. Navigates to the "View Problems" section of the platform. 3. Explores a categorized list of C programming problems. 4. Selects a specific problem of interest to view its statement and details.

Use Case No.	03
Use Case	Practice Problem
Description	Enables students to actively engage in practicing C programming by selecting andsolving problems from the platform's extensive collection. This feature fosters a hands-on approach to learning.
Primary Actors	Student
Preconditions	• Student must be logged in with a valid user ID. • A list of available problems must be accessible to the student.
Flows	1. Student navigates to the "Practice" section of the platform. 2. Browses through the available C programming problems. 3. Selects a specific problem of interest. 4. Attempts to solve the chosen problem using the programming interface. 5. Submits the solution for evaluation.
Alternative flows	If the solution is incorrect, the system provides feedback, including hints and suggestions for improvement.

Use Case No.	04
Use Case	Get hints
Description	Allows students to request guidance while solving C programming problems. User can take up to three hints. The gap between first and second hints will be30 minutes. After that user will be allowed to see the third one after one hour.
Primary Actors	Student
Preconditions	• Student must be logged in with a valid user ID. • Student must submit once before requesting hints
Flows	1. Student navigates to the problem-solving interface. 2. Attempts to solve a C programming problem. 3. Requests hints for additional guidance. 4. System checks if the user has received three hints. 5. If yes, system notifies the user that no more hints are available. 6. If no, system provides the first hint. 7. User continues solving the problem. 8. After 30 minutes, system checks if the user has received the second hint. 9. If yes, system notifies the user that the next hint is available. 10. If no, system waits until 30 minutes have passed from the first hint and then provides the second hint. 11. User continues solving the problem. 12. After one hour from the first hint, system provides the third hint. 13. User continues solving the problem or decides to stop.

Use Case No.	05
Use Case	View Rankings
Description	Allow users, both students, and teachers, to access and explore rankings basedon performance within the platform.
Primary Actors	Student, Teacher
Preconditions	• Users must be logged into the platform with valid user IDs. • Ranking data should be available for the respective user and class.

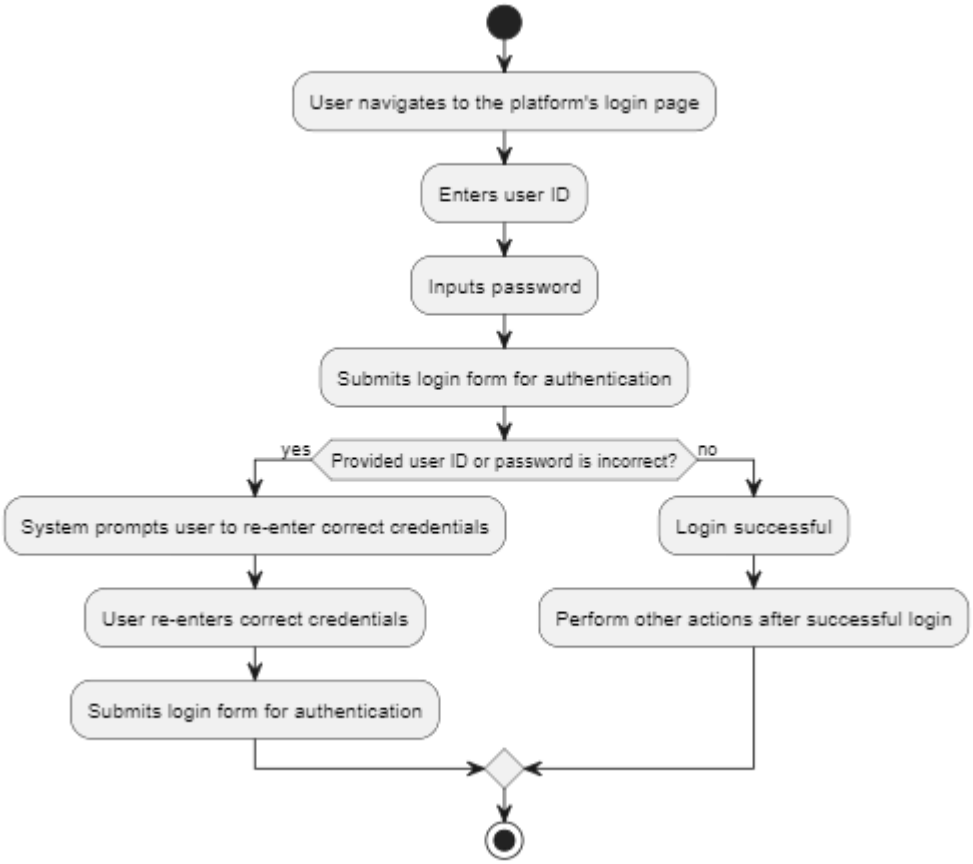
Flows	1. The user, whether a student or teacher, logs into the platform. 2. Navigates to the "View Rankings" section. 3. Explores individual and class rankings based on performance metrics.

Use Case No.	06
Use Case	Message Teacher
Description	Enables students to establish direct communication with teachers within the platform to discuss about a problem.
Primary Actors	Student, Teacher
Preconditions	• The student must be logged into the platform with a valid user ID. • Teachers must have accounts and be accessible for communication.
Flows	1. The student navigates to the "Message Teacher" section. 2. Selects the teacher they wish to communicate with. 3. Composes and sends a message, seeking clarification or guidance.
Alternate flows	If a teacher is not available for messaging, the system provides a notification andsuggests alternative communication methods or times.
Use Case No.	07
Use Case	Arrange Contest
Description	Empowers teachers to organize programming contests within the platform. Teachers, logged in with valid credentials, can define contest parameters, selectproblems, and invite students to participate.
Primary Actors	Teacher
Preconditions	• The teacher must be logged into the platform with a valid user ID. • Problems for the contest must be available in the platform's database.
Flows	1. The teacher navigates to the "Arrange Contests" section. 2. Defines contest parameters, including duration and rules. 3. Selects C programming problems from the platform's collection for thecontest. 4. Invites students to participate by specifying contest details.
Alternate flows	If no suitable problems are available, the system notifies the teacher and suggests alternatives or allows the addition of new problems.

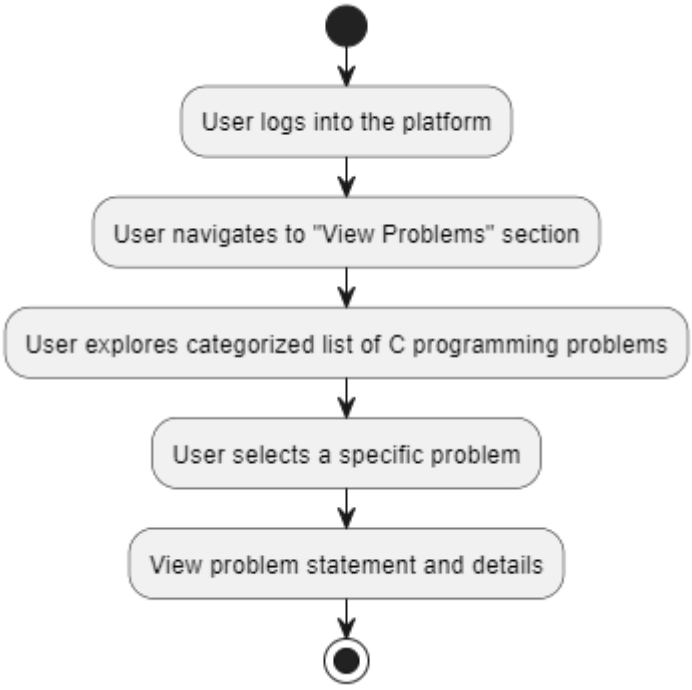
Use Case No.	08
Use Case	Evaluate Work
Description	Enables teachers to assess students' solutions.
Primary Actors	Teacher
Preconditions	• The teacher must be logged into the platform with a valid user ID. • Students must have submitted their solutions for evaluation.
Flows	1. The teacher navigates to the "Evaluate Work" section. 2. Selects a specific assignment or problem for evaluation. 3. Reviews submitted solutions, assigns grades, and provides constructivefeedback. 4. Updates the evaluation status, allowing students to view their assessedwork.
Alternative flows	If a student's solution requires additional clarification, the teacher may askfurther information before completing the evaluation.

Activity Diagram:

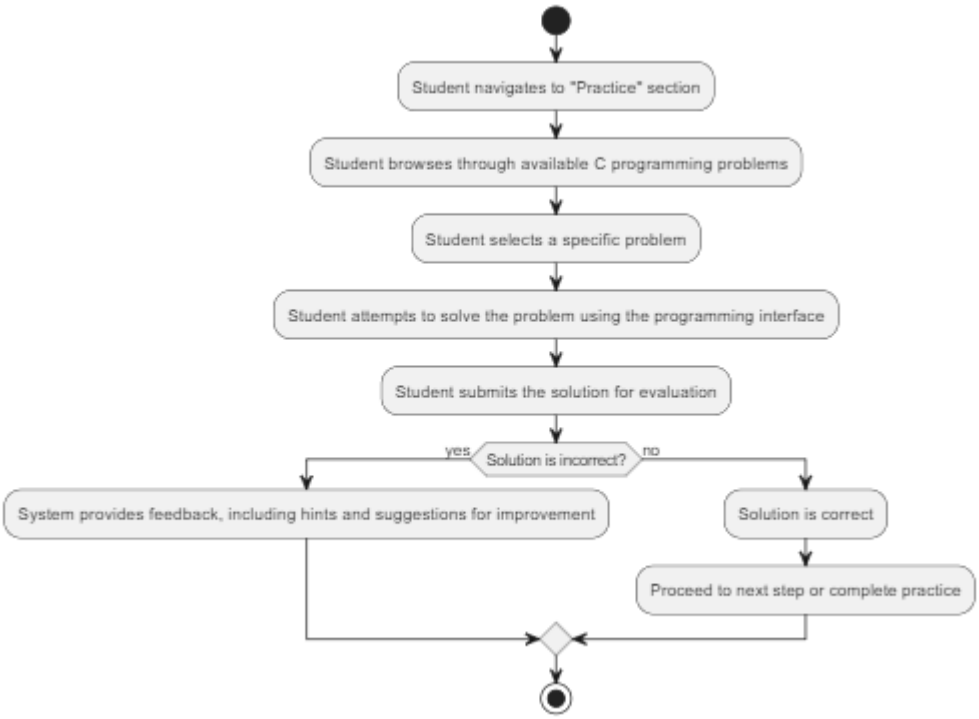
Login:



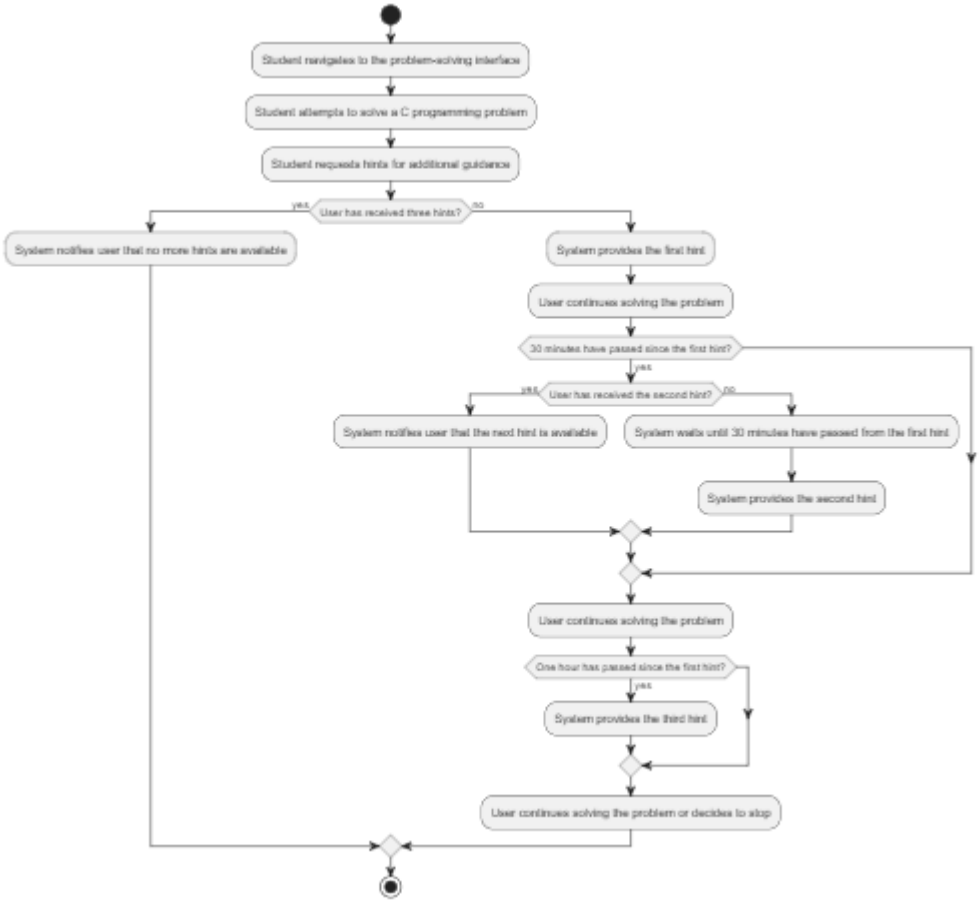
View Problem:



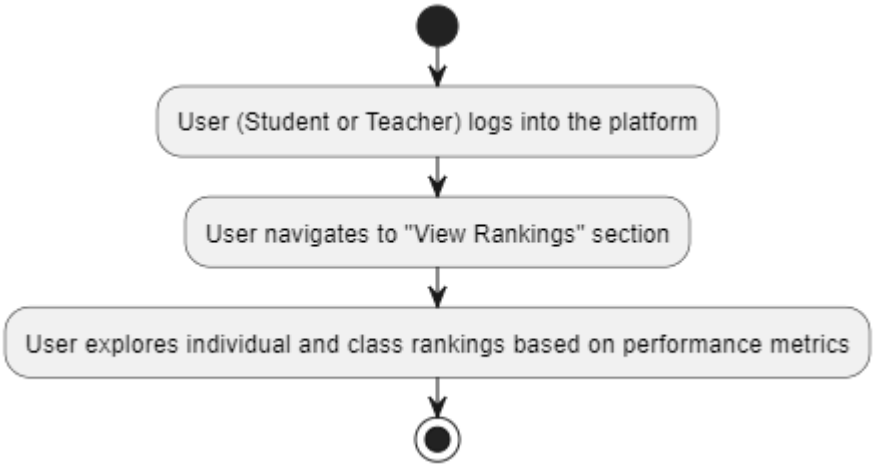
Practice Problem:



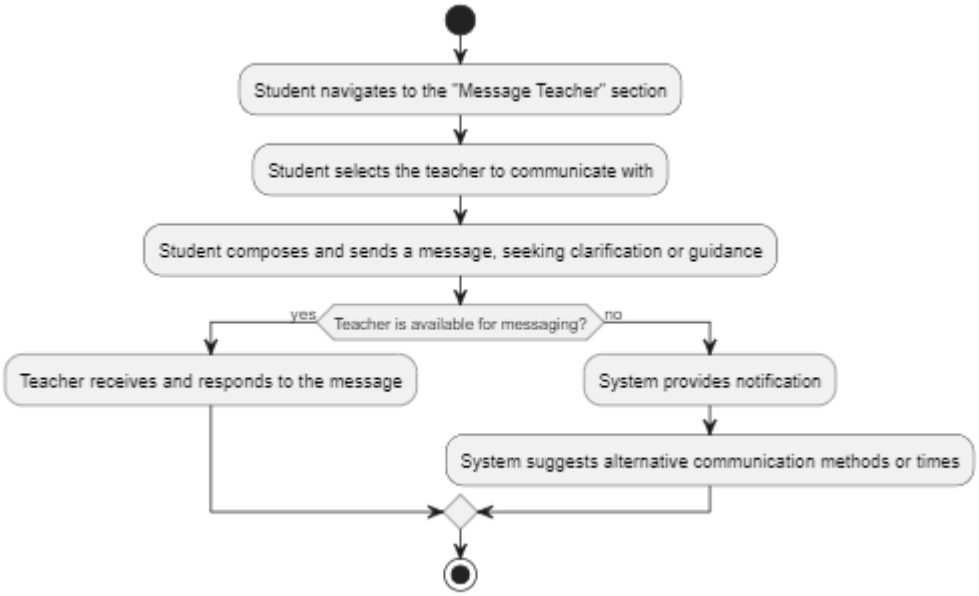
Get Hints:



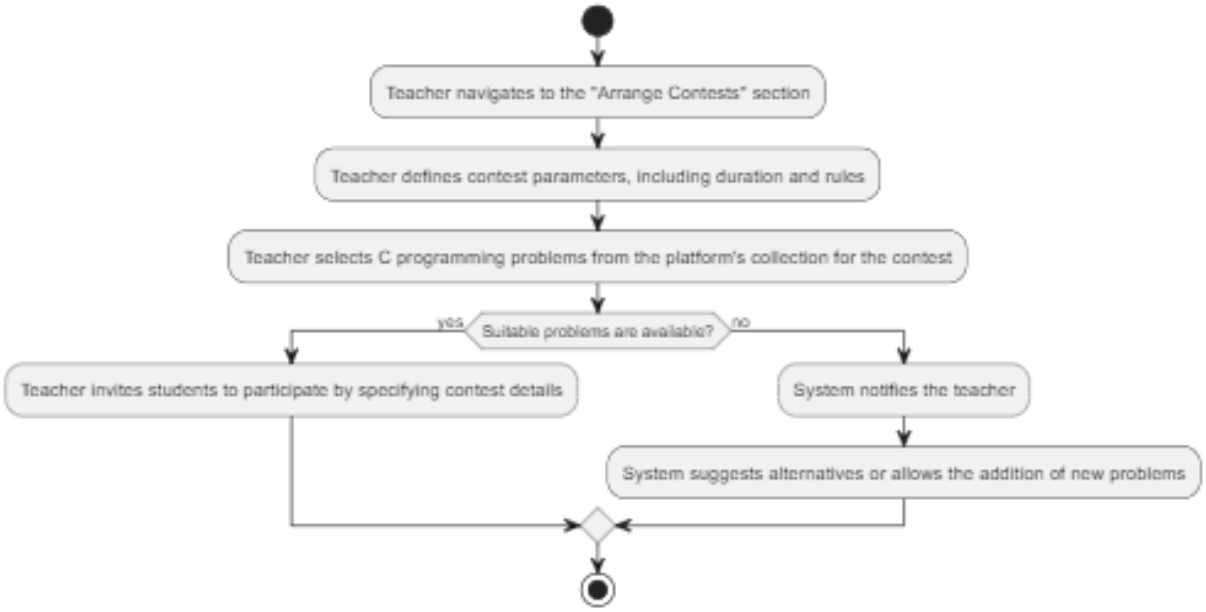
View Rankings:



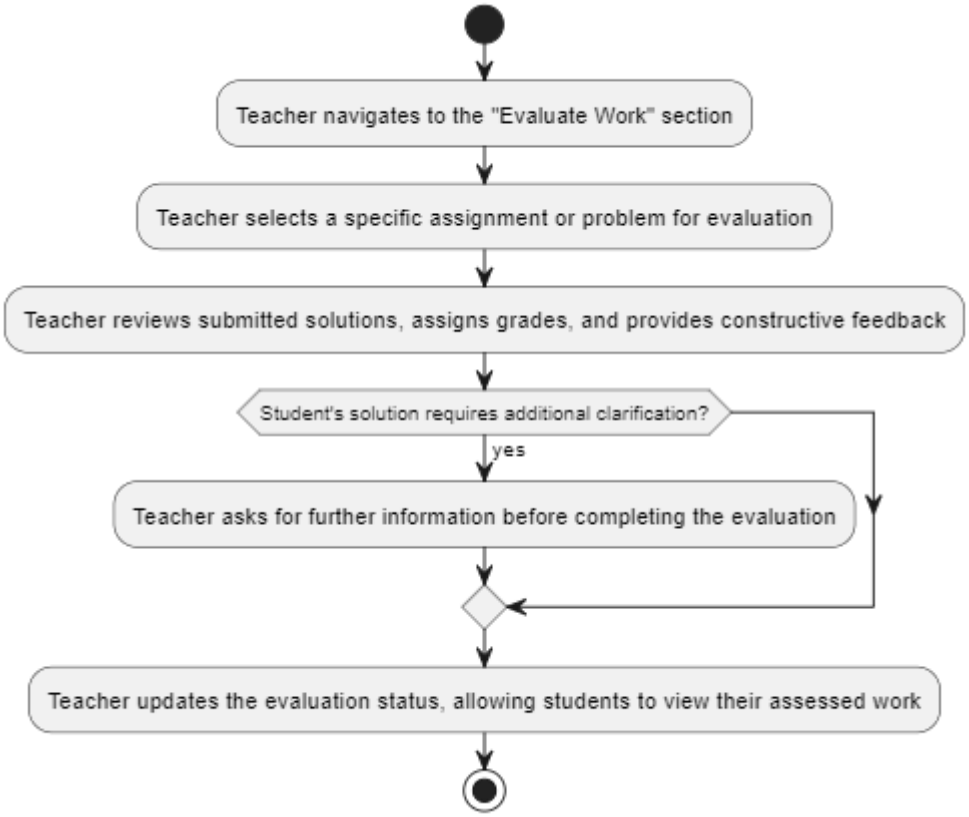
Message Teacher:



Arrange Contest:

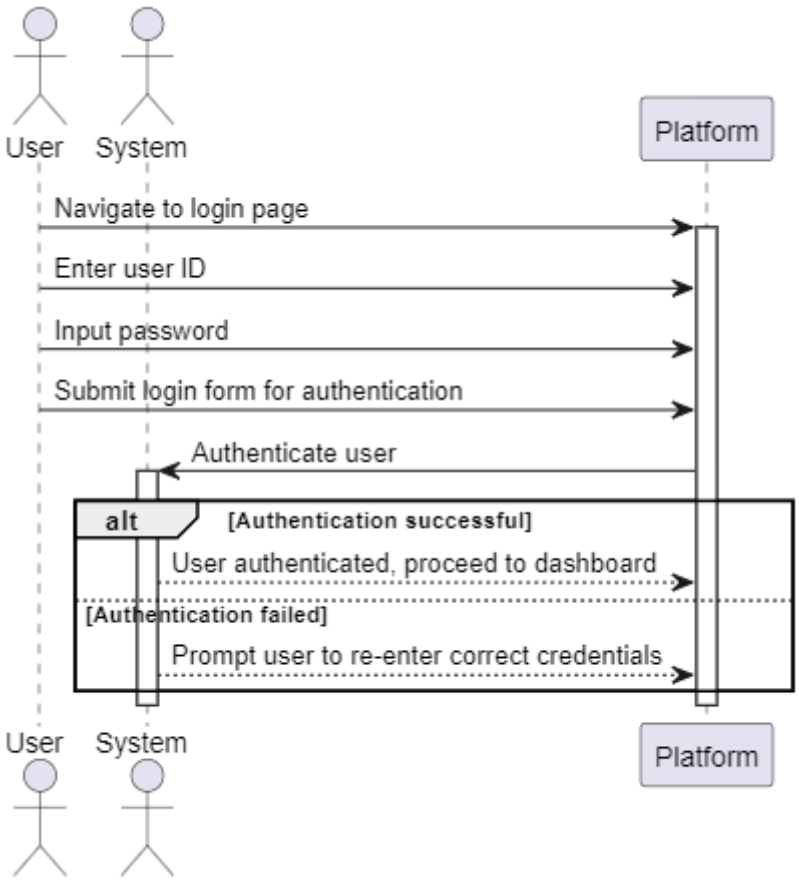


Evaluate Work:

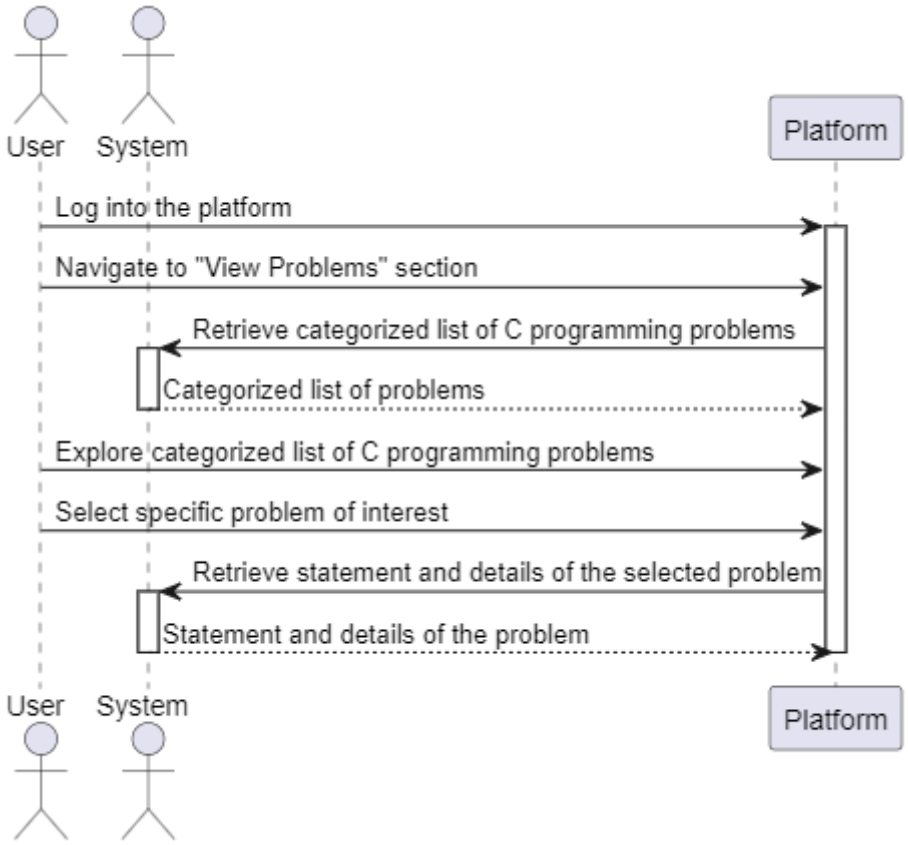


Sequence Diagram:

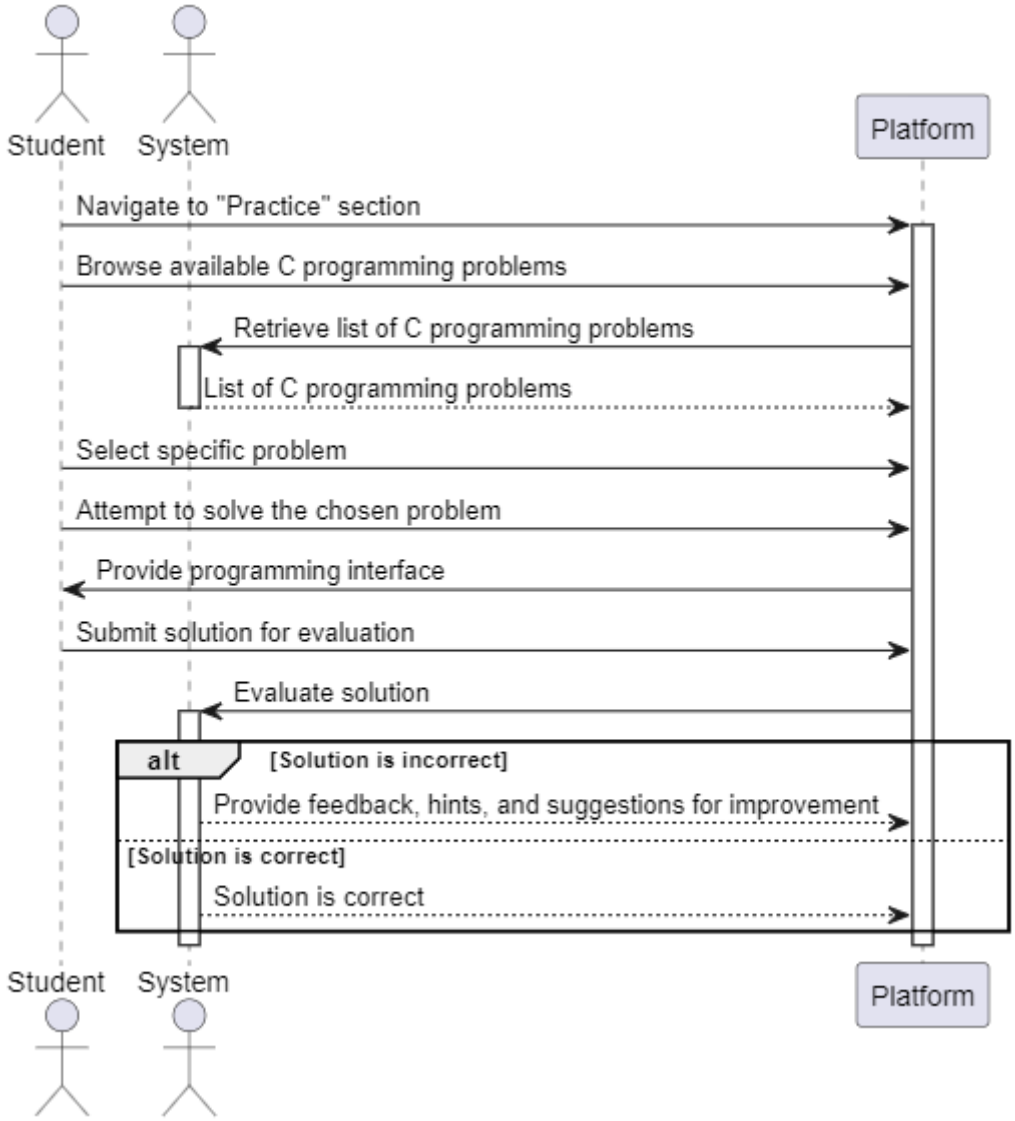
Login:



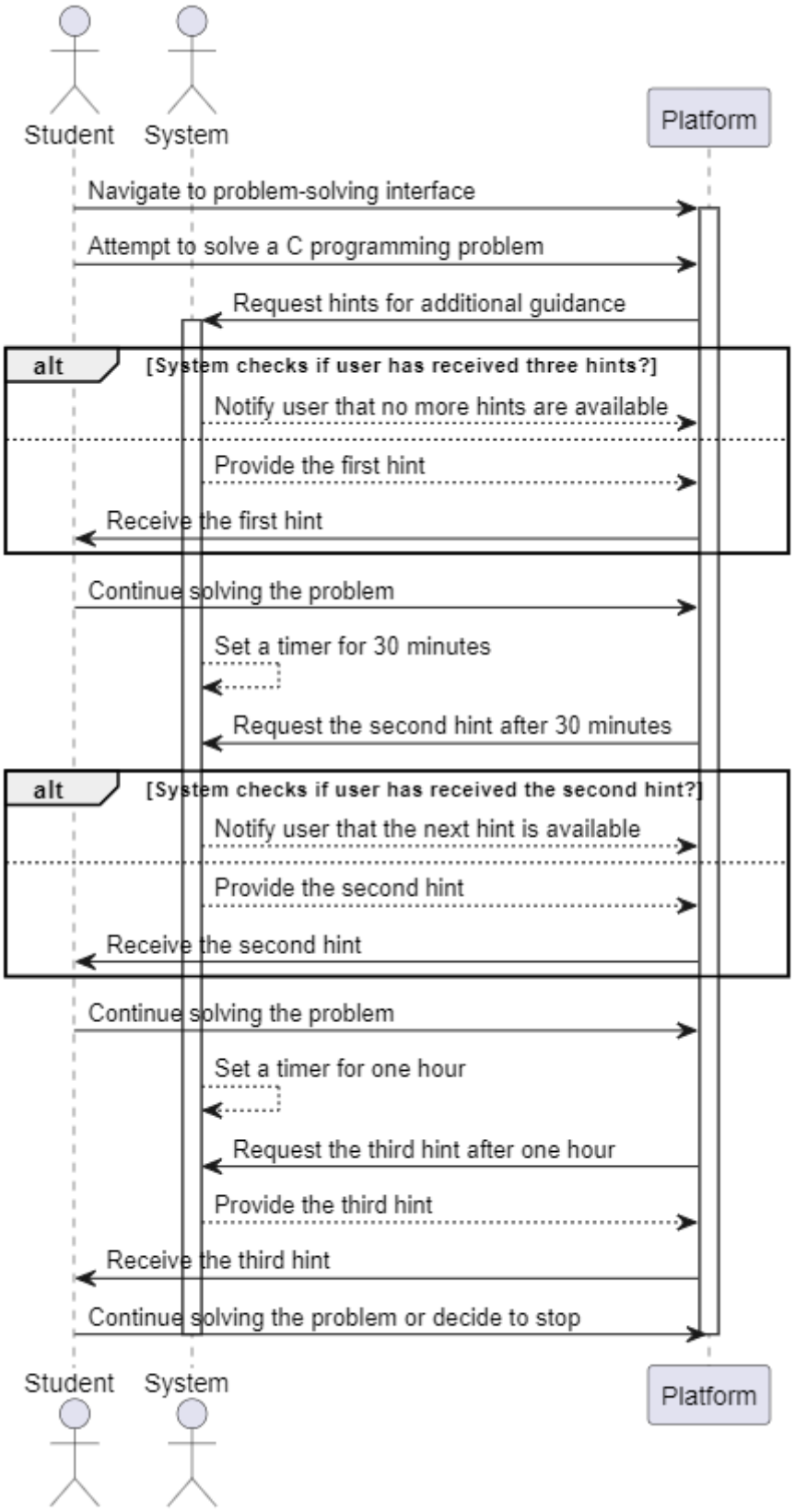
View Problem:



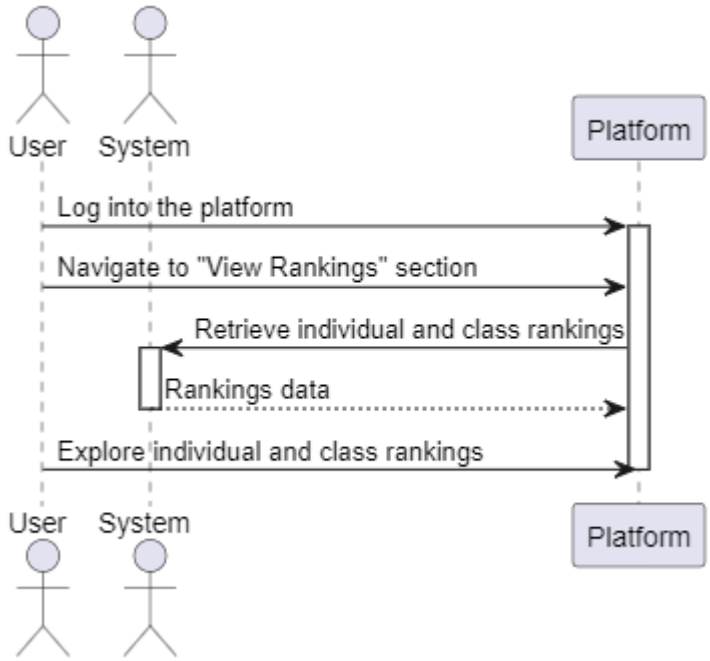
Practice Problem:



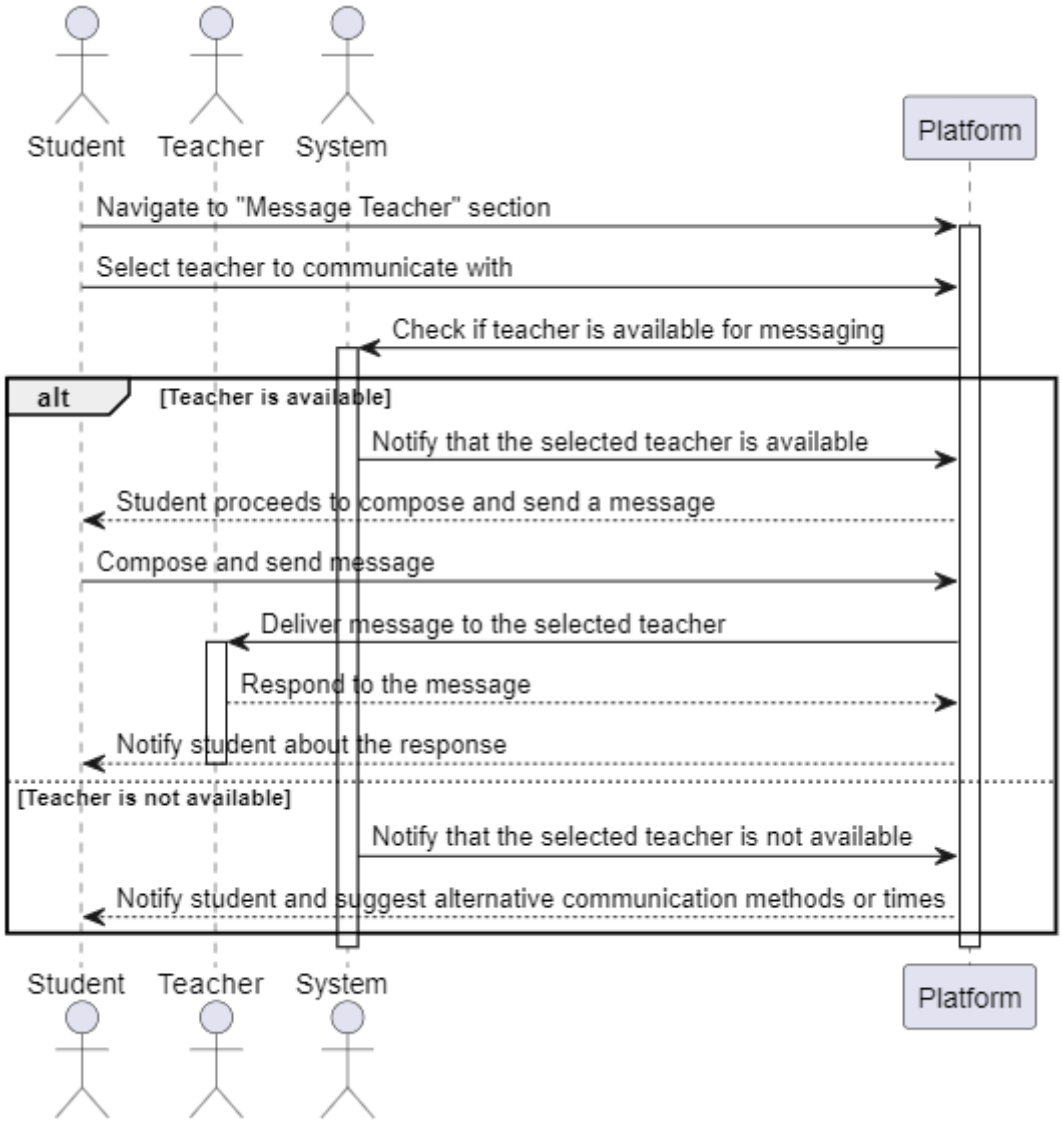
Get Hints:


Student
System
Platform
Navigate to problem-solving interface
Attempt to solve a C programming problem
Request hints for additional guidance
alt
[System checks if user has received three hints?]
Notify user that no more hints are available
Provide the first hint
Receive the first hint
Continue solving the problem
Set a timer for 30 minutes
Request the second hint after 30 minutes
alt
[System checks if user has received the second hint?]
Notify user that the next hint is available
Provide the second hint
Receive the second hint
Continue solving the problem
Set a timer for one hour
Request the third hint after one hour
Provide the third hint
Receive the third hint
Continue solving the problem or decide to stop
Student
System
Platform

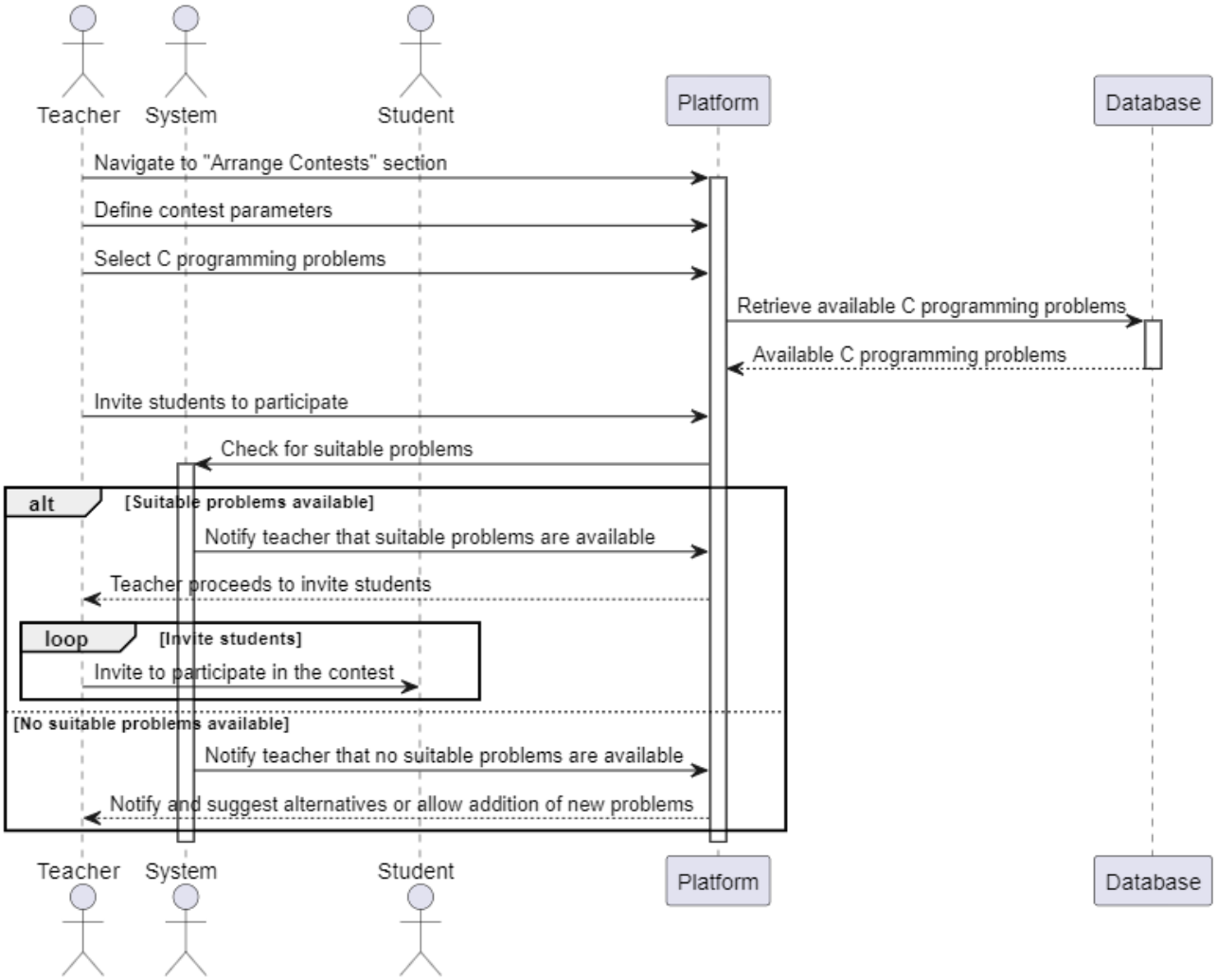
View Rankings:



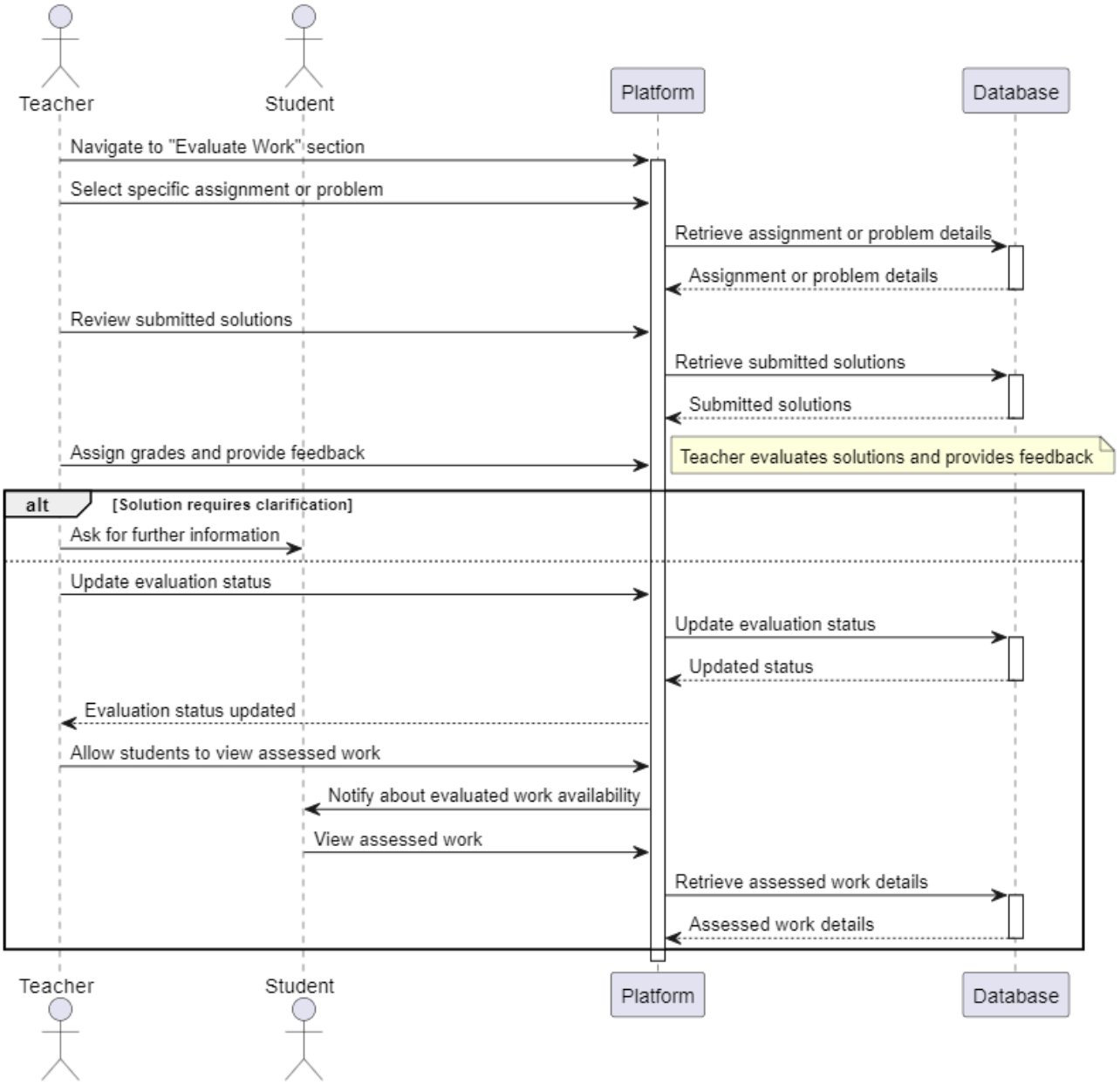
Message Teacher:



Arrange Contest:

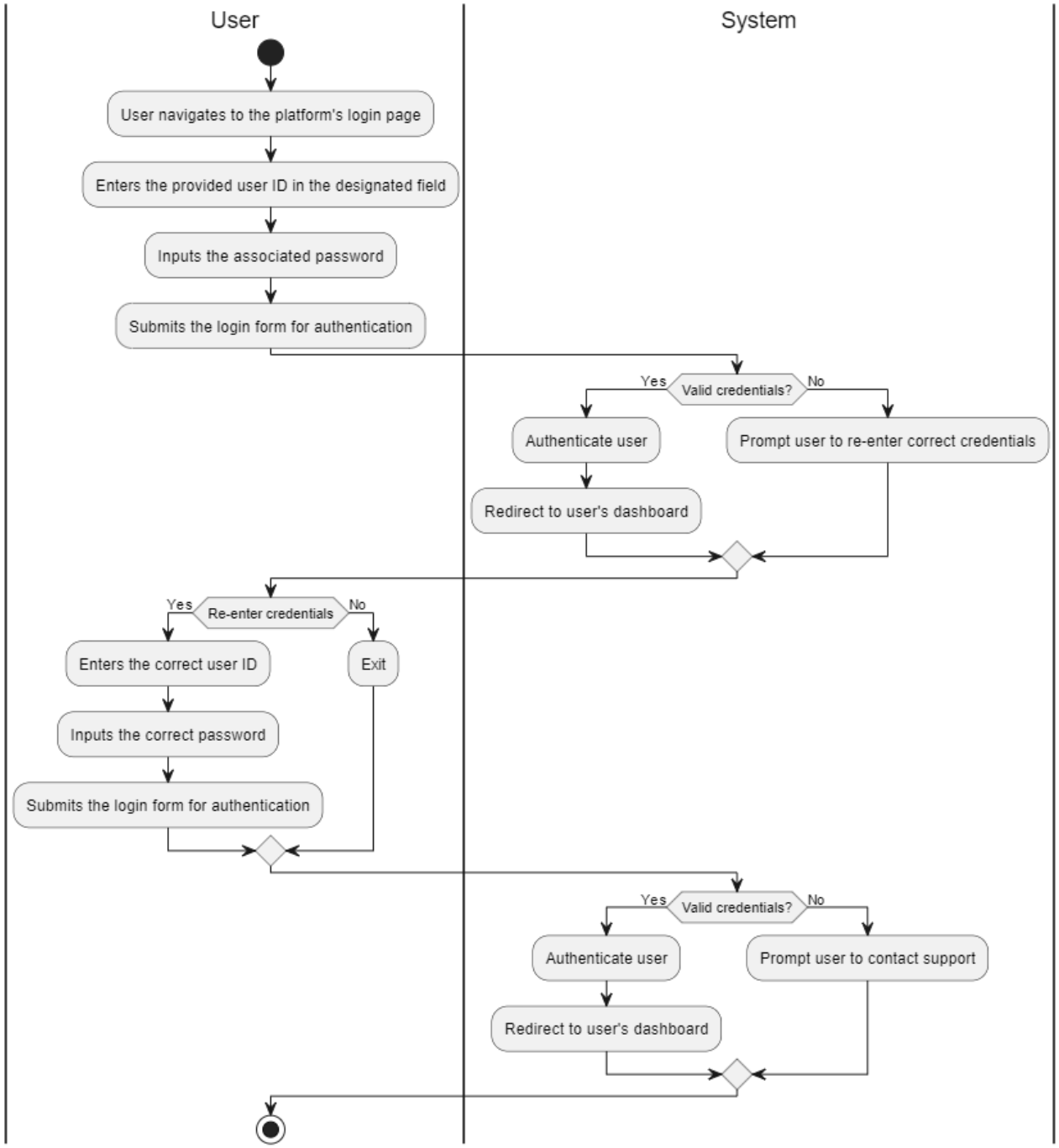


Evaluate Work:

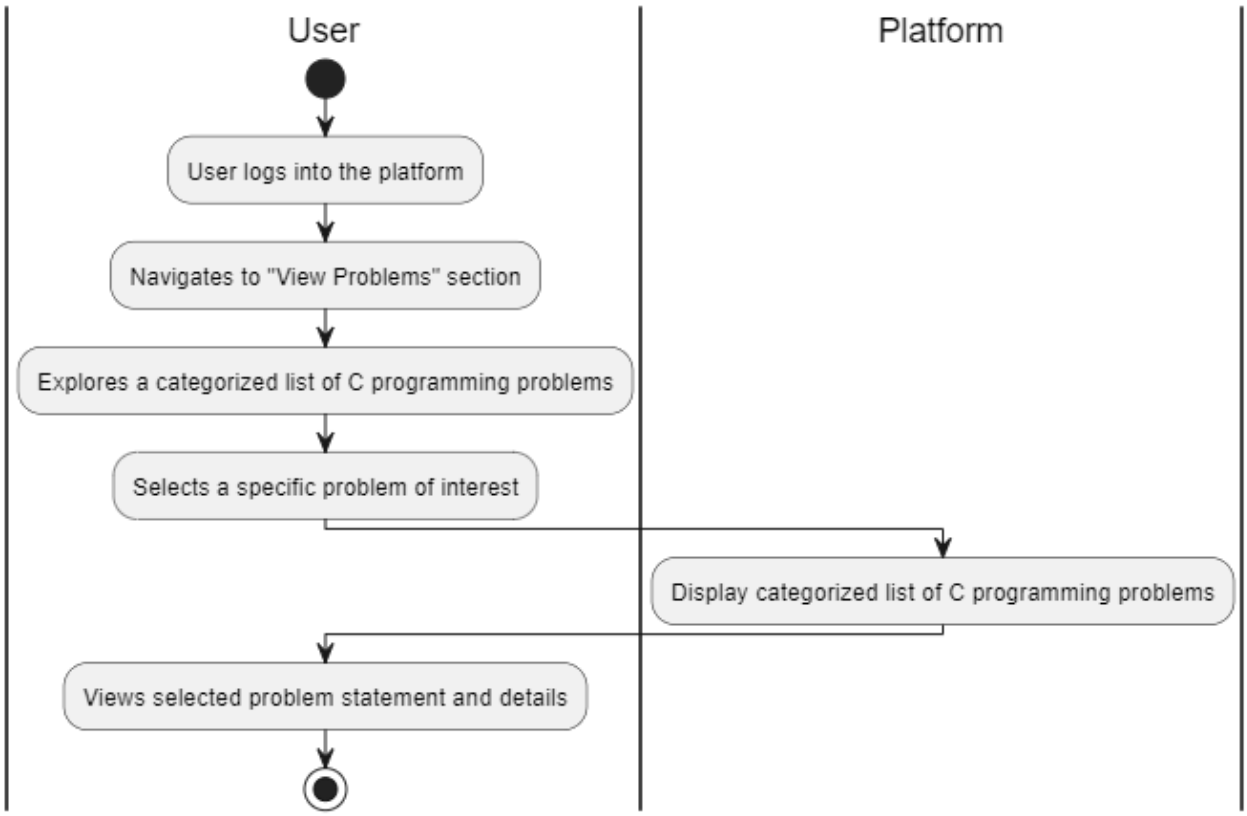


Swimlane Diagram:

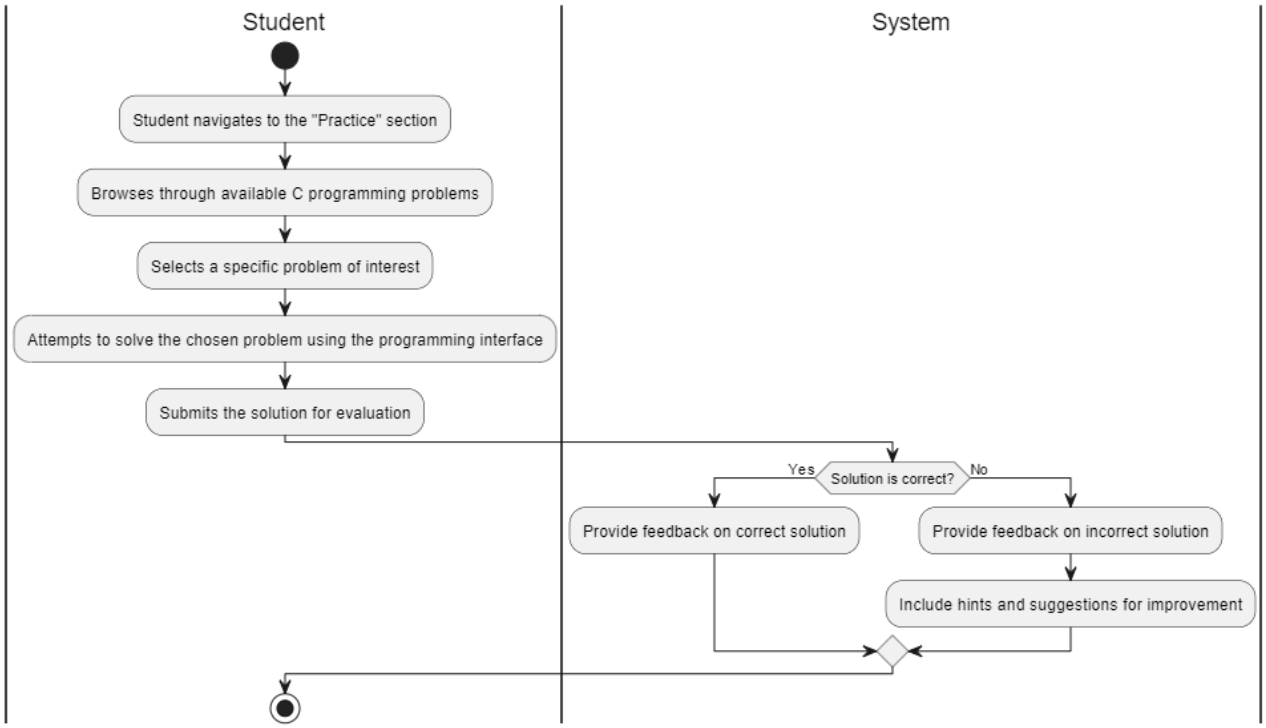
Login:



View Problems:



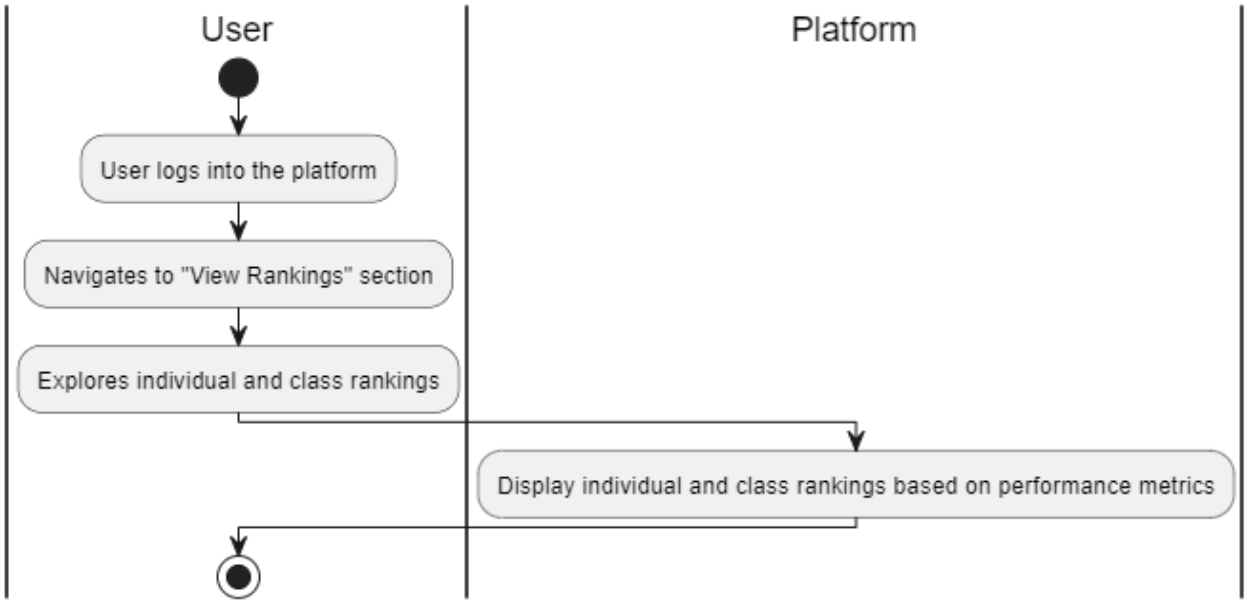
Practice Problems:



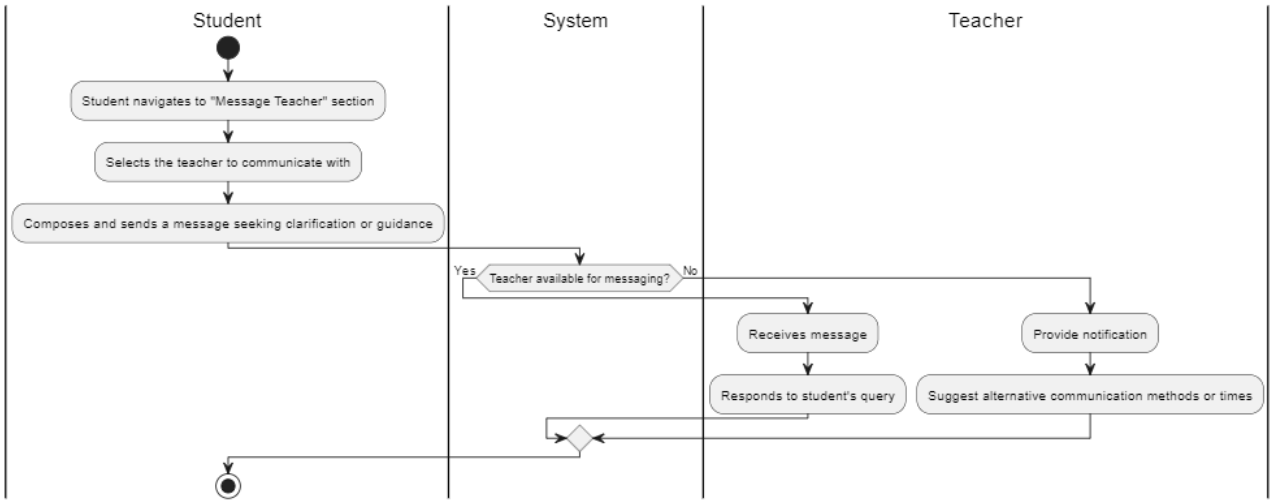
Get Hints:


Student
System
Student navigates to problem-solving interface
Attempts to solve a C programming problem
Requests hints for additional guidance
Check if user has received three hints
Yes
No
Notify user that no more hints are available
Provide the first hint
User continues solving the problem
After 30 minutes, check if user has received second hint
Yes
No
Notify user that the next hint is available
Wait until 30 minutes have passed from the first hint
Provide the second hint
User continues solving the problem
After one hour from the first hint, provide the third hint
User continues solving the problem or decides to stop

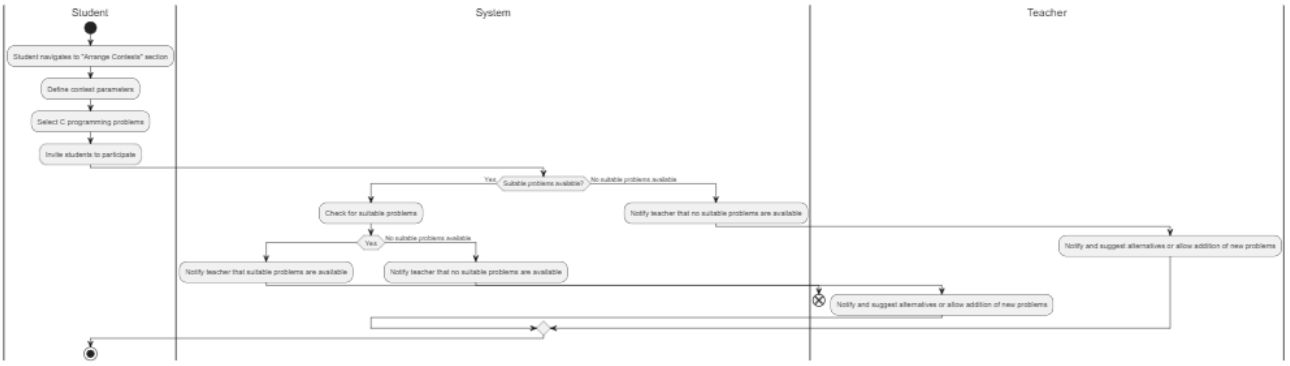
View Rankings:



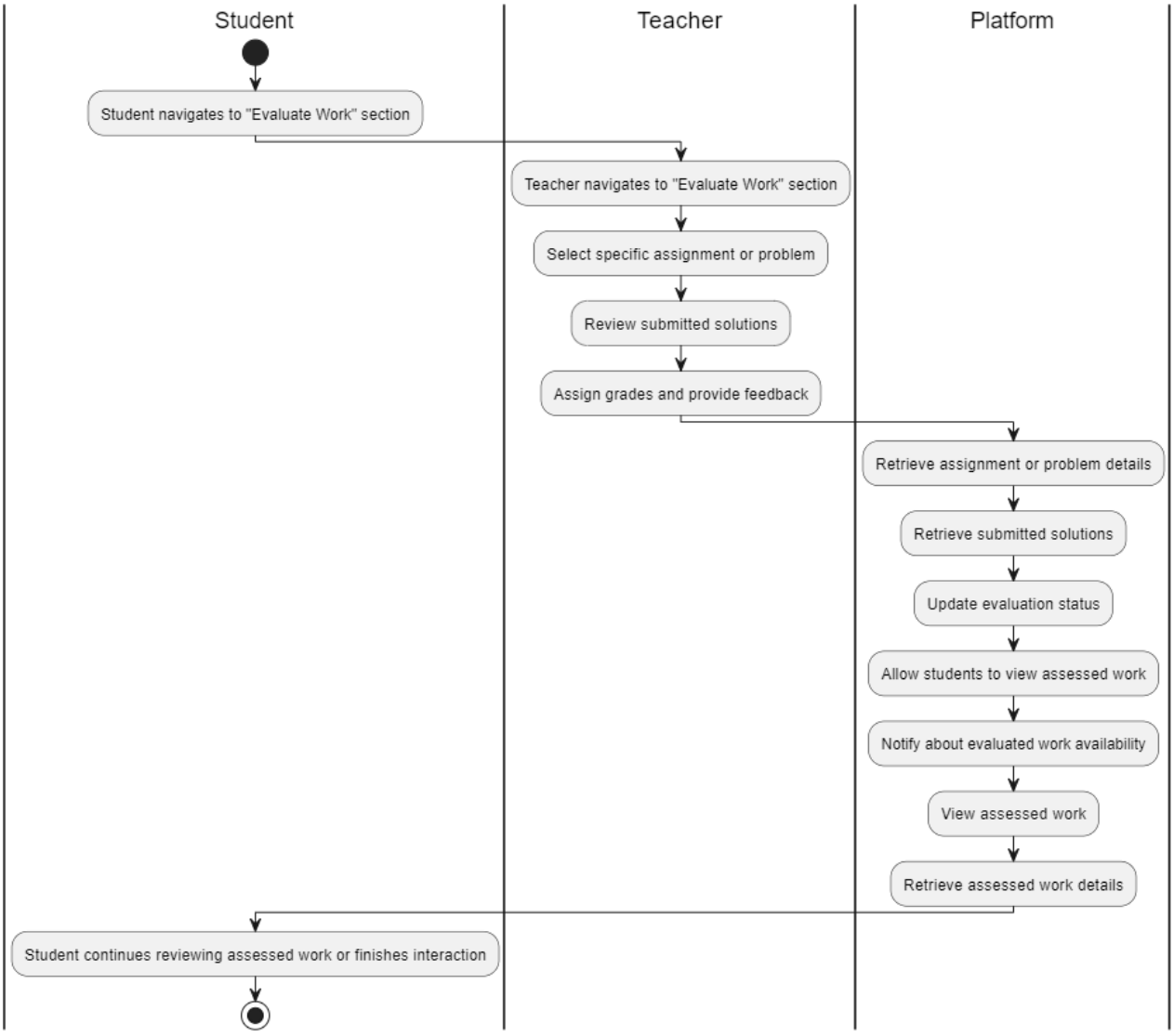
Message Teacher:



Arrange Contest:

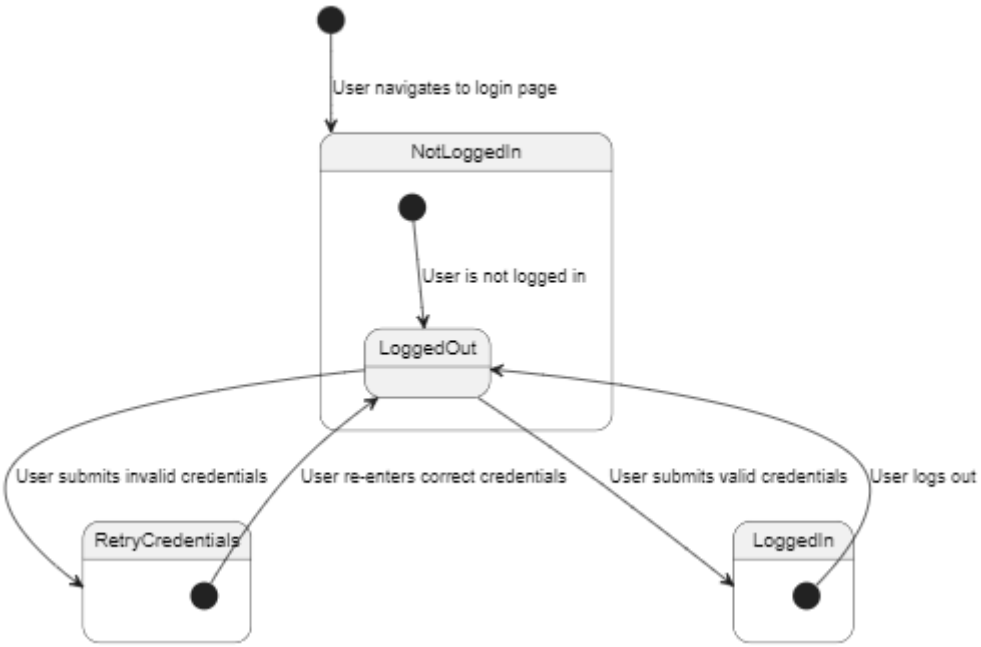


Evaluate work:

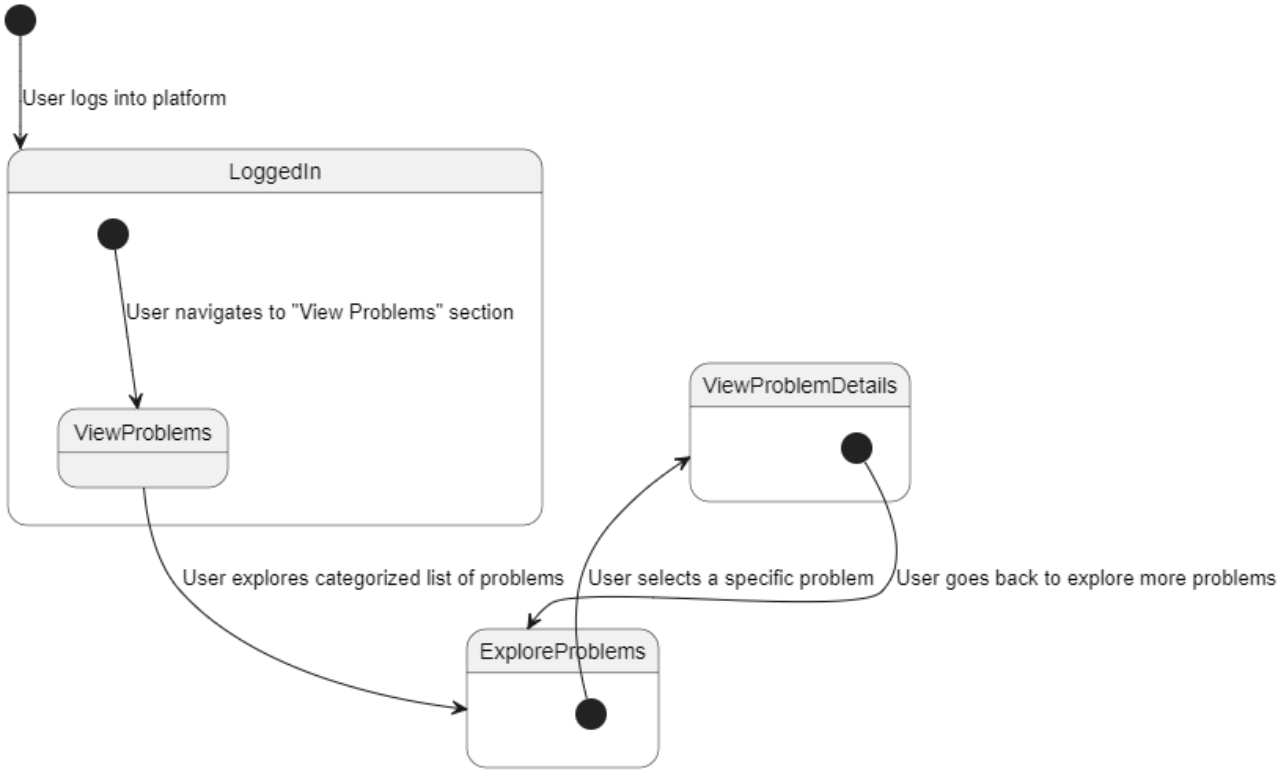


State Diagram:

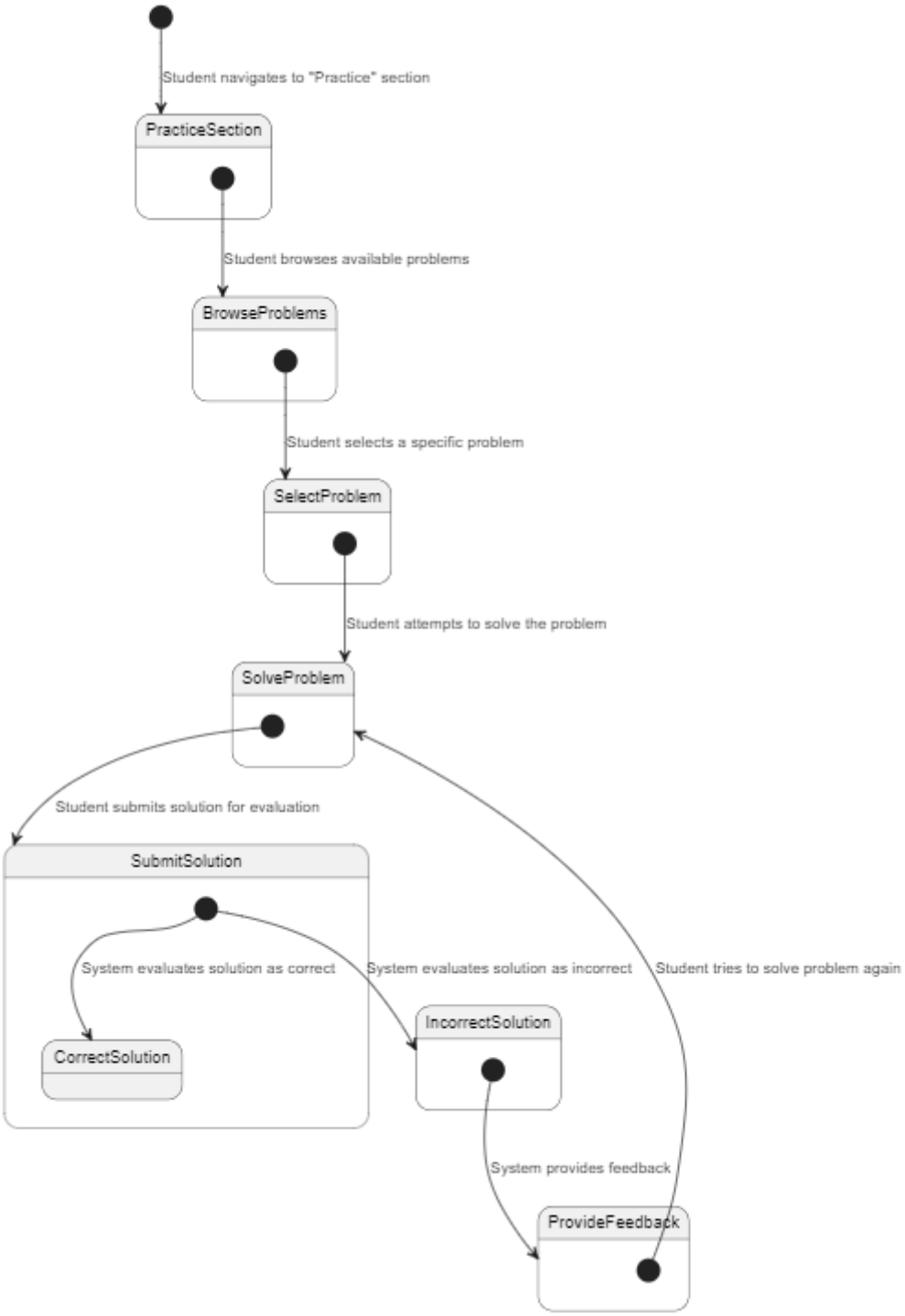
Login:



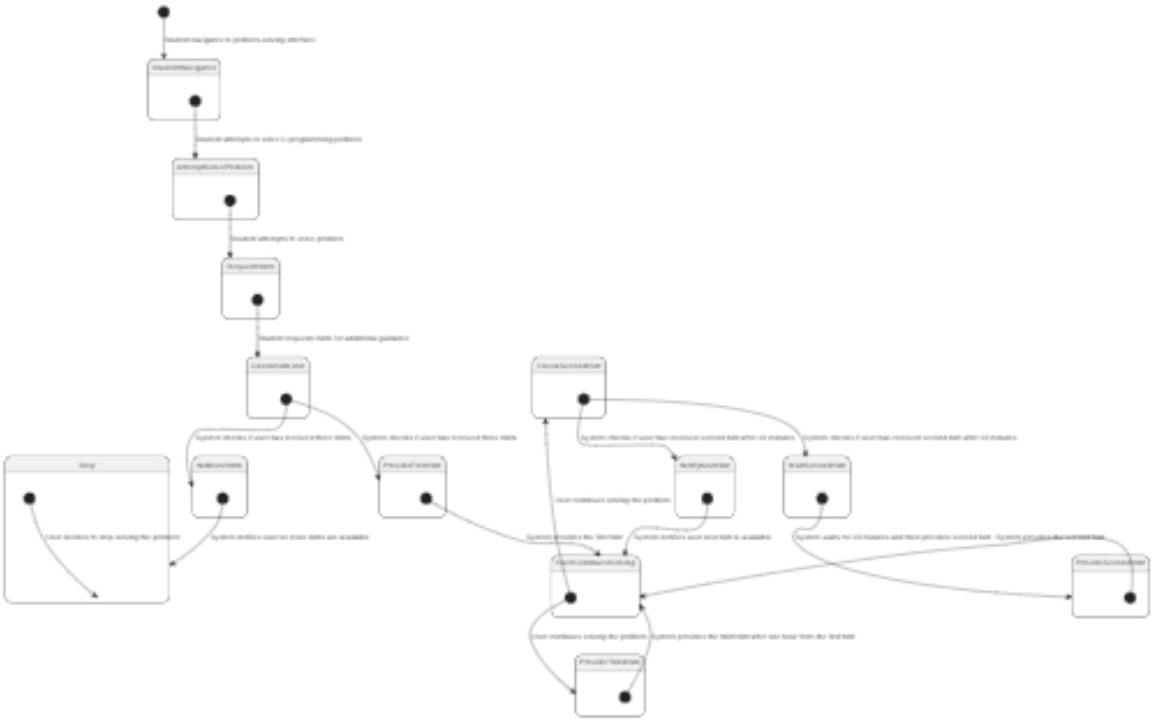
View Problem:



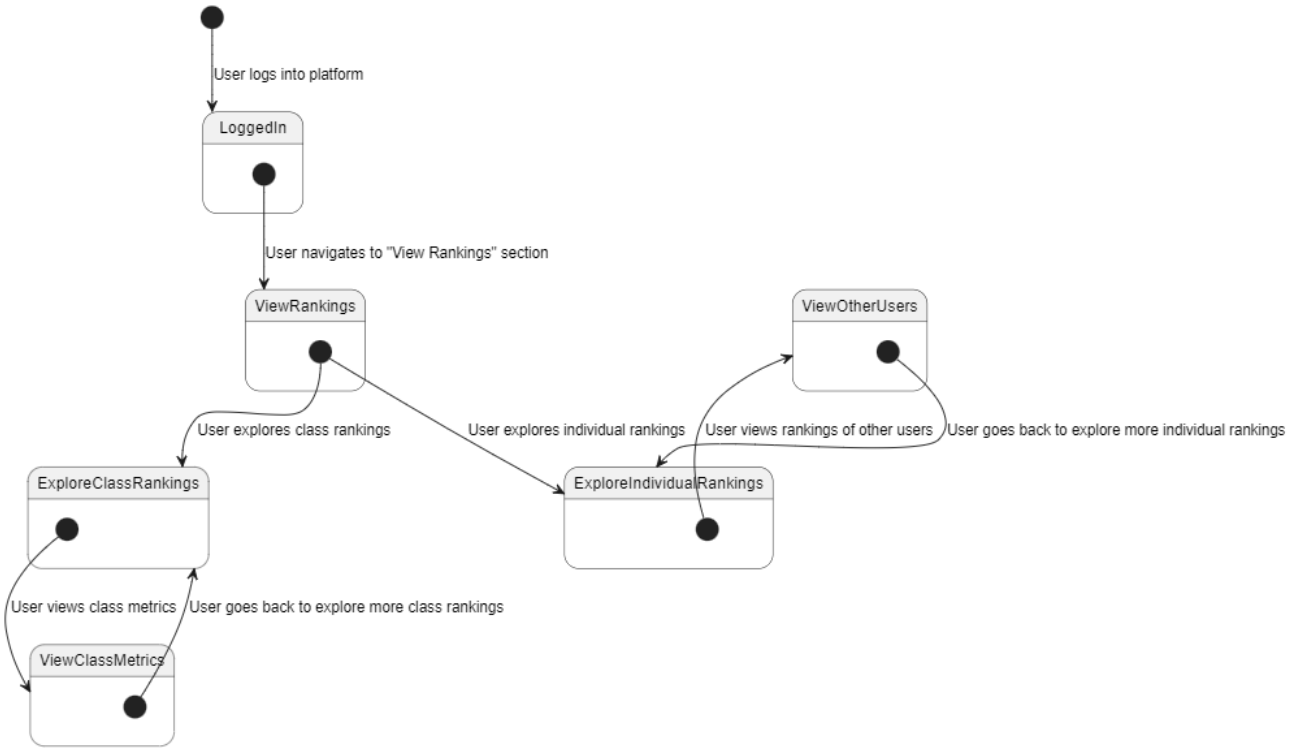
Practice Problem:



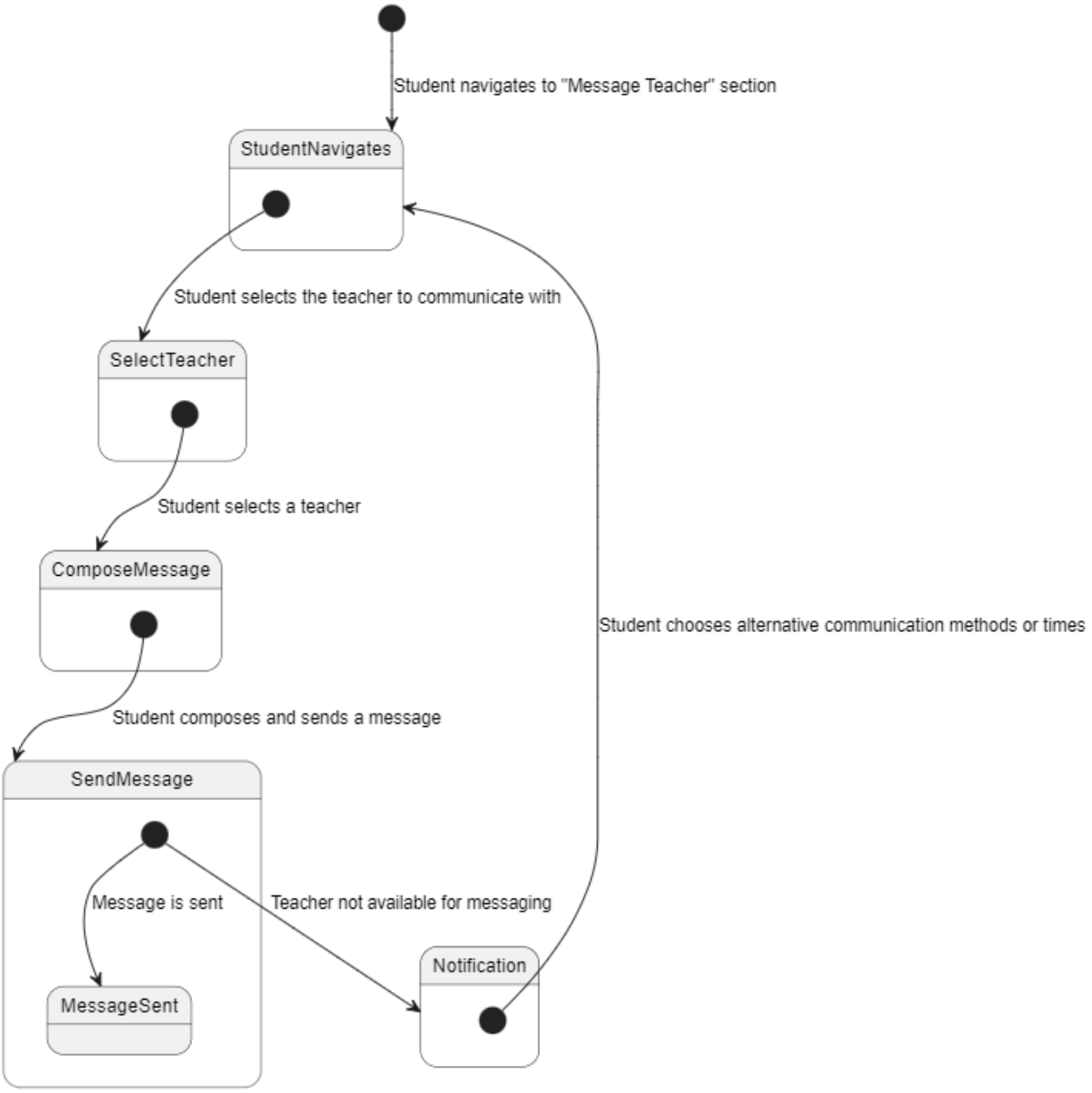
Get Hints:



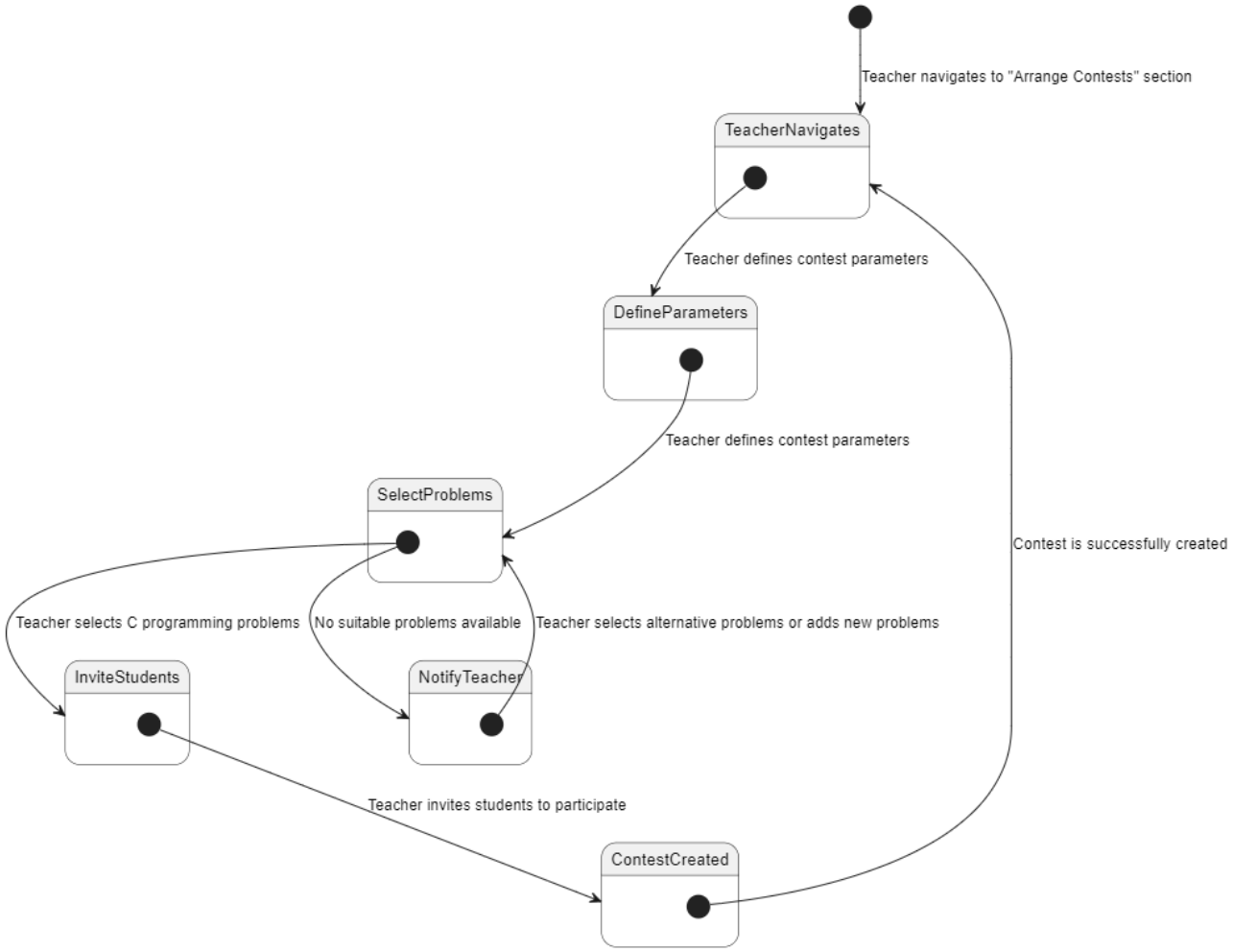
View Ranking:



Message Teacher:



Arrange Contest:



Evaluate Work: