**TABLE OF CONTENT**

1. Introduction

1.1 Project Details: Broad specifications of the work entrusted to you

1.2 Purpose

1.3 Scope

1.4 Objective

1.5 Technology and Literature Review

2. Project Management

2.1 Feasibility Study

2.1.1 Technical feasibility

2.1.2 Time Schedule feasibility

2.1.3 Operational feasibility

2.1.4 Implementation feasibility

2.2 Project Planning

2.2.1 Project development Approach and Justification

2.2.2 Project Plan

2.2.3 Role and Responsibilities

3. System Requirements Study

3.1 Study of Current System

3.2 Problems and Weakness of Current System

3.3 User Characteristics

3.4 Hardware and Software Requirements

3.5 Constraints

4. System Analysis

4.1 Requirements of New System (SRS)

4.1.1 Functional Requirements

4.1.2 Non-functional requirements

5. System Design

5.1 Use Case Diagram

5.2 Class Diagram

5.3 Sequence Diagram

5.4 Activity Diagram

6. Implementation Planning

6.1 Implementation Environment

6.2 Program Specification

7. Testing

7.1 Testing Plan

7.2 Testing Strategy

7.3 Testing Methods

7.4 Test Cases

8. User Manual

9. Limitation and Future Enhancement

10. Conclusion and Discussion

10.1 Conclusions

10.2 Discussion

10.2.1 Self Analysis of Project Viabilities

10.2.2 Problem Encountered and Possible Solutions

10.2.3 Summary of Project work

11. Bibliography

1.1 Project Details

The broad specification of work for a full-stack development project typically involves various tasks and responsibilities across the front-end and back-end development. Here's an outline of the work that may be entrusted for this project :

1. Project Planning and Requirement Analysis.
2. Develop responsive and user-friendly web pages using HTML, CSS, and JavaScript
3. Implement interactivity and dynamic content using front-end frameworks like React js.
4. Choose and implement an appropriate back-end technology stack (e.g., Node.js, Django, Flask, Ruby on Rails, etc.).
5. Design and implement the database schema based on the application's data model
6. Implement authentication and authorization mechanisms.
7. Conduct unit testing for both front-end and back-end components.
8. Optimize code and infrastructure for performance

1.2 Purpose

The main purpose of making this web application to have benefits of more than one application in one web application. Teachers have to use google classroom for uploading materials, for sending notice they use gmail and for uploading marks attendance use personal college websites thus we bring one application to get reed of using 4 application and use this one website to fulfil their needs efficiently.

1.3 Scope

The scope of website is broad and continually evolving as technology advances and user needs change.

This web application is educational website which encompasses various features which would be beneficial to not only student but faculties as well.

As the workload increases people find a way which can ease their word and would not be time consuming so this will bi helpful in full filling those needs as well.

1.4 Objective

The objective is to make life easier. To disseminate the educational content and resources. To make relationship of teacher and students more stronger and students can also connect with each other.

1.5 Technology and Literature Review

4. System Analysis

4.1 Requirements of New System (SRS)

4.1.1 Functional Requirements

1.Login

* Input: User credentials (username and password).
* Process: Validate user credentials against the stored data. Grant access upon successful authentication.
* Output: Successful login message. Error message for unsuccessful login attempts.

2. Attendance

* Input: User identification (e.g., student ID or employee ID). Date and time stamp.
* Process: Record attendance data in a database . Generate attendance reports.
* Output: Confirmation of attendance recorded. Attendance reports for users and administrators.

3. Avatar Selection:

* Input: User's choice of avatar or uploaded image.
* Process: Store user-selected avatars.
* Output: Confirmation message for successful avatar selection. Display selected avatar in the user profile.

4. Chatbox:

* Input: Text messages from users.
* Process: Real-time message delivery. Store chat history.
* Output: Display incoming messages. Notification for new messages.

5. Material Uploading Section:

* Input: Educational materials (documents, presentations, etc.).
* Process: Upload and store materials. Categorize materials based on user-provided metadata.
* Output: Confirmation of successful material upload. Material repository with search and categorization features.

6. Grades Section

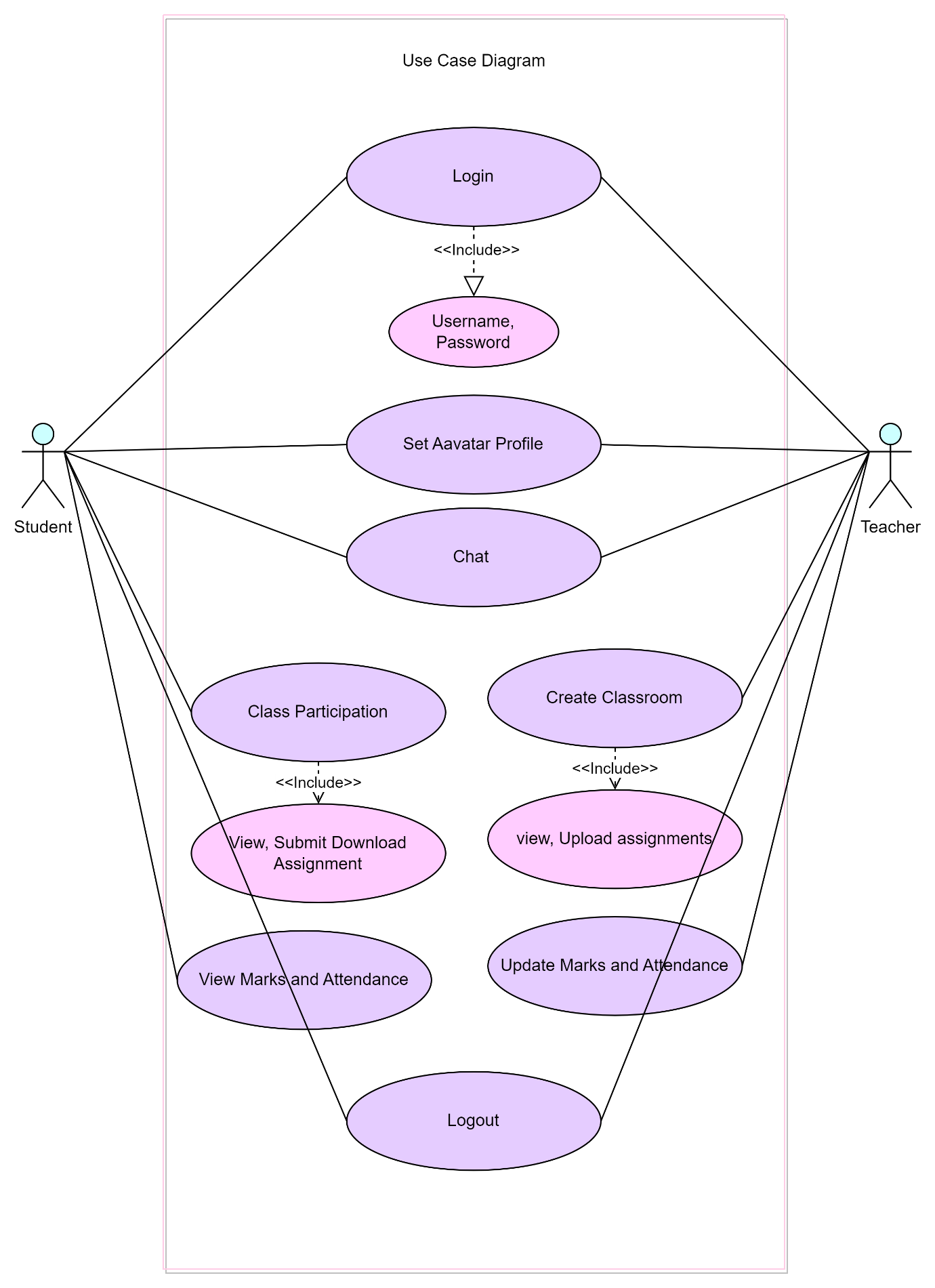
* Input: Student ID, course details, and grades.
* Process: Calculate and store grades. Generate grade reports.
* Output: Confirmation of successful grade input. Grade reports for users and administrators.

7.Search Functionality

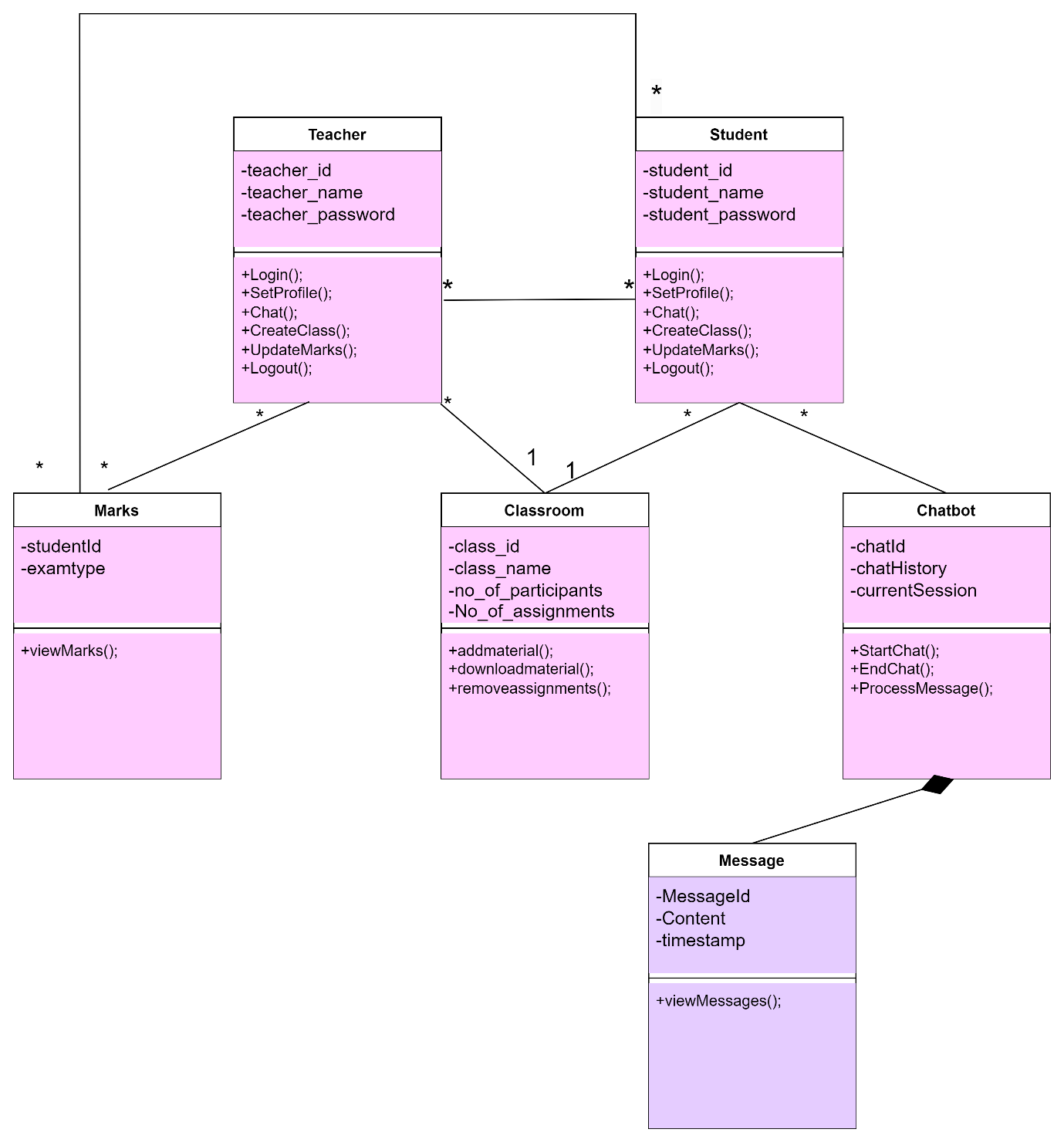
* Input: Search queries from users.
* Process: Search through materials, user profiles, or other relevant data.
* Output: Display search results.
  + 1. Non-Functional Requirements

1. Performance: The system should respond to user requests within 2 seconds.
2. Security: User passwords should be securely hashed and stored. SSL/TLS should be implemented for secure data transmission.
3. Scalability: The system should support up to 10,000 simultaneous users.
4. Usability: The user interface should be intuitive, requiring minimal training for users.
5. Reliability: The system should have a 99.9% uptime.
   1. UML DIAGRAMS

4.2.1 Use Case Diagram

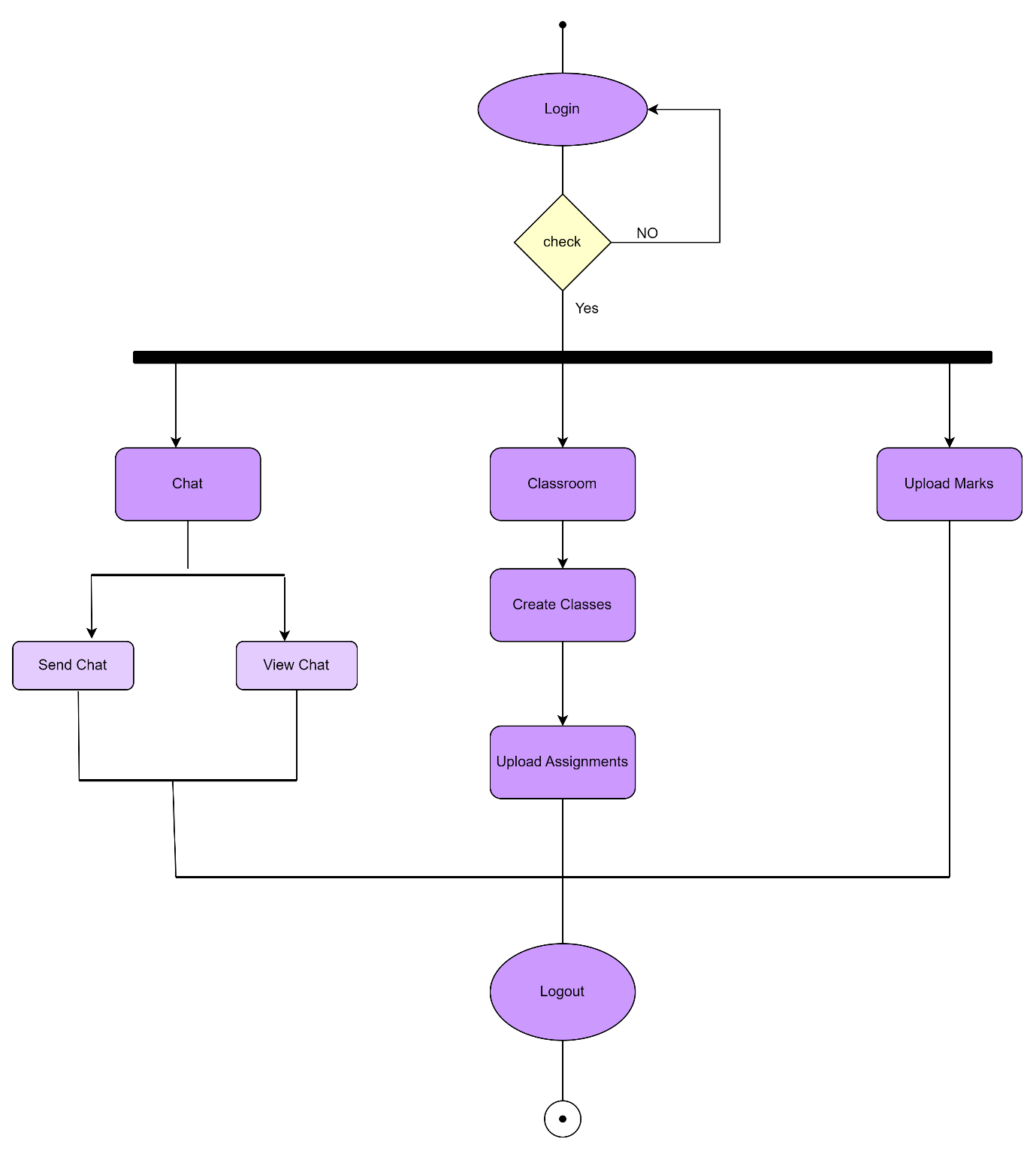


4.2.2 Class Diagram

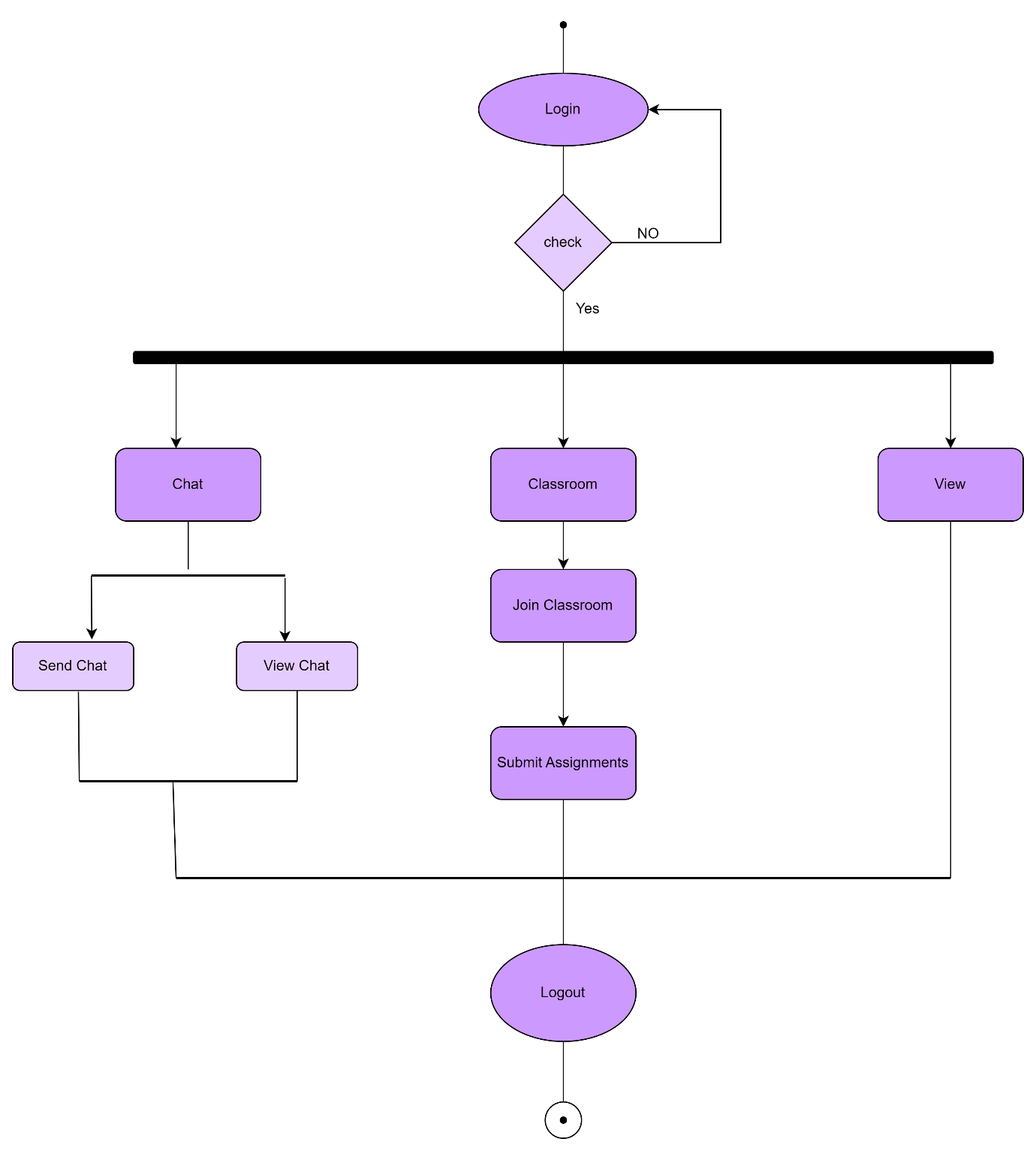


4.2.3 Activity Diagram

Teacher

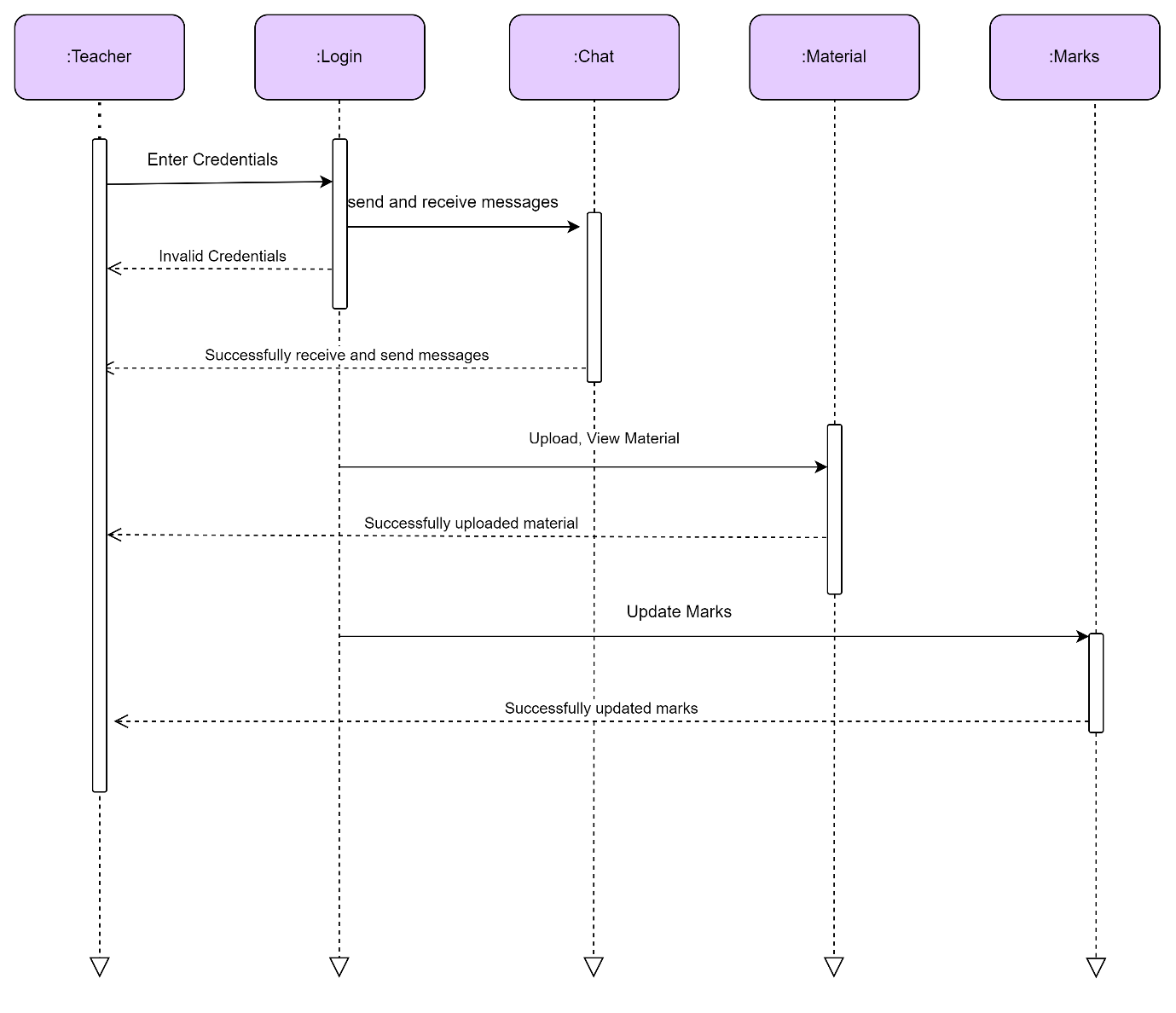


Student



4.2.4 Sequence Diagram

Teacher



Student

