Software Requirements Specification for

Online Examination System

Prepared by Hammad Javed
23CAMSA110
GK8310
MCA IIIrd semester

Aligarh Muslim University

Submitted to:
Prof. Aasim Zafar
Prof. Swaleha Zubair
Dr. Asif Irshad Khan
Ms. Priti Bala
Mr. Imshad Ahmad Khan

Contents

1. Introduction

1.1.Purpose

The purpose of the online examination system is to enhance the process of conducting and managing examinations in the Department of Computer Science at Aligarh Muslim University by leveraging digital technologies. Traditional examination methods, such as pen-and-paper tests, are often inefficient and require significant administrative effort in both preparation and evaluation. This system addresses these issues by providing an efficient, user-friendly, and secure platform that automates exam creation, distribution, and assessment.

One of the core goals of this system is to offer a platform that is especially suitable for conducting programming exams, which are often challenging to evaluate using conventional methods. The system allows students to solve programming questions directly on the platform, streamlining the evaluation process for teachers. This also ensures that the system remains highly relevant in an educational environment focused on computer science, where technical skills are crucial.

The system also aims to improve convenience for both students and teachers. Students can view upcoming exams, track their academic progress, and attempt exams in a controlled environment. Teachers benefit from tools to create and schedule exams, monitor student activity during exams, and evaluate submissions efficiently, ensuring that students receive timely feedback on their performance.

Security is another critical aspect of the system's purpose. The platform incorporates security measures such as password hashing, logging student activity during exams, and disabling unnecessary keys to prevent cheating. By creating a secure and reliable examination process, the system fosters academic integrity.

In summary, the online examination system focuses on enhancing efficiency, convenience, and security in the examination process, while providing a dedicated platform for the evaluation of programming skills and ensuring that all stakeholders benefit from a streamlined and modern approach to assessments.

1.2.Document Conventions
- This document follows the IEEE formatting requirements.
- Used Times New Roman, font size 12 throughout the document for text.
- Used italics for comments.
- Document text is single spaced and maintained the 1" margins.
- The format for headings is as followed:
- Major headings are in bold 16pt font and subheadings in bold 14spt font.

1.3. Intended Audience and Reading Suggestions

The primary audience for this document includes the following stakeholders:

Teachers: Responsible for creating, managing, and grading exams. This document will guide them on how to use the system to efficiently create exams, evaluate students, and generate progress reports.

Students: The system allows students to take exams, view their results, and track their academic progress. This document will help students understand the process of attempting exams and how to access their past performance.

System Administrators: They are responsible for deploying, maintaining, and troubleshooting the system. The document will serve as a reference for understanding the technical setup, configuration, and maintenance tasks.

Department of Computer Science, Aligarh Muslim University: As the primary implementing body, the department will use this document to understand the scope, functionality, and deployment requirements of the system.

Reading Suggestions:

Teachers and students can focus on the sections relevant to their respective roles, such as system features and user interfaces.

System administrators should review the sections on technical requirements, external interfaces, and system maintenance.

This document aims to provide comprehensive guidance on using the system efficiently, tailored to the needs of each audience group.

1.4. Product Scope

The online examination system is designed to facilitate and streamline the process of conducting exams in the Department of Computer Science at Aligarh Muslim University. Its primary goal is to create a platform that is efficient, secure, and user-friendly for both teachers and students. The system enables teachers to create exams with different types of questions, including multiple-choice questions (MCQs), subjective questions, and programming problems. Students can take exams in a controlled environment, and their submissions are evaluated by teachers through an integrated grading and feedback system.

The system is not limited to just exam creation and conduction but also extends its functionality to providing comprehensive progress reports. It allows students to view their past exam performances and track their academic progress, while teachers can monitor the performance of individual students or entire batches. This helps in identifying strengths and areas that need improvement, making the learning process more effective.

A key component of the system is its suitability for programming exams. Students can write and submit code directly within the platform, which is automatically tested or evaluated by teachers. This ensures a smooth workflow for handling technical assessments, which are critical in a computer science-focused curriculum.

In terms of accessibility, the system is designed to be used on department systems for conducting exams, ensuring a secure environment. However, other functionalities such as viewing exam schedules and progress reports can be accessed on any device with a web browser, providing flexibility and ease of use.

The system also incorporates essential security features like password hashing, logging student activity during exams, and disabling unnecessary keys during the exam period to maintain academic integrity. Overall, the system aims to enhance the examination process, reduce administrative burden, and create a robust platform for continuous learning and evaluation.

1.5.References

1. "IEEE Guide for Software Requirements Specifications," in IEEE Std 830-1984 , vol., no., pp.1-26, 10 Feb. 1984, doi: 10.1109/IEEESTD.1984.119205.

Retrieved September 17, 2022 from https://web.cs.dal.ca/~hawkey/3130/srs_template-ieee.doc

2. Rajarman, V. (1989), Analysis and Design of Information Systems, Prentice Hall of India. . Retrieved September 21, 2022.

3. Roger S. Pressman & David Lowe (2009), Web Engineering: A Practitioner's Approach, McGraw-Hill. Retrieved September 11, 2022

4. *SDLC - Agile Model*. (n.d.-b). Retrieved September 25, 2022, from https://www.tutorialspoint.com/sdlc/sdlc_agile_model.htm

5. *UML Use Case Diagram Tutorial*. (n.d.-b). Lucidchart. Retrieved October 25, 2022, from https://www.lucidchart.com/pages/uml-use-case-diagram

6. *UML Activity Diagram Tutorial*. (n.d.). Lucidchart. Retrieved October 27, 2022, from https://www.lucidchart.com/pages/uml-activity-diagram

7. Bobde, Shubham, et al. "Web based online examination system." Global Res Develop J Eng 2.5 (2017)

8. MongoDB Documentation. https://docs.mongodb.com/

9. Express.js Documentation. https://expressjs.com/

10. React Documentation. https://reactjs.org/docs/getting-started.html

11. Node.js Documentation. https://nodejs.org/en/docs/

12. Git Documentation. https://git-scm.com/docs/git

13. Computer Based Testing (CBT) exam. (n.d.). https://www.geeksforgeeks.org/cbt-fullform/#what-is-computer-based-testing-cbt-examination.

14. Data Flow Diagram (DFD). https://www.geeksforgeeks.org/what-is-dfddata-flowdiagram/

2. Overall Description

2.1. Product Perspective

The online examination system is developed to address the specific needs of the Department of Computer Science at Aligarh Muslim University. It replaces traditional paper-based exams with a more efficient and automated digital platform. This system operates as a standalone application, though it integrates seamlessly with the department's existing computer infrastructure to provide a controlled and secure environment for conducting exams.

The system's primary function is to facilitate exam creation, execution, and evaluation, particularly for programming-related assessments. Teachers can set up exams with various types of questions, including multiple-choice, subjective, and coding challenges. The platform allows students to attempt these exams, submit their responses, and receive feedback or grades.

Unlike general-purpose examination platforms, this system is specifically tailored for computer science exams, where coding problems are a critical component. The platform ensures that students can submit code for evaluation, and teachers can review it with ease, thereby streamlining the assessment process for programming subjects.

The system is designed to be accessible via any browser for non-examination functionalities, such as viewing exam schedules and results. However, exams themselves are conducted in a secure, department-monitored environment to maintain the integrity of the assessment process. The system is scalable and capable of handling a large number of users, ensuring that multiple exams can be conducted simultaneously without performance degradation.

2.2. Product Functions

The online examination system offers several core functionalities designed to streamline the process of conducting and managing exams for both teachers and students. These functions include:

1. **Exam Creation**:
   - Teachers can create exams by adding multiple types of questions, including multiple-choice questions (MCQs), subjective questions, and programming problems.
   - Teachers can specify the time duration for each exam, set deadlines, and assign weightage to different sections or questions.
2. **Exam Management**:
   - The system provides an interface for teachers to schedule exams and manage them through a centralized dashboard.
   - Teachers can view a list of upcoming, ongoing, and past exams, making it easier to manage multiple exams simultaneously.
3. **Student Participation**:
   - Students can log in to their accounts, view upcoming exams, and attempt exams at the scheduled time.

- During exams, students can respond to different types of questions and, in the case of programming questions, submit their code through the platform.

4. **Automatic and Manual Evaluation**:
    - For objective questions like MCQs, the system can automatically grade student submissions.
    - Teachers can manually evaluate subjective answers and programming submissions, providing detailed feedback.

5. **Progress Tracking**:
    - The system generates progress reports for students, displaying their performance across different exams. It also provides batch-level insights to help teachers analyze overall class performance.

6. **Security Features**:
    - The system logs student activities during the exam, disables unnecessary keys, and implements password hashing to ensure a secure testing environment.

2.3. User Classes and Characteristics

The online examination system is designed for use by three primary classes of users, each with distinct roles and characteristics:

1. **Teachers**:
    - **Role**: Teachers are responsible for creating exams, managing exam schedules, and evaluating student submissions. They can also generate progress reports for individual students or batches.
    - **Characteristics**: Teachers are expected to have a moderate level of familiarity with computer systems, including the ability to input exam data, manage exams, and evaluate responses. They need to be able to use the platform efficiently to create various types of questions, such as MCQs, subjective questions, and programming challenges.
    - **Access Level**: Teachers have administrative privileges to create and edit exams, view results, and provide feedback.

2. **Students**:
    - **Role**: Students are the primary users for whom the system is built. They use the platform to attempt exams, view their exam schedules, and track their academic progress.
    - **Characteristics**: Students are expected to have basic computer literacy and familiarity with web-based interfaces to navigate the system, attempt exams, and submit answers, particularly for coding challenges in programming subjects.
    - **Access Level**: Students have restricted access, allowing them to view their exam schedules, attempt exams, and check their results and progress reports.

3. **Admin**:

- **Role**: The admin oversees the overall operation of the system, including user management (both students and teachers), system configuration, and troubleshooting.
- **Characteristics**: Admin users typically possess advanced technical knowledge and are responsible for maintaining the platform, ensuring its smooth operation, and handling any technical issues that may arise.
- **Access Level**: Admin have full system access, including the ability to manage users, monitor system performance, and ensure the security of the platform.

2.4. Operating Environment

The online examination system is designed to operate in both controlled and flexible environments, depending on the type of user interaction and system functionality. The system has the following operating environment characteristics:

1. **Exam Conduction Environment**:
   - **Location**: The primary environment for conducting exams is within the Department of Computer Science at Aligarh Muslim University. Specifically, exams are conducted on department-managed computers to ensure a secure and controlled testing environment.
   - **Hardware Requirements**: The department systems used for exams should be equipped with standard desktop or laptop configurations, including a functional keyboard, mouse, and a stable internet connection. The systems should also be capable of running a modern web browser (such as Google Chrome, Mozilla Firefox, or Microsoft Edge).
   - **Security Measures**: During exams, features like keylogging, activity monitoring, and disabling of unnecessary keys (such as function keys and copy-paste shortcuts) will be enabled to maintain exam integrity.
2. **General Access Environment**:
   - **Location**: For non-exam functionalities, such as viewing exam schedules, progress reports, and accessing feedback, the system can be accessed from any device with a web browser. This allows students and teachers to interact with the platform from their personal devices (e.g., laptops, tablets, smartphones) outside of the controlled exam environment.
   - **Browser Requirements**: The system is designed to be cross-browser compatible, requiring any modern web browser that supports HTML5, JavaScript, and CSS3 for smooth functionality.
   - **Internet Requirements**: A stable internet connection is necessary for accessing the platform, especially during exams to ensure smooth data submission and real-time activity logging.

2.5.Design and Implementation Constraints

The design and implementation of the online examination system are subject to several constraints, which must be considered during its development and deployment:

1. **Limited Internet Connectivity**:

   - During exams, the system must function with minimal interruptions, which can be challenging in environments with unstable or low-speed internet connections. To mitigate this, the system needs to be optimized to handle temporary connectivity issues without losing data, especially during the submission of exam answers.

2. **Browser Compatibility**:

   - The system is designed to work on a variety of modern web browsers. However, maintaining cross-browser compatibility can introduce limitations in the use of advanced technologies or features. Ensuring that the system works seamlessly across different browsers, including Chrome, Firefox, and Edge, may restrict the adoption of certain browser-specific features.

3. **Security**:

   - Security is a top priority, particularly during exams. The system must implement strict security protocols such as password hashing, activity logging, and disabling unnecessary keyboard functions. These security measures must be carefully balanced with usability, ensuring that they do not overly complicate the user experience or cause performance lags.

4. **Real-time Monitoring and Logging**:

   - The system requires real-time logging of student activities during exams to prevent cheating. This necessitates efficient back-end processing to capture and store these logs without significantly impacting system performance.

5. **Scalability**:

   - The system must handle a potentially large number of users (students and teachers) accessing it simultaneously during exams. The design must ensure that it is scalable, with the ability to manage increased loads without performance degradation.

6. **Programming Environment**:

   - Since the system supports coding exams, there are constraints related to integrating a programming environment within the platform. It must be able to execute, test, and evaluate code submissions, which can be resource-intensive and requires a secure sandbox environment to avoid potential security risks.

2.6. User Documentation

The online examination system will be accompanied by comprehensive user documentation to guide different user groups (teachers, students, and administrators) through the system's functionalities. The documentation will cover the following aspects:

1. **Installation and Setup Guide**:

   - This section will provide step-by-step instructions for system administrators on how to install, configure, and deploy the online examination system on department servers or any other necessary infrastructure.

   - It will include details on software and hardware prerequisites, database setup, and system initialization processes.

2. **User Manual for Teachers**:

   - A detailed guide tailored to teachers, explaining how to create exams with different question formats (MCQs, subjective questions, programming problems), schedule exams, manage exams, evaluate submissions, and generate progress reports.

   - Instructions on how to provide feedback and remarks on student performances will also be included.

3. **User Manual for Students**:

   - A guide for students covering how to log in, view upcoming exams, attempt exams, submit answers (especially programming solutions), and check their exam results and progress reports.

   - The manual will also explain the system's security features and what actions are restricted during the exam to maintain exam integrity.

4. **Administrator Guide**:

   - This guide will help administrators manage user roles (teachers, students), monitor system performance, handle troubleshooting, and ensure data security.

   - It will also include instructions for maintaining the system, such as backing up the database and monitoring user logs.

5. **FAQ and Troubleshooting Section**:

   - A section dedicated to answering common questions that users may have and providing solutions to common technical issues.

2.7. Assumptions and Dependencies

The development and operation of the online examination system are based on certain assumptions and dependencies that influence its functionality and performance:

1. **Assumptions**:

   - **Reliable Internet Access**: It is assumed that all users, especially during exams, will have access to a stable and reliable internet connection. The system depends on this for real-time communication between the client and server, particularly for features such as live activity logging and timely exam submissions.

   - **User Familiarity with Technology**: It is assumed that teachers and students have a basic understanding of using computers and web applications. The system is designed to be user-friendly, but it assumes that users are comfortable with accessing a web-based platform for exam-related activities.

   - **Standardized Devices for Exams**: It is assumed that the department's computer systems used for exams are standardized and meet the necessary hardware and software requirements (e.g., modern browsers, adequate processing power, and memory).

   - **Scheduled Maintenance**: The system will require periodic maintenance, including software updates and database backups. It is assumed that there will be designated times for this maintenance to avoid disruptions during critical exam periods.

2. **Dependencies**:

   - **Browser Compatibility**: The system depends on modern web browsers (such as Chrome, Firefox, and Edge) for proper functionality. It is dependent on the continued support and updates from these browsers to ensure compatibility with the latest web technologies.

   - **Database Management System**: The system relies on a robust database management system (e.g., MySQL, PostgreSQL) to store exam data, user information, and logs securely. The performance and reliability of the database directly affect the system's functionality.

   - **Server Infrastructure**: The performance of the system is dependent on the server infrastructure used by the department. The server must be able to handle simultaneous user access during exams, ensuring smooth operation without lags or downtime.

   - **Third-Party Libraries and Tools**: The system may depend on third-party libraries or tools for certain functionalities, such as password hashing, real-time logging, and executing programming code for coding problems. The reliability and security of these libraries are crucial for the system's overall performance.

3. External Interface Requirements

3.1. User Interface

The online examination system features a user-friendly interface designed to cater to the needs of different user classes (teachers, students, and administrators). Key components of the user interfaces include:

- **Login Screen**: A secure login page for users to authenticate their credentials (username and password) before accessing the system. This screen includes options for password recovery.

- **Dashboard**:

  - **For Teachers**: A dashboard that displays upcoming exams, previously conducted exams, and options to create new exams. It also provides access to student performance reports and evaluation tools.

  - **For Students**: A dashboard showing upcoming exams, past exam results, and progress tracking. It allows students to easily navigate to available exams.

- **Exam Creation Interface**: An intuitive interface for teachers to design exams, allowing them to select question types, set timers, and customize exam settings.

- **Exam Attempt Interface**: A streamlined interface for students to attempt exams. This interface will include features for answering questions, submitting code, and saving progress.

- **Feedback and Evaluation Interface**: A section where teachers can view submitted answers, provide feedback, and assign grades.

3.2. Hardware Interface

The online examination system does not require specialized hardware; however, certain hardware configurations are recommended to ensure optimal performance:

- **Client Devices**: Users should have access to standard computers or laptops with:

  - A minimum of 4GB RAM for running a modern web browser.

  - A stable internet connection (minimum 5 Mbps) for seamless data transmission.

  - A functional keyboard and mouse for exam interaction.

- **Server Infrastructure**: The backend should be hosted on a server with:

  - Sufficient CPU and RAM to handle simultaneous user requests (at least 8GB RAM recommended).

  - Reliable storage for data management, preferably using SSDs for faster data retrieval.

  - Backup power supply or UPS to ensure uptime during exams.

3.3. Software Interface

The online examination system interacts with various software components to enhance functionality:

- **Web Server**: The system will run on a web server (such as Apache or Nginx) to handle user requests and serve web pages.

- **Database Management System (DBMS)**: The system will connect to a relational database management system (like MySQL or PostgreSQL) to store user data, exam questions, and results.

- **Programming Execution Environment**: For coding questions, the system may integrate with third-party APIs or services that allow code execution and evaluation. This interface must ensure secure execution to prevent unauthorized access or code injection.

- **Web Browsers**: The system will support standard web browsers (Chrome, Firefox, Edge) to ensure compatibility and accessibility.

3.4. Communications Interface

The system will employ various communication protocols to facilitate data exchange between users and the server:

- **HTTP/HTTPS**: The system will utilize HTTPS for secure communication between client devices and the server, ensuring data encryption and protection against eavesdropping.

- **Email Notifications**: The system will integrate with an email service to send notifications to users regarding exam schedules, results, and feedback.

4. System Features

The online examination system offers a range of features tailored to facilitate a seamless assessment experience for both students and teachers. The following are key system features:

1. **Exam Creation and Management**:

   o Teachers can create and configure exams, including setting parameters such as question type (MCQ, subjective, programming), time limits, and exam schedules.

   o Exams can be tailored to different classes or skill levels, allowing teachers to customize assessments according to their educational objectives.

2. **Student Exam Attempt and Submission**:

   o Students can view upcoming exams, select an available exam, and attempt it within the specified time frame.

   o Upon completion, the system automatically submits responses, preventing loss of data due to time constraints or technical issues.

3. **Automated and Manual Grading**:

   o Multiple-choice and programming questions are graded automatically based on predefined answers and test cases, reducing the grading workload for teachers.

   o Subjective answers are available for manual grading, with options for teachers to add detailed feedback and scores.

4. **Progress Report for Students and Batches**:

   o The system generates comprehensive progress reports, showing individual students' performance across exams and subjects, as well as aggregated performance metrics for each batch.

   o Students and teachers can view historical performance data, facilitating long-term tracking of academic progress.

5. **Optional Remark Feature for Each Question**:

   o Teachers have the option to add specific remarks for each question, providing personalized feedback on student responses. This feedback helps students understand areas of improvement and encourages a deeper understanding of the material.

6. **Activity Logging and Proctoring**:

   o To maintain exam integrity, the system logs student activities, including page navigation and idle time, allowing teachers to monitor any suspicious behaviors during exams.

o Proctoring controls, such as disabling copy-paste and limiting keyboard shortcuts, enhance exam security.

7. **Result Publication and Feedback**:

    o After grading, students receive their results with feedback and breakdowns by question. This transparency helps students reflect on their performance and identify strengths and areas for improvement.

    o Teachers can control when results are released, ensuring that evaluations are completed before scores are made available.

These system features collectively provide a structured, secure, and user-friendly examination platform that enhances the educational experience for both teachers and students.

5. Non-functional Requirements

5.1.Performance Requirements

The performance requirements for the online examination system are essential to ensure a smooth user experience, particularly during high-stakes examination periods. These requirements include:

1. **Response Time**:

    • The system should provide rapid response times for all user actions. For general navigation (e.g., moving between pages or sections), the system should respond within 2 seconds. During exams, it is crucial that actions like submitting answers or saving progress occur within 1 second to minimize disruption and anxiety for students.

2. **Concurrent User Support**:

    • The system must be capable of supporting a significant number of concurrent users, especially during peak exam times. It should handle at least 100 simultaneous users without degradation in performance. The architecture should be scalable to accommodate increased loads, such as when multiple classes or batches take exams concurrently.

3. **Load Handling**:

    • The system should be tested to ensure that it can sustain a load of at least 500 users accessing the system simultaneously during exam periods. Stress testing must be performed to validate this requirement, ensuring that performance remains stable under peak load conditions.

4. **Data Processing**:

    • The system must efficiently process exam submissions and grade evaluations. For multiple-choice questions, grading should occur in real time as submissions are made, providing instant feedback to students. For

programming questions, the system should evaluate code submissions within a maximum of 5 seconds.

5. **Availability**:

- The system should have an uptime of 99.5% during peak usage periods (especially during scheduled exams) to ensure that users can access it without interruptions. This includes considerations for scheduled maintenance, which should be conducted during off-peak hours to minimize impact on users.

6. **Scalability**:

- The architecture of the system must be designed for scalability, allowing for the addition of resources (e.g., servers, databases) to accommodate growing user demands without compromising performance. This may involve using cloud services or load balancing techniques.

7. **Backup and Recovery**:

- The system must implement robust backup and recovery protocols, ensuring that data is regularly backed up and can be restored quickly in the event of a failure. The recovery time objective (RTO) should be within 1 hour for critical functions.

5.2.Safety Requirements

Safety requirements for the online examination system focus on ensuring the protection of users and the integrity of the examination process. These requirements encompass physical safety measures, as well as safeguards against data breaches and system failures:

1. **Data Protection and Privacy**:

- The system must comply with relevant data protection regulations (e.g., GDPR, FERPA) to ensure the confidentiality and integrity of user data. Personal information, including student and teacher records, should be securely stored and encrypted in the database.

- Access to sensitive information must be restricted based on user roles, ensuring that only authorized personnel can view or modify user data.

2. **Secure Data Transmission**:

- All data transmitted between clients (students and teachers) and the server must be encrypted using SSL/TLS protocols. This prevents unauthorized access and eavesdropping during data transmission, safeguarding exam responses and personal information.

3. **Physical Security Measures**:

- For on-site examinations, the department should implement physical security measures to prevent unauthorized access to examination rooms and devices. This includes monitoring access points and ensuring that only authorized personnel are present during the exam.

4. **User Activity Monitoring**:

- The system must include activity monitoring features that log user actions during exams. This allows for the detection of suspicious behaviors (e.g., accessing unauthorized materials, multiple logins) and facilitates audits in case of any irregularities.

5. **Emergency Procedures**:

- The system should have defined emergency procedures to follow in case of unexpected events, such as server failures or natural disasters. This includes having a contingency plan for rescheduling exams or notifying users of system downtimes.

6. **Incident Response Plan**:

- An incident response plan must be established to address potential security breaches or system failures. This plan should outline the steps to be taken, including user notification, data recovery, and investigation procedures.

7. **User Education and Training**:

- Training materials and sessions should be provided to users (teachers, students, and administrators) to ensure they understand safety protocols. This includes best practices for password management, recognizing phishing attempts, and reporting suspicious activities.

8. **System Redundancy and Failover**:

- To enhance system reliability, redundant systems and failover mechanisms should be implemented. This ensures that if one component fails, an alternative is readily available to prevent service interruptions.

5.3. Security Requirements

Security requirements are critical for protecting the online examination system from unauthorized access, data breaches, and other threats. The following security measures are essential to ensure the integrity, confidentiality, and availability of the system:

1. **User Authentication**:

- The system must implement strong authentication mechanisms, including a combination of username and password, to verify user identities. Passwords should follow best practices, including complexity requirements (minimum length, use of special characters) and regular updates.

- Multi-factor authentication (MFA) should be considered to add an additional layer of security, requiring users to provide a second form of verification (e.g., a mobile authentication code).

2. **Access Control**:

- Role-based access control (RBAC) must be enforced to limit user permissions based on their roles (teacher, student, admin). This ensures that users can only access functionalities pertinent to their responsibilities, minimizing the risk of unauthorized actions.

- User roles should be reviewed regularly, with access rights updated promptly in response to personnel changes.

3. **Data Encryption**:

- Sensitive data, including user credentials, exam content, and results, must be encrypted both at rest (stored data) and in transit (data being transmitted). This will safeguard information from unauthorized access and breaches.

- The use of strong encryption algorithms (e.g., AES-256) is recommended to enhance data security.

4. **Regular Security Audits**:

- The system should undergo regular security assessments and audits to identify vulnerabilities. This includes penetration testing to simulate attacks and assess the system's resilience against potential threats.

- Any identified vulnerabilities should be addressed promptly with patches or updates to strengthen the system.

5. **Activity Logging and Monitoring**:

- Comprehensive logging of user activities, including login attempts, exam access, and submission actions, must be implemented. These logs should be monitored in real-time to detect suspicious behavior or potential breaches.

- An alert system should notify administrators of unusual activities, such as multiple failed login attempts or access from unfamiliar IP addresses.

6. **Session Management**:

- The system should enforce secure session management practices, including automatic session timeouts after periods of inactivity. This helps prevent unauthorized access to a user's session if they leave their device unattended.

- Session tokens should be securely generated and managed to protect against session hijacking attacks.

7. **Secure Code Practices**:

- Developers must follow secure coding practices to prevent common vulnerabilities (e.g., SQL injection, cross-site scripting). Code reviews and static analysis tools should be used to identify potential security flaws during the development process.

- Regular updates to third-party libraries and dependencies are essential to mitigate risks from known vulnerabilities.

8. **Backup and Recovery**:

- Regular backups of critical data must be performed to ensure that information can be restored in the event of data loss or corruption. Backup data should also be encrypted and stored securely.

- A disaster recovery plan should be in place to ensure that the system can be restored quickly in case of catastrophic failures or security incidents.

5.4. Software Quality Attributes

Software quality attributes define the characteristics that determine the overall quality and performance of the online examination system. Ensuring these attributes are met is crucial for delivering a reliable, efficient, and user-friendly application. The following key quality attributes are essential for the system:

1. **Usability**:

- The system must provide an intuitive and user-friendly interface for both students and teachers. Clear navigation, straightforward exam creation tools, and accessible help resources should be included to enhance user experience.

- User training and onboarding materials should be available to assist users in effectively utilizing the system's features, ensuring they can perform tasks without frustration.

2. **Reliability**:

- The online examination system must be reliable, with a high uptime percentage (99.5%) during peak usage periods. It should consistently perform its intended functions without failure, even under stress or heavy load.

- The system should include error-handling mechanisms that gracefully manage unexpected events, providing informative feedback to users without compromising data integrity.

3. **Performance**:

- The system should demonstrate high performance, characterized by fast response times (within 1-2 seconds) for user actions and efficient data processing during exams. Performance should be maintained under varying load conditions, ensuring responsiveness regardless of the number of concurrent users.

- Regular performance testing should be conducted to identify bottlenecks and optimize system responsiveness.

4. **Scalability**:

- The architecture of the system must be designed for scalability to accommodate increasing numbers of users and exams without sacrificing

performance. This includes the ability to add resources (e.g., servers, databases) as needed to handle growth.

- The system should support both horizontal (adding more machines) and vertical (upgrading existing machines) scaling options.

5. **Security**:

- Security is paramount for protecting sensitive user data and maintaining the integrity of the examination process. The system must incorporate robust security measures, including encryption, access control, and regular security audits, to safeguard against threats.

- Security features should be seamlessly integrated into the user experience, ensuring that users can focus on their exams without worrying about potential vulnerabilities.

6. **Maintainability**:

- The system must be designed for ease of maintenance, allowing for efficient updates and modifications without significant downtime. Code should be well-documented, and modular architectures should be utilized to facilitate easier debugging and enhancement.

- Regular code reviews and adherence to coding standards will help maintain high-quality code and reduce technical debt over time.

7. **Interoperability**:

- The online examination system should be able to integrate seamlessly with existing tools and systems used by the Department of Computer Science. This may include Learning Management Systems (LMS), student information systems, or analytics tools.

- Standardized APIs should be available to allow for smooth data exchange and communication between different software components.

8. **Portability**:

- The system should be accessible across various devices and operating systems, ensuring that users can access the platform from their preferred devices (desktops, laptops, tablets) without compatibility issues.

- The web-based nature of the application should allow users to connect from any device with a modern web browser, enhancing accessibility and convenience.

6. Other Non-functional Requirements

In addition to the primary functional and non-functional requirements outlined in previous sections, the online examination system must also adhere to several additional requirements that enhance its effectiveness, ensure compliance, and improve user experience. These include:

1. **Regulatory Compliance**:

   - The system must comply with relevant educational regulations and standards applicable to the institution, including data protection laws, accessibility standards (such as WCAG), and academic integrity guidelines. This compliance will ensure that the system meets the legal obligations for educational institutions.

2. **User Support and Help Resources**:

   - A comprehensive help section should be included within the system, providing users with FAQs, troubleshooting guides, and contact information for technical support. This support should be readily accessible to assist users in resolving any issues they encounter.

   - Training sessions or tutorials should be organized for both teachers and students to familiarize them with the system's features and functionalities.

3. **Backup and Disaster Recovery**:

   - Regular backups of the database and application should be performed to prevent data loss in case of system failures. Backup procedures must ensure that data can be restored quickly and accurately.

   - A disaster recovery plan should be in place to address potential catastrophic failures (e.g., server crashes, data breaches) to minimize disruption to the examination process.

4. **Audit Trails and Reporting**:

   - The system must maintain detailed audit trails for all user activities, including login attempts, exam creation, and submissions. This logging will provide a comprehensive history that can be reviewed for compliance and security purposes.

   - Administrative reporting features should allow for the generation of various reports, such as exam statistics, user activity logs, and performance metrics, to facilitate better decision-making and analysis.

5. **Performance Monitoring**:

   - Tools for performance monitoring should be integrated into the system to track key performance indicators (KPIs), such as system uptime, response times,

and user activity patterns. This data will help identify areas for improvement and optimize system performance over time.

6. **User Feedback Mechanism**:

   - A mechanism for gathering user feedback should be implemented, allowing users to submit suggestions or report issues directly through the system. This feedback can be used to identify areas for enhancement and guide future development efforts.

7. **Localization and Internationalization**:

   - The system should support localization features to accommodate users from diverse backgrounds. This may include multiple language options and cultural adaptations to ensure a user-friendly experience for all students and teachers.

8. **Integration with External Tools**:

   - The system should be designed to integrate with existing academic tools, such as Learning Management Systems (LMS), plagiarism detection software, and grading systems. This interoperability will streamline workflows and enhance the overall functionality of the examination process.

7. Analysis Models

7.1. SDLC Model

The Software Development Life Cycle (SDLC) model outlines the structured process that will guide the development, deployment, and maintenance of the online examination system. The chosen SDLC model for this project is the **Agile Model**, which emphasizes flexibility, iterative progress, and collaboration among stakeholders. The following phases outline the SDLC process for the online examination system:

1. **Requirements Gathering**:
   o During this initial phase, stakeholders—including teachers, students, and administrators—will collaborate to define the functional and non-functional requirements of the system.
2. **Planning**:
   o A detailed project plan will be developed, outlining timelines, resource allocation, and milestones. The project team will prioritize features based on stakeholder input, creating a backlog of tasks for subsequent iterations.
3. **Design**:
   o In this phase, the system architecture and user interface design will be created. The design will include wireframes, database schemas, and technical specifications to guide the development process. User feedback will be solicited on design prototypes to ensure usability.
4. **Development**:
   o The development team will implement the system based on the approved design. This phase will be iterative, allowing for incremental development and regular feedback from stakeholders. Each iteration will focus on delivering a functional subset of features.
5. **Testing**:
   o Comprehensive testing will be conducted throughout the development process, including unit testing, integration testing, and user acceptance testing (UAT).
6. **Deployment**:
   o Once testing is complete and the system meets quality standards, it will be deployed to the production environment. User training sessions will be conducted to familiarize stakeholders with the system and its features.
7. **Maintenance**:
   o Following deployment, the system will enter the maintenance phase, where ongoing support, bug fixes, and updates will be provided. User feedback will continue to be collected to guide future enhancements and ensure the system remains aligned with user needs.

By adopting the Agile SDLC model, the online examination system project aims to be responsive to changing requirements, enhance stakeholder engagement, and deliver a high-quality product that meets the needs of its users effectively.

7.2.Dataflow Diagram

Data flow diagram is the starting point of the design phase that functionally decomposes the requirements specification. A DFD consists of a series of bubbles joined by lines. The bubbles represent data transformation and the lines represent data flows in the system. A DFD describes what data flow rather than how they are processed, so it does not hardware, software and data structure. A data-flow diagram (DFD) is a graphical representation of the "flow" of data through an information system. DFDs can also be used for the visualization of data processing (structured design).

DFD Notations in Dia Diagram editor :

8.
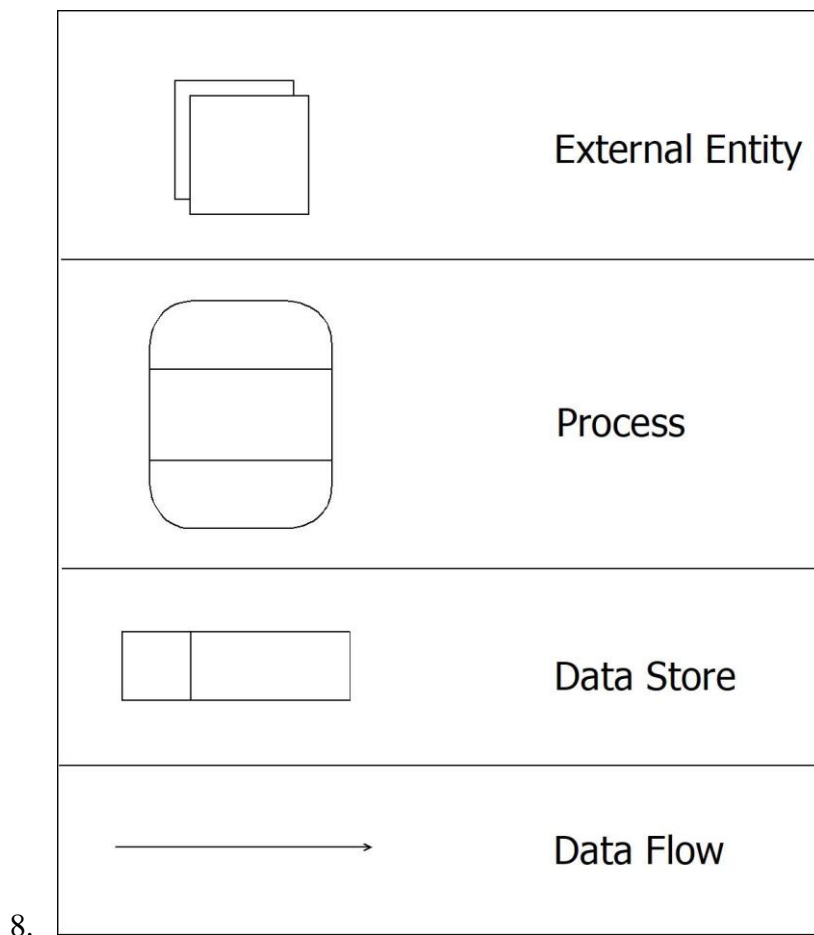
Level 0 Context level DFD:

DFD Level 0 is also called a Context Diagram. It's a basic overview of the whole system or process being analyzed or modeled. It's designed to be an at-a-glance view, showing the system as a single high-level process, with its relationship to external entities. It should be easily understood by a wide audience, including stakeholders, business analysts, data analysts and developers
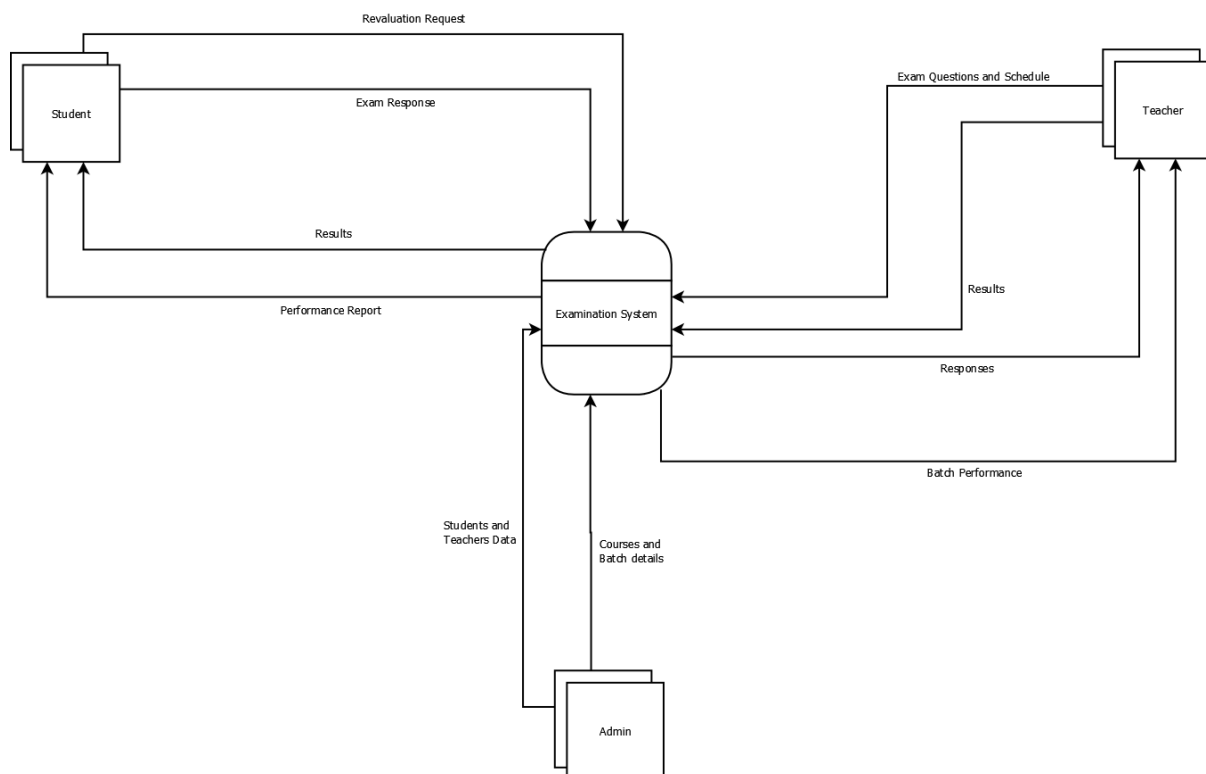
Level 1 DFD:

DFD Level 1 provides a more detailed breakout of pieces of the Context Level Diagram. You will highlight the main functions carried out by the system, as you break down the high-level process of the Context Diagram into its subprocesses.
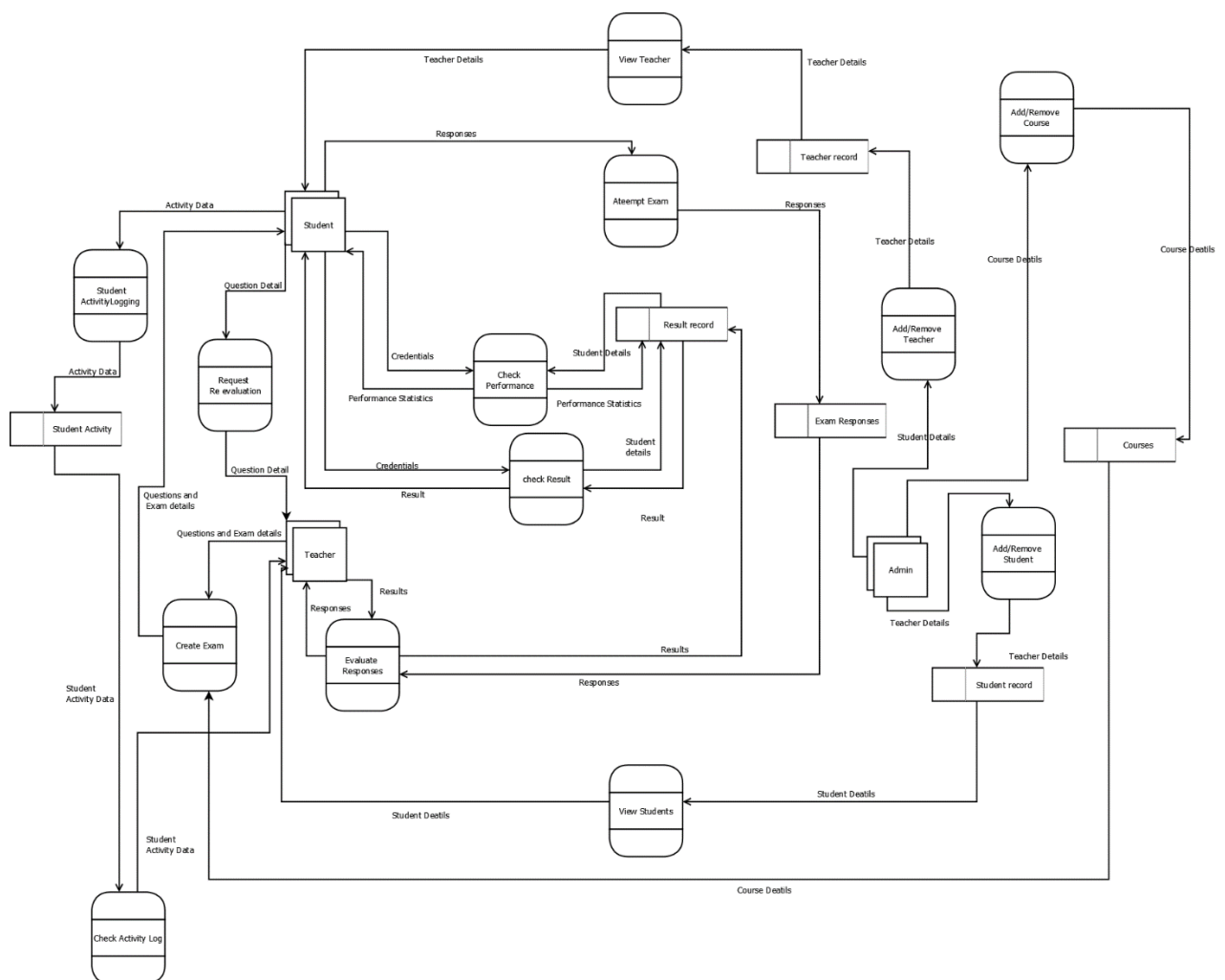
1. Entities

- **Student**: This represents users who attempt exams and request evaluations.

- **Teacher**: Teachers are responsible for creating exams, evaluating responses, and reviewing student activities.

- **Admin**: Admins manage teachers, students, and courses in the system.

- **Courses**: This represents the database of courses managed by the admin.

2. Processes and Interactions

- **Create Exam**: Teachers create exams with specific questions, which are then made available to students.

- **Attempt Exam**: Students use this process to take the exams. Their responses are recorded and sent to the teacher for evaluation.

- **Evaluate Responses**: This process is where teachers review students' answers and assign grades or feedback.

- **Check Result**: After evaluation, students can check their results through this process.

- **Check Performance**: This provides students with statistics or feedback on their performance across various exams, helping them understand their progress.

- **Request Re-evaluation**: Students can request a review of their answers if they feel the evaluation is inaccurate.

- **Student Activity Logging**: Tracks activities performed by the student during exams, such as login time, actions taken, and any suspicious behavior.

- **Check Activity Log**: Teachers can review the logs of student activities to monitor their behavior during exams.

3. Admin Processes

- **Add/Remove Teacher**: The admin can add new teachers to the system or remove existing ones. This includes updating teacher records.

- **Add/Remove Student**: Similarly, the admin can manage student records, adding or removing them from the system as needed.

- **Add/Remove Course**: Admins can also add or delete courses, which students are then able to enroll in and teachers can use as categories for exams.

4. Data Stores

- **Student Details, Teacher Details, Course Details**: These represent storage for essential records about students, teachers, and courses.

- **Responses, Result Record, Performance Statistics**: These data stores hold information about student exam responses, results, and statistical data on performance.

- **Question and Exam Details**: This data store contains information about exam questions and structures.

5. Data Flow

- Arrows in the diagram represent data flow, showing the transfer of information between entities, processes, and data stores. For example:

  o **Credentials**: Students provide their login credentials, which are validated before they can access exams.

  o **Exam Responses**: When students attempt an exam, their responses are sent to the teacher for evaluation.

  o **Results and Performance Statistics**: These are sent back to students, giving them insights into their achievements.

This DFD provides an overview of the main functions in your online examination system. It includes everything from exam creation and response evaluation to performance tracking and administrative tasks like managing teachers, students, and courses. Each entity, process, and data store is essential to maintaining the flow of information and ensuring that students have a smooth experience during exams, with robust monitoring and evaluation functionalities.

7.3 Entity-Relationship Diagram

An **ER (Entity-Relationship) diagram** is a type of visual representation used in database design to illustrate the structure of a database. It shows the **entities** (objects or concepts) within the system, their **attributes** (characteristics of those entities), and the **relationships** between them.

Key Components of an ER Diagram

1. **Entities**: These are the main objects or concepts, usually represented by rectangles. For instance, in a school database, entities could be "Student," "Course," and "Teacher."

2. **Attributes**: These are properties or details that describe an entity, represented by ovals. For example, the "Student" entity might have attributes like "StudentID," "Name," and "DateOfBirth."

3. **Relationships**: This show how entities are connected to each other, represented by diamonds. Common relationships include:

   o **One-to-One (1:1)**: Each instance of one entity is associated with a single instance of another entity (e.g., a person with a unique passport).

   o **One-to-Many (1)**: One instance of an entity is associated with multiple instances of another (e.g., one teacher teaches many courses).

   o **Many-to-Many (M)**: Multiple instances of one entity are associated with multiple instances of another (e.g., students enroll in multiple courses, and each course has many students).

4. **Primary Key**: A unique identifier for each entity, often underlined in ER diagrams (e.g., "StudentID" for "Student").

5. **Foreign Key**: A reference to the primary key of another entity, often used to establish relationships between tables in the database.

Purpose and Use of ER Diagrams

ER diagrams serve multiple purposes, especially in **database design**:

- **Database Planning**: ER diagrams help design and structure the database by identifying entities, attributes, and relationships before implementation.

- **Documentation**: They provide a visual map of the database structure, which can be used for documentation and understanding complex database schemas.

- **Communication**: ER diagrams simplify communication between developers, stakeholders, and users by providing a clear representation of how data is organized.

- **Normalization**: ER diagrams help in the process of normalization, ensuring efficient and error-free data storage by minimizing redundancy.

In essence, ER diagrams are foundational tools for organizing and planning a database. They ensure that all data requirements are understood and captured, and they support efficient database development by visually mapping out the logical structure of the data.

Following are the main components and its symbols in ER Diagrams:

- **Rectangles:** This Entity Relationship Diagram symbol represents entity types

- **Ellipse:** Symbol represent attributes

- **Diamonds:** This symbol represents relationship types

- **Lines:** It links attributes to entity types and entity types with other relationship types

- **Primary key:** attributes are underlined

- **Double Ellipses:** Represent multi-valued attributes
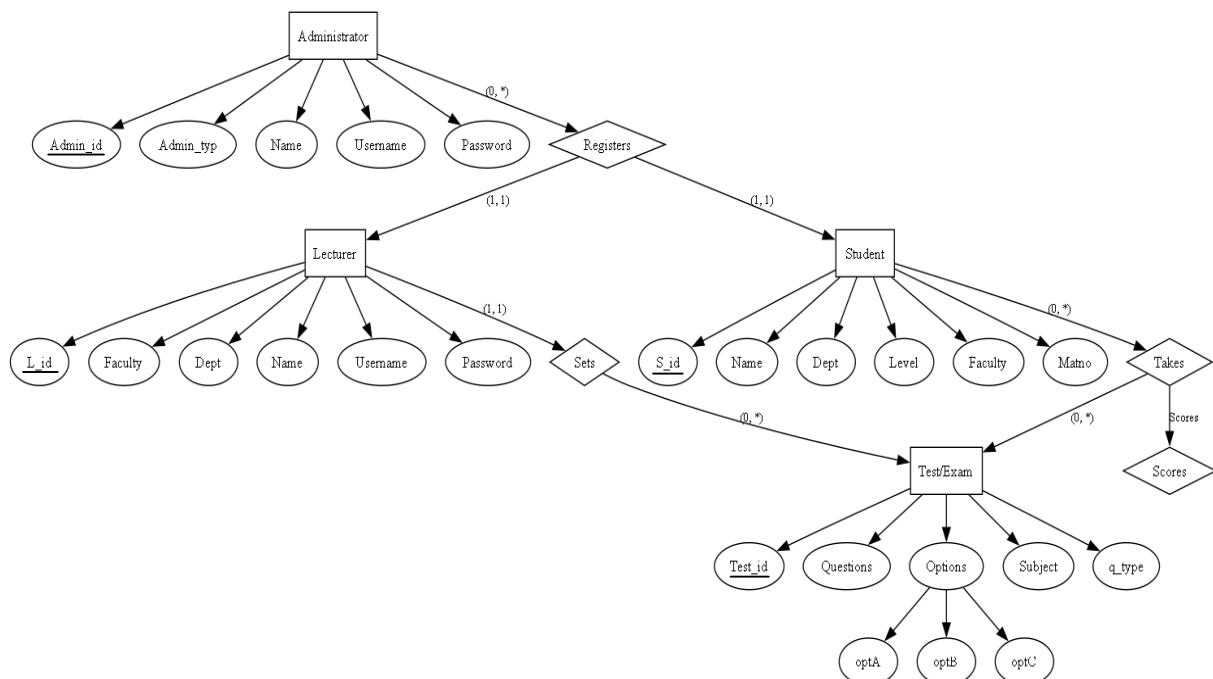
7.4   Class Diagram

A **Class Diagram** is a visual representation of the classes, attributes, methods, and relationships in an **object-oriented system**. Commonly used in software design, it's a central part of **UML (Unified Modeling Language)**.

Key Components of a Class Diagram

1. **Classes**: Represented by rectangles divided into three sections:

    o **Class Name** (top section)

    o **Attributes** (middle section, listing class properties/fields)

    o **Methods** (bottom section, listing operations/functions the class can perform)

2. **Relationships**:

    o **Association**: A generic connection between classes (e.g., a "Student" is associated with a "Course").

    o **Inheritance (Generalization)**: Shows a hierarchy where one class inherits attributes and methods from another (e.g., "GraduateStudent" inherits from "Student").

    o **Aggregation**: A whole-part relationship where the part can exist independently of the whole (e.g., a "Department" has "Professors" who can exist separately).

    o **Composition**: A whole-part relationship where the part cannot exist without the whole (e.g., a "House" contains "Rooms" that do not exist independently).

Purpose and Use of Class Diagrams

Class diagrams are essential in **object-oriented design** to:

- Define the structure of a system by showing classes, their attributes, and behaviors.
- Illustrate relationships between different parts of the system.
- Serve as a blueprint for coding, helping developers understand the system architecture.

In short, class diagrams help plan and visualize the code structure before actual development, making them crucial for complex software projects.
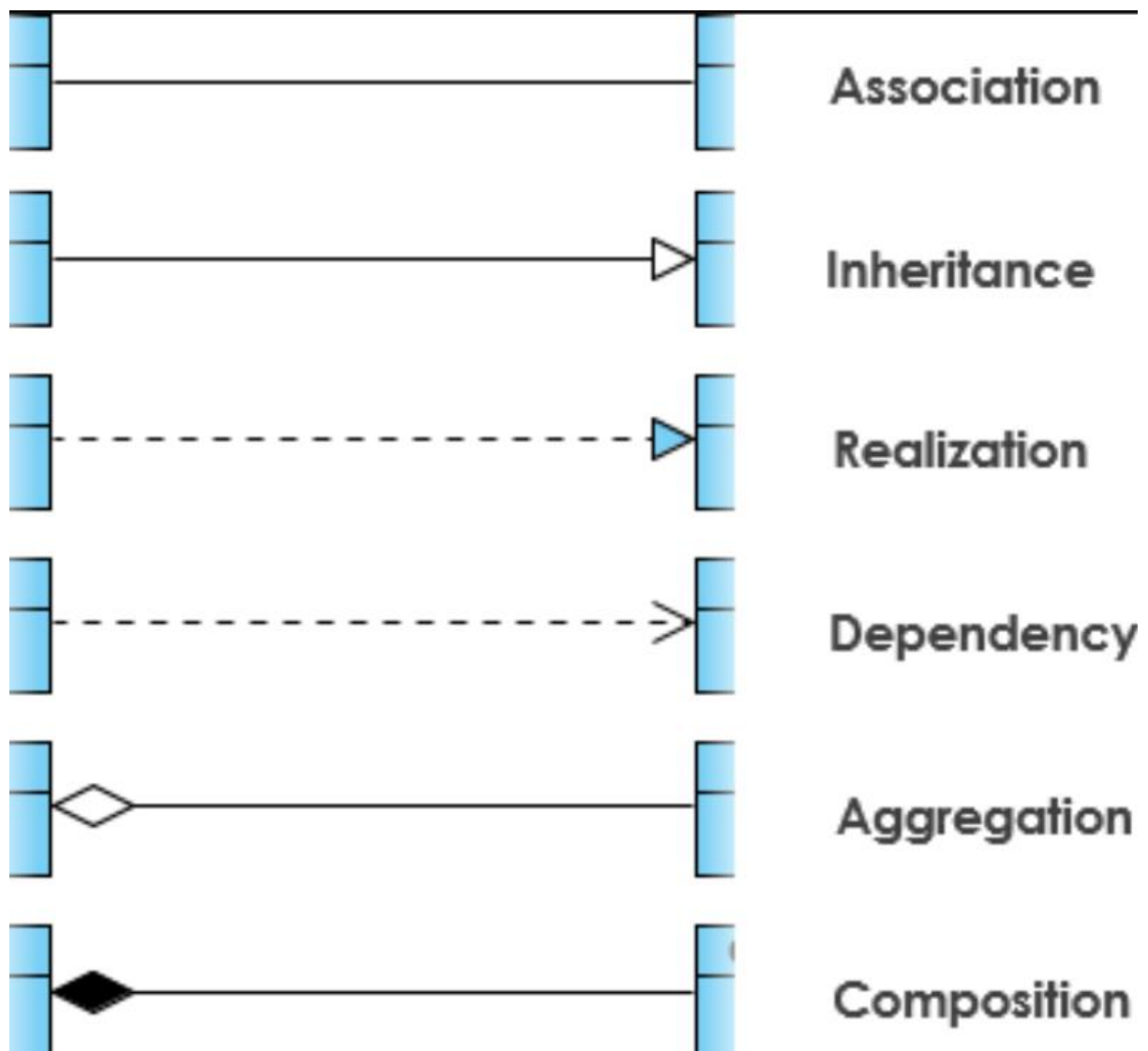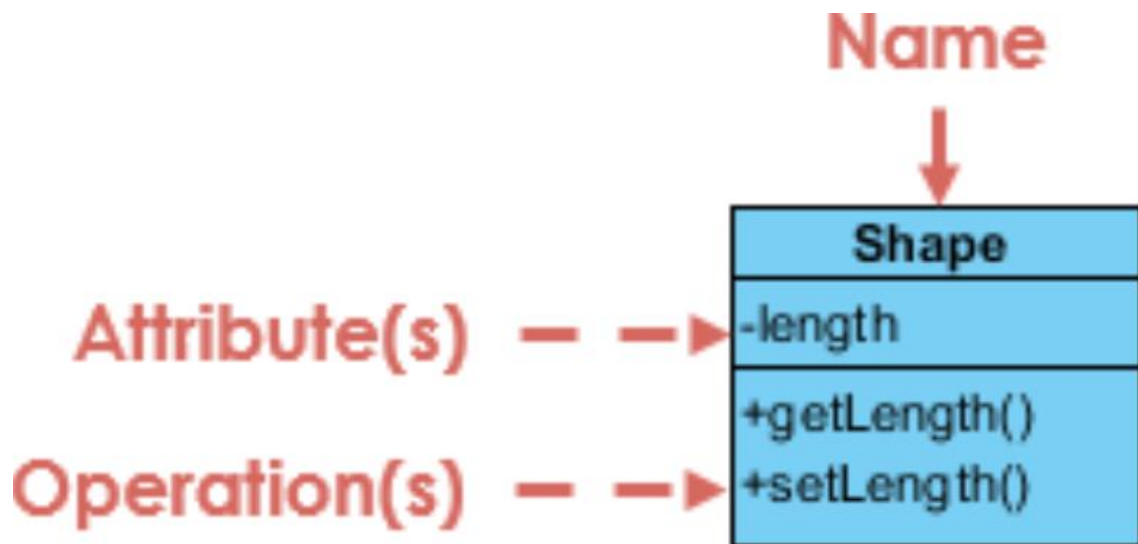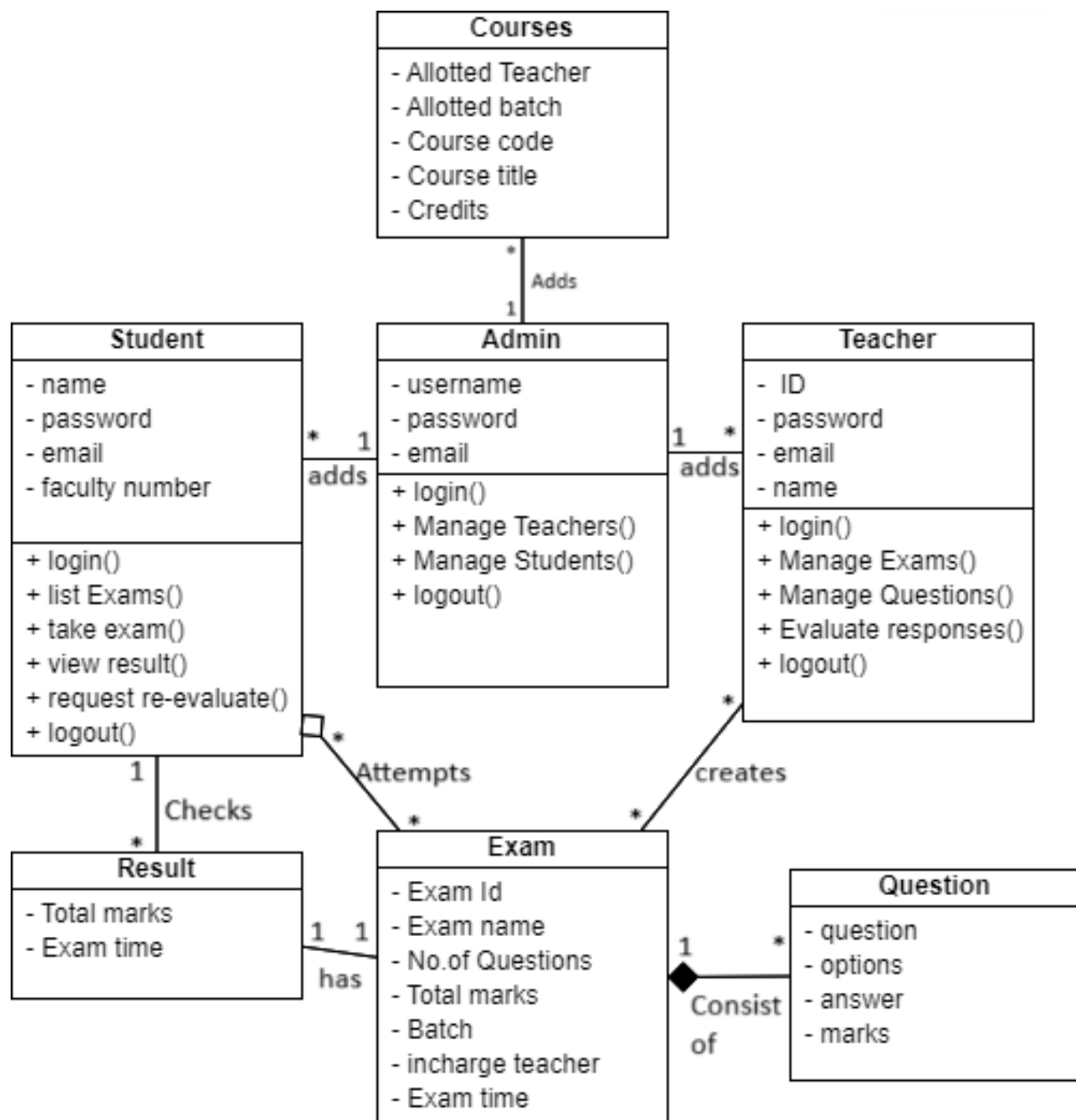
Class Diagram

7.5   Activity Diagram

An **Activity Diagram** is a type of UML diagram used to visually represent the **workflow** or **sequence of activities** in a system. It shows the **flow of control** from one activity to another, making it useful for modeling complex business processes or system functionalities.

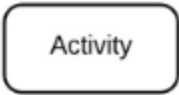**Key Components of an Activity Diagram**
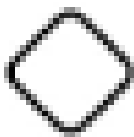
1. **Activities**: Represent tasks or actions that occur in the workflow, shown as rounded rectangles.

2. **Start Node**: Indicates the beginning of the workflow, represented by a filled black circle.

3. **End Node**: Marks the end of the workflow, shown as a filled black circle with an outer ring.

4. **Transitions (Arrows)**: Show the flow from one activity to the next.

5. **Decision Nodes**: Represent branching points, where the flow can split based on a condition (e.g., yes/no), shown as diamonds.

6. **Swimlanes**: Used to divide activities among different actors or departments, making it clear who performs each action.

7. **Fork and Join Nodes**: Represent parallel processes:

    o   **Fork**: Splits one flow into multiple parallel flows.

    o   **Join**: Combines multiple flows back into one.

**Purpose and Use of Activity Diagrams**

Activity diagrams help:

*   Visualize complex workflows and processes.

*   Clarify system functionality, showing step-by-step task sequences.

*   Communicate the flow of tasks and decision points to stakeholders and developers.

In summary, activity diagrams are valuable for understanding and documenting the flow of tasks within a system, making them especially useful for process modeling and workflow analysis.

| Symbol	Name	Description
●	Start symbol	Represents the beginning of a process or workflow in an activity diagram. It can be used by itself or with a note symbol that explains the starting point.
Activity	Activity symbol	Indicates the activities that make up a modeled process. These symbols, which include short descriptions within the shape, are the main building blocks of an activity diagram.
→	Connector symbol	Shows the directional flow, or control flow, of the activity. An incoming arrow starts a step of an activity; once the step is completed, the flow continues with the outgoing arrow.
▬	Joint symbol/ Synchronization bar	Combines two concurrent activities and re-introduces them to a flow where only one activity occurs at a time. Represented with a thick vertical or horizontal line.
▬	Fork symbol	Splits a single activity flow into two concurrent activities. Symbolized with multiple arrowed lines from a join.
◇	Decision symbol	Represents a decision and always has at least two paths branching out with condition text to allow users to view options. This symbol represents the branching or merging of various flows with the symbol acting as a frame or container.
◉	End symbol	Marks the end state of an activity and represents the completion of all flows of a process.

Student Activity Diagram
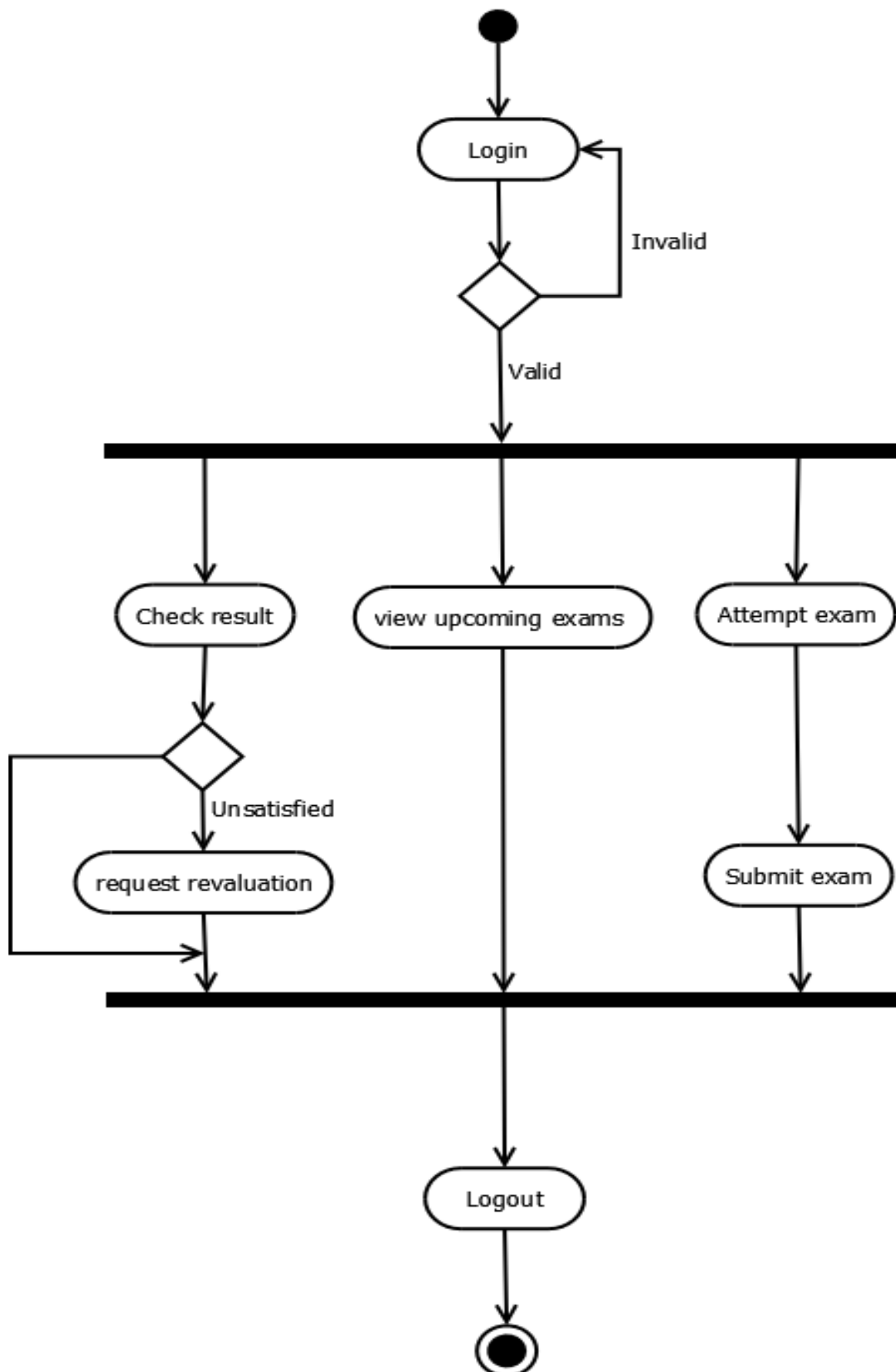
Teacher Activity Diagram
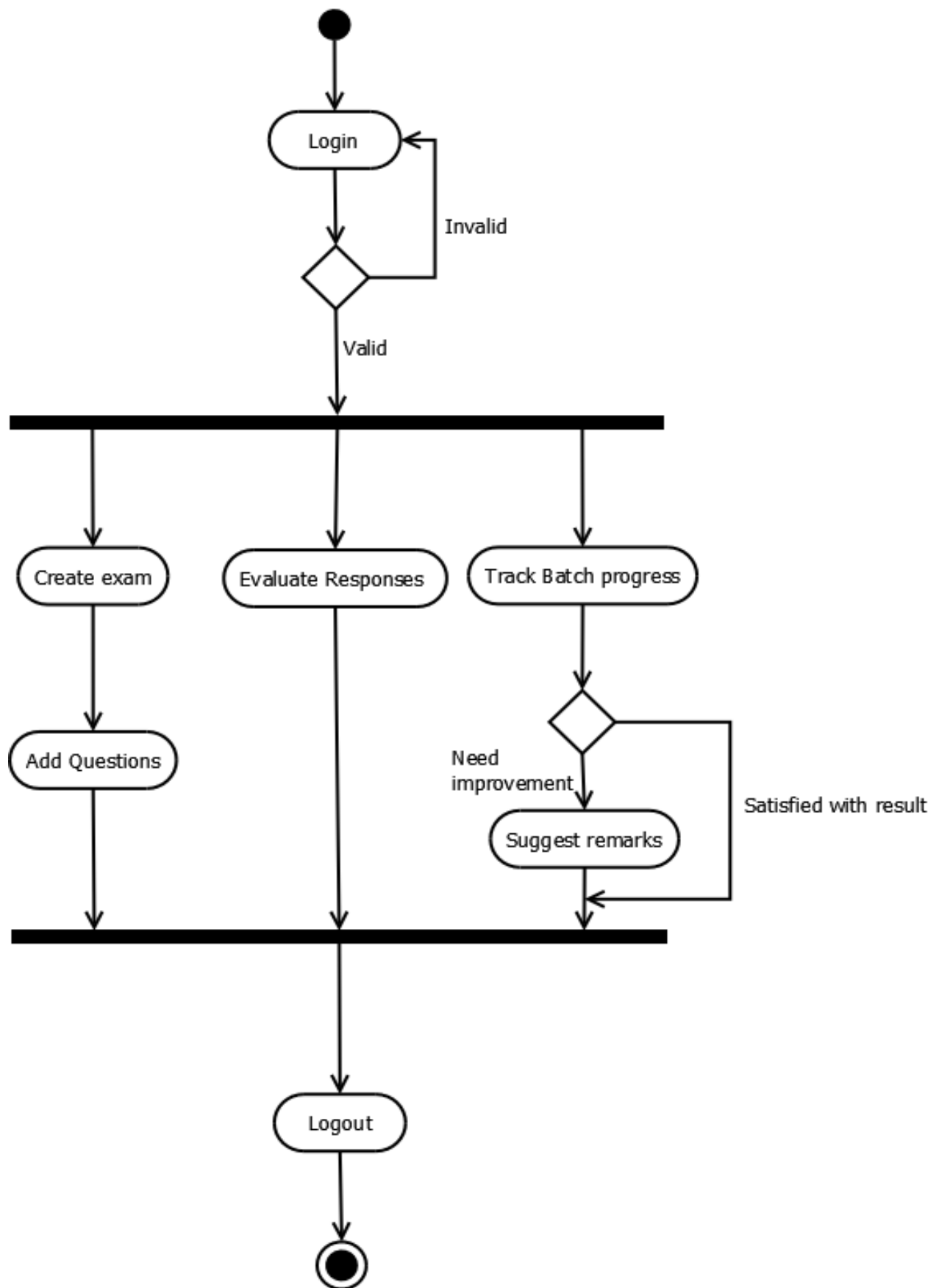
Admin Activity Diagram
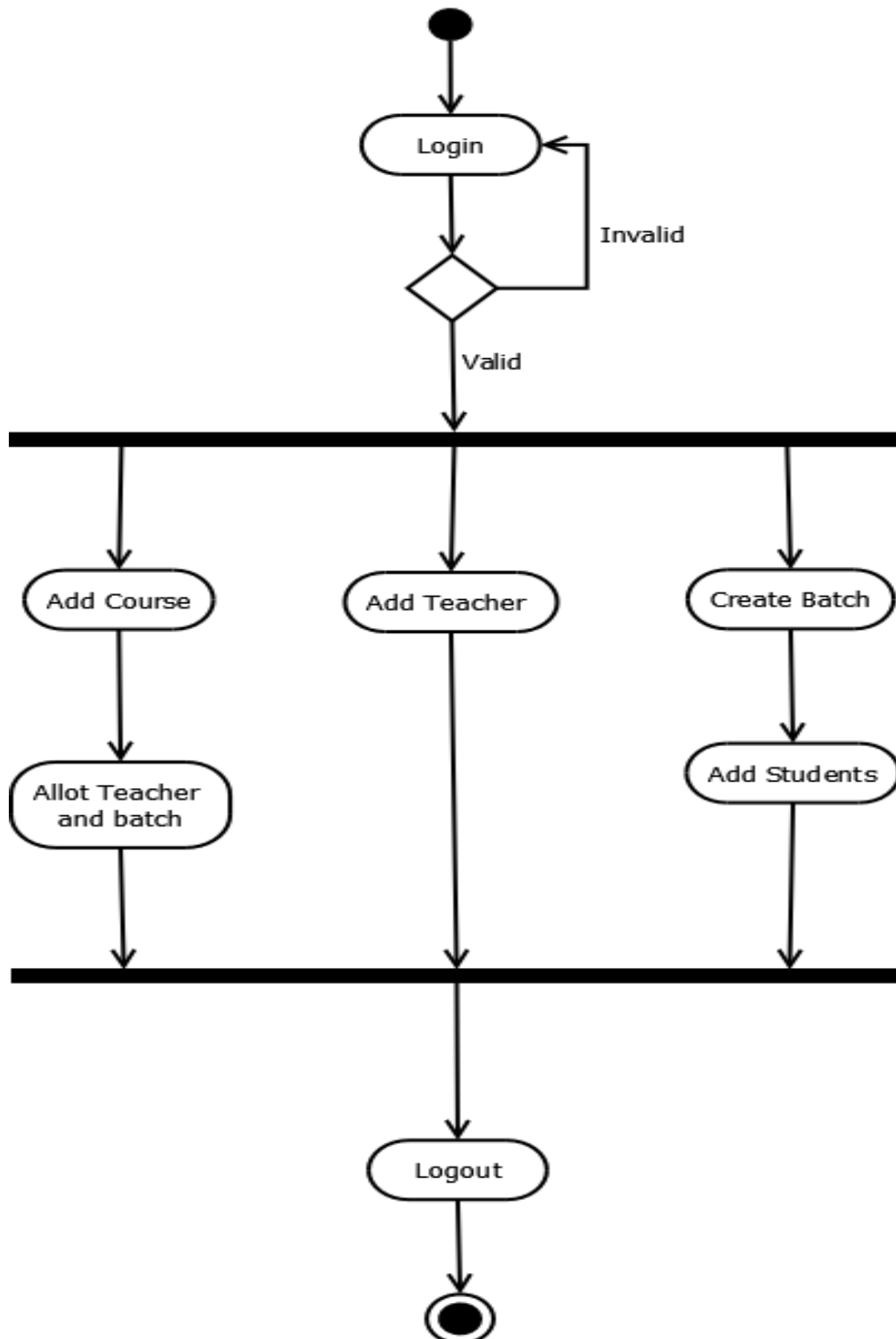
7.6 Use Case Diagram

A **Use Case Diagram** is a UML diagram that visually represents the **interactions between users (actors)** and a system. It outlines the **system's functionality** from an end-user perspective, showing what the system can do and who can interact with it.

Key Components of a Use Case Diagram

1. **Actors**: Represent users or external systems that interact with the system. Actors are typically shown as stick figures and can be **human users** or **other systems**.

2. **Use Cases**: Represent the system's functionalities or actions performed by actors, depicted as ovals with a description inside (e.g., "Login," "Register," "Place Order").

3. **System Boundary**: Defines the scope of the system, drawn as a rectangle around all the use cases.

4. **Relationships**:

   o **Association**: Links an actor to a use case, indicating interaction.

   o **Include**: Represents mandatory steps that a use case must invoke (e.g., "Process Payment" is included in "Place Order").

   o **Extend**: Represents optional or conditional behavior that extends a base use case (e.g., "Apply Discount" may extend "Place Order").

   o **Generalization**: Represents inheritance between actors or use cases (e.g., "Admin" inherits functionalities of a "User" actor).
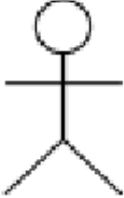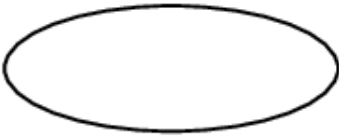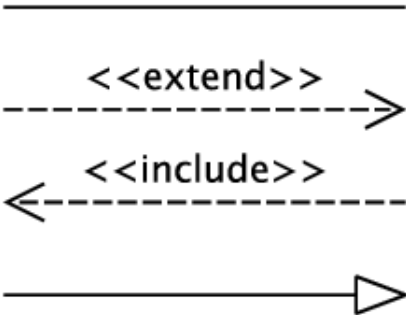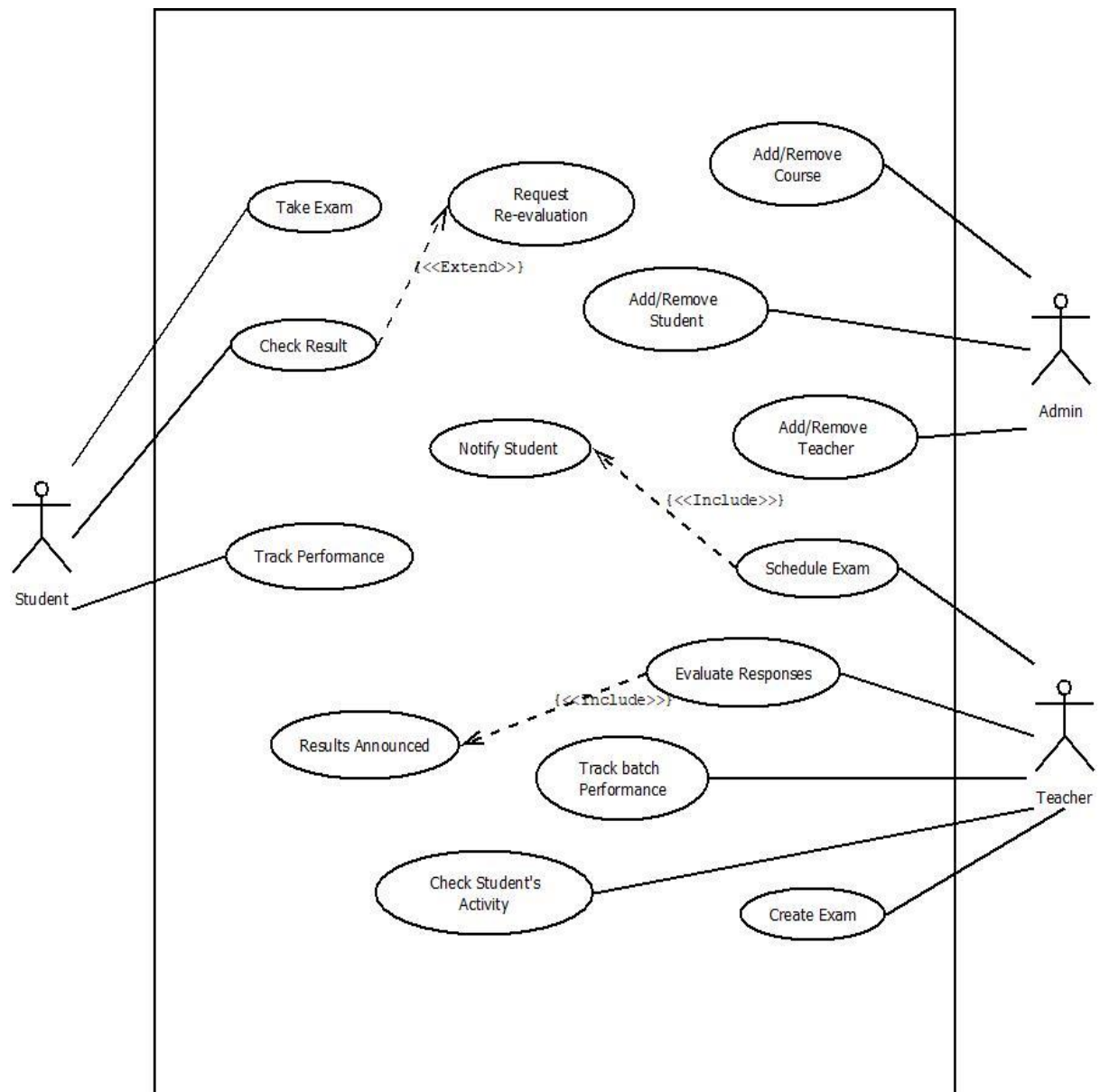
Purpose and Use of Use Case Diagrams

Use case diagrams help:

• Capture and visualize the **functional requirements** of a system.

• Show **who interacts with what functionality**, aiding communication between stakeholders.

• Provide a high-level **overview of system capabilities** without detailing internal processes.

In short, use case diagrams give a clear view of the system's functionality and user interactions, making them ideal for the early stages of system design.

Symbols representation in Use Case Diagram

Symbol	Reference Name
	Actor
	Use case
<<extend>> <<include>>	Relationship

8. Code and Screenshots