Student Management System Requirements Specification

Version 1.0

May 24, 2024

Team Members:

- Aurora Zhibaj
- Dea Molla
- Era Fejza
- Ema Kola
- Ersido Dingo
- Franko Kaloshi
- Herion Halilaj

Table of Contents

1. Executive Summary

1.1 Project Overview

This Student Management System is designed to support the needs of a university's diverse user base, which includes students, academic staff, and administrative staff. This system facilitates a wide range of functionalities to meet specific user needs: students can access their transcripts, grades, and attendance records, manage course selections, request documents, and engage with student clubs; academic staff can manage course materials, record grades and attendance, and access student surveys; and administrative staff can handle tasks such as syllabus review, course creation, timetable management, tuition fee updates, and disciplinary records. The Student Management System ensures secure, role-based access and compliance with privacy and data protection regulations. By providing an efficient, user-friendly interface and ensuring data integrity and security, the System Management System aims to make university operations as efficient as possible, enhance communication and collaboration, and improve the overall educational experience.

The primary users of this system include:

Students: This category includes both current students and alumni. Current students require extensive interaction with the system for their academic and extracurricular activities, while alumni need access primarily for viewing their academic records.

Academic Staff: This group includes main lecturers, advisors, and assistant lecturers who need to manage course materials, grades, attendance, and student surveys.

Administrative Staff: This category comprises various administrative roles such as Head of Departments, coordinators, finance office staff, Dean of Students office, registrar's office, and IT office staff, each responsible for specific administrative functions.

1.2 Purpose and Scope of this Specification

The purpose of this specification is to clearly outline the functional and nonfunctional requirements for developing and implementing the Student Management System at our university. Our objective is to build a system that caters to the varied needs of students, academic staff, and administrative staff, supports the university's academic and administrative workflows efficiently, and adheres to all relevant regulatory standards.

In Scope

Functional Enhancements:

- Adding and improving features for current students, such as course selection, tuition debt viewing, and involvement in student clubs.
- Implementing functionalities for academic staff, including loading grades, recording attendance, and accessing student survey responses.
- Administrative capabilities like syllabus review, timetable management, tuition fee updates, and creating and maintaining student accounts.

Compliance and Security:

- Ensuring the system adheres to data protection regulations and university privacy policies.
- Implementing strong authentication and authorization mechanisms and encrypting sensitive data.

User Experience:

- Enhancing the user interface to facilitate ease of use for students, academic staff, and administrative personnel.

Out of Scope

Future Legislative Mandates:

- Changes to comply with new legislative mandates.

Major System Overhauls:

- Significant changes or complete overhauls of the system planned for future stages.

Additional Functionalities:

- Introduction of new features or modules to be considered in future phases.

2. Product/Service Description

The Student Management System is a critical tool for managing student information, academic records, and administrative processes. It integrates various functions to support the daily operations of a university, ensuring data consistency, accessibility, and security.

2.1 Product Context

The Student Management System interacts with multiple components, such as: course management (course selection, grade and attendance), financial management, human resource management, and authentication. The system is designed to be modular and scalable, allowing for future enhancements and integrations.

2.2 User Characteristics

Students
- Technical Expertise: Basic computer and internet skills.
- Other Characteristics: Includes current students needing extensive system interaction and alumni requiring limited access.

Academic Staff
- Technical Expertise: Moderate to high proficiency with educational tools and systems.
- Other Characteristics: Includes main lecturers, advisors, and assistant lecturers, each with specific system interactions.

Administrative Staff
- Technical Expertise: Proficient with office and administrative software.
- Other Characteristics: Includes a variety of roles with specific system responsibilities.

2.3 Assumptions

Users will have access to the necessary hardware and software to use the system. Training will be provided to ensure users are familiar with the system functionalities. The operating system and network infrastructure will support the Student Management System.

2.4 Constraints

- Audit/Financial Functions: The system must maintain comprehensive logs for auditing purposes.

- Access, Management, and Security: Extensive security measures must be in place to protect sensitive information.
- Criticality: The Student Management System is critical for daily university operations; any downtime must be minimized.
- System Resource Constraints: Must operate efficiently within the existing hardware limitations.

2.5 Dependencies

- Module Dependencies: Certain modules (e.g., student accounts) need to be fully functional before others (e.g., course selection) can be implemented.

3. Requirements

3.1 Functional Requirements

Req#	Requirement	Comments	Priority	Date Rvwd	SME Reviewed / Approved
BR_01	All users must be able to log in to the system..	The process by which a registered user logs into the system.	1	04/04/2024	Herion Halilaj
BR_02	All users must be able to view and filter timetable records.	The process in which the user views and filters the public timetable	2	04/04/2024	Herion Halilaj
BR_03	Students must be able to view their transcripts.	The process of how the student can see their final grades and gpa (transcript).	1	04/04/2024	Franko Kaloshi
BR_04	Students must be able to view their course grades.	The process of how the student can see their grades for each course.	1	04/04/2024	Franko Kaloshi
BR_05	Students must be able to view all their attendance records.	The process of how the student can see their attendance for each course.	1	04/04/2024	Franko Kaloshi
BR_06	Current students must be able to make document requests.	The process by which a current student can make document requests in the system.	2	04/04/2024	Era Fejza
BR_07	Current students must be able to select courses for each semester	The process by which a current student selects courses for each semester in the system.	1	04/04/2024	Era Fejza

Req#	Requirement	Comments	Priority	Date Rvwd	SME Reviewed / Approved
BR_08	Current students must be able to view their tuition debt	The process by which a current student views their student debt within the system.	2	04/04/2024	Era Fejza
BR_09	Current students must be able to fill out surveys for lecturer evaluation	The process by which a current student fills out a survey for lecturer evaluation within the system.		04/04/2024	Era Fejza
BR_10	Current students must be able to apply to student clubs	The process of users applying to student clubs.	3	04/04/2024	Herion Halilaj
BR_11	Current students must be able to view student club activities	The process of users viewing activities organized by their respective clubs	3	04/04/2024	Herion Halilaj
BR_12	Current students must be able to view their disciplinary records	The process of how the student can see their disciplinary records if they have one.	2	04/04/2024	Franko Kaloshi
BR_13	Current students must be able to view course syllabus	The process of accessing published and approved course syllabus for the courses in which they are enrolled.	2	04/04/2024	Ersido Dingo
BR_14	Current students must be able to apply for resit exams	The process of applying for the resit exam/s, only when final grade is less than CC.	2	04/04/2024	Franko Kaloshi
BR_15	Academic staff must be able to load grades into the system	The process of uploading grades for each student in a course by the academic staff.	1	04/04/2024	Dea Molla
BR_16	Academic staff must be able to record student attendance	The process of recording the attendance of each student in the system.	1	04/04/2024	Dea Molla
BR_17	Academic staff must be able to view course syllabus	The process of accessing and viewing the course syllabus that has previously been published in the system.	2	04/04/2024	Dea Molla

Req#	Requirement	Comments	Priority	Date Rvwd	SME Reviewed / Approved
BR_18	Academic staff must be able to access survey responses	The process of accessing the survey responses submitted by students for an academic staff's respective courses.	3	04/04/2024	Dea Molla
BR_19	Main lecturers must be able to publish the course syllabus	The process of publishing a syllabus for review by main lecturers.	1	04/04/2024	Ersido Dingo
BR_20	Advisors must be able to approve students' course selections	The process of approving every student's course selection by each program advisor.	1	04/04/2024	Ersido Dingo
BR_21	The Head of Department must be able to review, monitor, and evaluate syllabus material	The process in which the head of department reviews syllabus material of a specific lecturer.	2	04/04/2024	Aurora Zhibaj
BR_22	The Head of Department must be able to create new courses	The process of creating new courses.	1	04/04/2024	Aurora Zhibaj
BR_23	The Head of Department must be able to assign lecturers to courses	The process of assigning lecturers to an already existing course.	1	04/04/2024	Era Fejza
BR_24	Coordinators must be able to upload and make changes to timetable records	The process of uploading and making changes to the timetable records within the student management system.	2	04/04/2024	Aurora Zhibaj
BR_25	The Finance Office must be able to manage and update tuition fees	the process of managing tuition fees for each student, keeping track of the payments according to scholarship and debt.	1	04/04/2024	Aurora Zhibaj
BR_26	The Finance Office must be able to manage document request payments	The process managing external document request payments within the student management system.	2	04/04/2024	Aurora Zhibaj
BR_27	The Dean of Students Office must be able to record disciplinary/misconduct measures.	The process by which the Dean of Students Office records any disciplinary/misconduct measures.	2	04/04/2024	Ersido Dingo

Req#	Requirement	Comments	Priority	Date Rvwd	SME Reviewed / Approved
BR_28	The Dean of Students Office must be able to create new student clubs	The process of creating new student clubs.	3	04/04/2024	Ema Kola
BR_29	The Dean of Students Office must be able to post student club activities	The process of posting activities for a student club.	3	04/04/2024	Ema Kola
BR_30	The Registrar's Office must be able to create new student accounts.	The process of creating new student accounts/profiles.	1	04/04/2024	Ema Kola
BR_31	The Registrar's Office must be able to edit/update student account	The process of updating/editing student information.	1	04/04/2024	Ema Kola
BR_32	The Registrar's Office must be able to process document requests	The process of processing the requested document based on their personal student information.	2	04/04/2024	Ema Kola
BR_33	The IT Office must be able to create academic staff accounts	The IT office should be able to create new academic staff accounts in the system.	1	04/04/2024	Dea Molla

3.2 Non-Functional Requirements

3.2.1 Product Requirements

3.2.1.1 User Interface Requirements

The interface for all users should first and foremost be intuitive and user-friendly, with clear navigation menus and organized layouts for every functionality to be used accordingly.

Students:

- Screens should display relevant information concisely, with easy access to features such as viewing transcripts, grades, attendance records, course selections, and student club activities.
- Transcript reports should be well-structured, displaying GPA, final grades, and course details in a format that is easy to read and understand.
- Menu options should be logically categorized, allowing students to easily access different features such as transcripts, grades, course selections, and document requests.
- Error messages should be descriptive and user-friendly, providing clear guidance on how to resolve issues such as incorrect login credentials or unavailable services.

Academic Staff:

- Screens should provide academic staff with access to features such as grade loading, attendance recording, course syllabi, and survey responses in an organized and easily accessible manner.
- Grade and attendance reports should be presented in a structured format, allowing academic staff to quickly review and analyze student performance data.
- Menu structures should be tailored to the specific needs of academic staff, providing easy access to functions such as grade management, attendance tracking, and survey analysis.
- Error messages should provide clear guidance on resolving issues related to data entry errors, system failures, or other technical issues that may arise during academic tasks

Administrative Staff:

- Administrative staff interface is dependent on their varied functionalities:
  - ➔ Head of Department: Review, monitor, and evaluate syllabus material, create new courses, assign lecturers.
  - ➔ Coordinators: Upload and update timetable records.
  - ➔ Finance Office: Manage and update tuition fees, handle document request payments.
  - ➔ Dean of Students: Record disciplinary actions, create new student clubs, post club activities.
  - ➔ Registrar's Office: Create and update student accounts, process document requests.
  - ➔ IT Office: Create academic staff accounts.

3.2.1.2 **Usability**

- The user documentation and help should be complete
- The help should be context sensitive and explain how to achieve common tasks
- The system should be easy to learn

Efficiency

3.2.1.2.1 Performance Requirements

The system shall support at least 2000(rough minimum estimate of nr of students in a university) simultaneous users during peak times without performance degradation. (Priority 1)

98% of the transactions shall be processed in less than 1 second. (Priority 1)

The system shall handle up to 15,000 transactions per hour during peak load conditions. (Priority 1)

3.2.1.2.2 Space Requirements

The system shall not exceed 10 GB of storage for application data and 100 GB for user-generated content within the first year of operation. (Priority 2)

3.2.1.3 Dependability

Availability

- The system shall be available 99.9% of the time, 24/7, excluding scheduled maintenance. (Priority 1)
- Scheduled maintenance shall not exceed 4 hours per month and must be communicated to users at least 48 hours in advance. (Priority 1)

Reliability
The software must run correctly at all times, producing expected outputs for given inputs consistently. It must be robust enough to handle many concurrent users without crashing or slowing down. Timetable records and other dynamic data must update in real-time as users interact with the system.

- The system shall have a mean time between failures (MTBF) of at least 1000 hours. (Priority 1)
- The system shall automatically log and notify the IT department of any failures or errors. (Priority 1)

Maintenance
The system shall allow for modular updates and maintenance without requiring a complete system shutdown. (Priority 2)
The system shall provide detailed error messages and logs to facilitate troubleshooting. (Priority 1)

Monitoring

After deployment, developers will continue testing the system alongside users to ensure it handles concurrent use and to identify any bugs. The developers will remain in direct contact with the system administrator to address any issues promptly, minimizing downtime.

The system should run independently without developer intervention, operated efficiently by administrators and librarians. Each student can create their account and use the application without needing assistance from librarians or developers.

3.2.1.3 Efficiency

Efficiency of the software ensures that it not only completes all tasks but also operates optimally under various conditions. Efficiency is determined by performance and space requirements

3.2.1.3.1 Performance Requirements

Response Time: The system should respond to user requests in real-time with no lag or delay, ensuring a seamless user experience.

Throughput: The system should handle a high flow of data simultaneously as multiple users may access it concurrently. It must respond to every request immediately without compromising speed or performance.

Compatibility: The application should be compatible with various hardware, including computers and mobile devices, as long as they have internet access. Performance and functionality should remain consistent across different devices.

3.2.1.3.2 Space Requirements

Our software is a web application, meaning users do not need to download anything to their devices. This eliminates any storage space requirements on user devices. Users will access the application via a web browser, ensuring minimal impact on device storage.

3.2.1.5 Integrity & Security

Maintaining software integrity is crucial for reliability, security, and trustworthiness. Developers must ensure the software is secure and maintained throughout its lifecycle.

Access Control: Users will access interfaces based on their user group.

Restricted Access: Students cannot directly access or modify the database.

Password Security: Passwords must follow specific rules to enhance security.

Data Privacy: No sensitive personal data is stored within user accounts.

Security Measures: Continuous implementation and revision of security measures will protect the system.

Code Review: Regular code reviews will identify and fix errors or vulnerabilities.

3.2.2 Organizational Requirements

3.2.2.1 Environmental Requirements

Internet Access: The web application requires a reliable and stable internet connection for proper functionality.
Web Browsers: The application should be compatible with all popular web browsers, including Chrome, Firefox, Safari, and Edge.
Devices: The application should be accessible on various devices, including laptops, desktops, mobile phones, and tablets.
Security: The application must ensure that sensitive user data is well protected through encryption and secure access protocols.
Technical Support: Any glitches or bugs affecting the application's functionality should be addressed promptly.
User-friendly Interface: The application should have intuitive and user-friendly features, making it easy to navigate and use, thereby encouraging student engagement.

3.2.2.2 Operational Requirements

User Authentication: Students should be able to create an account and log in using their university email to access the application.
Timetable Management: The application should provide a detailed list of available timetable records for courses.
Reservation System: The application should allow users to reserve the records they need, with a clear indication of availability.
Search and Filter: Students should be able to search and filter timetable records easily by course title, instructor, time, and availability.
Notification System: The application should notify students about the status of their reservations or any changes to the timetable records.
User Feedback: The application should allow users to provide feedback on their overall experience to facilitate continuous improvement.

3.2.2.3 Development Requirements

The application will be accessed via an internet connection, such as Wi-Fi or mobile data. Any network issues will be managed by the system's robust network handling protocols.

3.2.3 External Requirements

3.2.3.1 Regulatory Requirements

Personal information such as names, email addresses, and reservation history should be gathered, processed, and stored safely and lawfully. The application must comply with security standards, such as SSL and two-factor authentication, to prevent unauthorized access or security breaches.

3.2.3.2 Ethical Requirements

Request Permission Only When Necessary: Only request necessary permissions to avoid user discomfort and ensure trust.
Promote Responsible Usage: Encourage users to engage with the application only when necessary, avoiding addictive design practices.
Avoid Dark Patterns: Ensure transparency and honesty in the design. For instance:
Avoid asking ambiguous questions to sway user responses.
Refrain from using ads disguised as regular content.
Provide clear notifications before any subscription starts, ensuring users are aware of trial expirations.

3.2.3.3 Legislative Requirements

Copyright Laws: The application must comply with copyright laws and policies, including fair use guidelines and the Copyright Act.
Protection of Personal Information Laws: Follow best practices regarding data protection and privacy, ensuring data security and confidentiality.
User Compliance: Users must agree to the terms of use and privacy policies before using the application. If they disagree, they can choose not to use the application.

3.2.3.3.1 Accounting Requirements
Due to the nature of our web application, no accounting requirements are necessary.

3.2.3.3.2 Security Requirements
Authentication: Users must authenticate themselves before gaining access to the application.
Authorization: The application should grant access based on user roles and privileges. For instance, students can view and reserve timetable records, while librarians have additional administrative privileges.
Encryption: All sensitive data transmitted between users' devices and the application should be encrypted using protocols like HTTPS.
Secure Passwords: Passwords should be hashed and salted according to industry standards to ensure security.
Session Management: Users should be logged out after a period of inactivity to prevent unauthorized access.
Regular Updates: The application should be regularly updated to fix bugs and patch security vulnerabilities.

3.3 Domain Requirements

- The system must provide functionality to manage student information, including enrollment status, personal details, academic history, and contact information.
- The system should support the management of courses offered by the university, including course creation, scheduling, enrollment, and tracking of course-related data such as syllabus, prerequisites, and lecturer assignments(which lecturer/s teaches a course).

- The system must facilitate the maintenance and management of academic records for students, including transcripts, grades, attendance records, and disciplinary actions.
- The system should allow for the management of faculty information, including details such as lecturer assignments, qualifications, contact information, and survey evaluations.
- The system should include functionality to manage financial aspects related to student accounts, tuition fees, scholarships, and payment processing.
- The system should provide access to student services such as advising (course selection) and extracurricular activities management.
- The system should comply with relevant regulatory requirements and standards for data privacy, security, and accessibility, ensuring the protection of sensitive student information.

4. User Scenarios/Use Cases

Log in

UC Name	***UC01 - Log in***
Summary	This use case describes the process by which a registered user **logs into** the system.
Dependency	**UC30** or **UC33** (Registering account)
Actors	Primary Actor: **Registered User**
Preconditions	The user must have a registered account on the platform.
Description of the Main Sequence	● The user navigates to the login page. ● The user enters their email and password. ● The user submits the login form. ● The system verifies the provided credentials. ● If the credentials are valid, the system grants access to the user's account. ● The user is redirected to their dashboard or the designated landing page.
Description of the Alternative Sequence	**If** the user enters invalid credentials in step 2: ● a. The system prompts the user with an error message indicating invalid credentials. ● b. The user retries entering their credentials.
Non functional requirements	Performance: The login process should be completed within 10 seconds under normal server load conditions. Security: User passwords must be securely hashed and compared against the stored hash to prevent unauthorized access.

	Error Handling: The system should provide clear error messages for incorrect login attempts to assist users in troubleshooting.
Postconditions	The user is logged into their account. Access to the user's account features and functionalities is granted.

View timetable

UC Name	***UC02 - View timetable***
Summary	This use case describes the process in which the user **views and filters** the public timetable
Dependency	**UC01**(Log into the system)
Actors	Primary Actors: All registered users
Preconditions	The user must have a registered account on the platform
Description of the Main Sequence	● The user logs into the system. ● The user navigates to the timetable tab. ● There the user can see the public timetable and filters for specifying the sort of timetable(Graduate or Undergraduate), Class(Degree and field of study for example BA SWE2), Lecturer and Classroom. ● The user can choose to view the timetable of his choice applying each filter individually. ● An interface displaying a timetable is shown after the filters have been applied.
Description of the Alternative Sequence	● The user should have all timetable information already published in the system however in the case that there is missing timetable information or the timetable is not approved the user must wait for the administrative staff and coordinators who are responsible for publishing the correct information.
Non functional requirements	Performance: The timetable filters should be fast and efficient, showing a result in less than 4 seconds under normal server load. Security: The timetable should be accessible to all users, however it must be confirmed that they are logged in, in the case that their session has expired they need to log in again before

	they can access their timetable. Error Handling: The system should provide the user with clear information in the case that incorrect filters are applied. Reliability:The system should ensure that the information regarding the timetable is publicly accessible and accurate.
Postconditions	The user is able to view a user interface displayed with the details of the specified timetable and is informed about the requested timetable.

View Transcript

UC Name	***UC03- View Transcript***
Summary	This use case describes the process of how the student can **see their final grades and gpa (transcript)**.
Dependency	**UC01**(Log in into the system)
Actors	Primary Actor : Student
Preconditions	The user must be logged into their account. The user must be identified as a current student or alumni into the system.
Description of the Main Sequence	● The student opens the application and logs in. ● Press the transcript button. ● The student can now see the gpa and final grades.
Description of the Alternative Sequence	If the current student does not have any grades in the system yet. ● The student opens the application and logs in. ● Press the transcript button. ● They cannot see their grades since none appear.
Non functional requirements	Security: Access to transcript should be restricted to registered users only. (students in this case) Error Handling: The system should provide clear error messages for any issues encountered Reliability: The system should ensure that the transcript page is accurately recorded.
Postconditions	The students are able to see their final grade of each course separately in each semester, and their gpa/cgpa.

View grades

UC Name	**UC04- View grades**
Summary	This use case describes the process of how the student can **see their grades** for each course.
Dependency	**UC01**(Log in into the system)
Actors	Primary Actor : Student
Preconditions	The user must be logged into their account. The user must be identified as a current student into the system.
Description of the Main Sequence	● The student opens the application and logs in. ● Press the Courses button. ● Choose the Course you want to see your grades in. ● The student can see their grades in this course.
Description of the Alternative Sequence	If the current student does not have any grades in the system yet. ● The current student opens the application and logs in. ● Press the Courses button. ● Choose the Course. ● They cannot see their grades since none appear.
Non functional requirements	Security: Access to grades should be restricted to registered users only. (students in this case) Error Handling: The system should provide clear error messages for any issues encountered Reliability: The system should ensure that grades page is accurately recorded.
Postconditions	Now the students can see their grades like : midterm exam,final exam,project etc for every course that they are taking or they took before in the university

View Attendance

UC Name	***UC05- View Attendance***
Summary	This use case describes the process of how the student **can see their attendance** for each course.
Dependency	**UC01**(Log in into the system)
Actors	Primary Actor : Student
Preconditions	The student must be logged into their account.
Description of the Main Sequence	● The student opens the application and logs in. ● Press the Courses button ● Select the course you want to see the attendance ● You press the Attendance Button ● The student can now see the attendance of this course for each lesson
Description of the Alternative Sequence	If the student does not start the university yet he can not have access to the course.
Non functional requirements	Security: Access on attendance should be restricted to registered users only. (current students and alumni in this case) Reliability: The system should ensure that all attendance for each course is accurately recorded.
Postconditions	The students can now view their attendance percentage and see the specific instances where they were absent during the lesson.

Make Document Request

UC Name	***UC06-* Make Document Request**
Summary	This use case describes the process by which a current student can **make document requests** in the system.
Dependency	**UC-01**(Logging into the system) **UC-32** (Processing document requests) **UC-26** (Managing document request payments).
Actors	Primary Actor: **Current Student** Secondary Actor: Registrar's Office
Preconditions	The user must be logged into their account. The user must be identified as a current student into the system.
Description of the Main Sequence	● The current student navigates to the document request section within the dashboard of their account. ● The system presents the list of available types of documents that can be requested by the user as well as with their specifications(language, price). ● The current student selects the type of document they want to request. ● The system prompts the student to provide any necessary details or information required for the document request. ● The student submits the document request form. ● The system acknowledges the request for the student.
Description of the Alternative Sequence	**If** the current student has any penalty debt to pay(usually from 50 €) they can not proceed to make a document request into the system. In order to do so, current students should: ● visit the Finance Office and pay the debt. ● proceed with the steps normally(mentioned in the main sequence)
Non functional requirements	Reliability: The system should ensure that all document requests are accurately recorded and processed. Performance: The document request process should be completed within 15 seconds under normal server load conditions. Usability: The document request interface should be easy to navigate for current students. Scalability: The system should be able to handle an increasing number of document requests without experiencing significant

	degradation in performance.
Postconditions	-The document request is successfully submitted and recorded into the system. -The student receives confirmation of their document request by acknowledging the request and being provided details regarding the document(date of application ,number of copies, total price, payment, status and actions) - The Registrar's Office is notified of the new document request they have to process, from the system.

Course selection

UC Name	***UC07 - Course selection***
Summary	This use case describes the process by which a current student **selects courses for each semester** in the system.
Dependency	**UC-01** (Logging into the system) **UC-20** (Course selection approval)
Actors	Primary Actor: **Current Student** Secondary Actors: -Advisor(to validate and confirm the selection)
Preconditions	-The user must be logged into the system. -The user must be identified as a current student. -Current students must have paid the tuition fee of the respective semester, as well as any other penalty fee, if existent. -Students are eligible to select courses only during the period of time the applications are open.
Description of the Main Sequence	● The current student navigates to the course selection section in the system dashboard. ● The system presents to the student the available courses for the upcoming semester . ● The student selects desired courses from the list. ● The advisor validates the selected courses for any prerequisites and if any issue is existent. ● If no issues are found, the advisor confirms the course selection. ● The student receives confirmation of successful course

	selection (usually by email).
Description of the Alternative Sequence	**If** the student attempts to select a course for which they do not meet prerequisites or if any issue is occured: ● The advisor has the right to not accept a student's course selection. ● Advisor informs the student for steps she/ he should take in order to achieve a successful course selection. ● Current student performs the steps suggested by the advisor (or the system itself). ● Current student does course selection again. (Main sequence)
Non functional requirements	Security: Access to course selection should be restricted to registered users only. (current students in this case) Error Handling: The system should provide clear error messages for any issues encountered during course selection. Reliability: The system should ensure that all courses selected are accurately recorded.
Postconditions	-The selected courses by the students are associated with the student's account for the upcoming semester. -The student should access the course syllabus and other information and functionalities related to the selected courses. (attendance, grades)

View student's debt

UC Name	***UC08 - View Debt***
Summary	This use case describes the process by which a current student **views their student debt** within the system.
Dependency	**UC-01**(Log into the system) **UC-25** (Managing and updating tuition fees)
Actors	Primary Actor: **Current Student** Secondary Actors: Finance Office(to validate and confirm the

	selection)
Preconditions	-The user must be successfully logged into the system. -The user must be identified as a current student.
Description of the Main Sequence	● The current student navigates to the “Finance”section within the dashboard of the system. ● The system retrieves the student's debt information from the database. (Finance Office) ● The system displays the student’s debt. ● The student views the displayed debt information.
Description of the Alternative Sequence	*No alternative sequence. This is just a “view” process.*
Non functional requirements	Security: Access to student’s debt information should be restricted to authenticated users only. Reliability: The system should ensure that all student debts are accurately recorded and processed from the Finance Office. Usability: The student debt interface should be easy to navigate for current students who want to access this feature.
Postconditions	-The current student proceeds to the payment process of the debt (in case of having one),in order to gain access to several actions taken on the system.

Fill out Survey

UC Name	***UC09 - Fill out Survey***
Summary	This use case describes the process by which a current student **fills out a survey** for lecturer evaluation within the system.
Dependency	**UC01** (Logging into the system) **UC07** (Course selection)
Actors	Primary Actor: **Current student**

Preconditions	-The student must be successfully logged into the system. -The current student must be enrolled in a course with a lecturer to evaluate. -The survey application period must be initiated (usually before the final exams).
Description of the Main Sequence	● The student navigates to the system,and clicks the button that contains the survey information (usually the survey can be found at the “Interim grades” dashboard, as the grades cannot be accessed without filling the survey). ● The student selects each survey for each subject. ● The system presents the survey form containing questions and ratings related to the lecturer's performance, teaching style, and course material. ● The student fills out the survey by providing ratings and an optional written feedback for each lecturer/assistant lecturer. ● After completing the survey, the student submits the survey form. ● The system stores the survey responses anonymously for further analysis.
Description of the Alternative Sequence	*No alternative sequence*
Non functional requirements	Security: Access to the survey for lecturer evaluation should be restricted to authenticated students only. Reliability: The system should ensure that all student survey answers are accurately recorded. Usability: The survey should be easy to navigate and complete, as well as it should contain comprehensive content for the course and lecturers.
Postconditions	-The student has successfully filled out and submitted the survey for lecturer and course evaluation anonymously. -The survey responses are stored securely in the system database anonymously. -The student can access other functionalities of the system like; interim grades, finance or attendance, after completing the

	survey.

Apply to Student Clubs

UC Name	***UC10 - Apply to Student Clubs***
Summary	This use case involves the process of users **applying to student clubs.**
Dependency	**UC01** (Log in into the system) **UC28** (Creating student clubs/student clubs already exist)
Actors	Primary Actor: Current Students Secondary Actor: Dean of Students
Preconditions	Student is currently studying at the university, has a registered account and is logged in into the system. Clubs should have been created ahead of time by Administrative staff(Dean of Students). Club application period should be open.
Description of the Main Sequence	● When club applications open, the student navigates to the Clubs tab and can view all the active clubs that have been created & registered. ● Students can select a club they're interested in joining. ● Then they must fill in the form with their personal information such as: Name, Surname, Phone Number and additional information that may be needed and is within the scope of the club. ● Afterwards students need to press the Submit button to confirm and submit the application. ● The information of the application is received by the system and sent to the Dean of Students
Description of the Alternative Sequence	● If the user has provided incomplete information the application is not submitted ● Rather the user is prompted to enter the information again, correctly.
Non functional requirements	Security: The system needs to ensure that only current students who are logged in can access the club application form Reliability: The system needs to ensure that data is passed correctly and reliably from the students applying to the Dean of

	Students and people who are responsible for the club. Usability: The application form should be accessible and use friendly design language to make sure that information is filled correctly by users.
Postconditions	The reviewed application information is saved by the Dean of Students staff in the system and the student is accepted into the club.

View Club Activities

UC Name	***UC11 - View Club Activities***
Summary	This use case involves the process of users **viewing activities** organized by their respective clubs
Dependency	**UC01**(Login) **UC10**(Applying/Joining a student club) **UC28**(Student clubs already exist) **UC29(**Creating and posting activities in Student clubs)
Actors	Primary Actor: Current Students Secondary Actor: Dean of Students
Preconditions	Student is currently studying at the university, has a registered account and is logged in into the system. Clubs should have been created ahead of time by the Dean of Students and the Student must be a registered member of that club. Club activity should have been posted ahead of time.
Description of the Main Sequence	● The Student navigates to the Clubs tab. ● Under the clubs section the Student selects the specific club whose activities the student is interested in. ● The Student can then view in a list all the activities that have been posted in that club by the organizers sorted by the time posted. ● The most recent activities will be at the top of the list of activities so the user can see which activities he can attend.
Alternative Sequence	No alternate sequence as the student can only view the activities.
Non functional requirements	Performance: A fast response rate of less than 3 seconds will make sure that there is a negligible bounce rate and that users

	stay interested in the club's activities. Usability: The user interface of club activities should use friendly design language to make sure that activities get their point through, meaning that information such as date, time, location, description are well communicated.
Postconditions	The user is well informed regarding the activities(and their details) of the clubs that he participates in and can choose to attend if interested.

View Disciplinary/Misconduct records

UC Name	***UC12 - View Disciplinary/Misconduct records***
Summary	This use case describes the process of how the student can **see their disciplinary records** if they have one.
Dependency	**UC01**(Log in into the system.)
Actors	Primary Actor : Students
Preconditions	-The student must be successfully logged into the system. -This student must be a current student or Alumni - This students must have a disciplinary records
Description of the Main Sequence	● The student open the application and logs in. ● Press the profile button ● Press the disciplinary button ● The student can see the disciplinary records with the date and some details
Description of the Alternative Sequence	● If the student does not have a disciplinary record the button does not appear
Non functional requirements	Performance: The system should provide quick and clear-cut access to view the disciplinary record . Security: Access to the discipline page should be restricted to the head of department , ensuring data confidentiality and integrity.
Postconditions	The students are able to see the disciplinary record if they have

	at least one in the system.They can see the date when this problem happened and some details about description or If you're fined by this problem

View Course Syllabus

UC Name	***UC13 - View Course Syllabus***
Summary	Current students should be able to **view the course syllabus** for the courses they are enrolled in.
Dependency	The course syllabus must be prepared and available in a suitable format within the system. The system needs to efficiently retrieve and present the syllabus to the students. **UC01** (Log in)
Actors	Current students who are enrolled in specific courses.
Preconditions	Authentication and authorization of the student. The student must be enrolled in the course for which they want to view the syllabus.
Description of the Main Sequence	● The student logs into the system using their credentials. ● They navigate to the section or module within the system related to course information or syllabi. ● The system displays a list of courses in which the student is currently enrolled. ● The student selects the specific course for which they want to view the syllabus. ● The system retrieves and displays the course syllabus to the student.
Description of the Alternative Sequence	**If** the course syllabus is not available or cannot be found for the selected course, the system displays a message indicating that the syllabus is not available at the moment.
Non functional requirements	Performance: The system should efficiently retrieve and present the course syllabus, even with a large number of enrolled courses and students. Response times should be minimal to provide a seamless user experience. Security: Access to the course syllabus should be restricted to authorized users only, ensuring that students can only view

	syllabus for courses in which they are enrolled
Postconditions	The student successfully views the course syllabus. The system may provide an option for the student to download or print the syllabus for reference.

Apply for Resit Exam

UC Name	***UC14 - Apply for Resit Exam***
Summary	This use case describes the process of **applying for the resit exam/s.**
Dependency	**UC01**(Log in into the system) **UC15**(Load grades into the system)
Actors	Primary actor : Current student Secondary Actors : Academic Staff
Preconditions	-The student must be successfully logged into the system. -The student must be a current student. - -The student must have a final grade of DC or less to apply for the resit exam.
Description of the Main Sequence	● The student opens the application and logs in. ● Press the Resit Application button to have access on it ● You see the courses that you have taken as final grades Dc or less. ● The student select the course he want to apply for resit ● You can see a confirmation by the system that you are approved for this request exam.
Description of the Alternative Sequence	If the student has not yet completed the final exams and the professors have not finalized the grades, you do not have access to this page. ● The system prompts the user with an error message “Resit application has not started yet”. ● The system returns you in the menu page

Non functional requirements	Usability: The system interface should be user-friendly and intuitive, allowing students to easily navigate through the application and submit resit exam requests without confusion. Reliability: The system should be available and functional at all times to accommodate resit exam requests from students whenever they need to access it. Security: The system should ensure the security and privacy of student data, requiring authentication for logins and encrypting sensitive information such as grades and personal details. Performance: The system should have fast response times, minimizing loading times and delays when students log in, access their grades, and submit resit exam requests. Scalability: The system should be able to handle a potentially large volume of concurrent users during peak times, ensuring that all students can access it without experiencing performance degradation.
Postconditions	Upon successful submission of a resit exam request, the system confirms approval to the student and updates their record accordingly.

Load grades into the system

UC Name	***UC15 - Load grades into the system***
Summary	Academic staff members should be able to **upload grades into the system**
Dependency	The grading information must be prepared and available in a suitable format.The upload procedure needs to be handled by the system effectively(verifying the data being uploaded, storing it securely, updating the database correctly).**UC01** (Log in)
Actors	The primary actor is the **academic staff member**(can be either main or assistant lecturer), which is responsible for the particular course.

Preconditions	-Authentication and authorization of the staff member
Description of the Main Sequence	● The staff member logs into the system and goes to the “Courses” section(the courses she/he teaches appears) ● He/she selects the specific course for which they want to upload grades. ● Within the “Course” section, the teacher selects the activity they want to perform(“Upload Grades”, in this case) ● The system then shows all non-finalized grades the teacher has uploaded (for modification)or the teacher can add a new grade(for example, a new quiz or a new assignment) ● The system prompts the teacher to specify the type of grade(e.g. Midterm exam, final exam, assignment etc.) and its weight(e.g. 20%, 30%). ● The system then displays a list of students enrolled in the selected course. ● The teacher uploads the respective points for each student. ● After entering all the points the teacher needs to confirm the upload. ● The system validates for accuracy and consistency. ● The system updates the grades in the database.
Description of the Alternative Sequence	● After uploading the grades, the teacher identifies an error or discrepancy in one of the grades. ● He/she navigates back to the “Grade Uploading” section and now he/she selects the particular non-finalized grades that have been uploaded(not a new one). ● The teacher edits the grade for that student and the system validates the changes. ● The changes are reflected on the database.
Non functional requirements	PERFORMANCE: Efficiency of grade upload, even when a large number of students enrolled in a particular course, to ensure minimal waiting times for teachers. The system should allow for quick and seamless modification of grades. SECURITY: Access to the grades uploaded should be restricted to authorized users only, and data transmission should be encrypted to protect sensitive information.
Postconditions	● The user interface reflects the updated grades. ● The system sends a confirmation message to the teacher indicating the upload was successful.

Record Attendance

UC Name	***UC16 - Record Attendance of Students***
Summary	Academic Academic staff members should be able to **record the attendance** of students in the system.
Dependency	**UC01** (Log in into the system)
Actors	The primary actor is the **academic staff member**(main or assistant lecturer) responsible for the course.
Preconditions	Authentication and authorization of the staff member.
Description of the Main Sequence	● Staff member logs into the system and navigates to the “Course” section. ● Selects the specific course for which they want to record the attendance. ● Within the “Course” section, selects the activity they want to perform, in this case, “Record Attendance” ● The system displays all the attendance records previously uploaded for the selected course, showing the date, the type of teaching and description (e.g, Feb 23 2024 - Lecture, Introduction) ● Staff members should choose to modify an existing attendance record or add a new one. ● If adding a new record, the system prompts the staff member to specify the date, type (lesson, practice, lab), and hours of the session. ● The system then displays a list of enrolled students in the course ● Staff member uploads the attendance record for each student. ● After entering all the attendance records the teacher needs to confirm the upload. ● The system validates for accuracy and consistency. ● The system updates the attendance records in the database
Description of the Alternative Sequence	● After recording attendance, staff member identifies an error or discrepancy. ● Navigates back to the "Record Attendance" section and selects the specific attendance record to modify. ● Edits the attendance record for the session.

	● System validates changes. ● Changes are reflected in the database.
Non functional requirements	● PERFORMANCE: Efficient attendance recording, even with a large number of students enrolled in the course, to ensure minimal waiting times for staff. ● SECURITY: Access to attendance records restricted to authorized users, with encrypted data transmission for protection.
Postconditions	User interface reflects the updated attendance records. System sends confirmation messages indicating successful recording of attendance.

View Syllabus (Academic)

UC Name	***UC17*** **-** ***View Course Syllabus***
Summary	Academic staff members should be able to **access and view the course syllabus** that have previously been published in the system.
Dependency	The course syllabus must have been previously published by the academic staff member in the system.**UC01** (Log in)
Actors	Primary actor:Academic staff member (main or assistant lecturer) responsible for the course.
Preconditions	Authentication and authorization of the staff member. The course syllabus must be previously published by the staff member.
Description of the Main Sequence	● Staff member logs into the system. ● Navigates to the “Course” section. ● Selects the specific course for which they want to view the syllabus. ● Within the course section, select the option to view the syllabus. ● The system displays the course syllabus previously published by the staff member.

Description of the Alternative Sequence	• N/A
Non functional requirements	PERFORMANCE: The system should provide quick access to the syllabus information. SECURITY: Access to the syllabus should be restricted to the authorized users only.
Postconditions	The system maintains security and privacy of the syllabus content.

Access Survey Responses

UC Name	***UC18 - Access Survey Responses***
Summary	Academic staff members should be able to **access the survey responses** submitted by students for their respective courses.
Dependency	Survey responses must be submitted by students for their teachers. Access to survey responses should be enabled after the midterms exams week. **UC01** (Log in)
Actors	Primary Actor:**Academic staff member** (main or assistant lecturer) responsible for the course.
Preconditions	Authentication and authorization of the staff member. Survey responses must have been submitted by students for the respective courses. Midterm exams week must have passed.
Description of the Main Sequence	• After the midterm exams week, staff members log into the system. • Navigates to the “Course” section. • Selects the specific course they teach. • Within the course section, select the option to access survey responses. • The system displays the survey responses submitted by students for that course.The survey show an overall assessment, without revealing any student name or identification.
Description of	• N/A

the Alternative Sequence	
Non functional requirements	PERFORMANCE: The system should efficiently retrieve and display survey responses, even for courses with a large number of enrolled students. SECURITY: Access to survey responses should be restricted to authorized users only.
Postconditions	The academic staff member successfully accesses the survey responses for each course. Survey ensures anonymity. The system maintains security and privacy of the survey responses.

Publish Syllabus

UC Name	**UC19 - *Publish Syllabus***
Summary	The main lecturer should be able to **publish the course syllabus**.
Dependency	The course syllabus must be prepared and available in a suitable format within the system. The system needs to handle the upload procedure effectively, verifying the data being uploaded, storing it securely, and updating the database correctly.**UC01** (Log in)
Actors	Main lecturer
Preconditions	Authentication and authorization of the main lecturer. The course for which the syllabus will be published is selected.
Description of the Main Sequence	● The main lecturer logs into the system. ● They navigate to the course management module. ● The main lecturer selects the specific course for which they want to publish the syllabus. ● The system displays the current draft or template of the course syllabus. ● The lecturer updates or modifies the syllabus content as necessary.

	• The main lecturer publishes the updated syllabus.
Description of the Alternative Sequence	• If the main lecturer encounters an error or discrepancy while publishing the course syllabus, such as an invalid file format or incomplete content, the system displays an error message. • The main lecturer is prompted to correct the error or provide missing information. • The lecturer makes the necessary adjustments to the syllabus. • After making the corrections, the lecturer attempts to publish the syllabus again. • The system validates the updated syllabus content. • If the validation is successful, the system publishes the syllabus. • If the validation fails again, the system displays another error message, prompting the lecturer to review and correct the issues. • This loop continues until the syllabus is successfully published or the lecturer cancels the operation.
Non functional requirements	Performance: The system should efficiently handle the upload and publication of course syllabi, ensuring minimal waiting times for lecturers. Security: Access to the syllabus publication feature should be restricted to authorized main lecturers only. Data transmission should be encrypted to protect sensitive information.
Postconditions	• The course syllabus is published and made available to students enrolled in the course. • The system sends a confirmation message to the main lecturer indicating the publication was successful.

Approve Course Selection

UC Name	***UC20 - Approve Course Selection***
Summary	Advisors should be able to **approve students' course selection**.
Dependency	The course selection data must be available in the system. The system needs to handle the approval process effectively, verifying the selected courses and updating the student's record accordingly. **UC01** (Log in into the system)
Actors	Advisor
Preconditions	Authentication and authorization of the advisor. The student has submitted their course selection for approval.
Description of the Main Sequence	● The advisor logs into the system. ● They navigate to the advisor dashboard or student management module. ● The advisor selects the student whose course selection they want to approve. ● The system displays the student's submitted course selection. ● The advisor reviews the selected courses and verifies their suitability. ● The advisor approves the student's course selection.
Description of the Alternative Sequence	● If the advisor identifies any issues or discrepancies with the student's course selection, such as missing prerequisites or course conflicts, the advisor contacts the student to discuss potential modifications. ● The advisor works with the student to resolve the issues and adjust the course selection as necessary. ● After resolving the issues, the advisor repeats the approval process by reviewing and approving the modified course selection.
Non functional requirements	Performance: The system should efficiently handle the approval process, ensuring minimal waiting times for advisors. Security: Access to the course selection approval feature should

	be restricted to authorized advisors only.
Postconditions	● The student's course selection is approved, and their record is updated accordingly. ● The system sends a notification to the student informing them that their course selection has been approved.

Review Syllabus

UC Name	***UC21 - Reviews syllabus material***
Summary	This use case describes the process in which the head of department **reviews syllabus material** of a specific lecturer.
Dependency	**UC01**(Login) **UC19**(Main Lecturer publishes syllabus)
Actors	Primary Actor: Head of Department Secondary Actors: Main lecturer(who published the syllabus)
Preconditions	The head of department is authenticated and authorized in the system. Syllabus material is available for review, therefore a main lecturer must have published it beforehand.
Description of the Main Sequence	● The head of department logs into the student management system. ● The head of department navigates to the syllabus dashboard. ● The head of department can view any published course syllabus up for review. ● The system displays the syllabus material for a selected course(s). ● The head of department reviews the syllabus content, making any necessary changes. ● The head of department approves the revised syllabus material.

Description of the Alternative Sequence	**If** the head of department encounters any errors or inconsistencies in the syllabus material: -The head of department flags the issues, and denies submission of the syllabus. -The system notifies the syllabus' main lecturer of his rejection. -The head of department proceeds with the review process after/if the issues are addressed.
Non functional requirements	Security: Access to the syllabus section should be restricted to the head of department only.
Postconditions	The reviewed syllabus material is saved in the system, reflecting any changes made by the head of department.

Create new Course

UC Name	***UC22 - Create new Course***
Summary	This use case involves the process of **creating new courses**.
Dependency	**UC01**(Login)
Actors	Primary Actor: Head of Department Secondary Actors: Academic Staff (Main Lecturer and Assistant Lecturers)
Preconditions	The head of department is authenticated and authorized in the system.
Description of the Main Sequence	● The head of department logs into the student management system. ● The head of department navigates to the "New Course" tab. ● The head of department fills out the necessary details for the new course, including course title, description,

	prerequisites, and credit hours. ● The head of department submits the course creation form. ● The system notifies any relevant academic/administrative personnel of the newly created course.
Description of the Alternative Sequence	If the entered information for the new course is incomplete or invalid: -The system prompts the head of department to provide the missing or corrected information.
Non functional requirements	Error handling:The system should provide clear error messages for any issues or invalid information during course creation. Security: Access to the course creation functionality should be restricted to authorized personnel, ensuring data integrity and preventing unauthorized modifications.
Postconditions	A new course is successfully created and added to the course catalog within the system, further utilized through course selection.

Assigning lecturer to teach Course

UC Name	***UC23 - Assigning lecturers to courses***
Summary	This use case describes the process of **assigning lecturers to a course.**
Dependency	**UC01** (Logging into the system) **UC22** (Creating new course)
Actors	Primary Actors: Head of Department Secondary Actors: Academic Staff
Preconditions	The Head of Department must be authenticated and authorized to access the system. The lecturer/s are required to meet the requirements and qualifications for teaching the course.
Description of the Main	● The Head of Department logs into the system.

Sequence	● The Head of Department navigates to the course management. ● The system displays a list of available lecturers who are qualified to teach a selected course. ● The Head of Department reviews the list of available lecturers and their qualifications and selects one or more lecturers to assign to the course. ● The Head of Department updates the course information with the assigned lecturers. ● The system notifies the assigned lecturers of their new course assignments.
Description of the Alternative Sequence	If there are no available lecturers qualified to teach the selected course: -The system notifies the Head of Department of the unavailability. -The Head of Department may choose to proceed with other arrangements.
Non functional requirements	Performance: The system should provide a responsive interface for selecting lecturers, ensuring a smooth and efficient process for the Head of Department. Security: Access to lecturer assignment features should be restricted to authorized personnel, maintaining confidentiality of lecturer information. Reliability: The system should accurately reflect the current availability and qualifications of lecturers for assignment, minimizing errors in lecturer selection.
Postconditions	The selected course is successfully updated with the assigned lecturers within the system. Lecturers are notified of their new course assignments, and the lecturer/s-course pairings will be available for student course selection.

Upload Timetable

UC Name	***UC24 - Upload Timetable***
Summary	This use case involves the Coordinator **uploading and making changes to the timetable** records within the student management system.
Dependency	**UC01**(Login)

Actors	Primary Actor: Coordinator Secondary Actors: None
Preconditions	The Coordinator is authenticated and authorized within the system.
Description of the Main Sequence	● The Coordinator logs into the student management system. ● The Coordinator navigates to timetable management. ● The Coordinator selects the timetable records to upload or modify. ● For uploading new timetable records: a. The Coordinator prepares the timetable data in the required format. b. The Coordinator uploads the timetable data into the system. c. The system validates the uploaded data. d. If the validation is successful, the system adds the new timetable records to the database. ● For making changes to existing timetable records: a. The Coordinator selects the timetable record to be modified. b. The Coordinator makes the necessary changes to the timetable details. c. The system validates the modified data. d. If the validation is successful, the system updates the timetable record in the database. ● The system notifies Students, Academic Staff and Administrative Staff of any changes to the timetable.
Description of the Alternative Sequence	If the uploaded timetable data contains errors or inconsistencies(for example multiple courses overlapping for the same class): -The system notifies the Coordinator of the errors. -The Coordinator corrects the errors in the timetable data.

Non functional requirements	Performance: The system should handle timetable uploads and modifications efficiently, minimizing processing time to ensure timely updates. Security: Access to timetable management features should be restricted to the Coordinators only.
Postconditions	Timetable records are successfully uploaded or modified within the system, reflecting any changes made by the Coordinator. Notifications are sent to relevant stakeholders regarding the updates.(Students, Academic Staff and Administrative Staff)

Manage Tuition fees

UC Name	***UC25 - Manage Tuition fees***
Summary	This use case involves the Finance Office **managing tuition fees** for each student, keeping track of the payments according to scholarship and debt.
Dependency	**UC01**(Login) **UC30**(create student account)
Actors	Primary Actor: Finance Office Secondary Actors: Students
Preconditions	The Finance Office is logged into the system. It retrieves information from external sources like the bank account of the university, and all occurring transactions.
Description of the Main Sequence	● The Finance Office logs into the student management system. ● The Finance Office navigates to tuition fee management. ● Based on their transcripts, The Finance Office/system updates tuition fees for each scholarship student accordingly. ● The Finance Office submits the fees as debt to be viewed on the students' side.

	● The Finance Office, through external sources, checks which payments have been processed within the deadline, and updates corresponding student debt.
Description of the Alternative Sequence	-If a student does not pay their tuition fees on time, The Finance Office will be obligated to charge a late payment fee until a second deadline, consequently updating the debt on the system. -If they still haven't paid by the second deadline, they will not be able to attend any courses for the term.
Non functional requirements	Security: Access to tuition fee management features should be restricted to authorized personnel, ensuring data confidentiality and integrity.
Postconditions	Tuition fees for the selected academic term are successfully managed and updated within the system, reflecting any changes made by the Finance Office. Invoices or fee statements are generated for enrolled students based on the updated fee structure. (In case of debt higher than 50 Euro, other UC-s, e.g. the document request sent from the students, are affected.)

Document Request Payment

UC Name	***UC26 - Document Request Payment***
Summary	This use case involves the Finance Office **managing document request payments** within the student management system.
Dependency	**UC01**(Login) **UC06**(Student makes a document request) **UC32**(Registrar processes a requested document)
Actors	Primary Actor: Finance Office Secondary Actors: Students, Registrar's Office
Preconditions	The Finance Office is logged in the system.

	The student must have requested a document.
Description of the Main Sequence	● The Finance Office logs into the student management system. ● The Finance Office navigates to the document request payment management section. ● The Finance Office views the list of pending document requests that require payment. ● The Finance Office selects a document request for payment processing. ● The Finance Office verifies the details of the document request and the associated fee. ● The Finance Office initiates the payment process, by manually entering payment details. (The student must have paid their fee at the school premises.) ● The system updates the status of the document request to indicate that payment has been received. ● The Finance Office generates a receipt or confirmation for the payment transaction, notifying the Registrar's Office of the payment confirmation.
Description of the Alternative Sequence	● If the student does not pay their fee, their document request simply remains pending for a short period of time, until it expires.
Non functional requirements	Performance: The system should handle document request payment processing efficiently, ensuring timely confirmation of payments. Security: Access to document request payment management features should be restricted to authorized personnel, ensuring secure handling of payment information.
Postconditions	Payment for the selected document request is successfully processed within the system, and the status of the document request is updated. A receipt is generated for the transaction, and the Registrar's office is notified accordingly. The latter will proceed to provide the requested document for the student.

Record disciplinary/misconduct cases

UC Name	***UC27- Record disciplinary cases***
Summary	This use case describes the process by which The Dean of Students Office **records any disciplinary/misconduct measures.**
Dependency	**UC01** (Log in)
Actors	Primary Actor: Dean of Students personnel
Preconditions	- Dean of Students Office personnel are logged in the system. - Student information is already stored in the system.
Description of the Main Sequence	● Dean of Students logs into the system. ● Dean selects the student involved in the disciplinary/misconduct issue. ● Dean records details of the incident, including date, nature of the misconduct, and any actions taken. ● System saves the recorded information.
Description of the Alternative Sequence	None
Non functional requirements	**Security:** Access to disciplinary records should be restricted to authorized personnel only. **Performance:** The system should handle a large volume of records efficiently. **Reliability:** Data recording should be reliable and accurate.
Postconditions	The disciplinary/misconduct measures are successfully recorded in the system for future reference.

Create new Student Clubs

UC Name	***UC28 -* Create Student Clubs**

Summary	This use case describes the process of **creating new student clubs.**
Dependency	**UC01** (Log in)
Actors	Primary Actor: Dean of Students personnel
Preconditions	Dean of Students personnel is logged into the system with appropriate permissions. There are no existing clubs with conflicting names in the system.
Description of the Main Sequence	● The Dean of Students logs into the system. ● They navigate to the section for creating new student clubs. ● They input the necessary information for the new club (e.g., club name, purpose, advisor). ● The system validates the information provided. ● If validation is successful, the system creates the new club.
Description of the Alternative Sequence	If any required information is missing or invalid, the system prompts the Dean of Students to correct the errors before proceeding with club creation.
Non functional requirements	**Security:** Access to club activity posting should be restricted to the Dean of Students or authorized personnel only. **Reliability:** Club creation should be reliable, ensuring accurate data entry and validation.
Postconditions	A new student club is successfully created in the system.

Post Student Club Activities

UC Name	***UC29 - Post Student Club Activities***
Summary	This use case describes the process of **posting activities for a**

	student club.
Dependency	**UC01** (Log in) **UC28** (a student club already exists)
Actors	Primary Actor: Dean of Students personnel
Preconditions	Dean of Students personnel is logged into the system with appropriate permissions. Student clubs are already registered in the system.
Description of the Main Sequence	● Dean of Students logs into the system. ● Dean navigates to the section for managing student club activities. ● Dean selects the club for which they want to post activities. ● Dean inputs details of the activity, including date, time, location, and description. ● System saves the posted activity.
Description of the Alternative Sequence	If you want to post an activity for a club that is not registered, you would have to go to the registering clubs section of the system first.
Non functional requirements	**Security:** Access to club activity posting should be restricted to the Dean of Students or authorized personnel only. **Reliability:** The system should reliably save posted activities and ensure they are displayed accurately.
Postconditions	The activity is successfully posted for the selected student club, and the information is saved in the system for students to view.

Create new Student Account

UC Name	***UC30 - Create new Student Account***
Summary	This use case describes the process of **creating new student accounts.**

Dependency	**UC01** (Log in)
Actors	Primary Actor: Registrar's Office Staff
Preconditions	Staff members of the Registrar's Office have logged into the system with appropriate permissions.
Description of the Main Sequence	● Staff members of the Registrar's Office log into the system. ● They navigate to the section for creating new student accounts. ● They input the required student information ● The system validates the information and generates a unique student ID. ● Staff members review and confirm the creation of the new account. ● The system saves the student information and assigns the new student ID.
Description of the Alternative Sequence	If any required information is missing or invalid, the system prompts staff members to correct the errors before proceeding with account creation.
Non functional requirements	**Reliability:** The account creation process should be reliable, ensuring accurate data entry and ID generation. **Security:** The system must adhere to security protocols to protect student data from unauthorized access or breaches. **Privacy:** The system should handle student information in compliance with relevant privacy regulations, ensuring confidentiality and data protection.
Postconditions	A new student account is successfully created with a unique student ID assigned, and the student information is saved in the system.

Update/Edit student information

UC Name	***UC31 - Update/Edit student information***
Summary	This use case describes the process of **updating/editing student information.**
Dependency	**UC01** (Log in) **UC30(**The student account to be edited should

	already exist)
Actors	Registrar's Office personnel
Preconditions	- Registrar's Office personnel have logged into the system with appropriate permissions. - Student accounts are already created and accessible in the system.
Description of the Main Sequence	● Registrar's Office personnel log into the system. ● They search for the student account whose information needs to be edited/updated. ● They select the specific student account. ● The system displays the current student information. ● Registrar's Office personnel make the necessary edits/updates to the information. ● User clicks to save the changes. ● If the edited information is not valid, the system prompts the user to correct the errors. ● The system updates the student account with the validated edited/updated information.
Description of the Alternative Sequence	If the student account cannot be found, Registrar's Office personnel may search using different criteria or initiate the creation of a new student account. If the edited information is not valid, the user is prompted by the system.
Non functional requirements	**Security:** Access to editing/updating student account information should be restricted to authorized personnel only. **Privacy:** The system should handle student information in compliance with relevant privacy regulations, ensuring confidentiality and data protection. **Reliability:** Changes made to student account information should be accurately reflected in the system without errors.
Postconditions	The student account information is successfully edited/updated in the system, and the changes are saved and reflected in the updated student records.

Process Document Requests

UC Name	***UC32 - Process Document Requests***
Summary	This use case describes the process of **processing document requests.**
Dependency	**UC01** (Log in) **UC26** (finance office manages document request payments) **UC06** (student makes document requests)
Actors	Primary Actors: Registrar's Office personnel Secondary Actors: Students, Finance Office
Preconditions	Registrar's Office personnel are authenticated and authorized to access the system. Student records and document templates are stored in the system. Finance Office has approved payment for student request.
Description of the Main Sequence	● Registrar's Office personnel log into the system. ● They navigate to the document request section. ● The system generates the requested document based on the student's information and the selected template. ● Registrar's Office personnel review the document for accuracy. ● If everything is correct, the system sends the document request approval to the specified recipient. (Ready status)
Description of the Alternative Sequence	-If the document is generated incorrectly, the Registrar's Office personnel can initiate a review process. -Registrar's Office personnel review the document again, identifying errors and making necessary corrections. -Once corrected, the system sends the document request approval to the specified recipient. (Ready status)
Non functional requirements	Performance: The system should efficiently process document requests, minimizing wait times for students. Reliability: Document generation should be accurate and dependable. Security: Access to student records and document templates should be restricted to authorized personnel only.
Postconditions	The requested document is successfully generated and the specified recipient is notified of the “Ready” status.

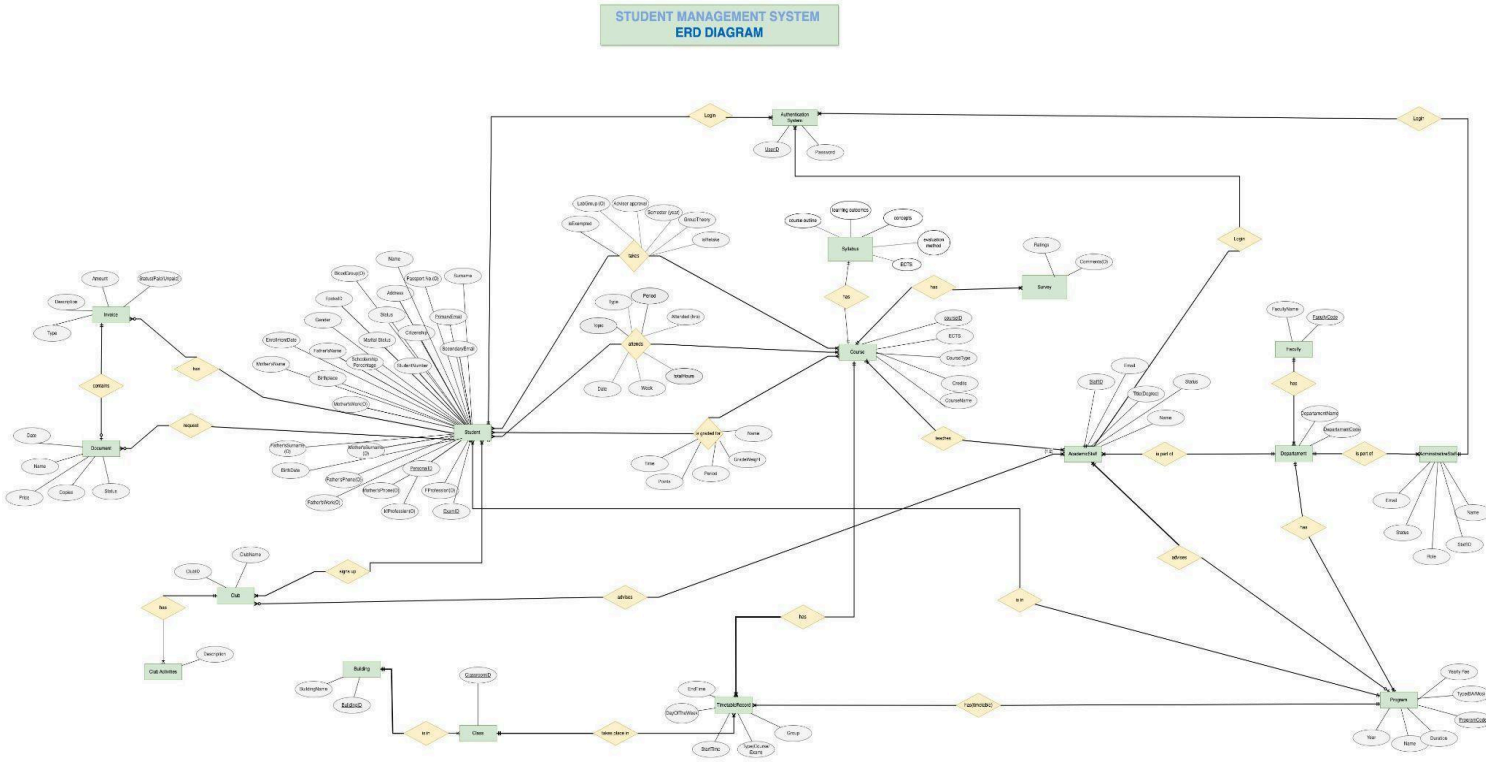
Create staff account

UC Name	***UC33 - Create Academic Staff Account***
Summary	The IT office should be able to create new academic staff accounts in the system.
Dependency	**UC01** (Log in)
Actors	Primary actor:IT administrator.
Preconditions	Authentication and authorization of the IT professional to perform administrative tasks. Access to administrative privileges or permission to create, modify, or delete user accounts.
Description of the Main Sequence	● The IT admin logs into the system with administrative credentials. ● They navigate to the user management. ● Within the user management section, they select the option to create a new user account. ● They enter the required information for the new academic staff member, including username, password, contact information, and role designation. ● After entering the necessary details, they submit the form to create the new account. ● The system validates the information provided and securely saves the username and password in the database, and confirms the successful creation of the new academic staff account.
Description of the Alternative Sequence	● N/A
Non functional requirements	PERFORMANCE: The system handles account creation requests efficiently to ensure minimal waiting times for the IT office or system administrator. SECURITY:The system enforces strong password policies and encrypts sensitive data to ensure the security of newly created accounts. USER MANAGEMENT: The system provides a user-friendly interface with clear prompts and error messages for input validation.

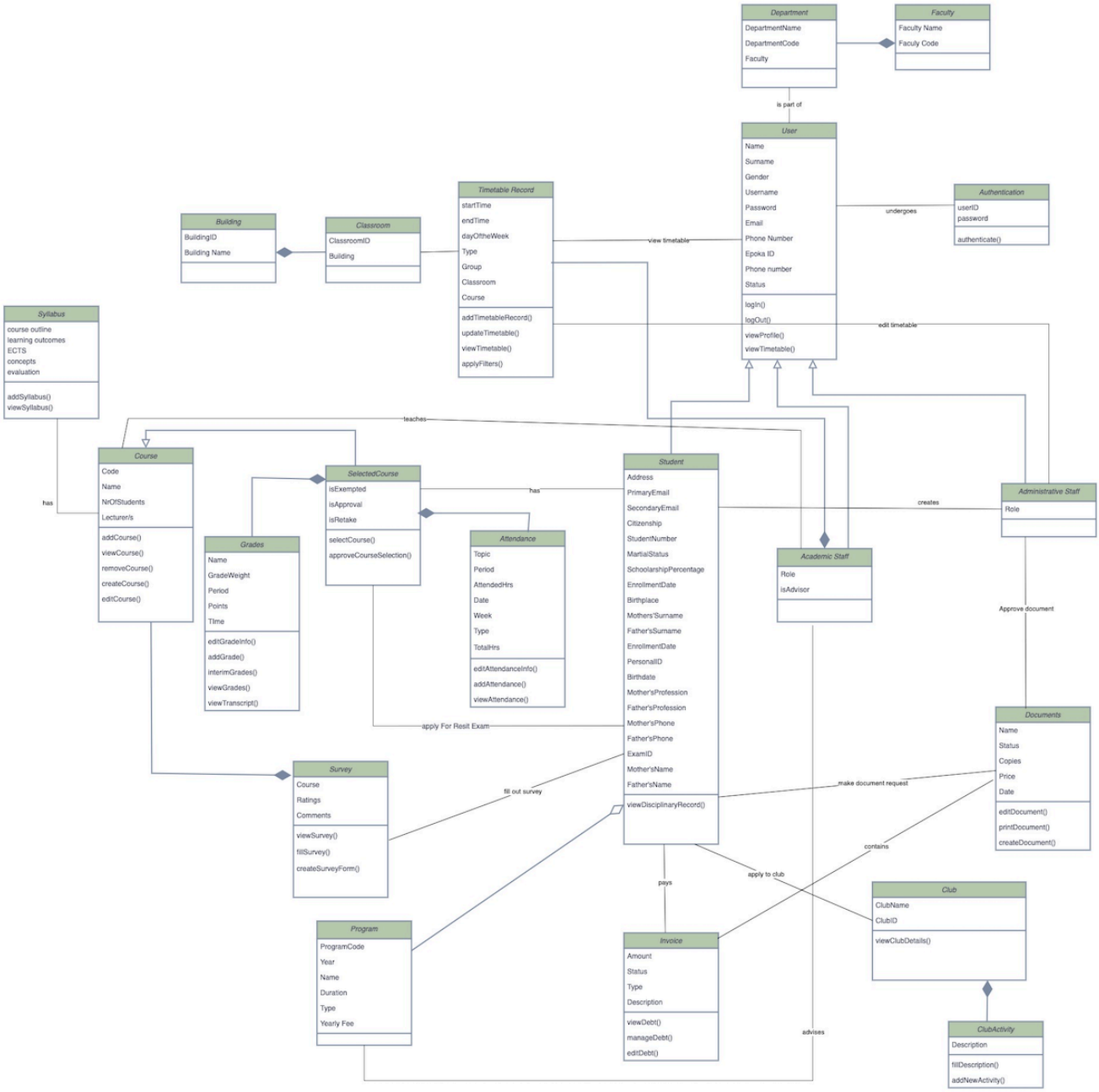
	SCALABILITY: The system accommodates an increasing number of academic staff members over time without compromising performance or usability.
Postconditions	A new academic staff account is successfully created in the system, and the username and password details are securely saved in the database. The newly created account is available for the academic staff member to use for authentication and access to system features.

Diagrams

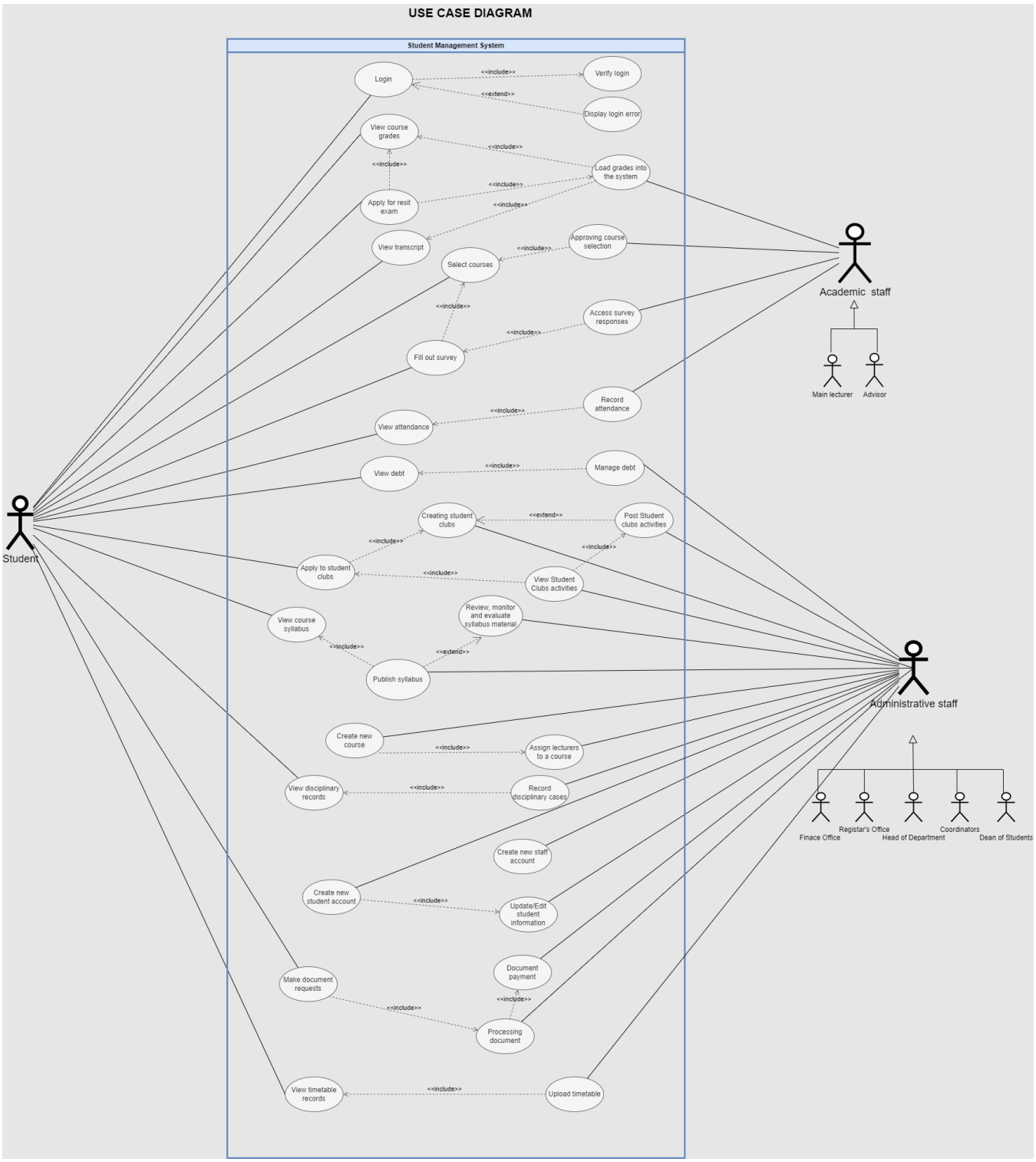
ER diagram



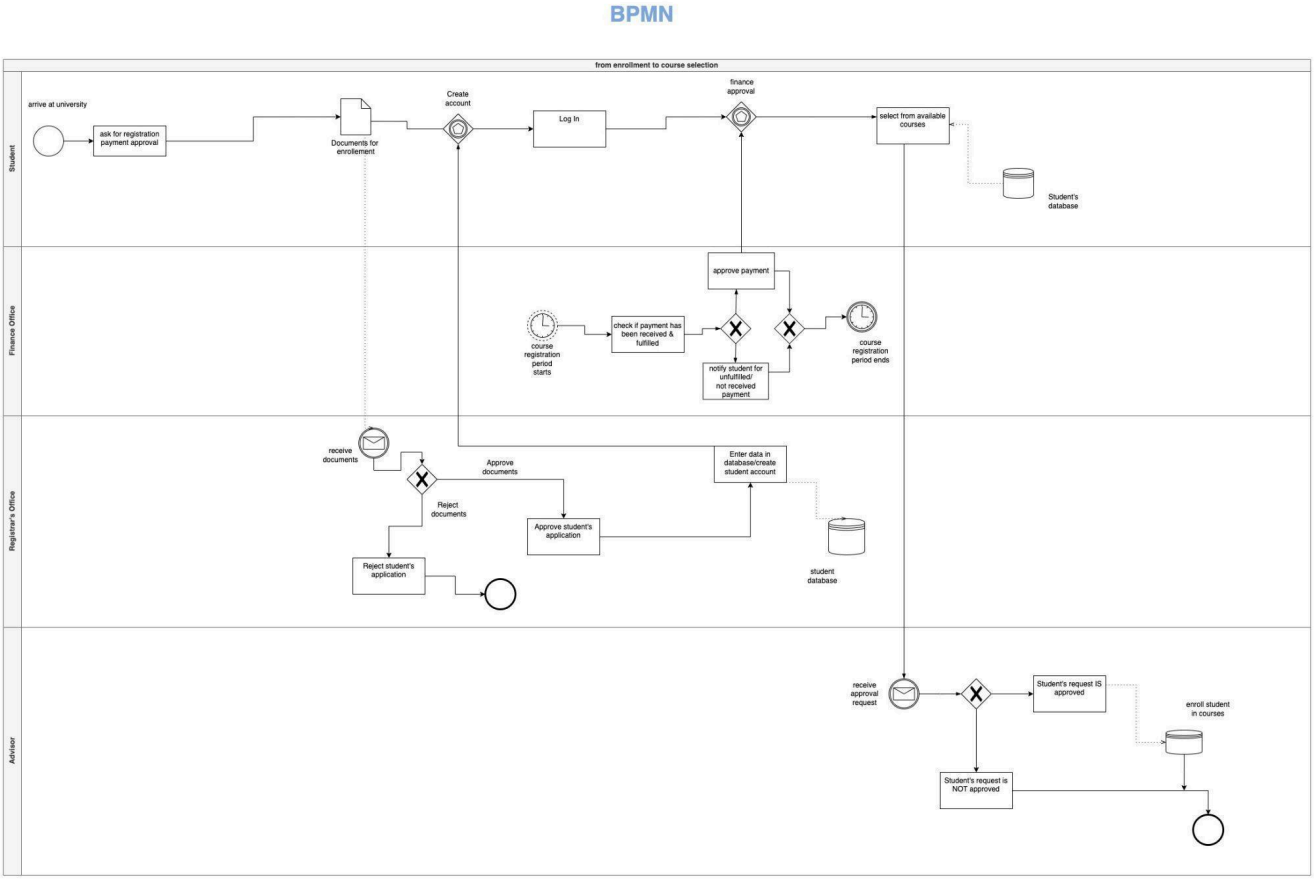
Class diagram



Use Case diagram

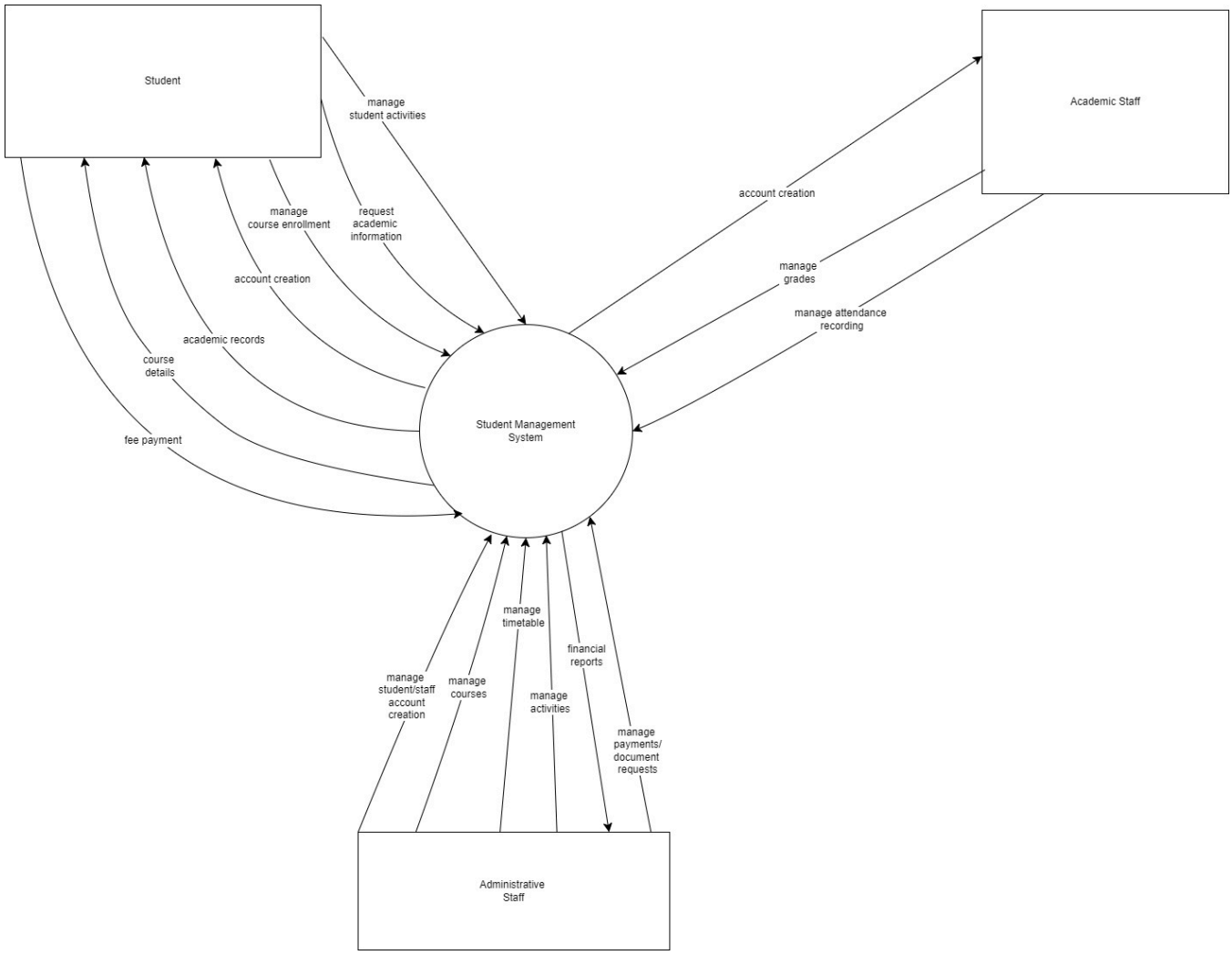


BPMN

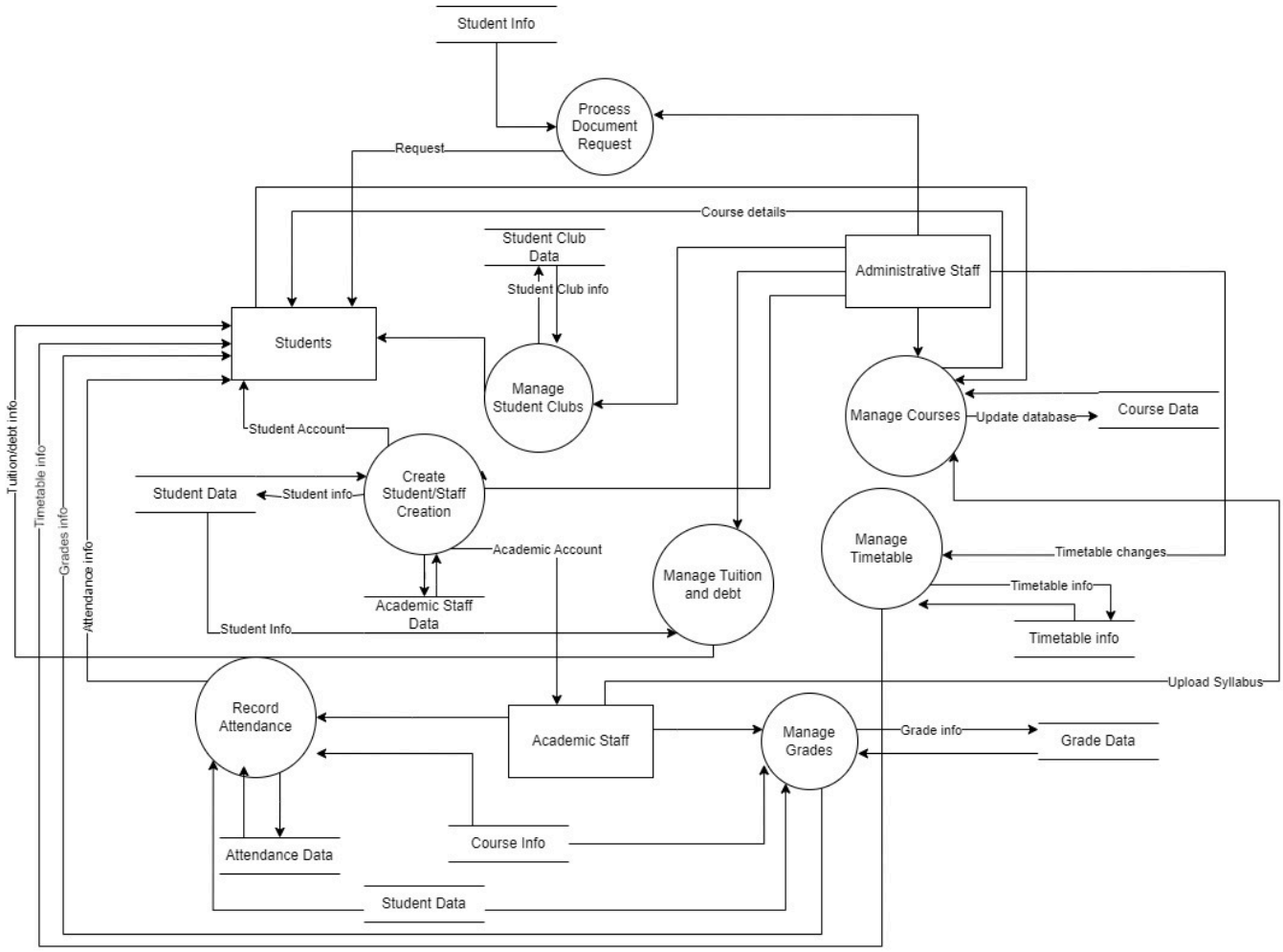


DFD

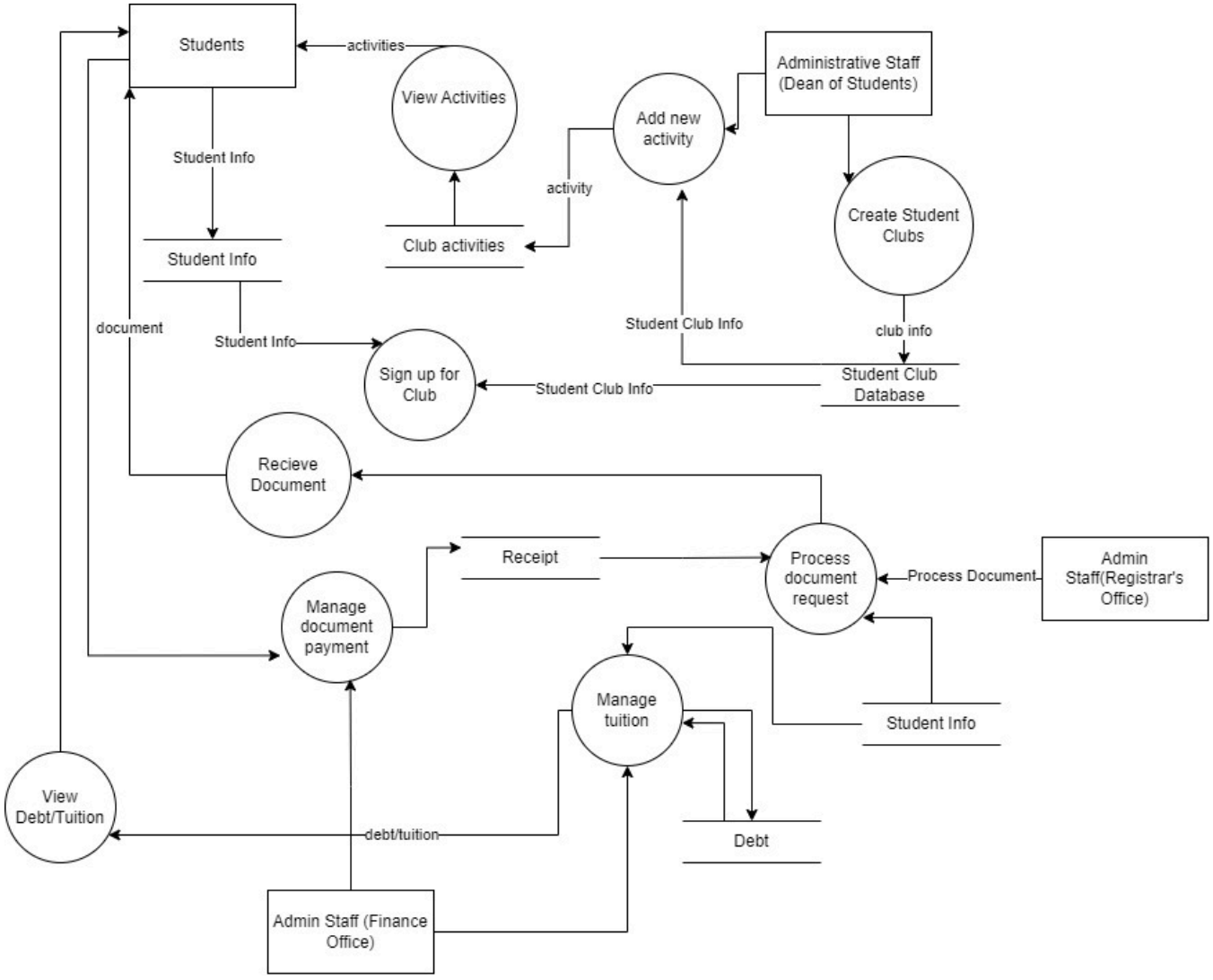
Level 0

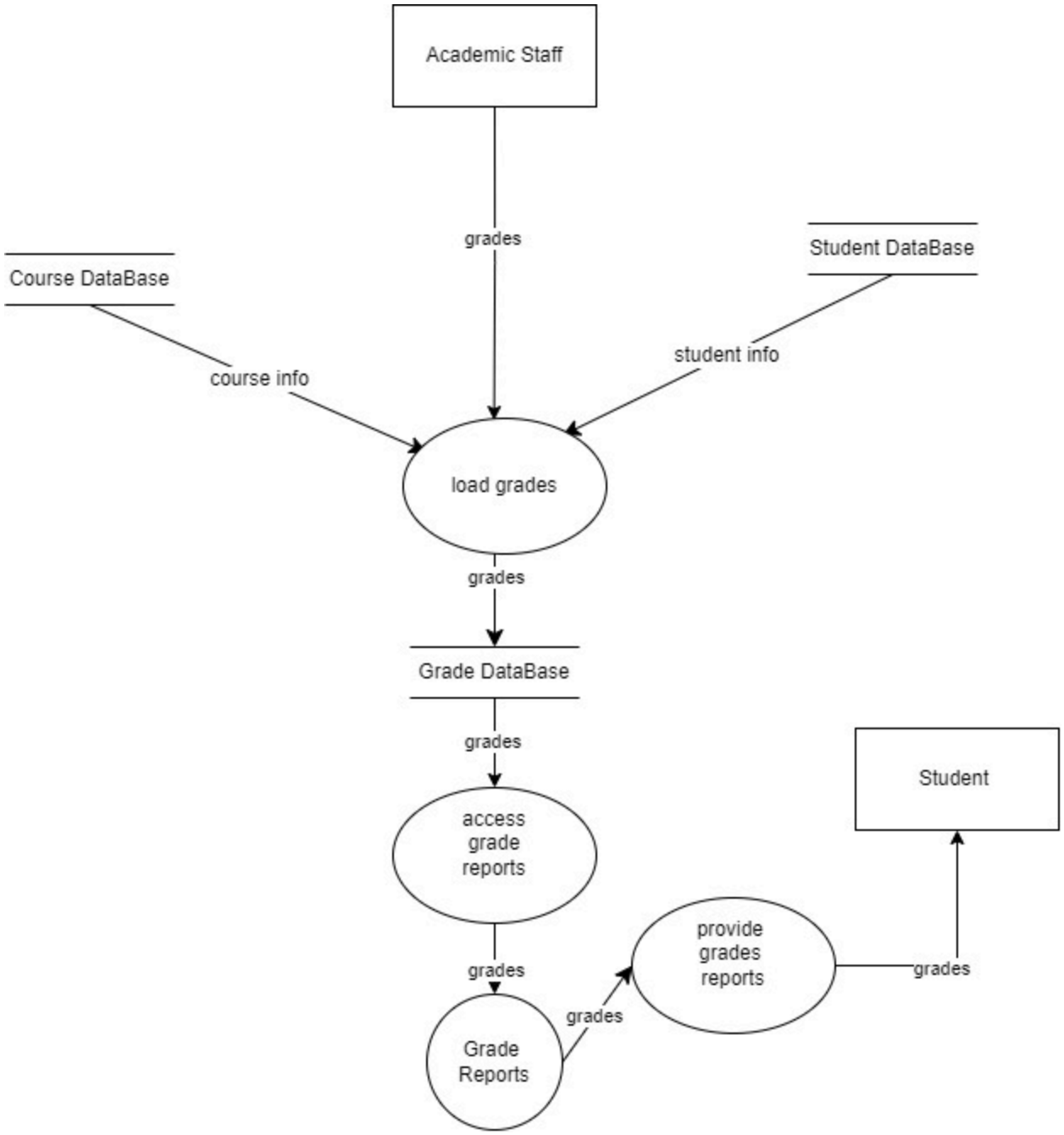

Student
Academic Staff
Student Management System
Administrative Staff
manage student activities
manage course enrollment
request academic information
account creation
academic records
course details
fee payment
account creation
manage grades
manage attendance recording
manage timetable
financial reports
manage student/staff account creation
manage courses
manage activities
manage payments/ document requests

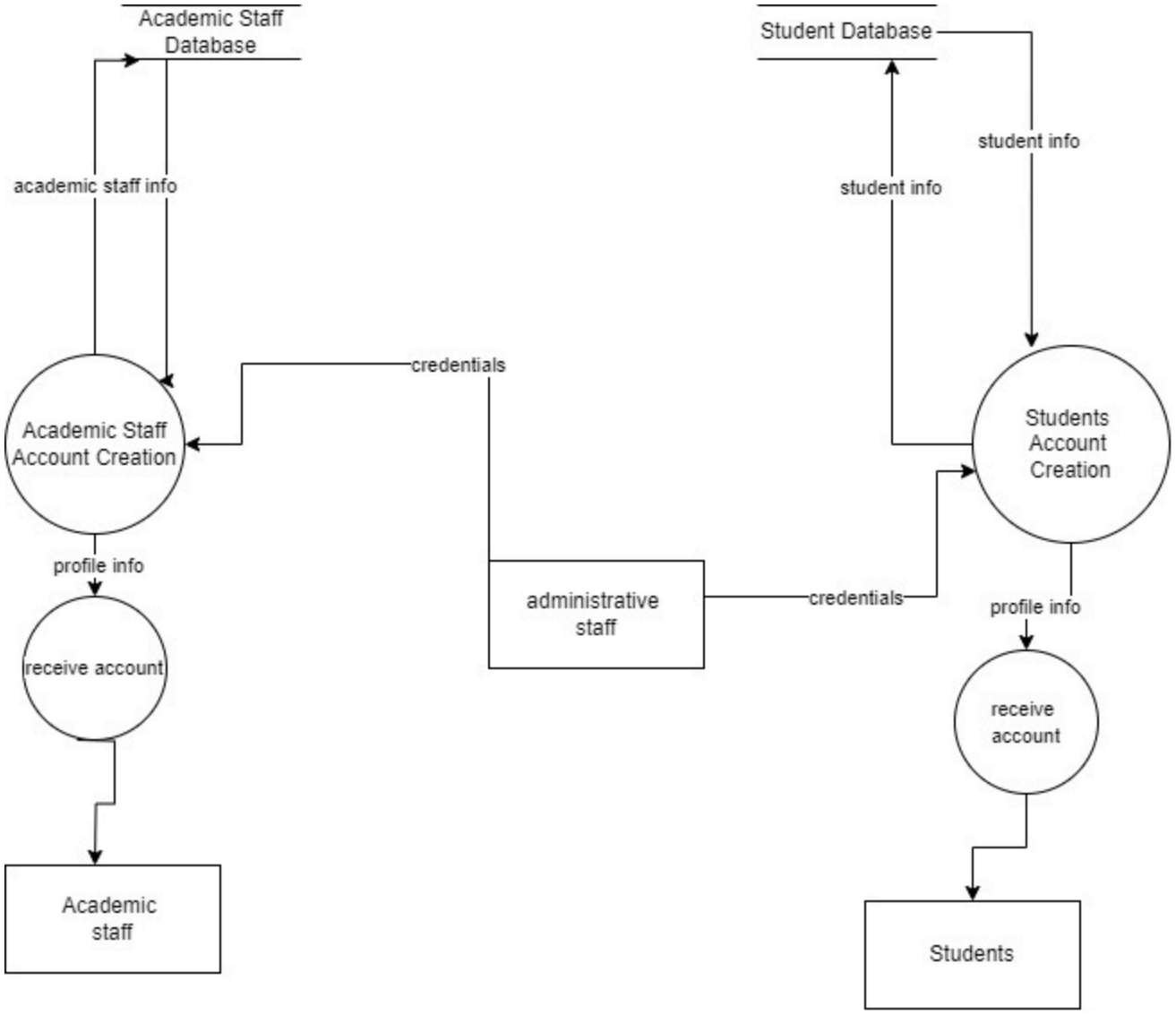
Level 1

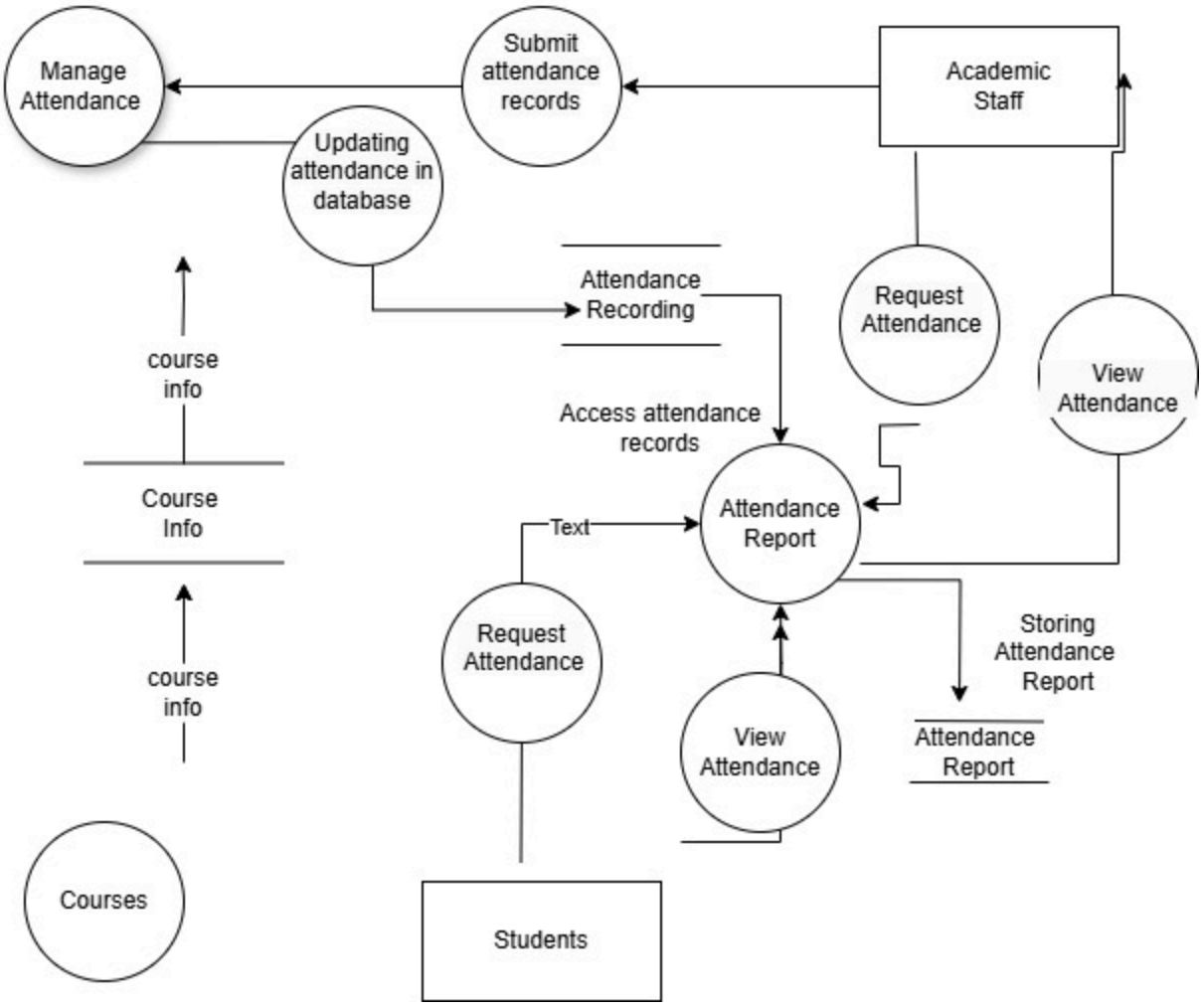

Student Info
Process Document Request
Request
Course details
Student Club Data
Student Club info
Administrative Staff
Students
Manage Student Clubs
Manage Courses
Update database
Course Data
Tuition/debt info
Timetable info
Grades info
Attendance info
Student Account
Create Student/Staff Creation
Student Data
Student info
Academic Account
Academic Staff Data
Manage Tuition and debt
Manage Timetable
Timetable changes
Timetable info
Timetable info
Student Info
Upload Syllabus
Record Attendance
Academic Staff
Manage Grades
Grade info
Grade Data
Attendance Data
Course Info
Student Data

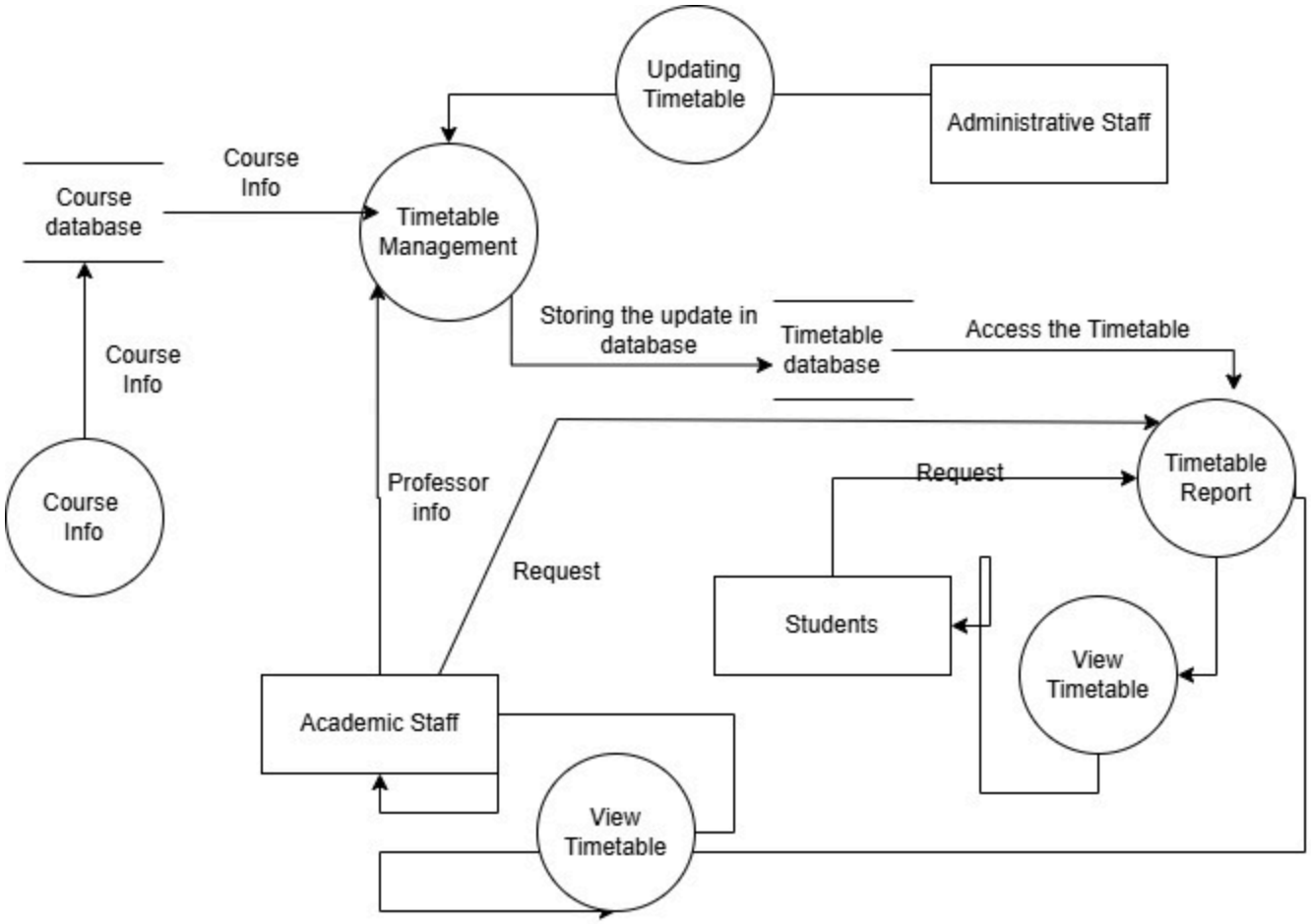
Level 2


Students
activities
View Activities
Administrative Staff (Dean of Students)
Add new activity
Student Info
activity
Create Student Clubs
Student Info
Club activities
document
Student Info
Student Club Info
club info
Sign up for Club
Student Club Info
Student Club Database
Recieve Document
Receipt
Process document request
Process Document
Admin Staff(Registrar's Office)
Manage document payment
Manage tuition
Student Info
View Debt/Tuition
debt/tuition
Debt
Admin Staff (Finance Office)


Academic Staff
grades
Course DataBase
Student DataBase
course info
student info
load grades
grades
Grade DataBase
grades
Student
access grade reports
provide grades reports
grades
grades
grades
Grade Reports

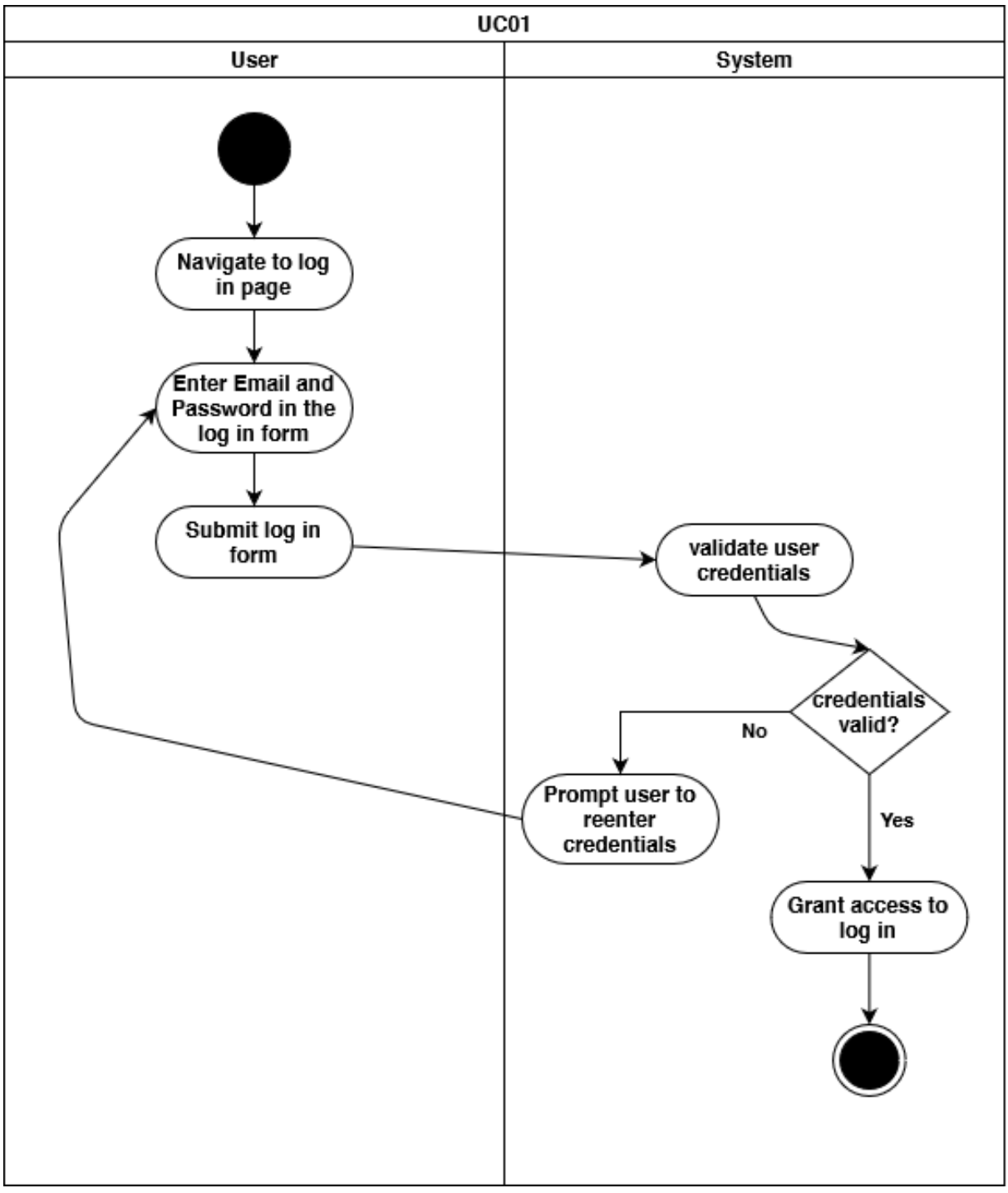

Academic Staff Database
Student Database
academic staff info
student info
student info
credentials
Academic Staff Account Creation
Students Account Creation
profile info
administrative staff
credentials
profile info
receive account
receive account
Academic staff
Students


Manage Attendance
Submit attendance records
Academic Staff
Updating attendance in database
Attendance Recording
Request Attendance
View Attendance
course info
Access attendance records
Course Info
Attendance Report
Text
course info
Request Attendance
Storing Attendance Report
View Attendance
Attendance Report
Courses
Students

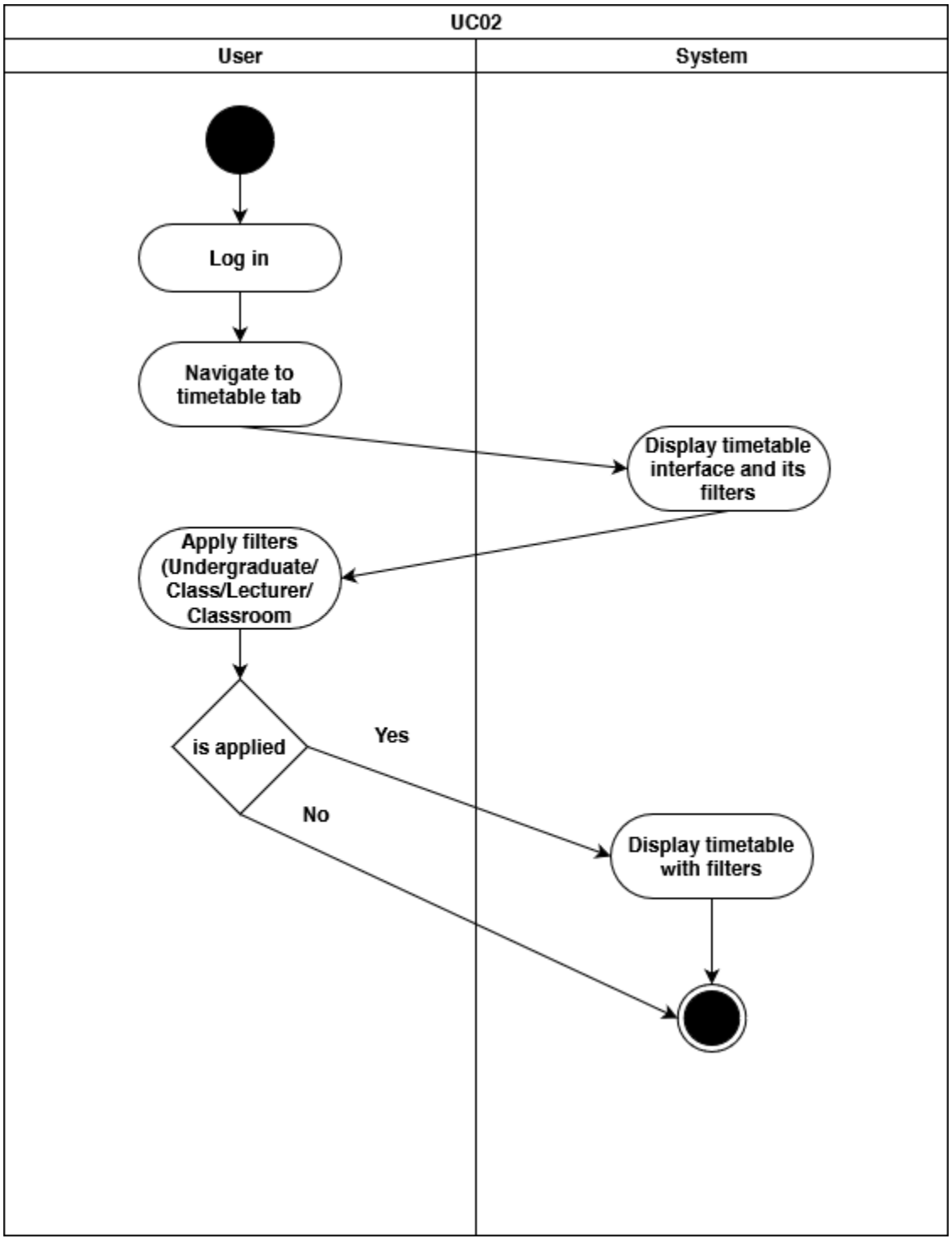

Updating Timetable
Administrative Staff
Course database
Course Info
Timetable Management
Storing the update in database
Timetable database
Access the Timetable
Course Info
Course Info
Professor info
Request
Timetable Report
Request
Students
View Timetable
Academic Staff
View Timetable

Activity Diagrams

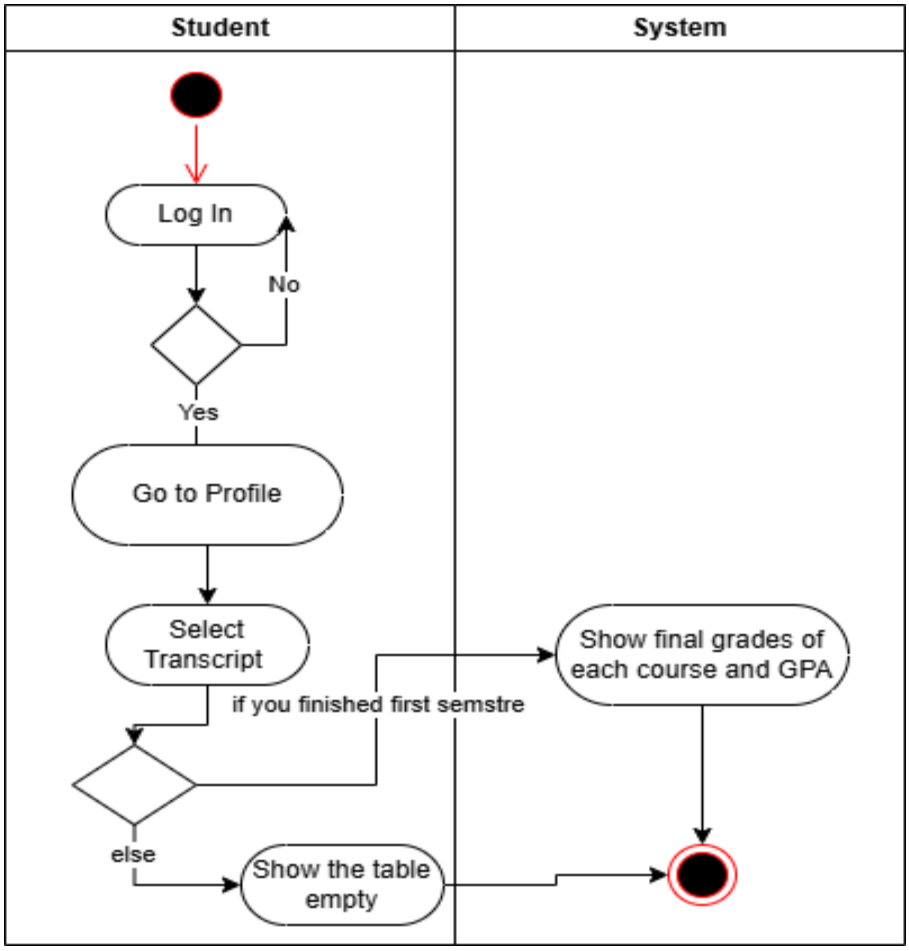
UC01 - Log in



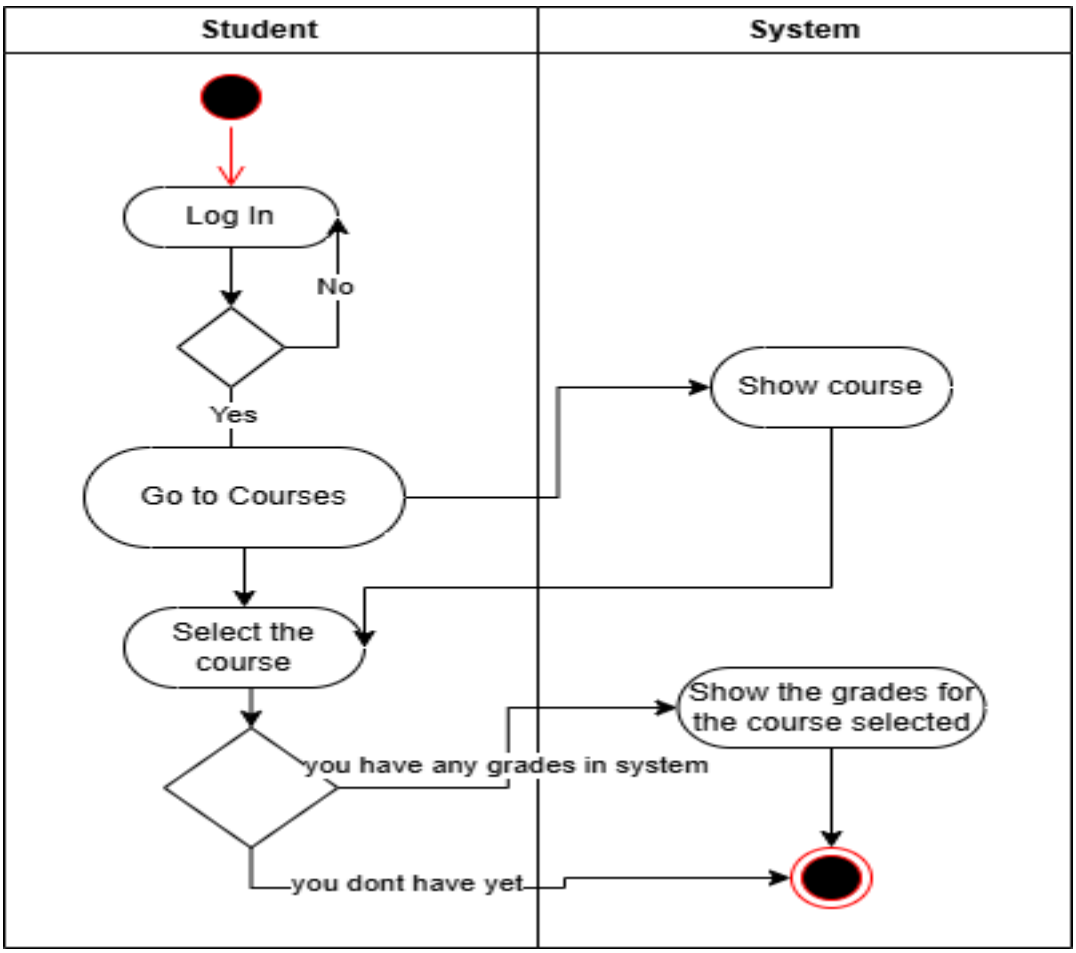
UC02 -View timetable



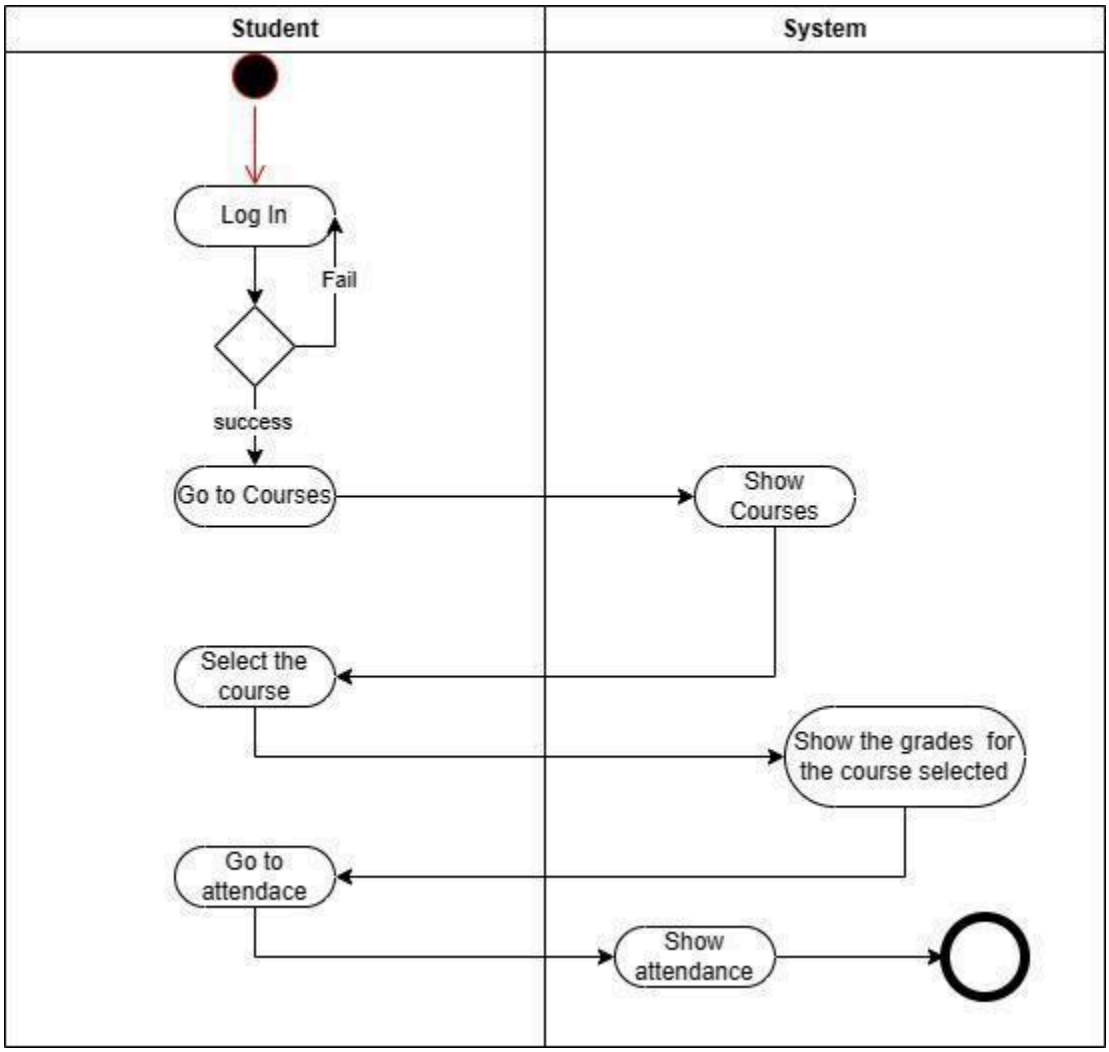
UC03- View Transcript



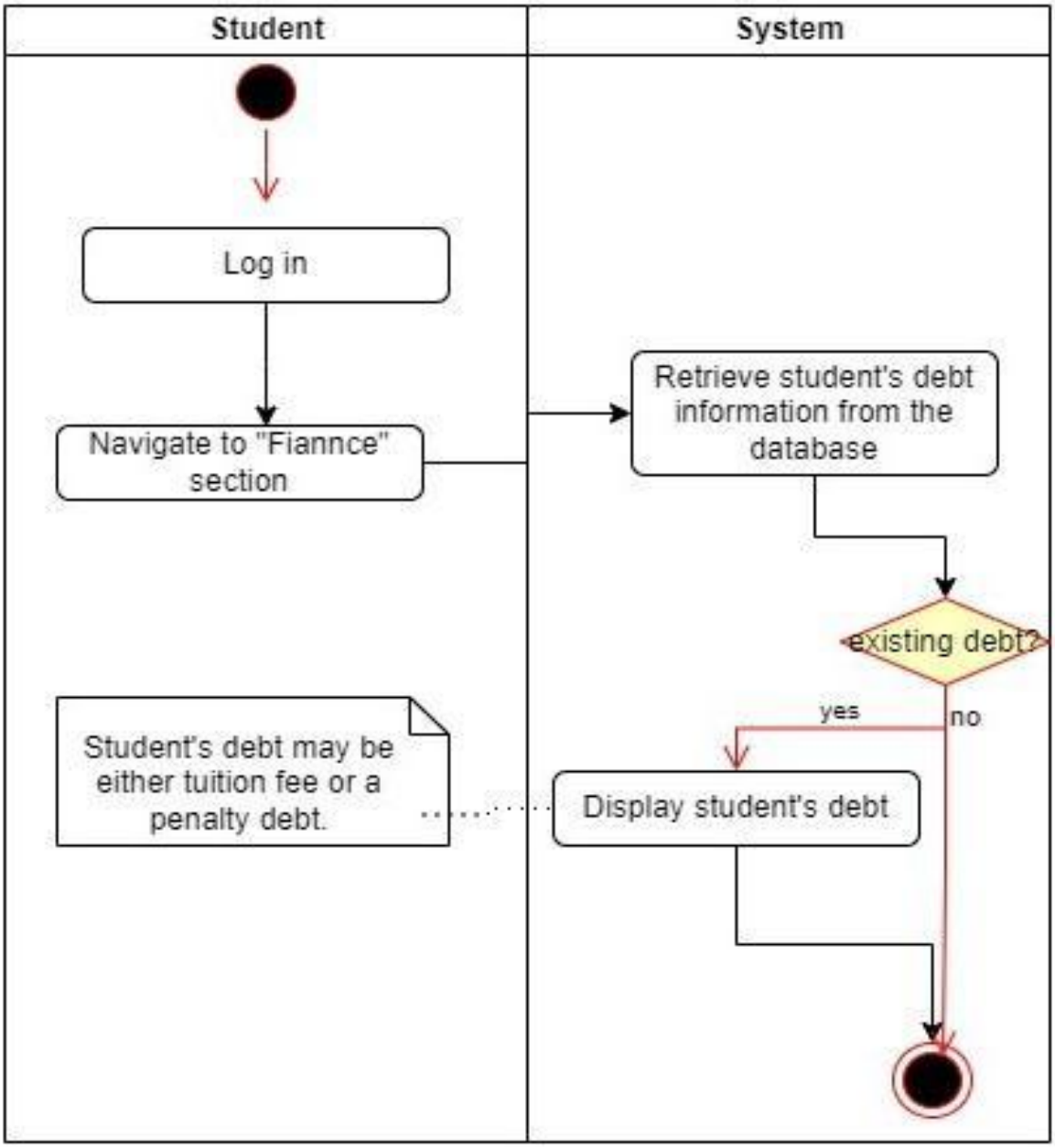
UC04- View grades



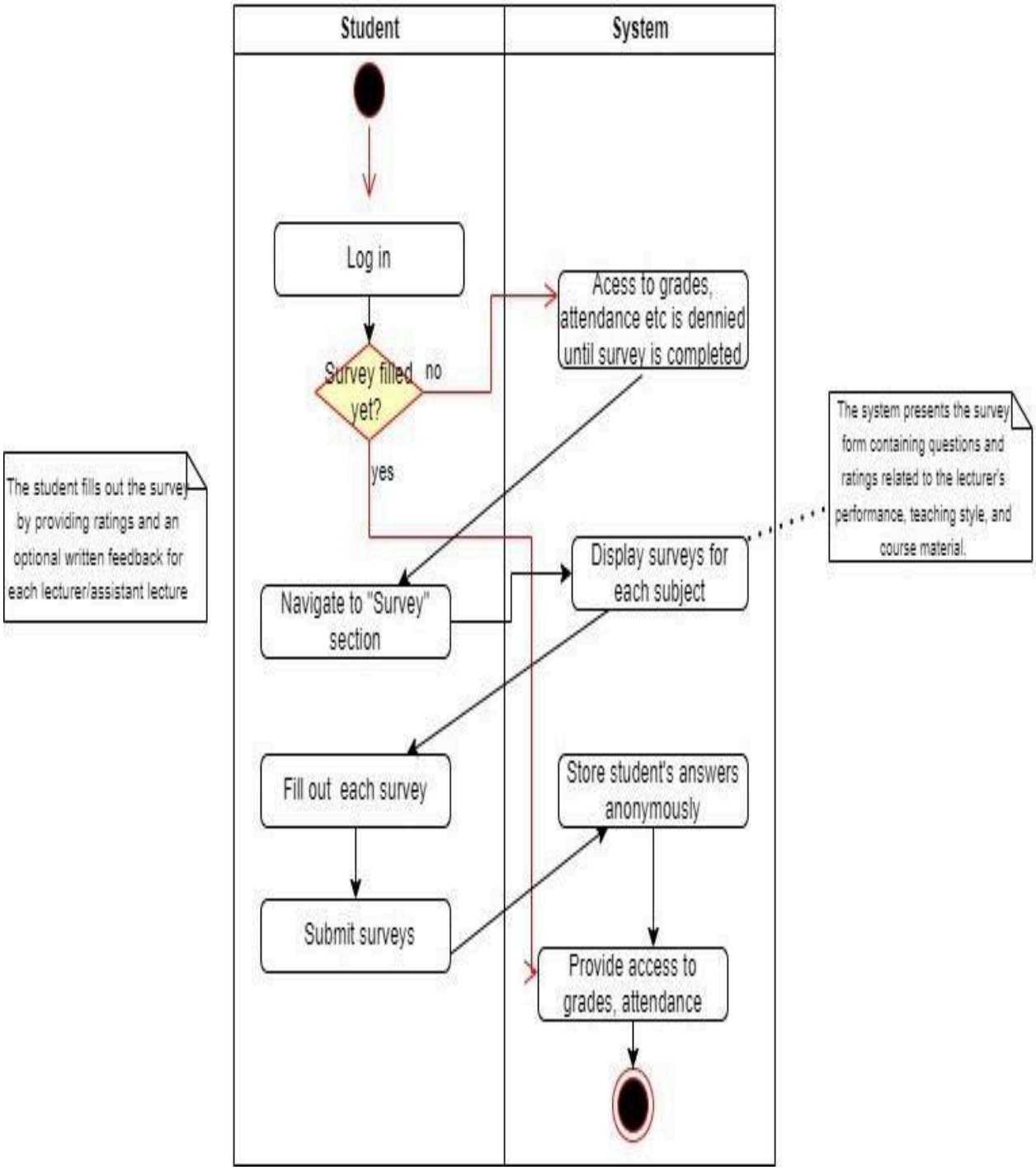
UC05- View Attendance



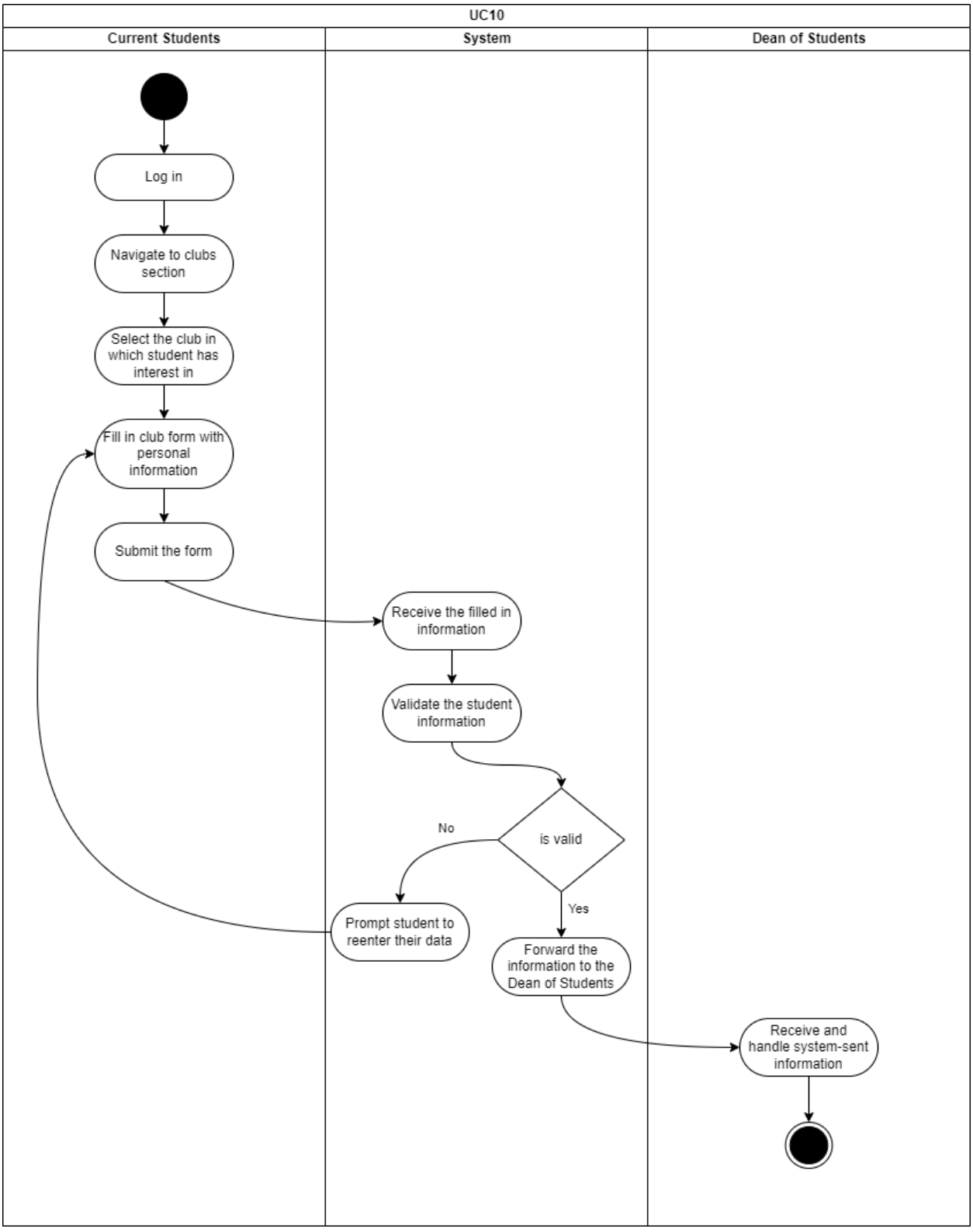
UC08: View Debt



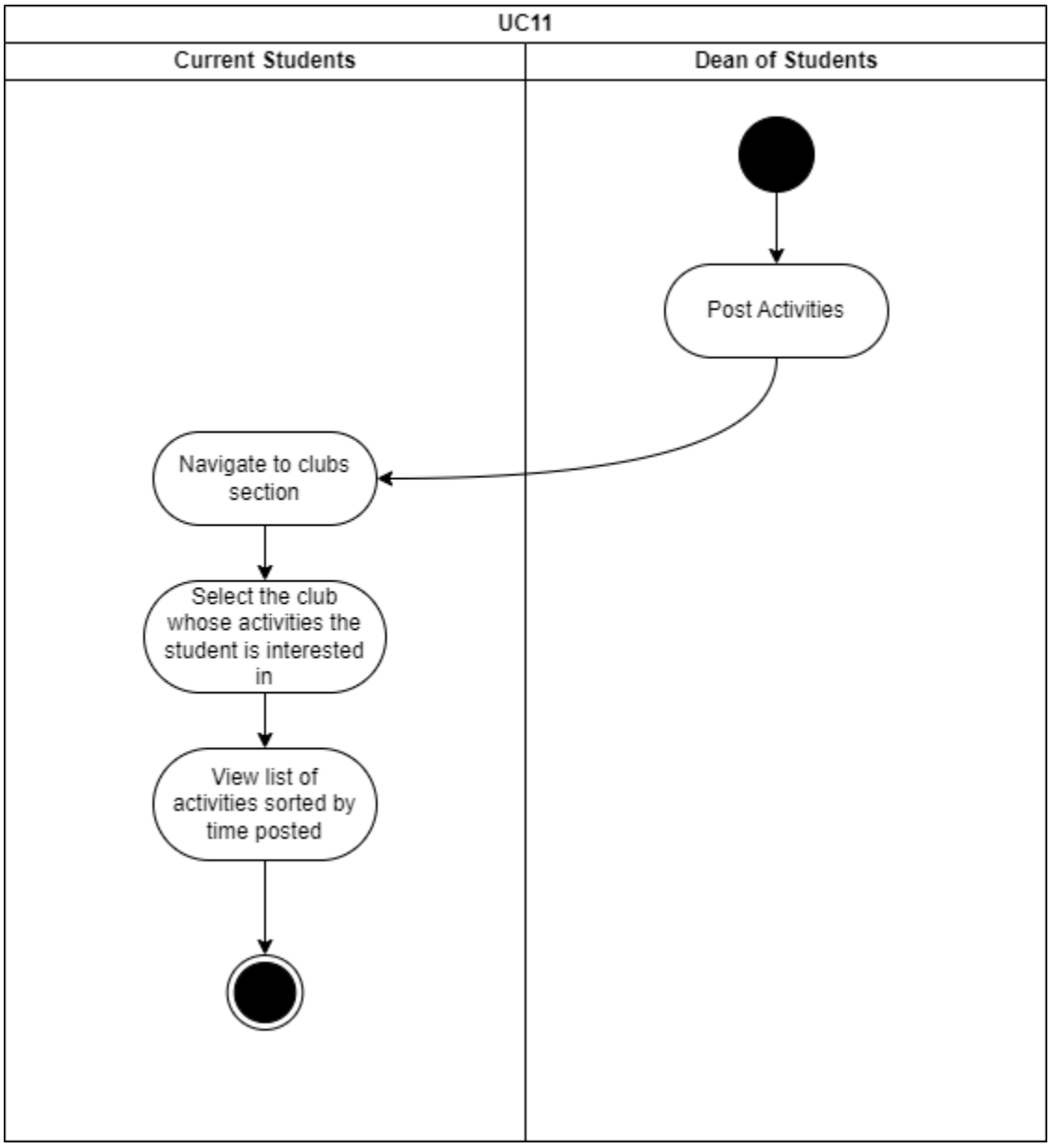
UC09: Fill out Survey



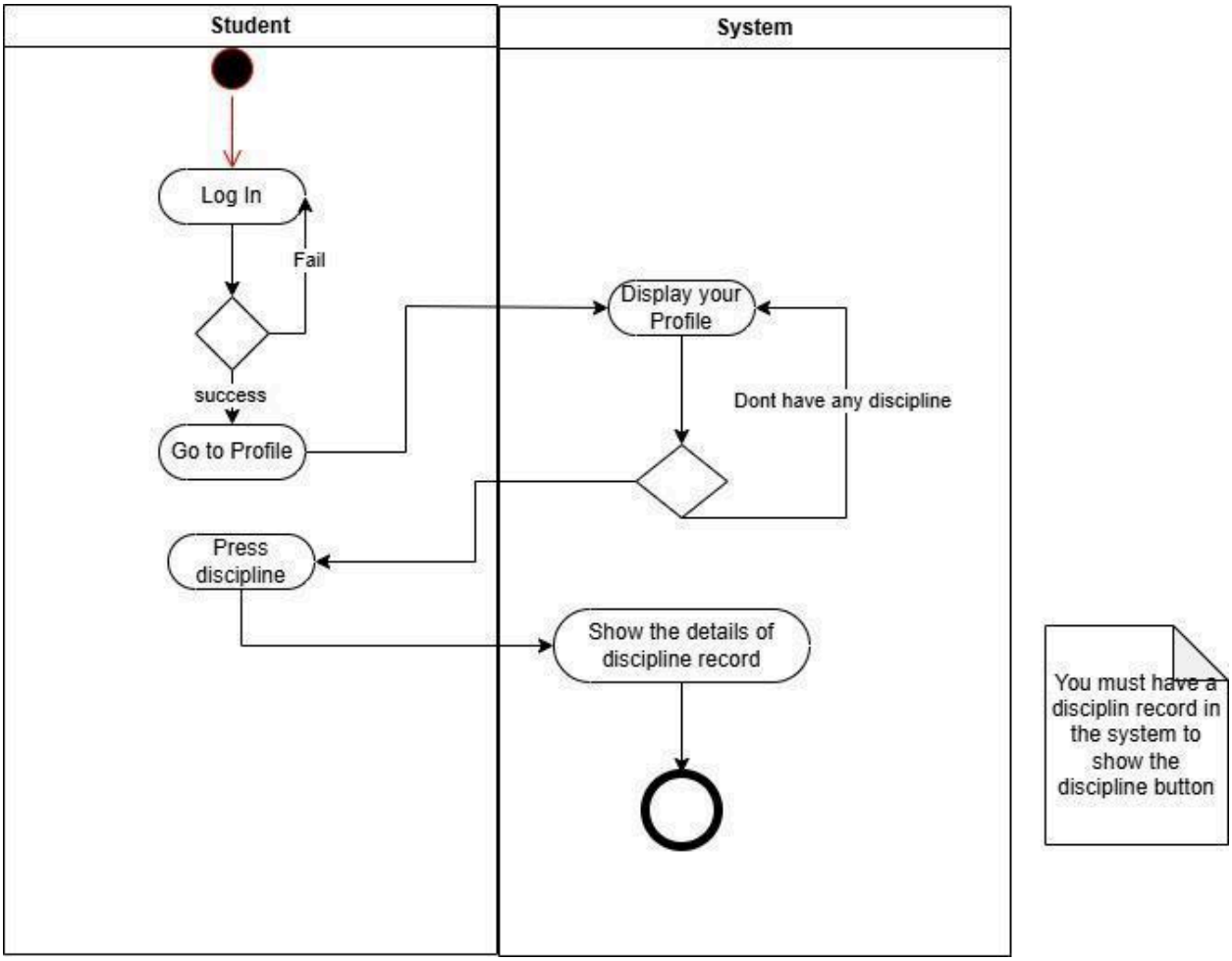
UC10 -Apply to Student Clubs



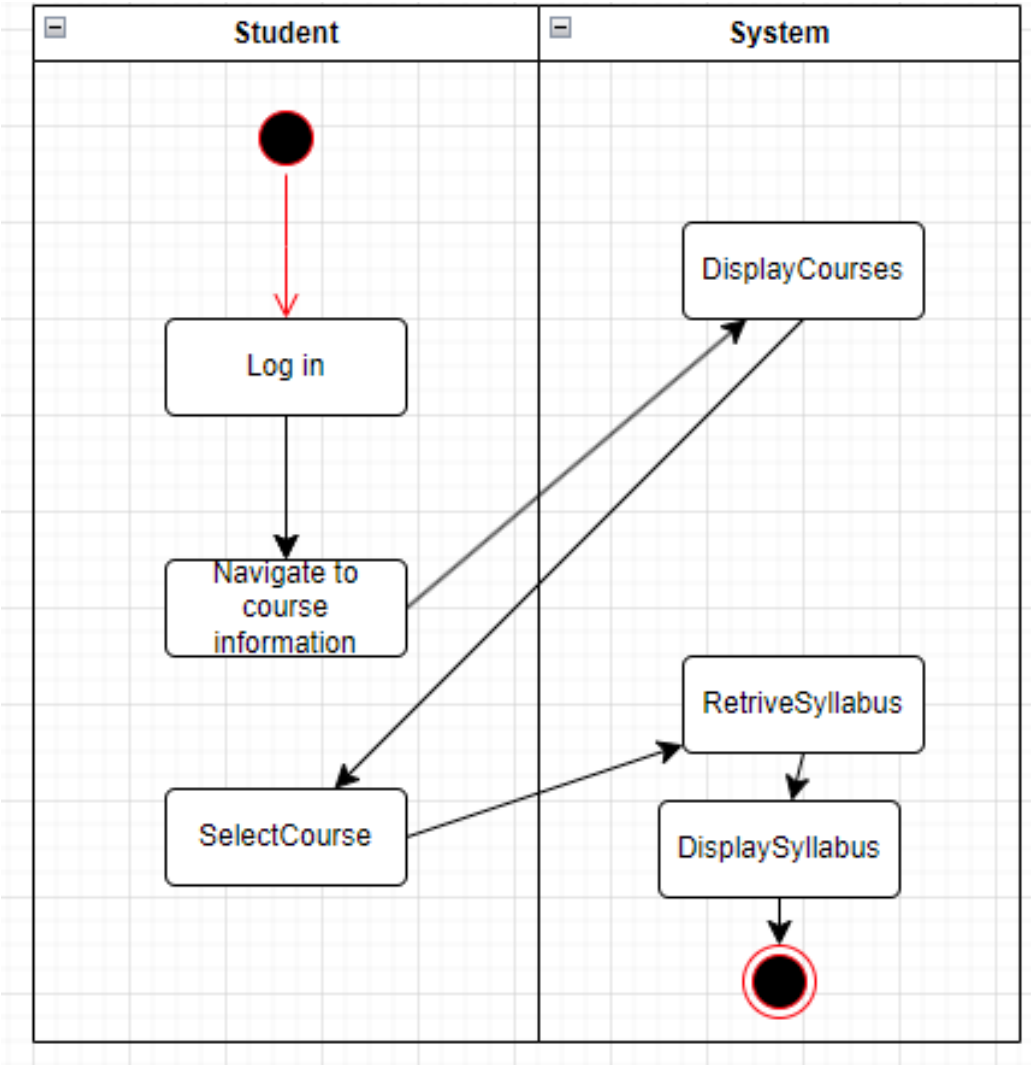
UC11-View Club Activities



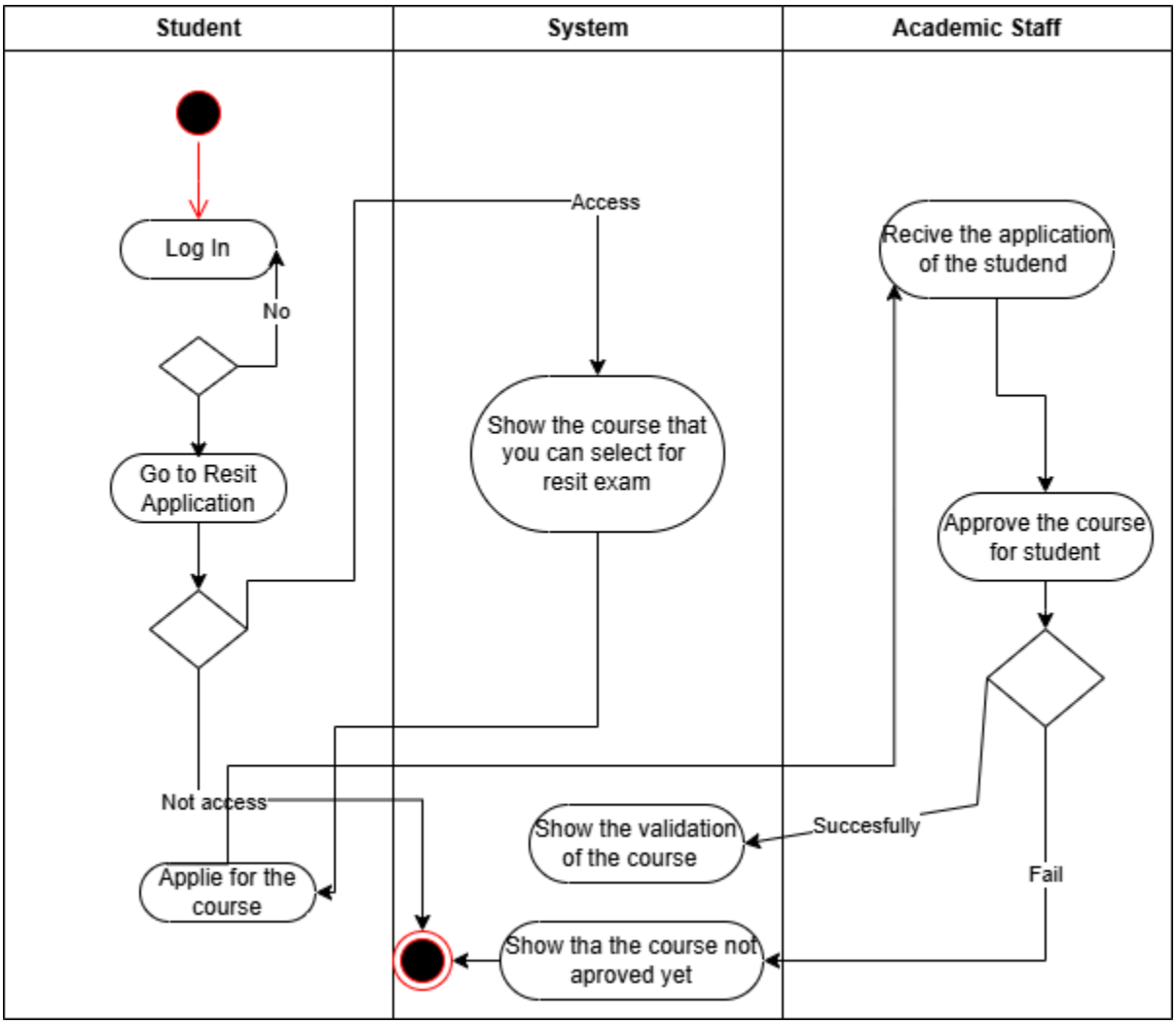
UC12-View Disciplinary/Misconduct records



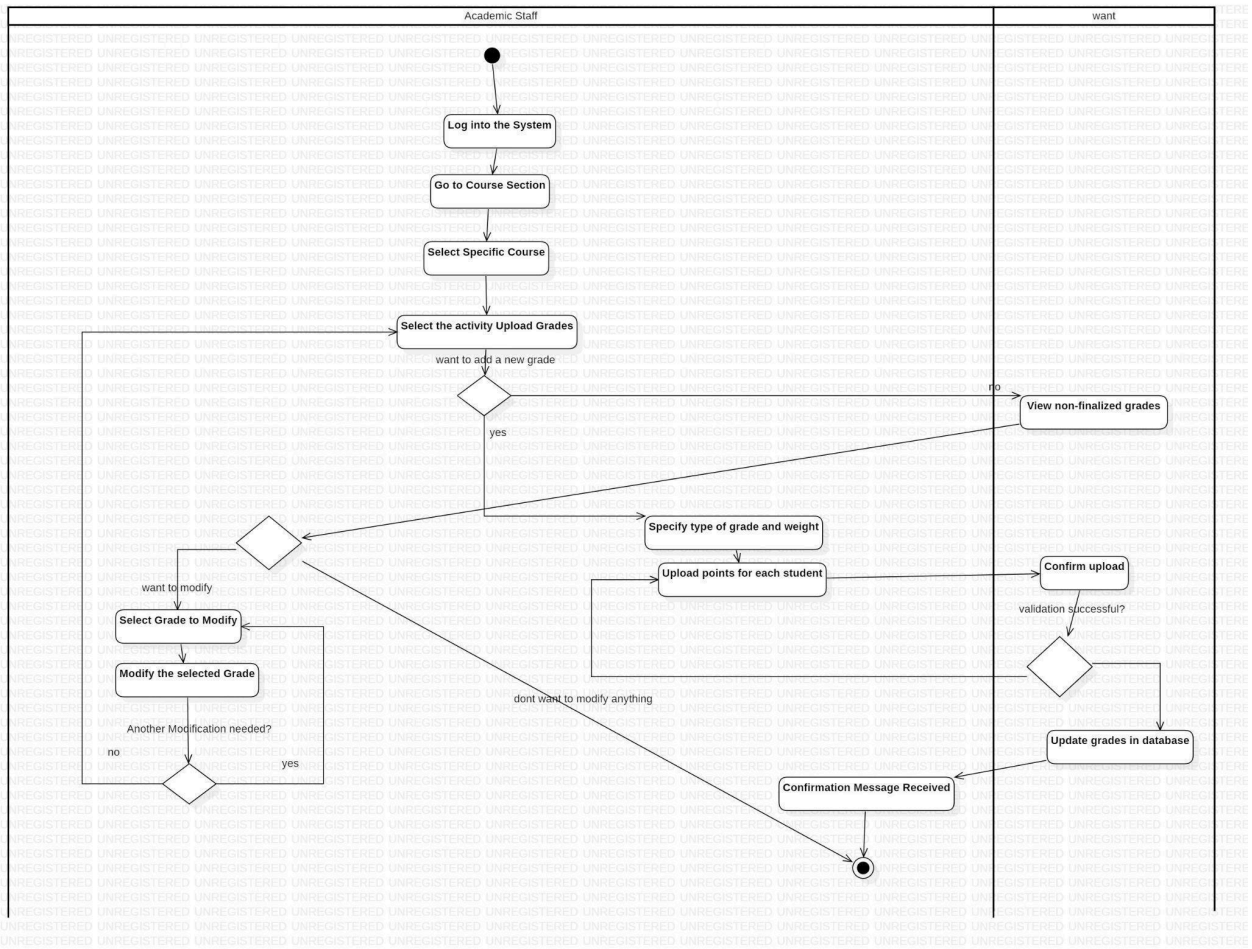
UC13- View Course Syllabus



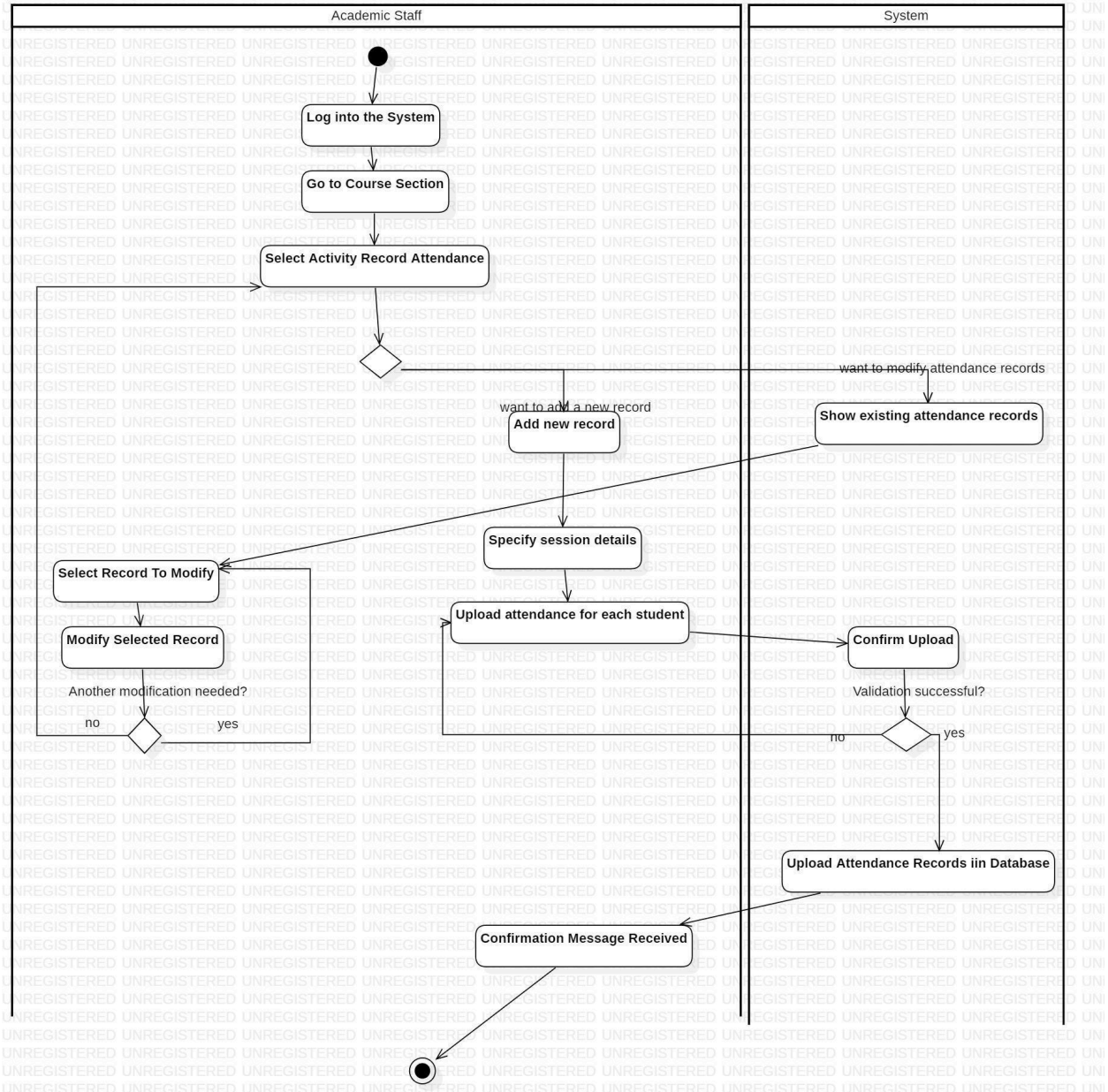
UC14 : **Apply for Resit Exam**



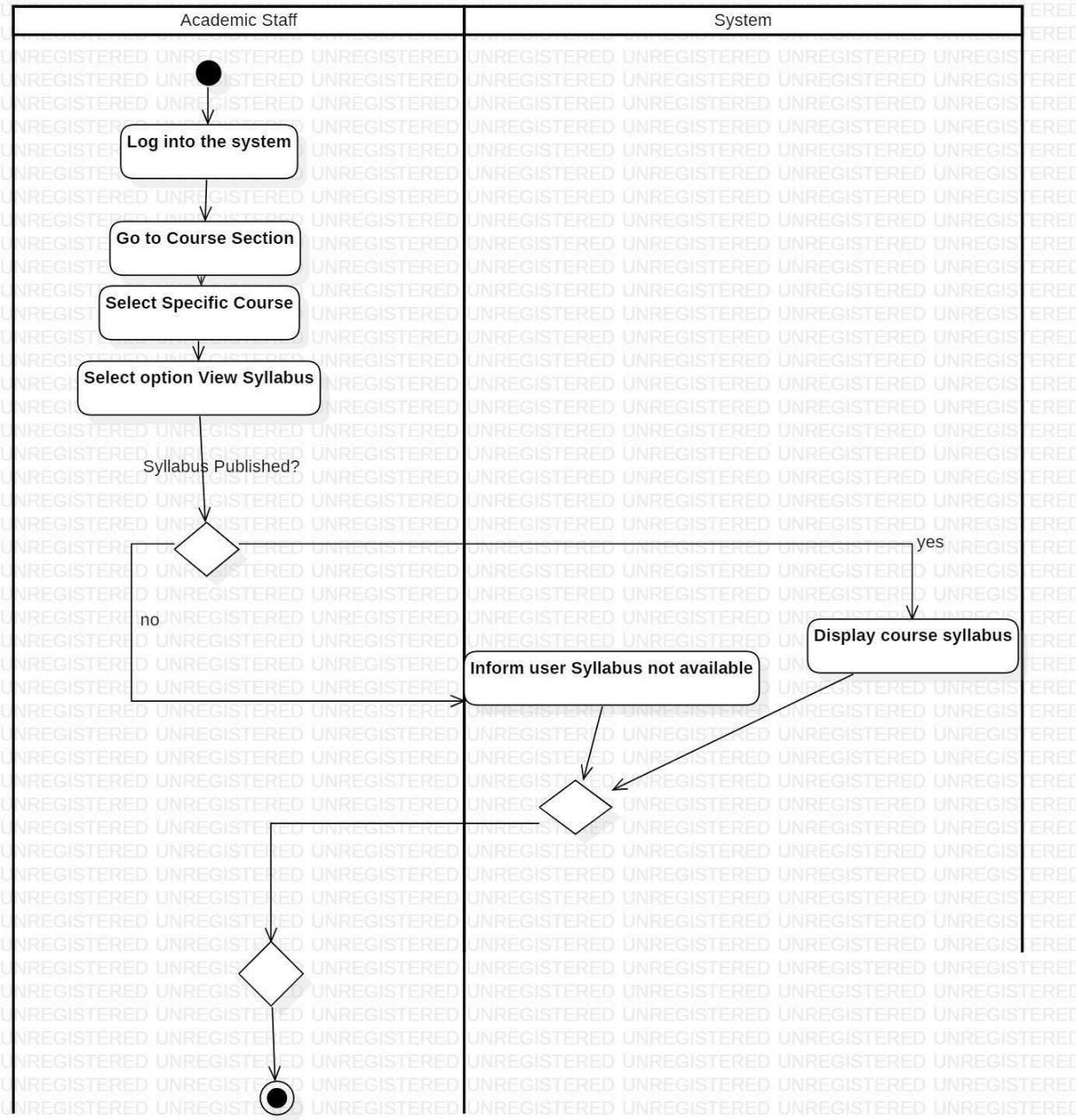
UC15- **Load grades into the system**


Academic Staff
want
Log into the System
Go to Course Section
Select Specific Course
Select the activity Upload Grades
want to add a new grade
yes
no
View non-finalized grades
Specify type of grade and weight
Upload points for each student
Confirm upload
validation successful?
want to modify
Select Grade to Modify
Modify the selected Grade
Another Modification needed?
no
yes
dont want to modify anything
Update grades in database
Confirmation Message Received

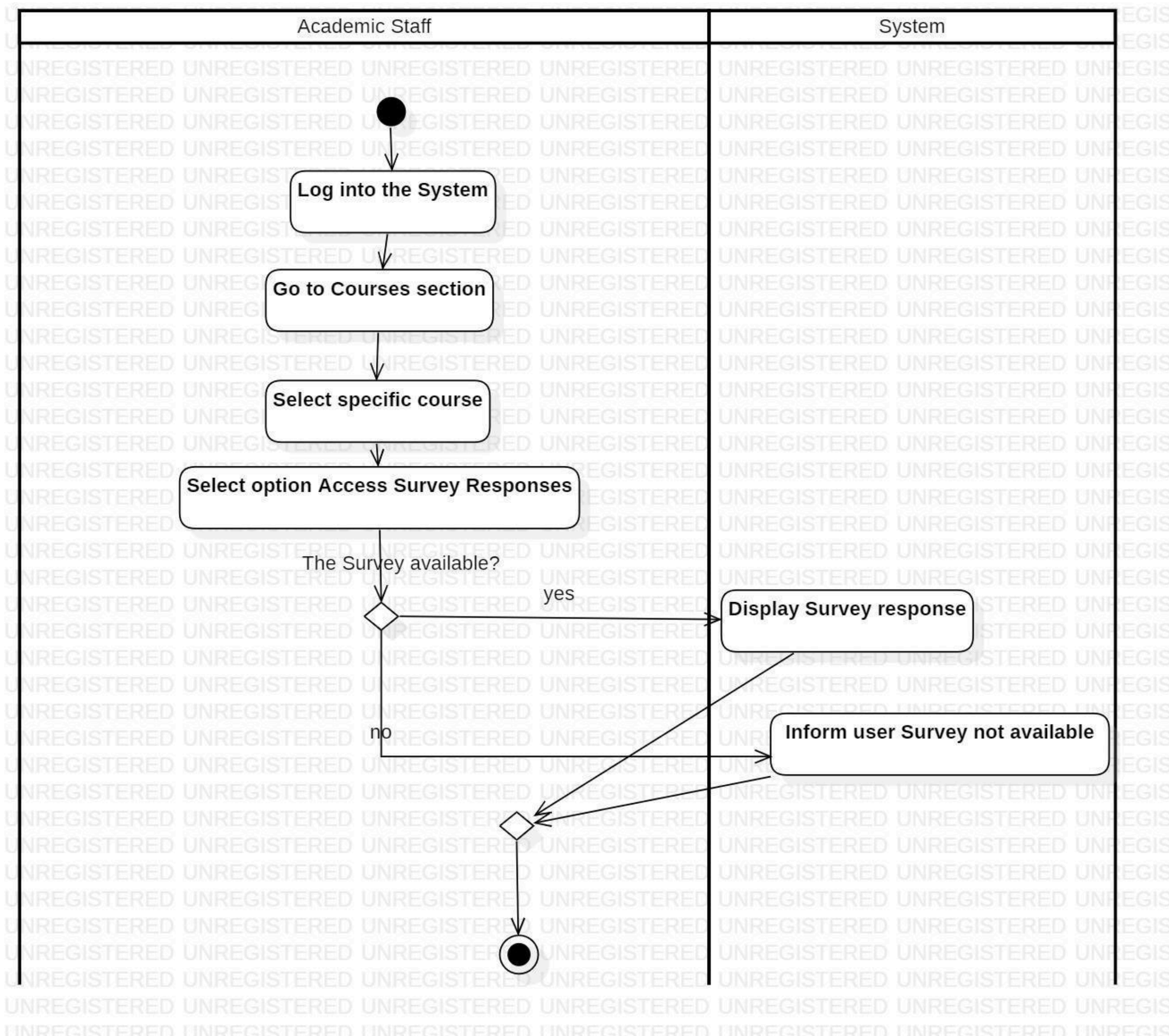
UC16- *Record Attendance of Students*



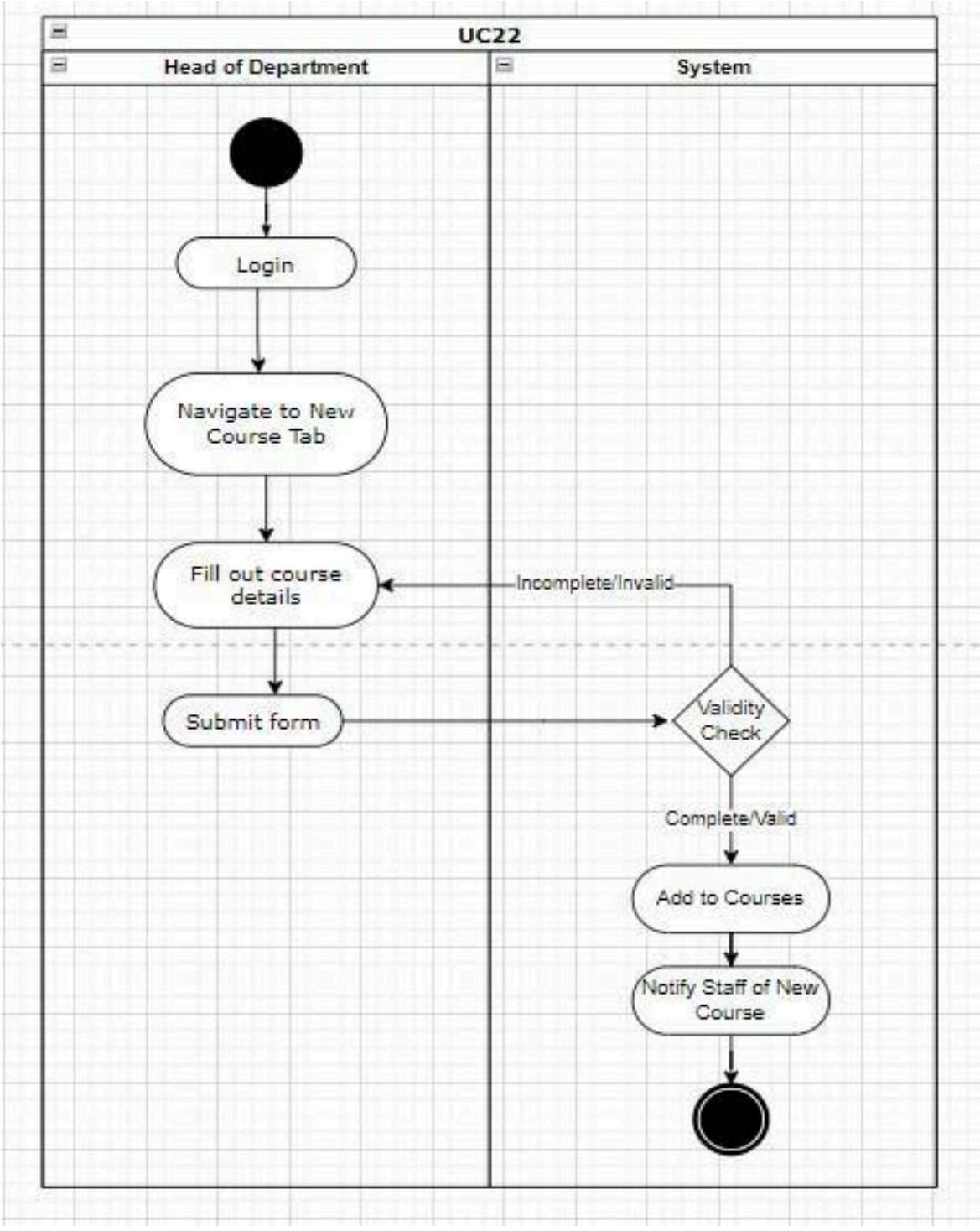
UC17- View Course Syllabus



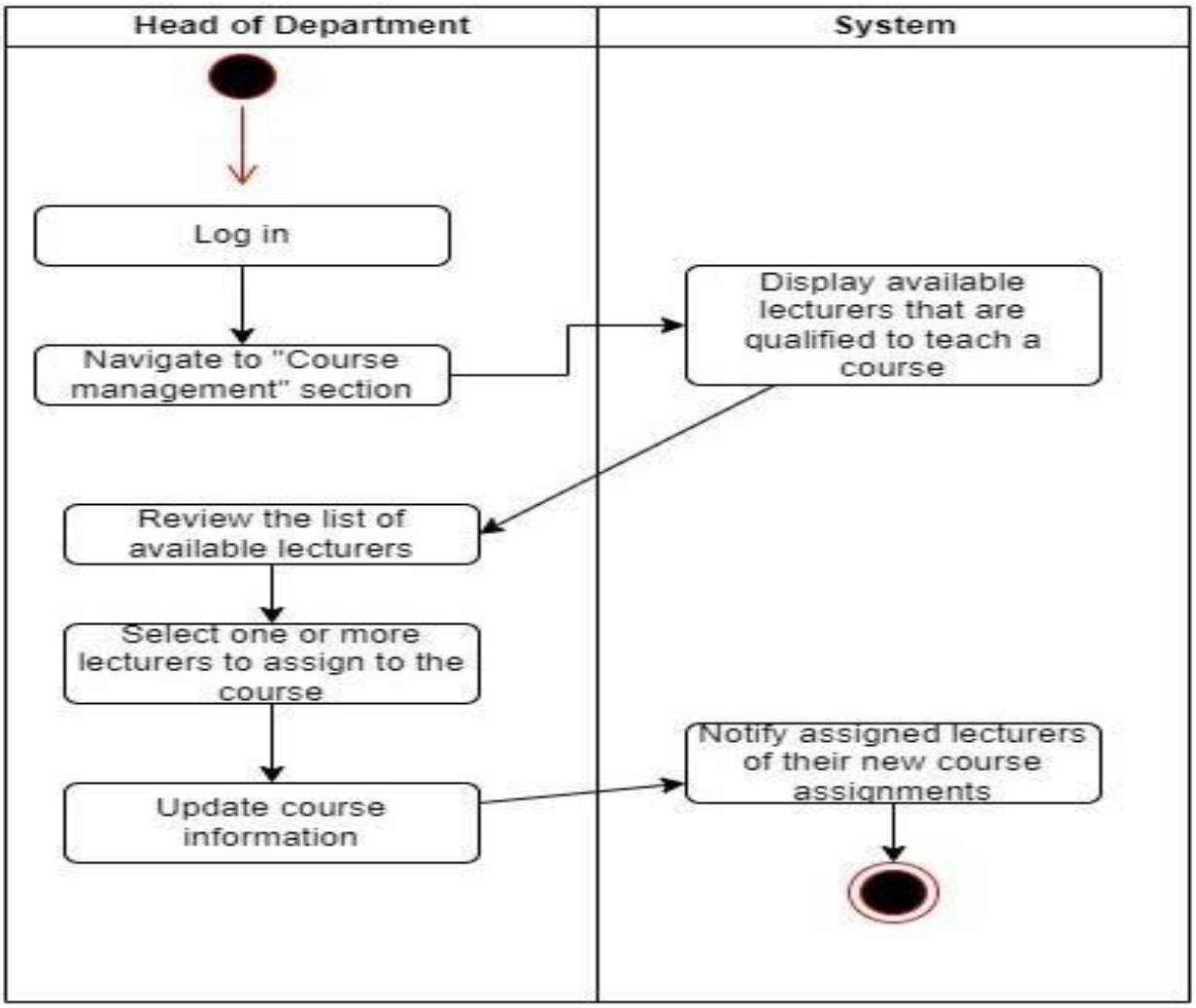
UC18-Access Survey Responses

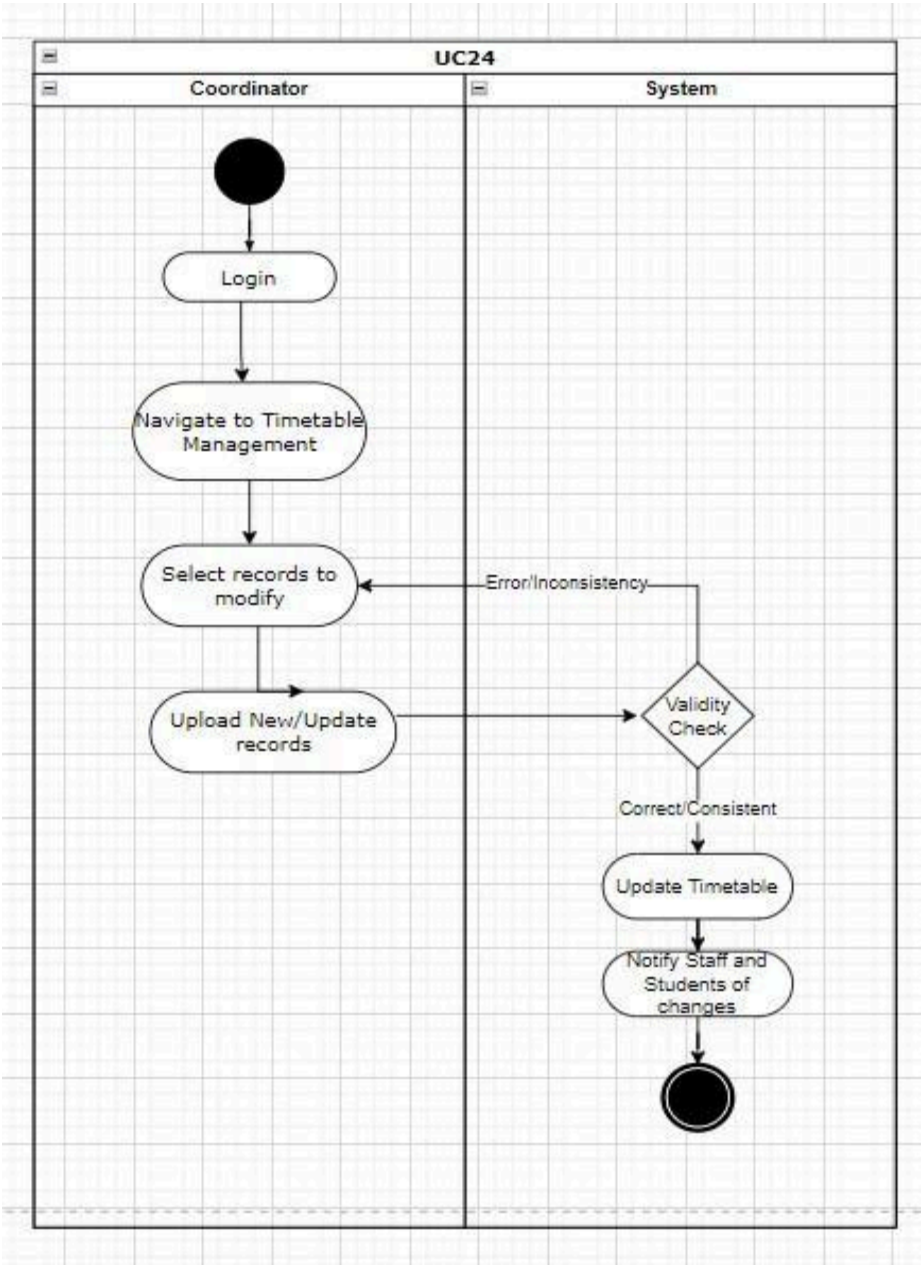


UC22- Create new Course



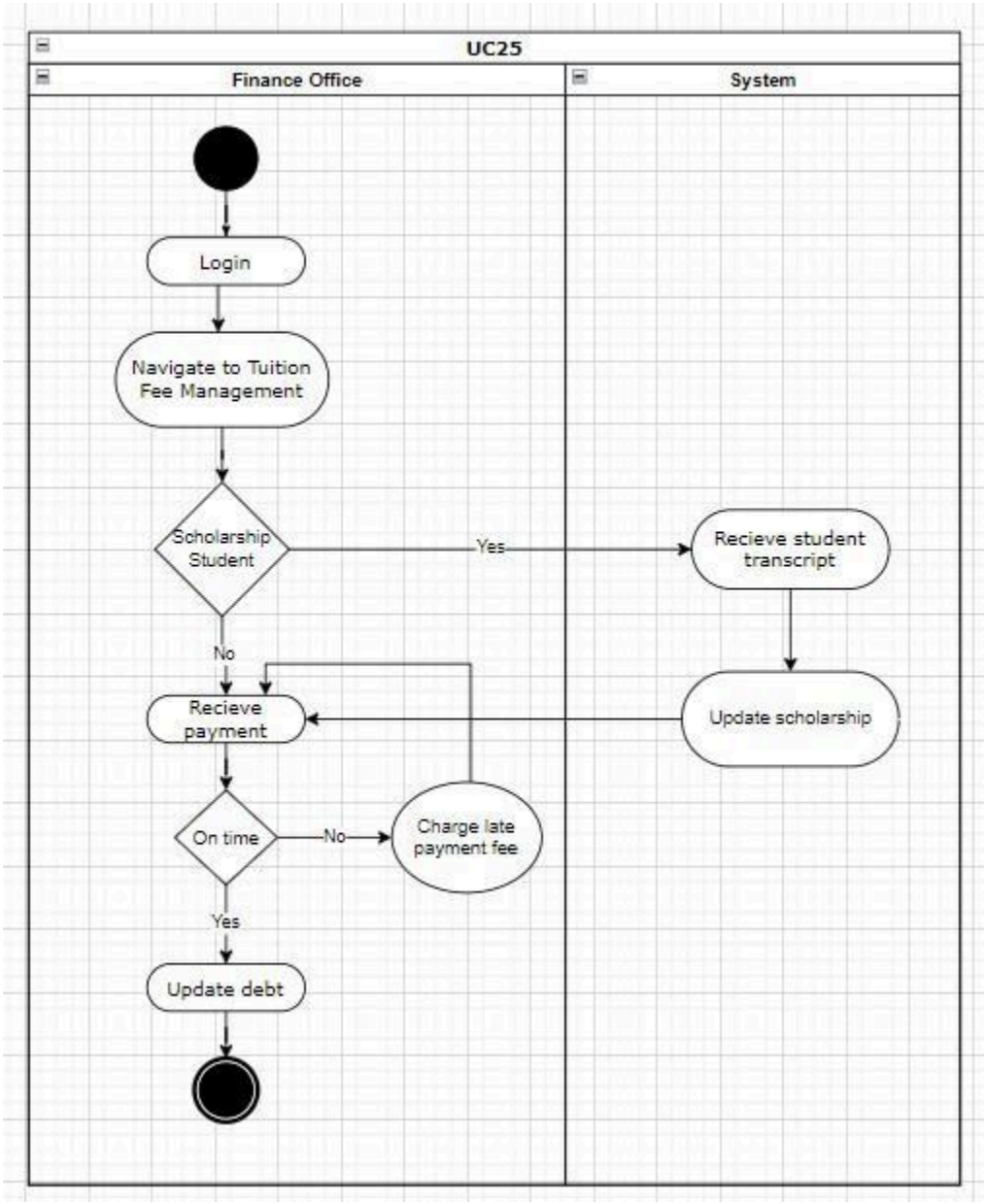
UC23- Assigning lecturers to courses



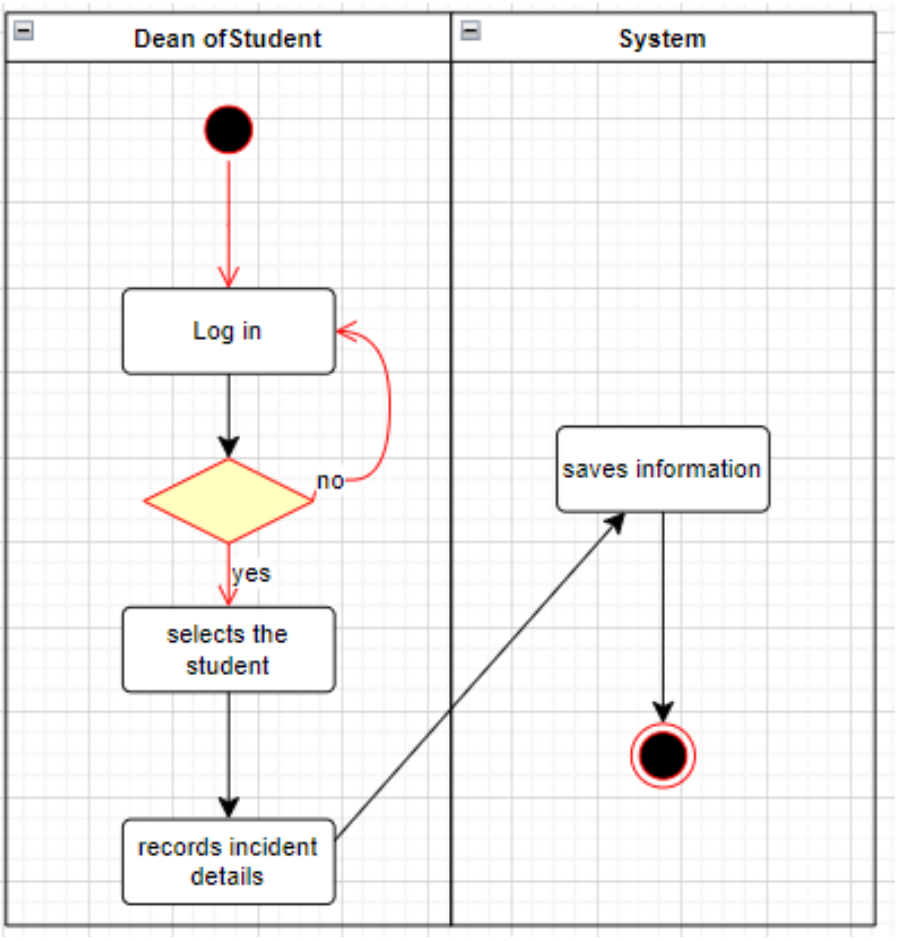


UC24- **Upload Timetable**

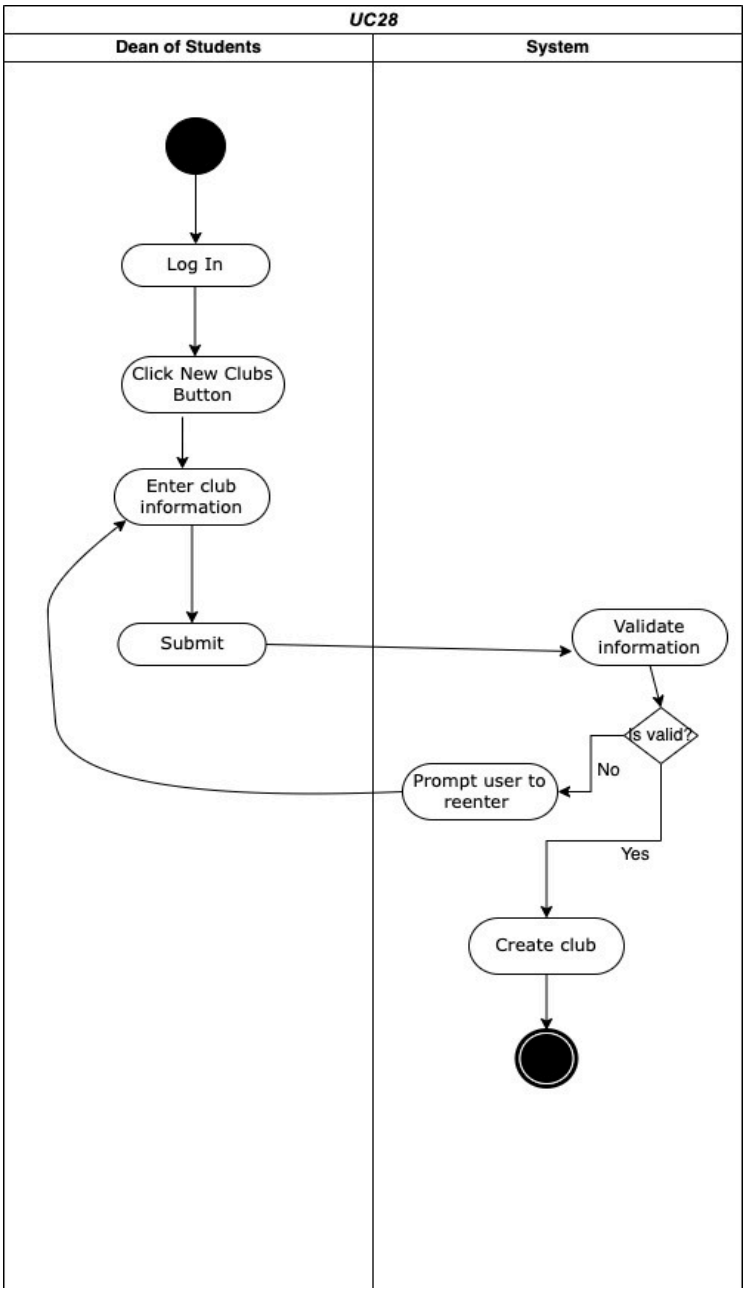
UC25-**Manage Tuition fees**



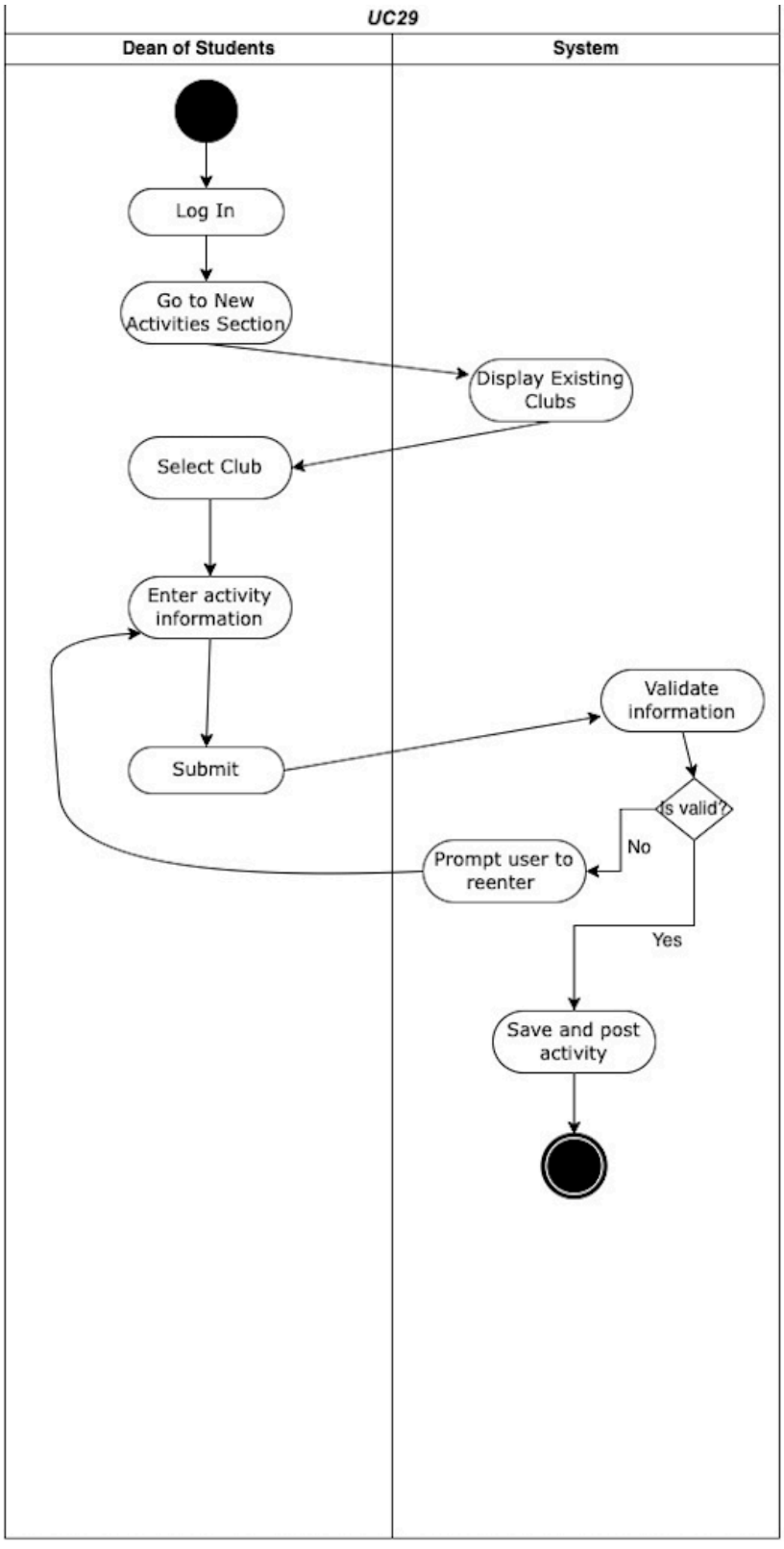
UC27- **Record disciplinary cases**


Dean ofStudent
System
Log in
no
yes
selects the student
records incident details
saves information

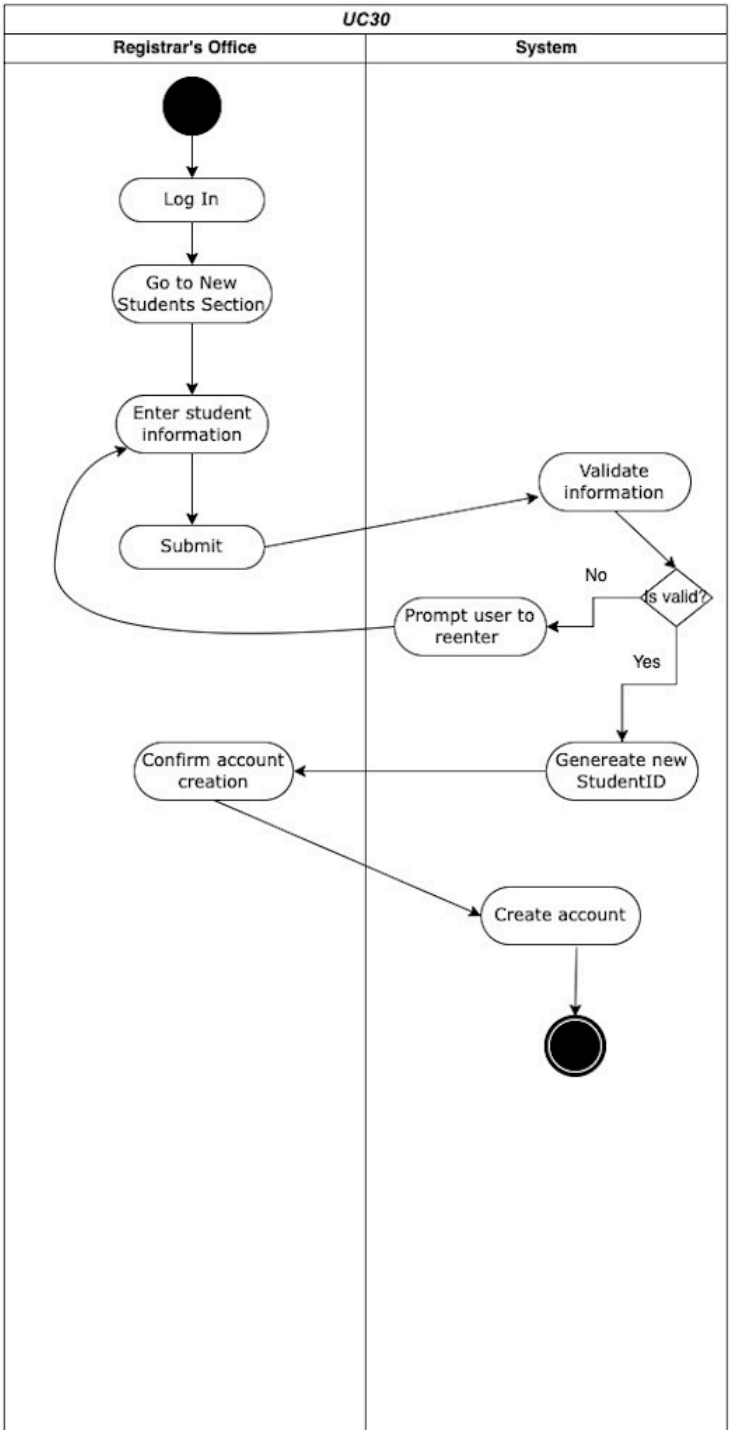
UC28 - Create Student Clubs



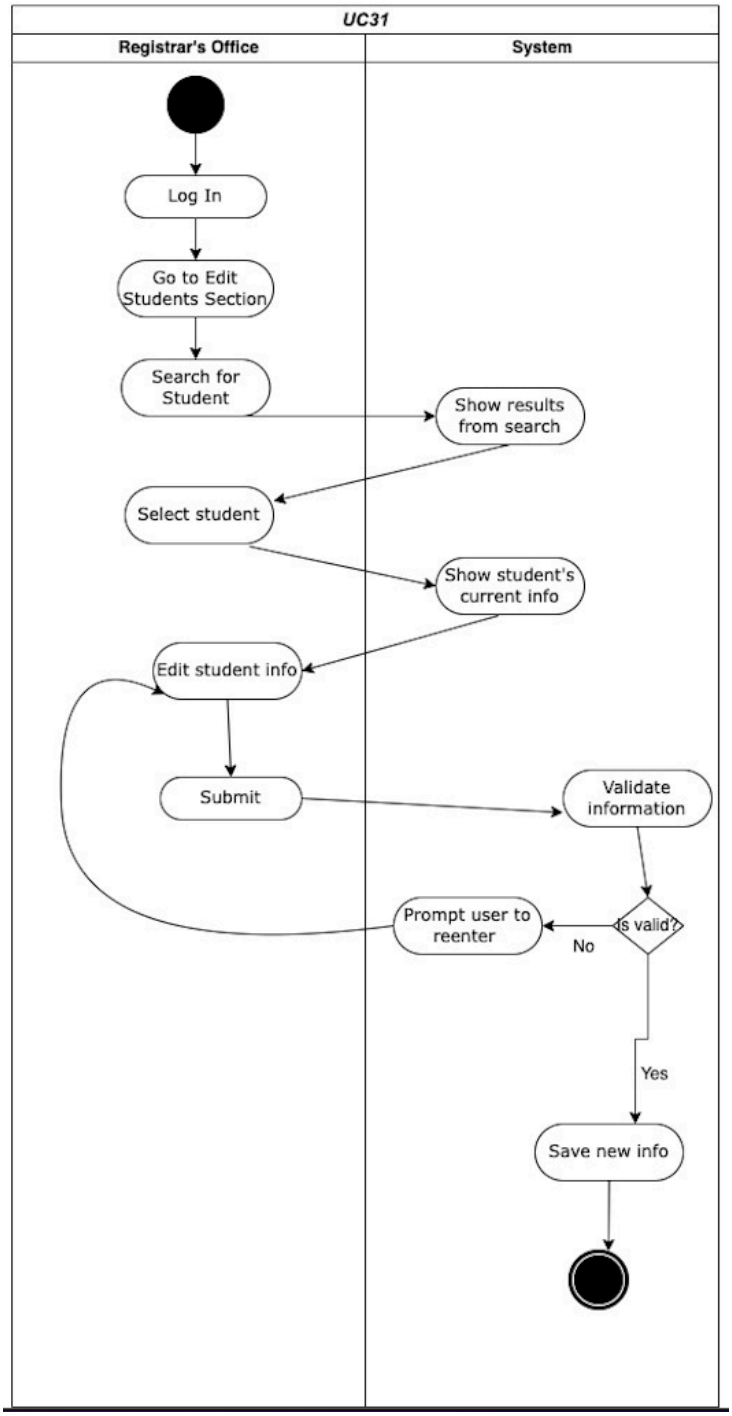
UC29 - Post Student Club Activities



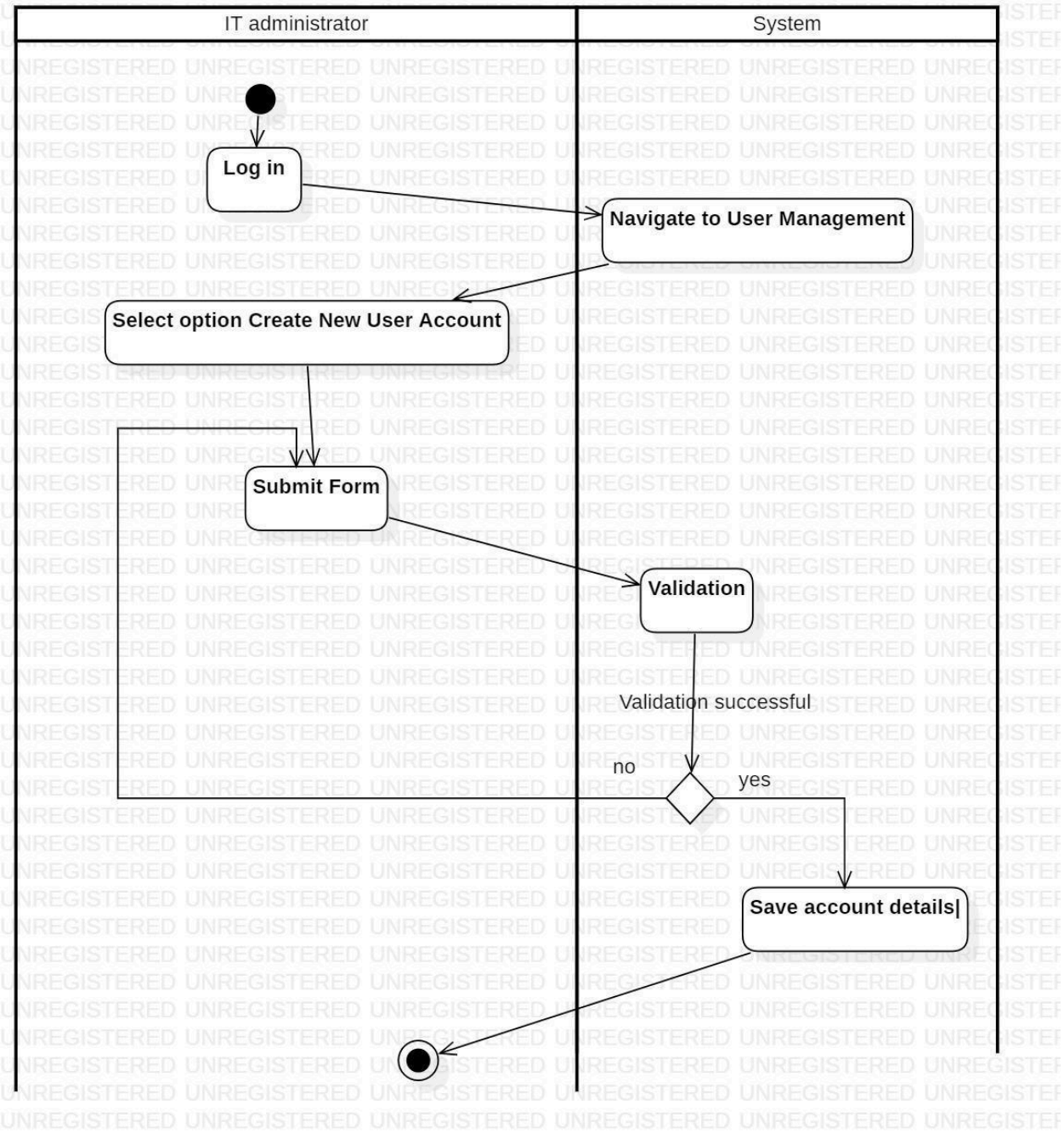
UC30 - Create new Student Account



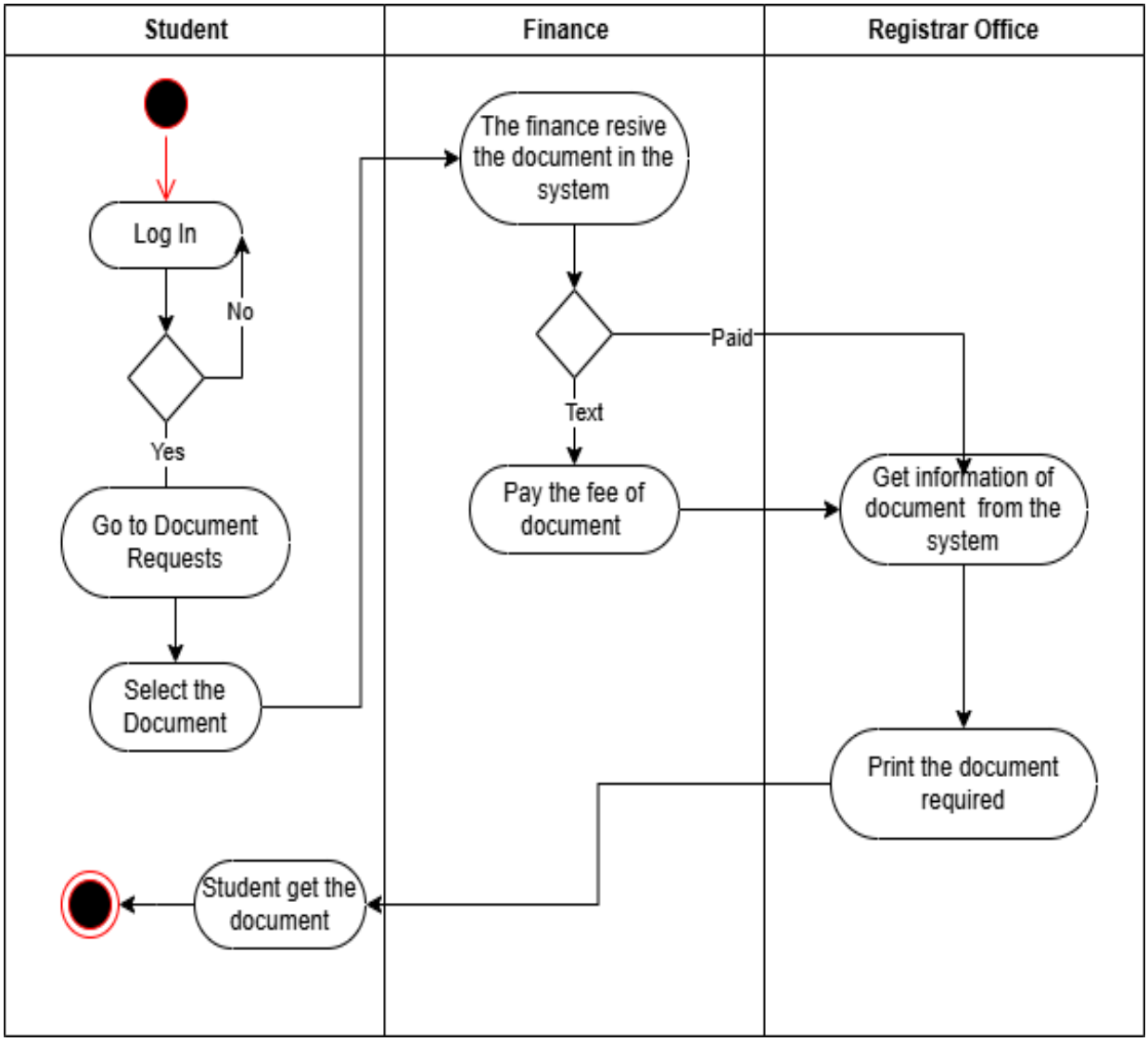
UC31 - Update/Edit student information



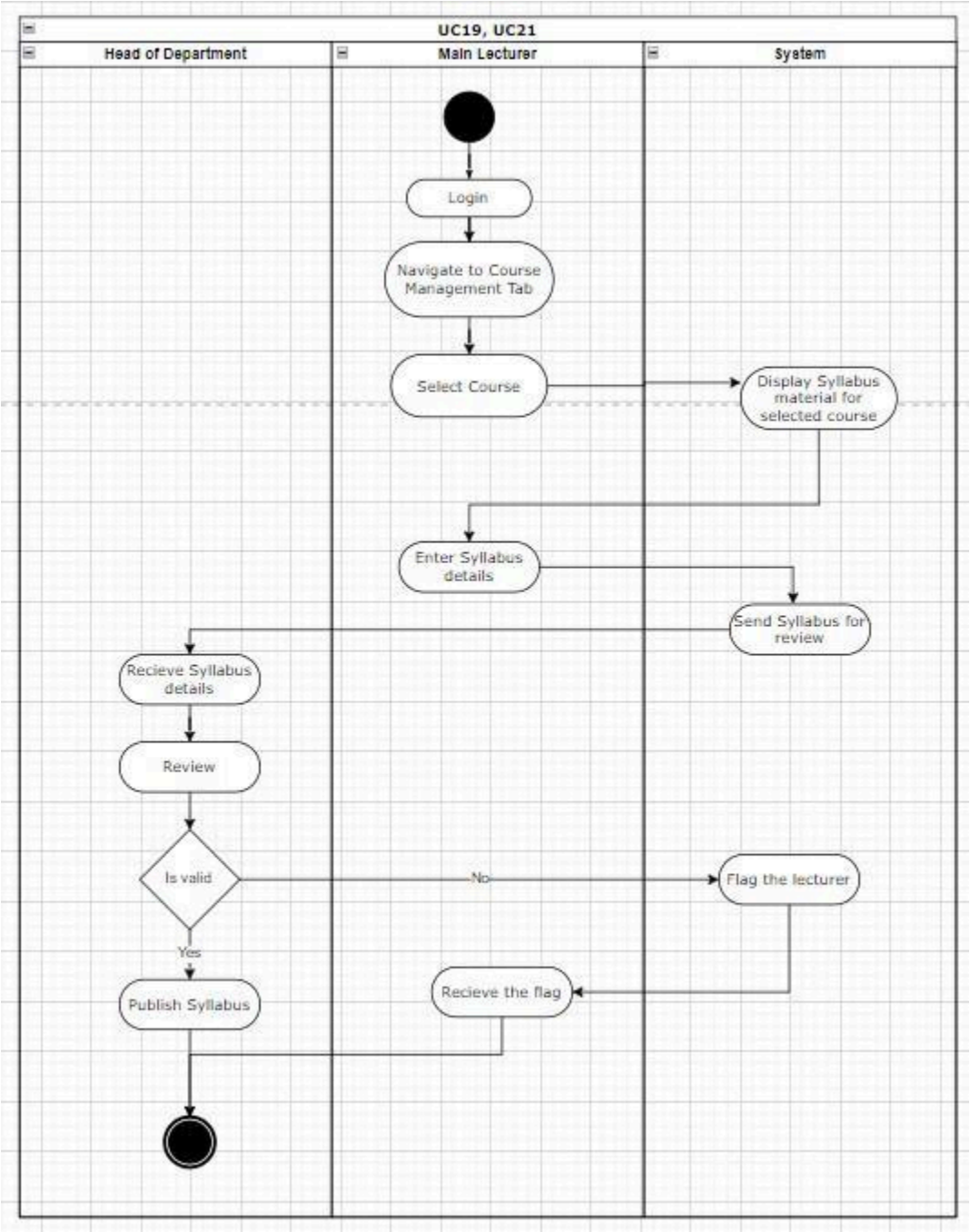
UC33- Create Academic Staff Account



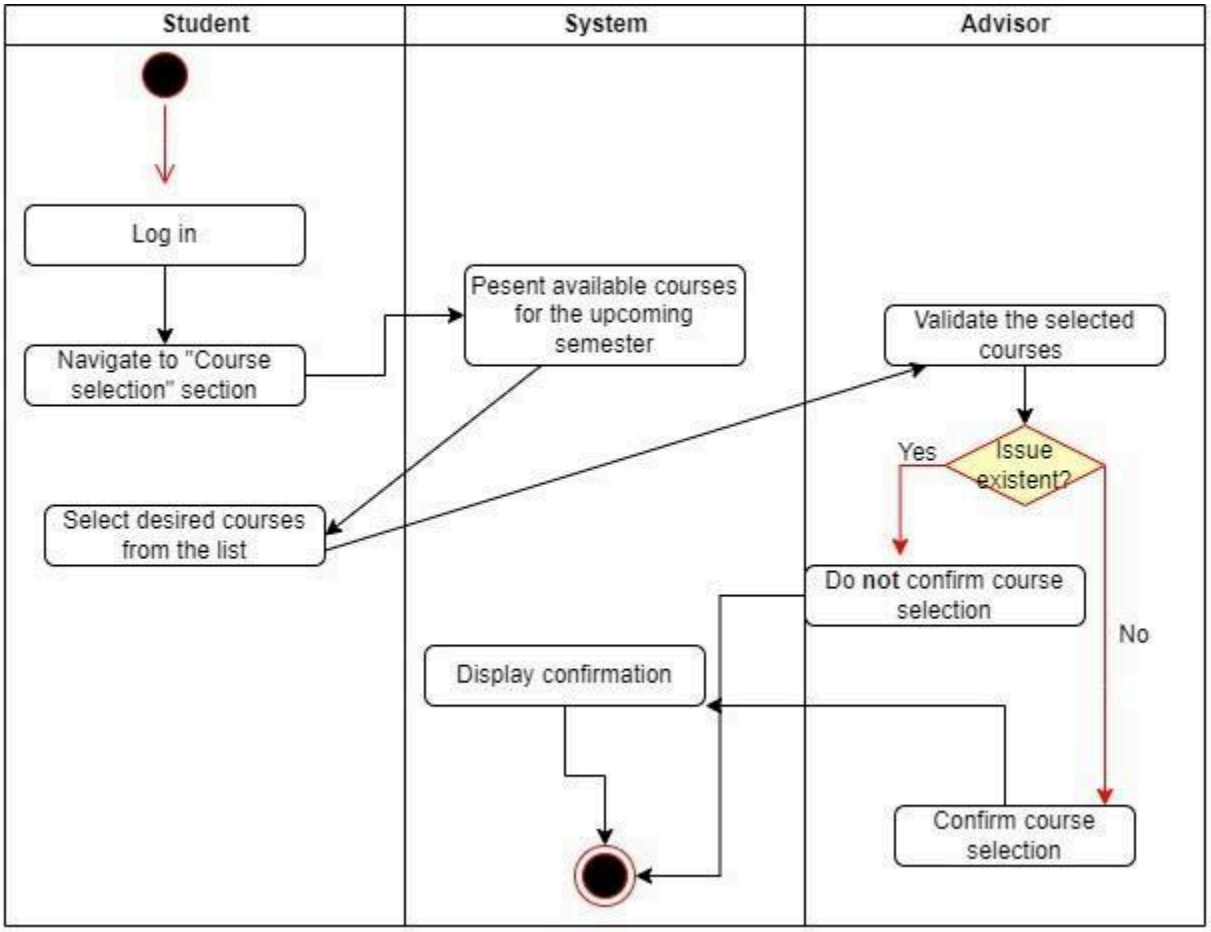
UC06, UC26, UC32 - Document Request



UC19, UC21- Syllabus Publishing

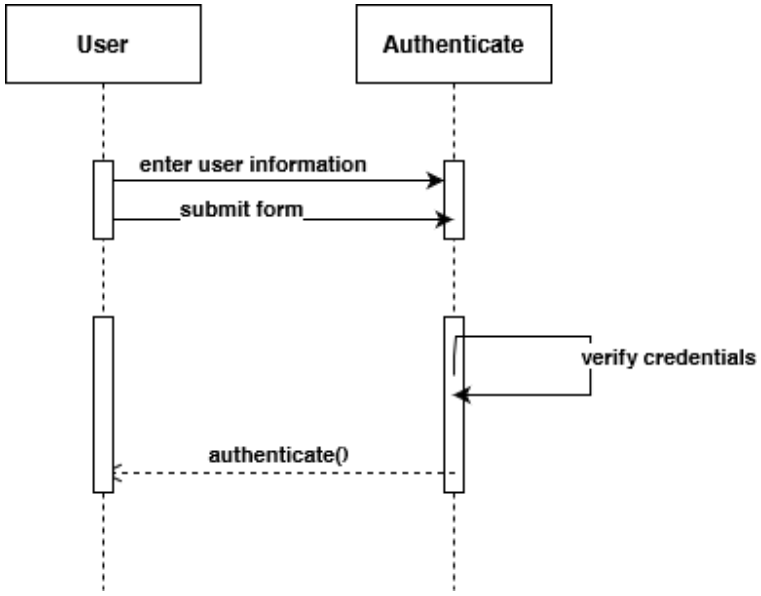


UC07, UC20: Course Selection

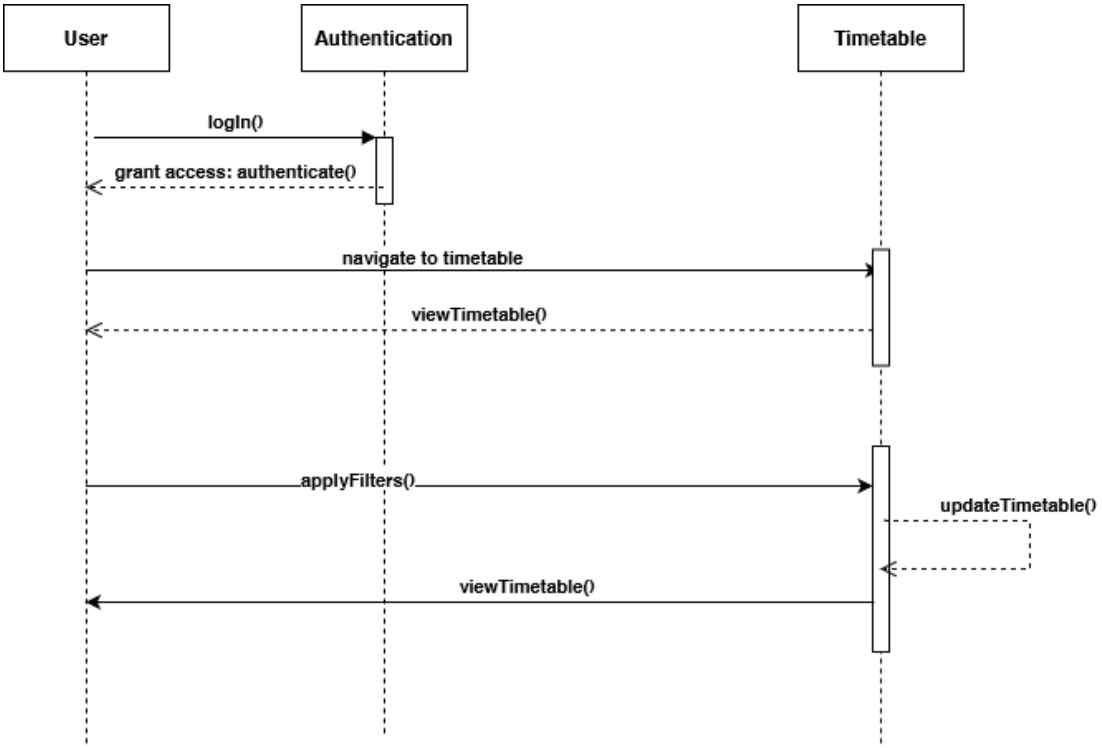


Sequence diagrams

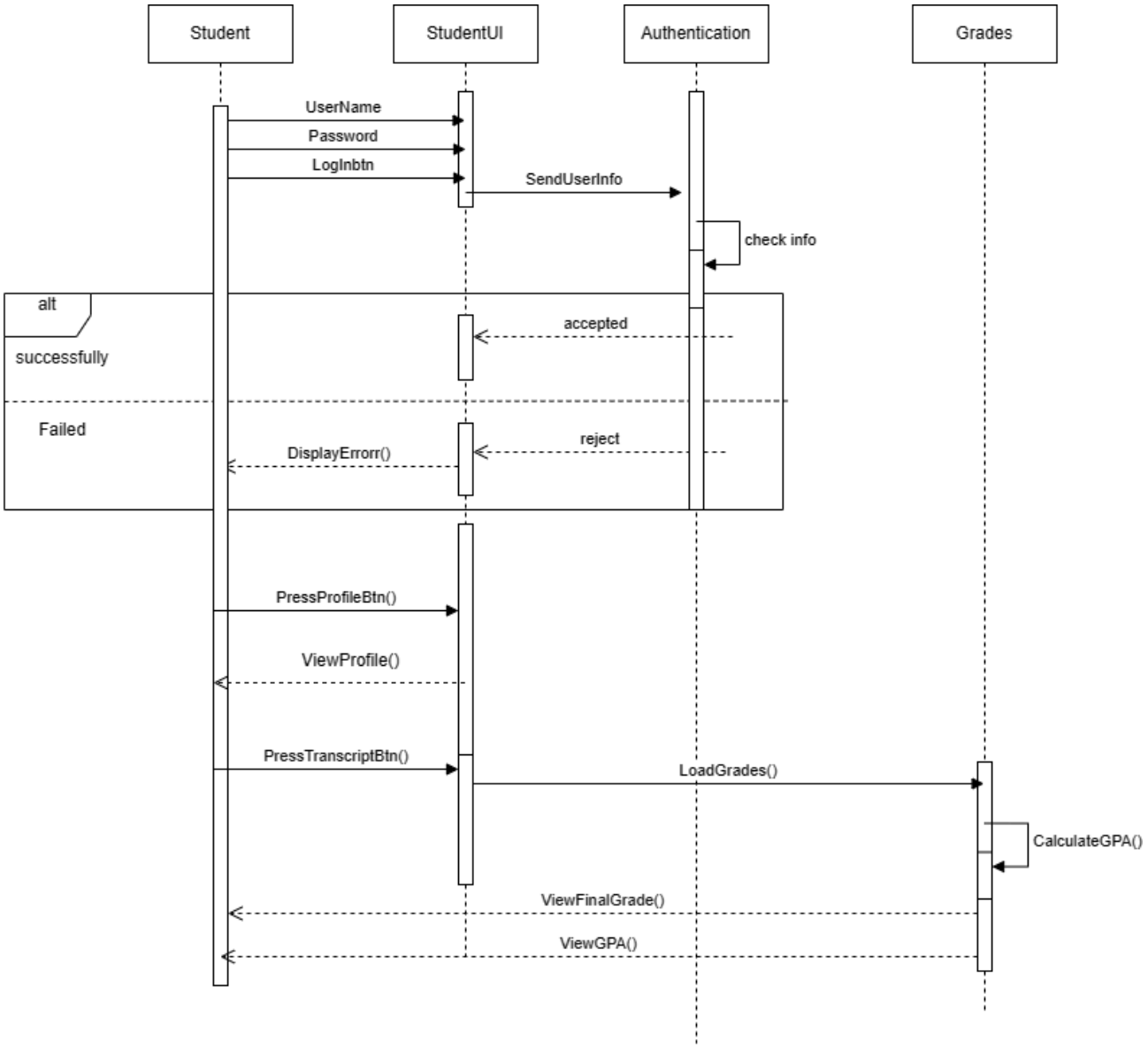
UC01 - Log in



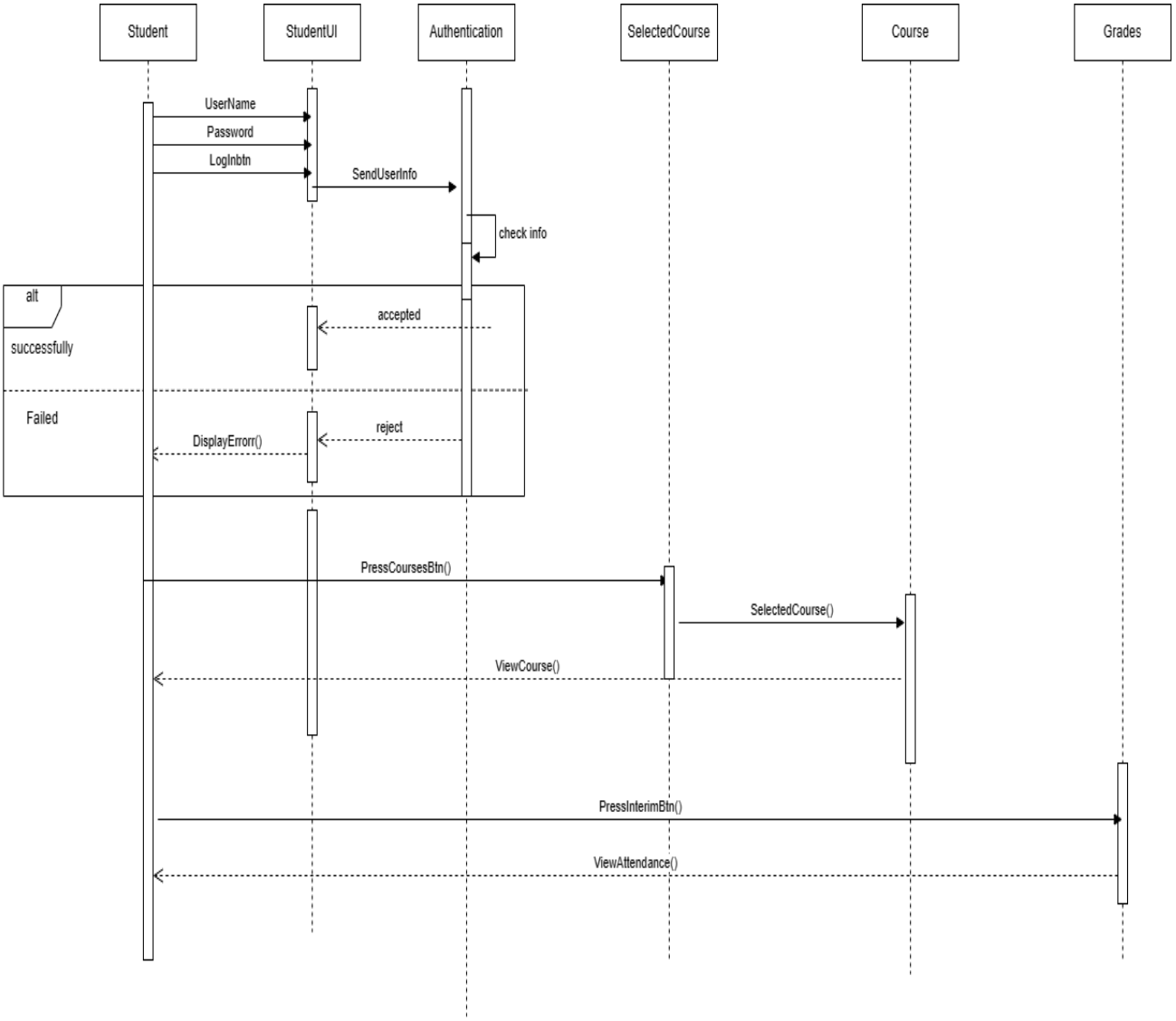
UC02 - View timetable



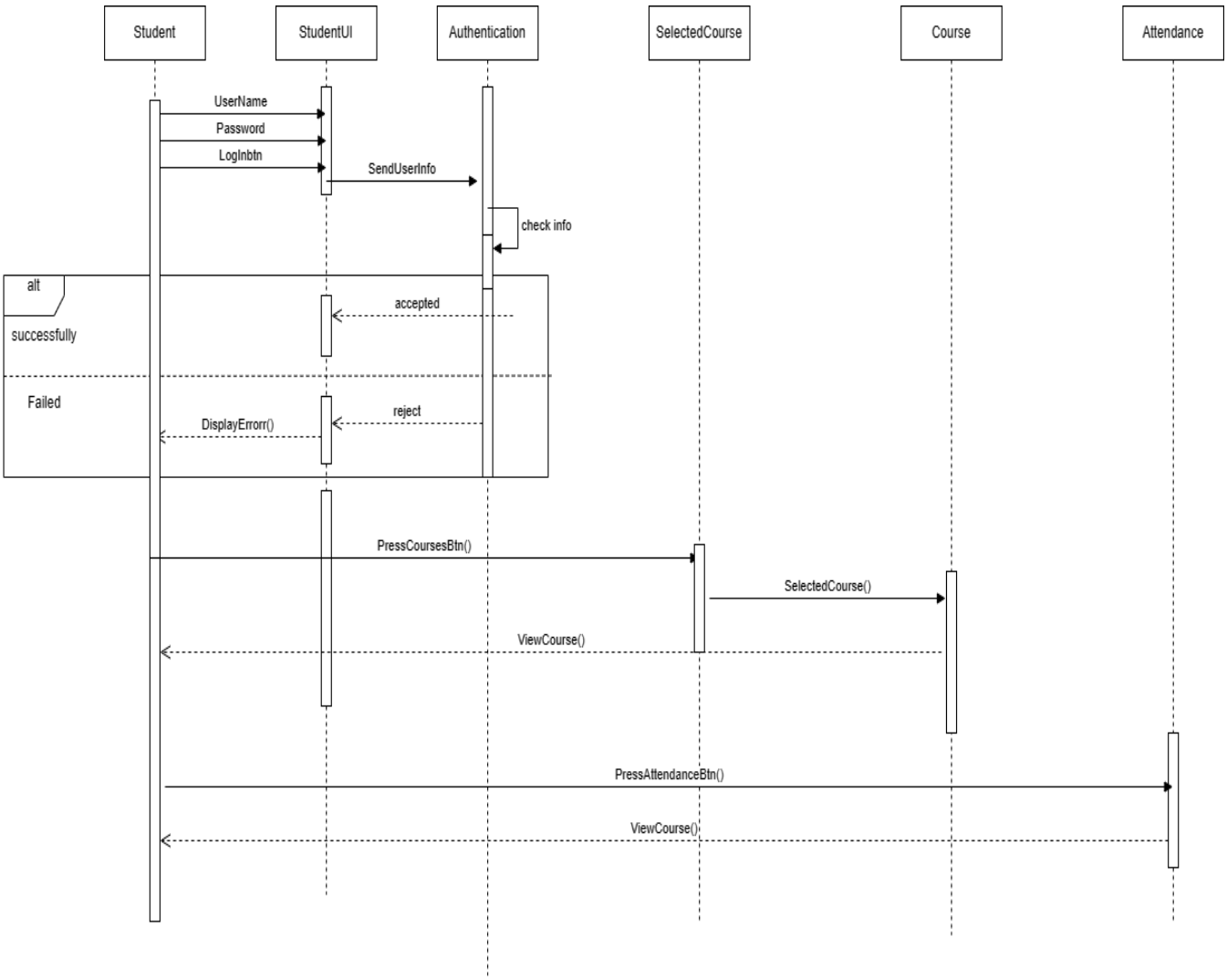
UC03- View Transcript



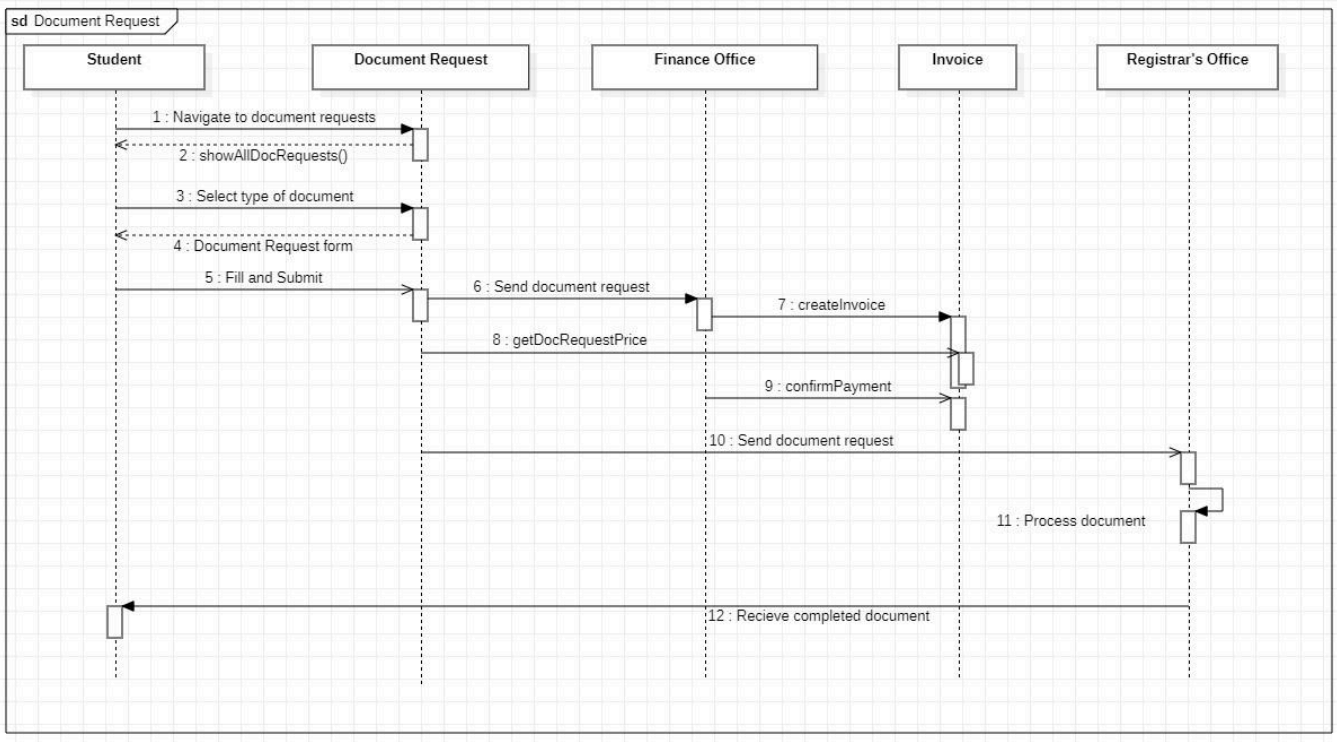
UC04- View grades



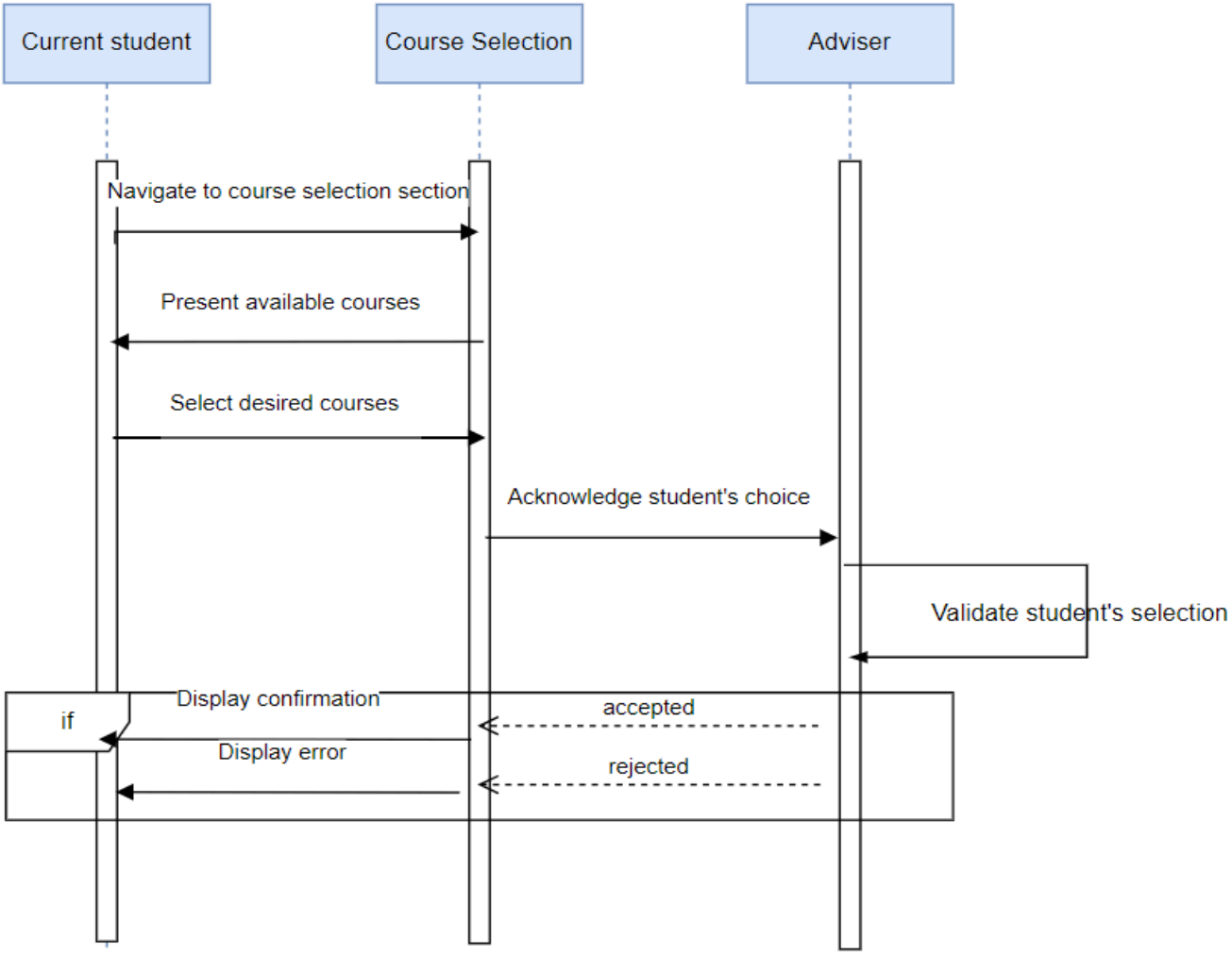
UC05- View Attendance



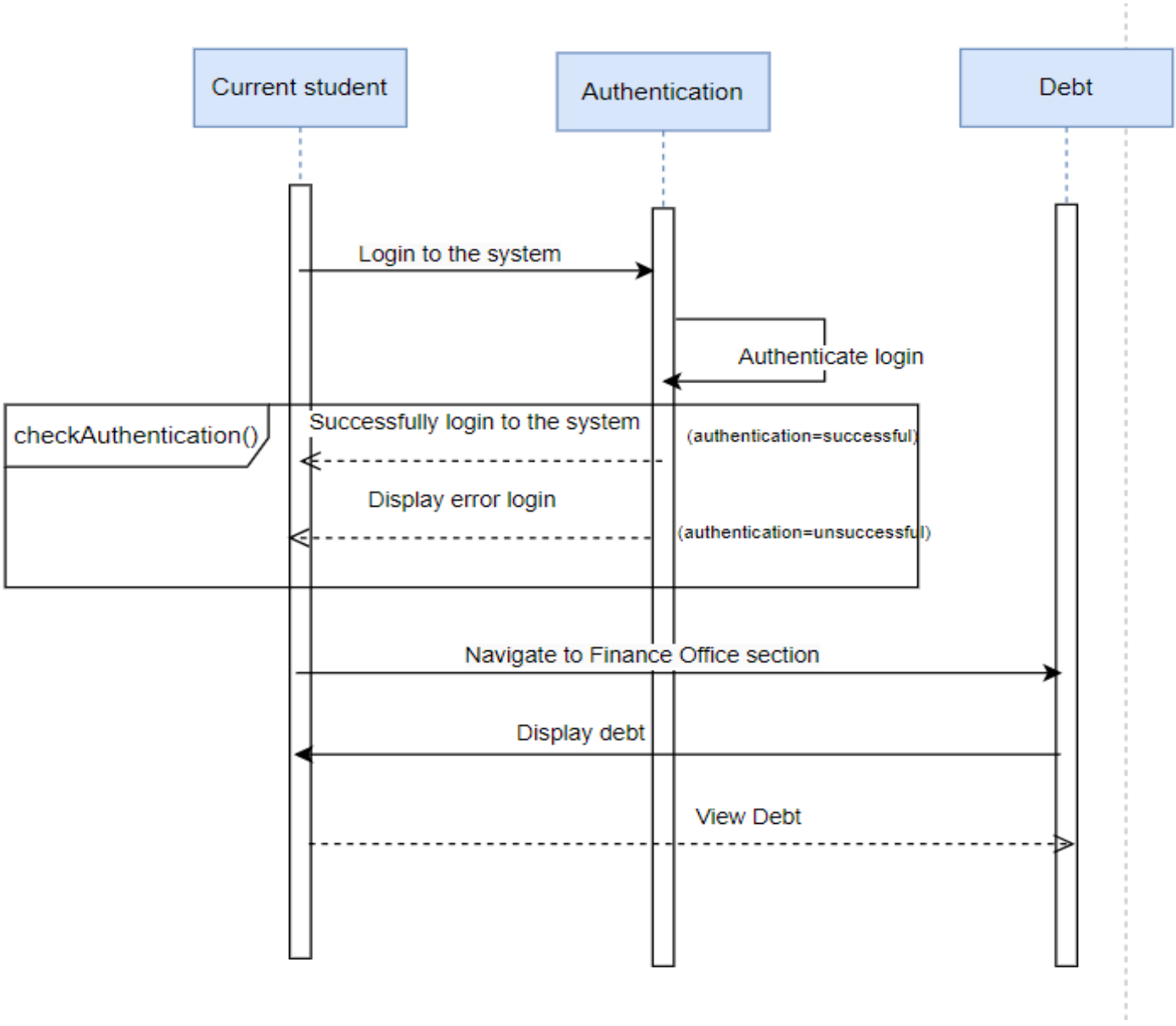
UC06, UC26, UC32 - Document Request



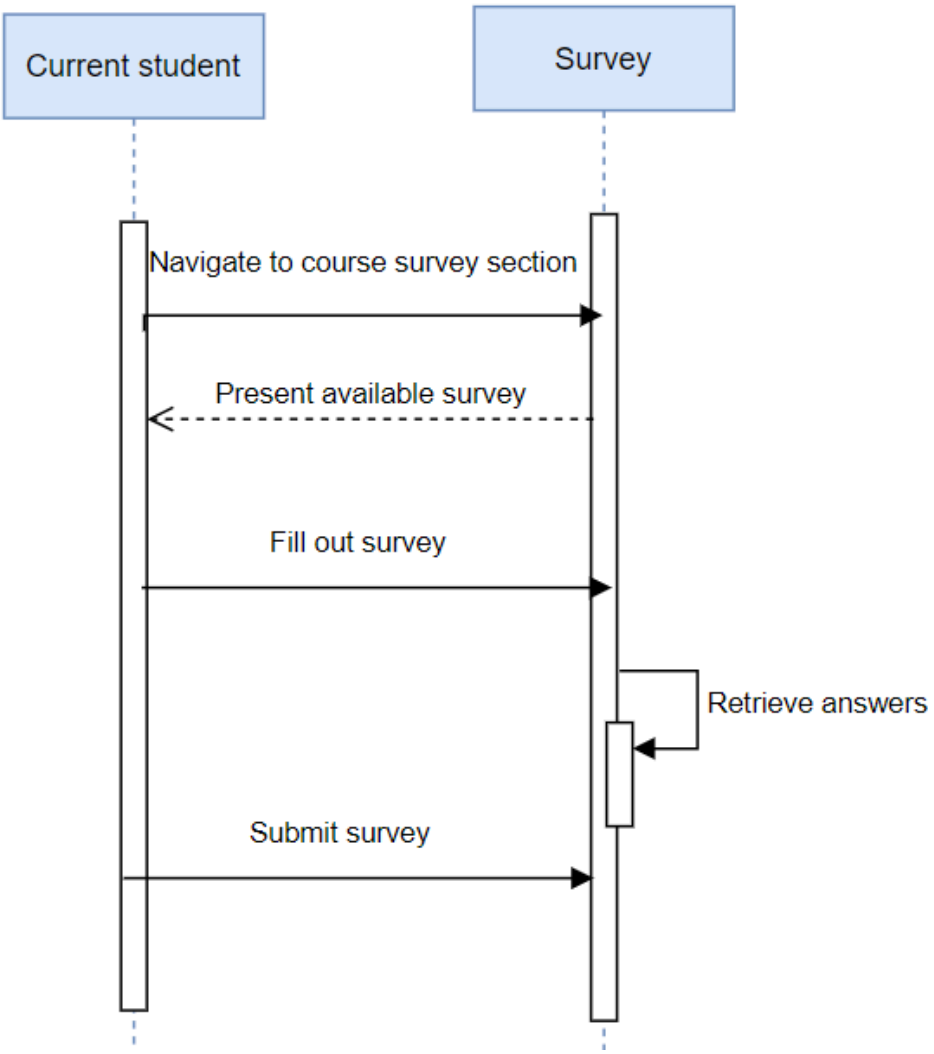
UC07 ,UC20- Course selection



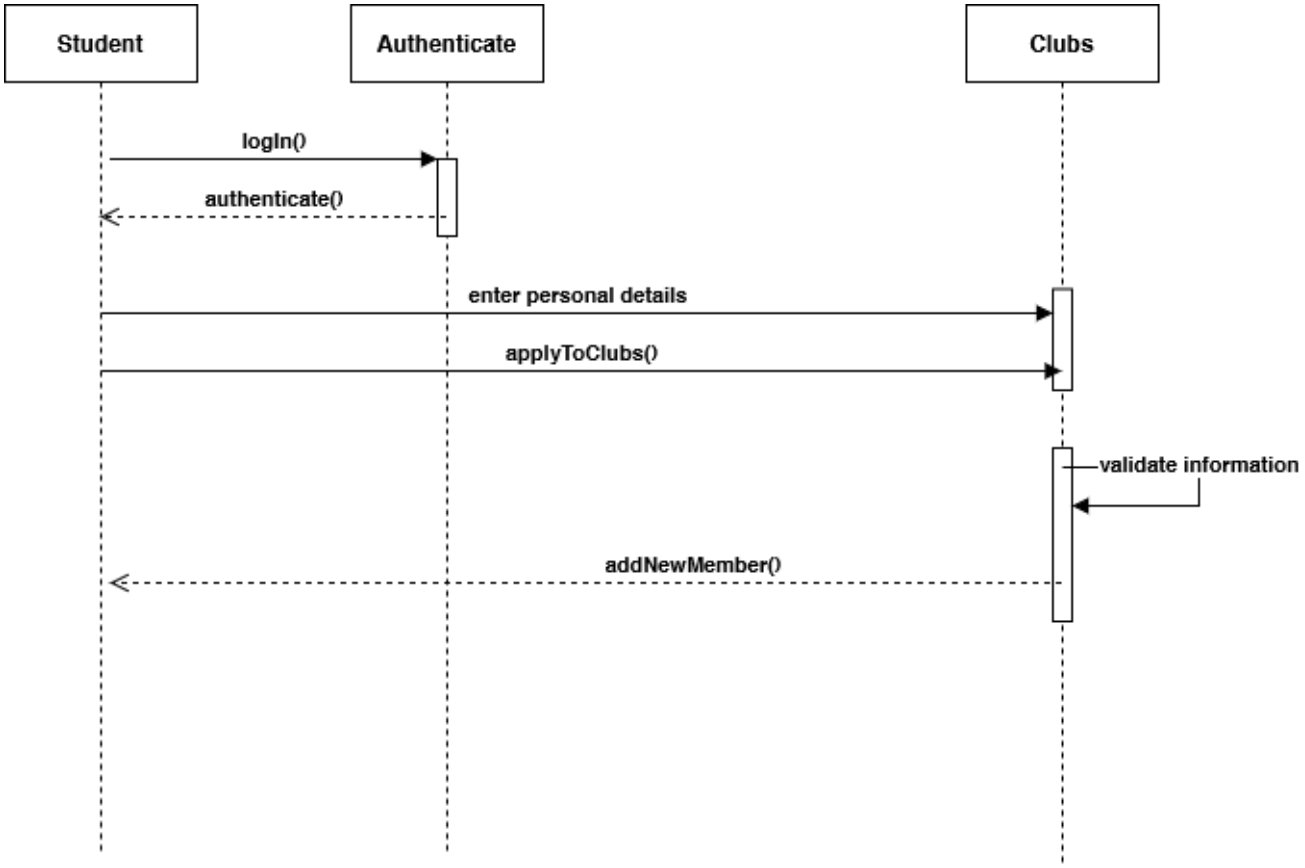
UC08 - View Debt



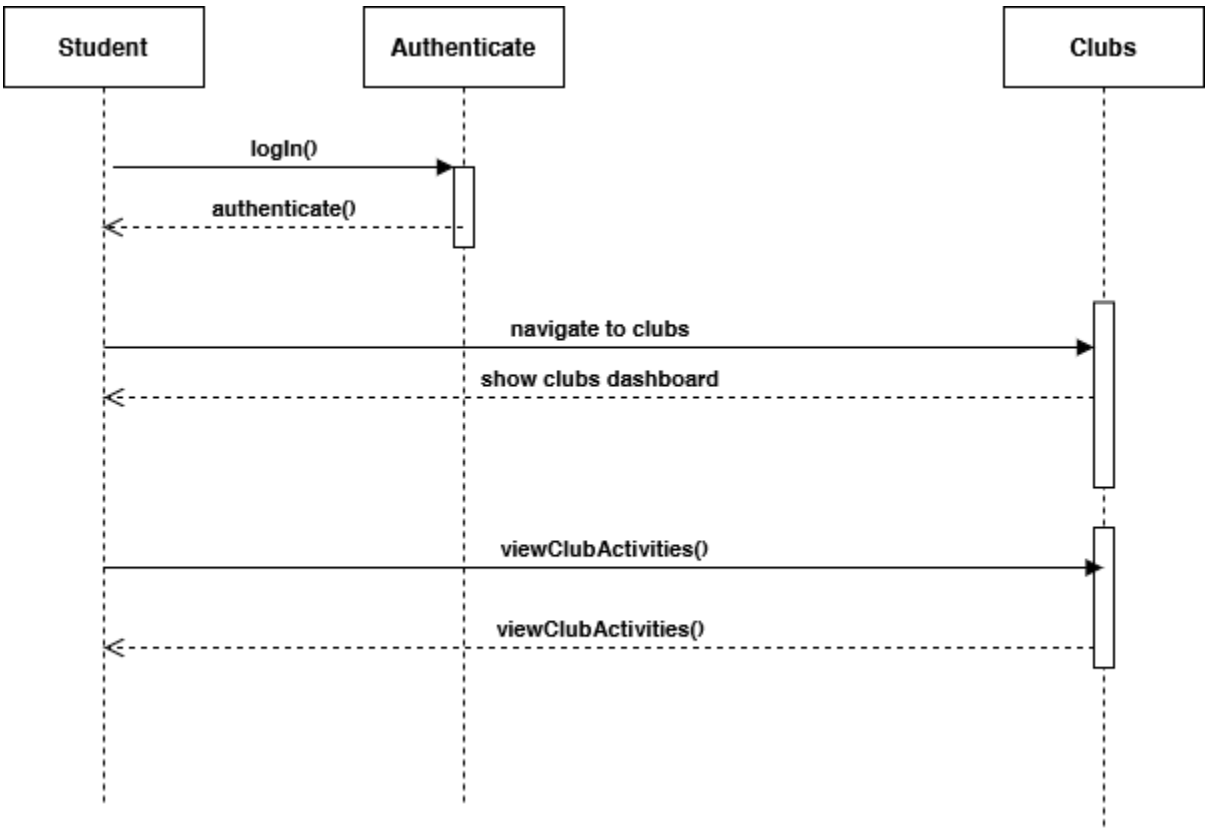
UC09 - Fill out Survey



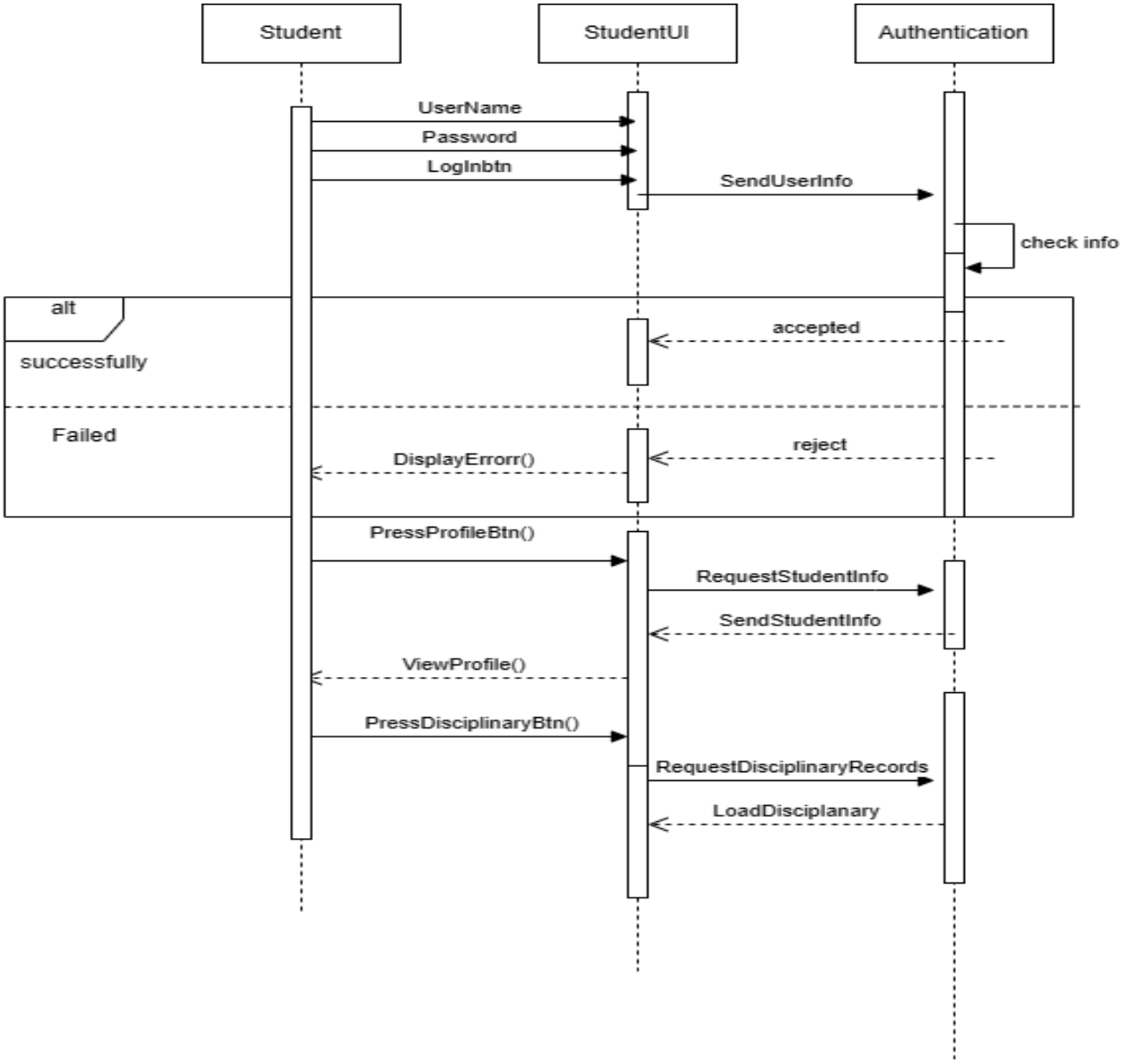
UC10 - Apply to Student Clubs



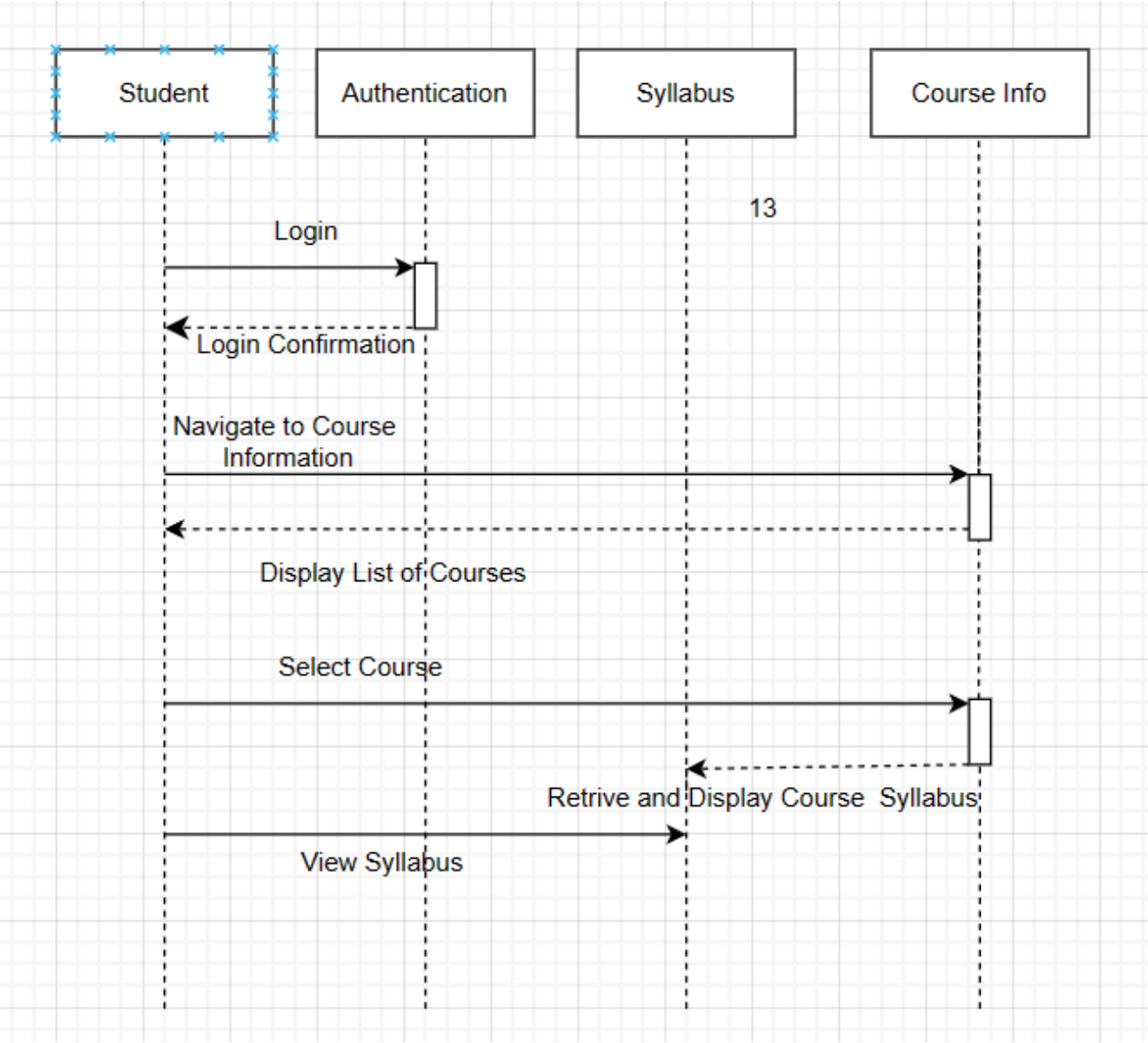
UC11 - View Club Activities



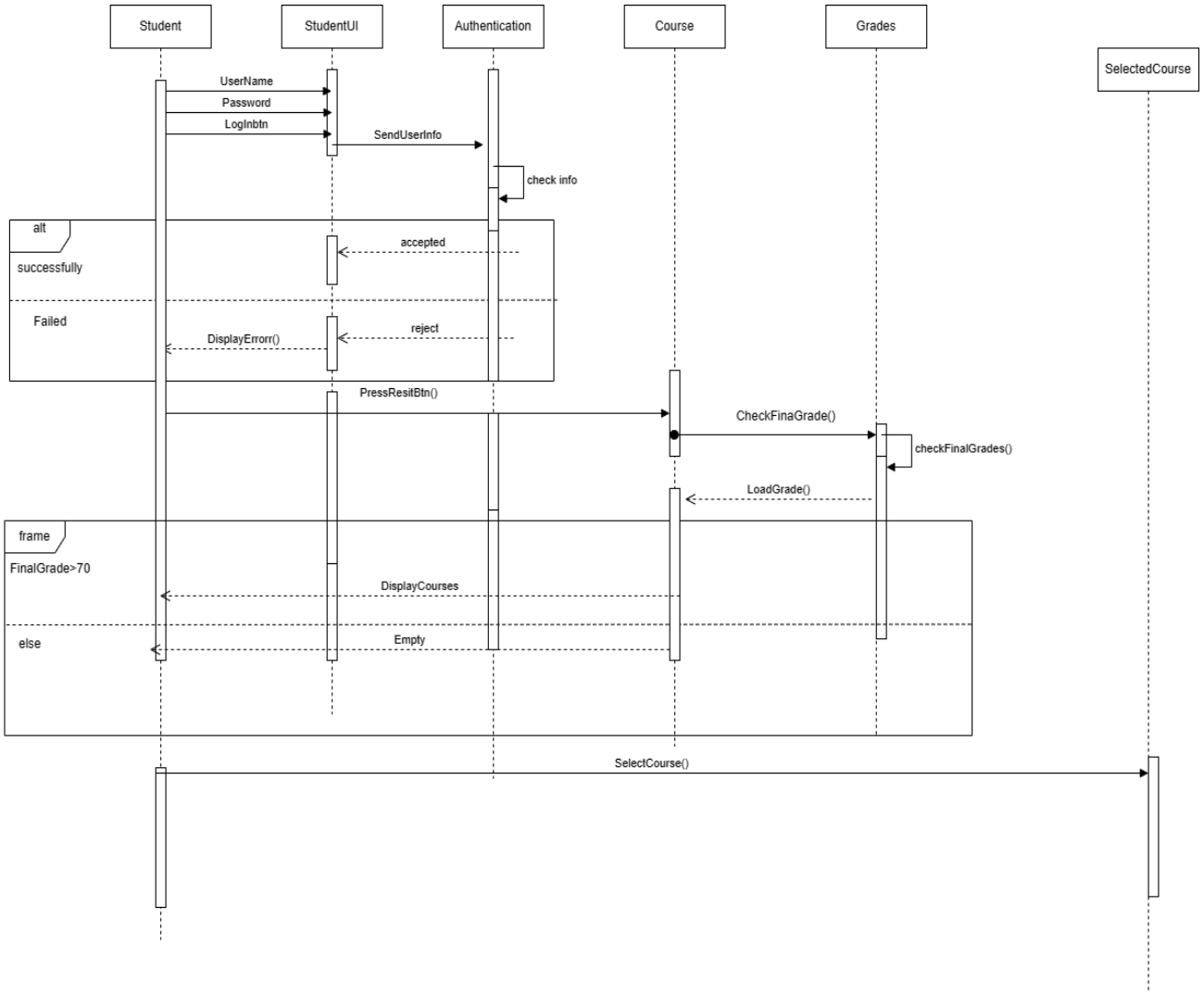
UC12-Disciplinary records



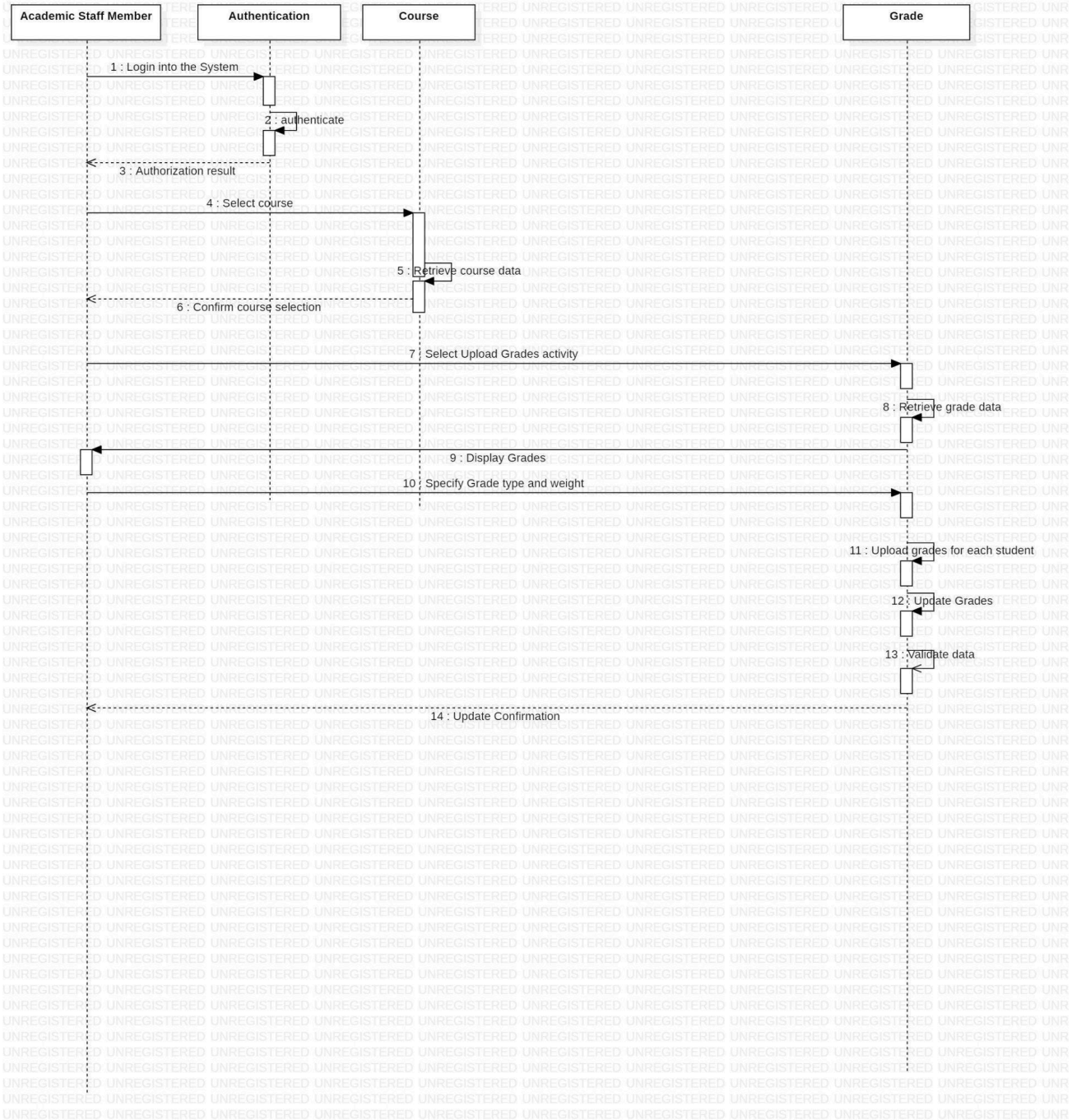
UC13 - View Course Syllabus



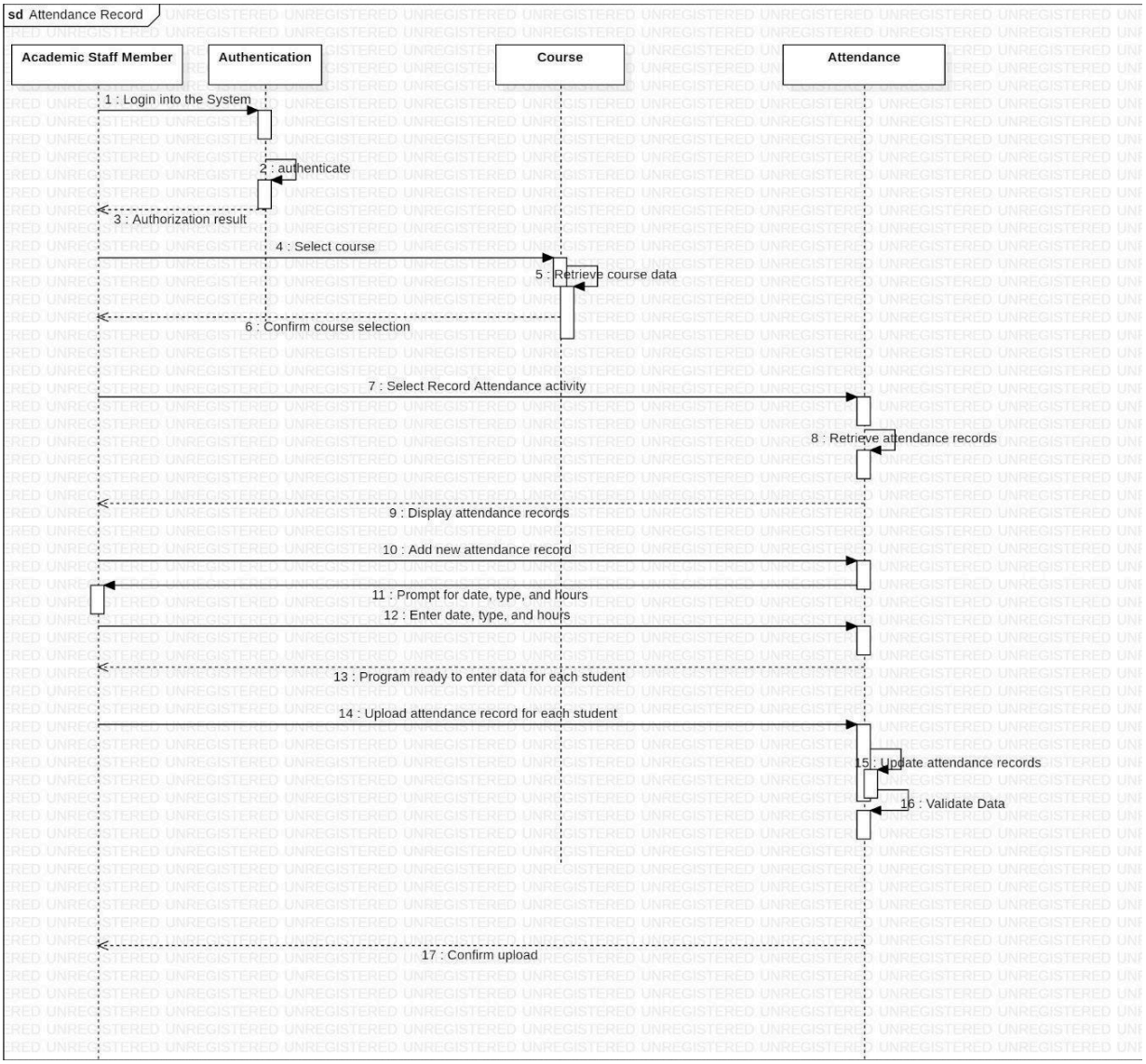
UC14-Apply for Resit Exam



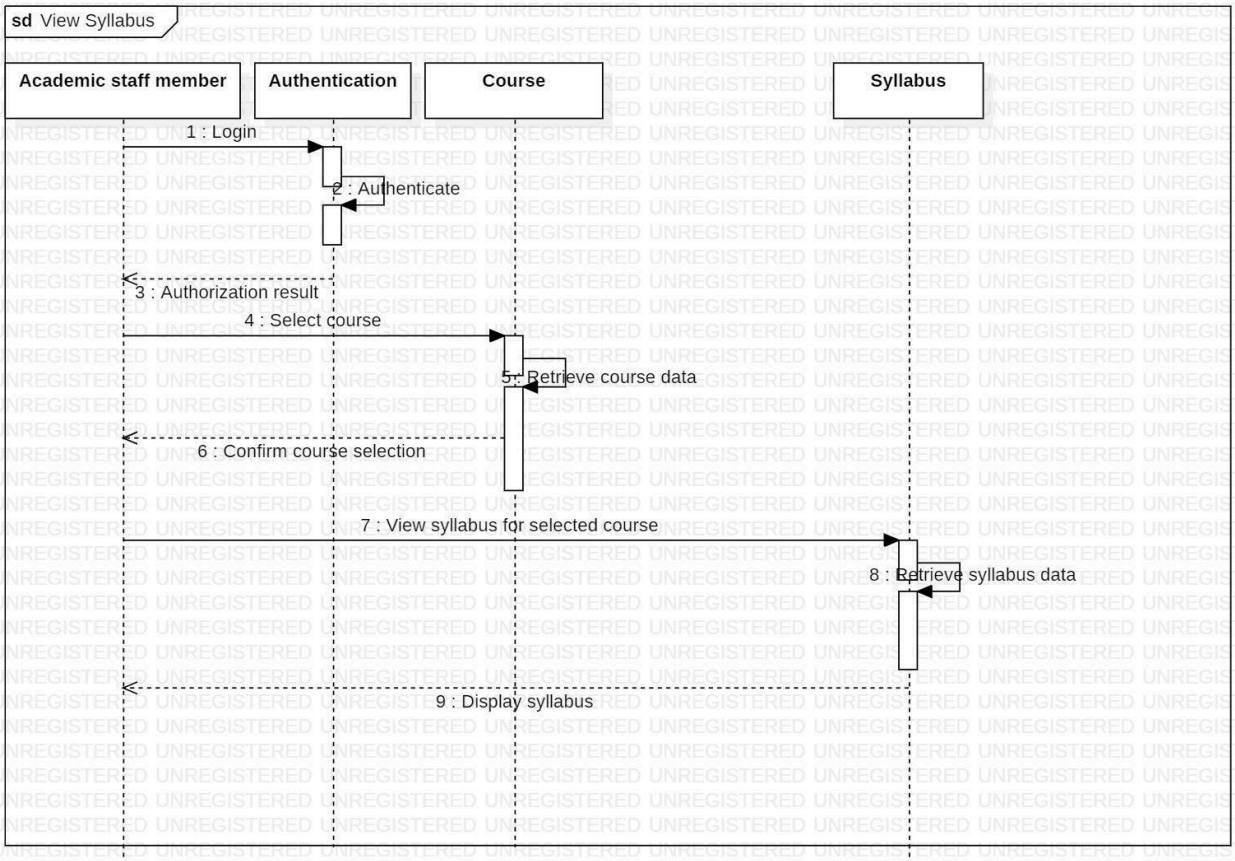
UC15 - Load grades into the system



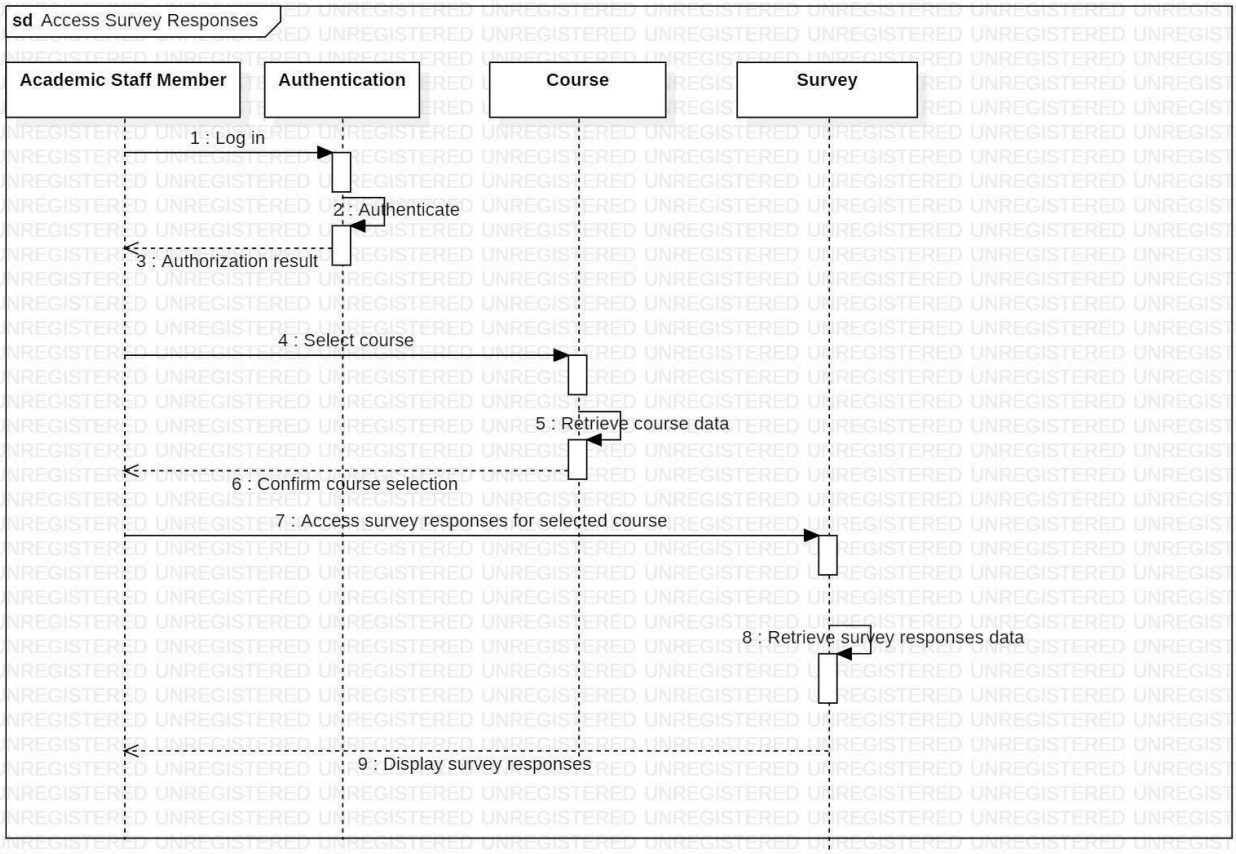
UC16 - Record Attendance of Students



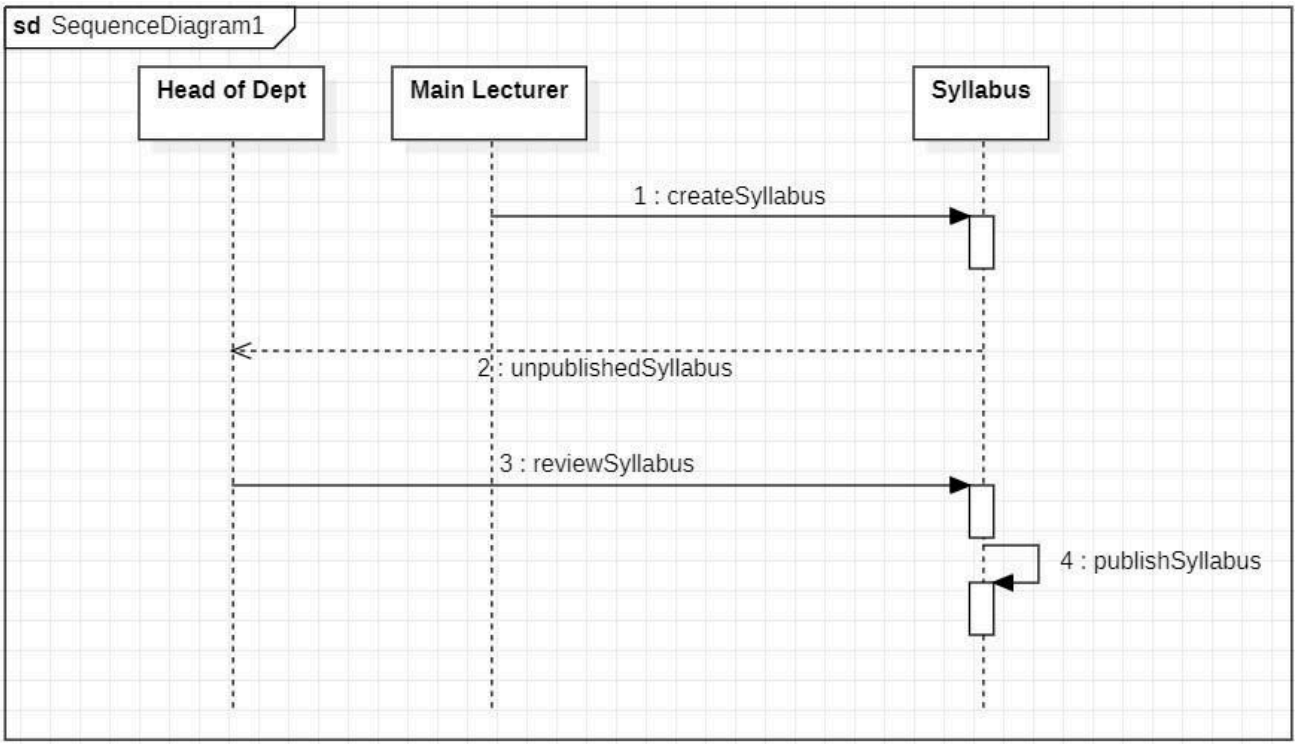
UC17 - View Course Syllabus



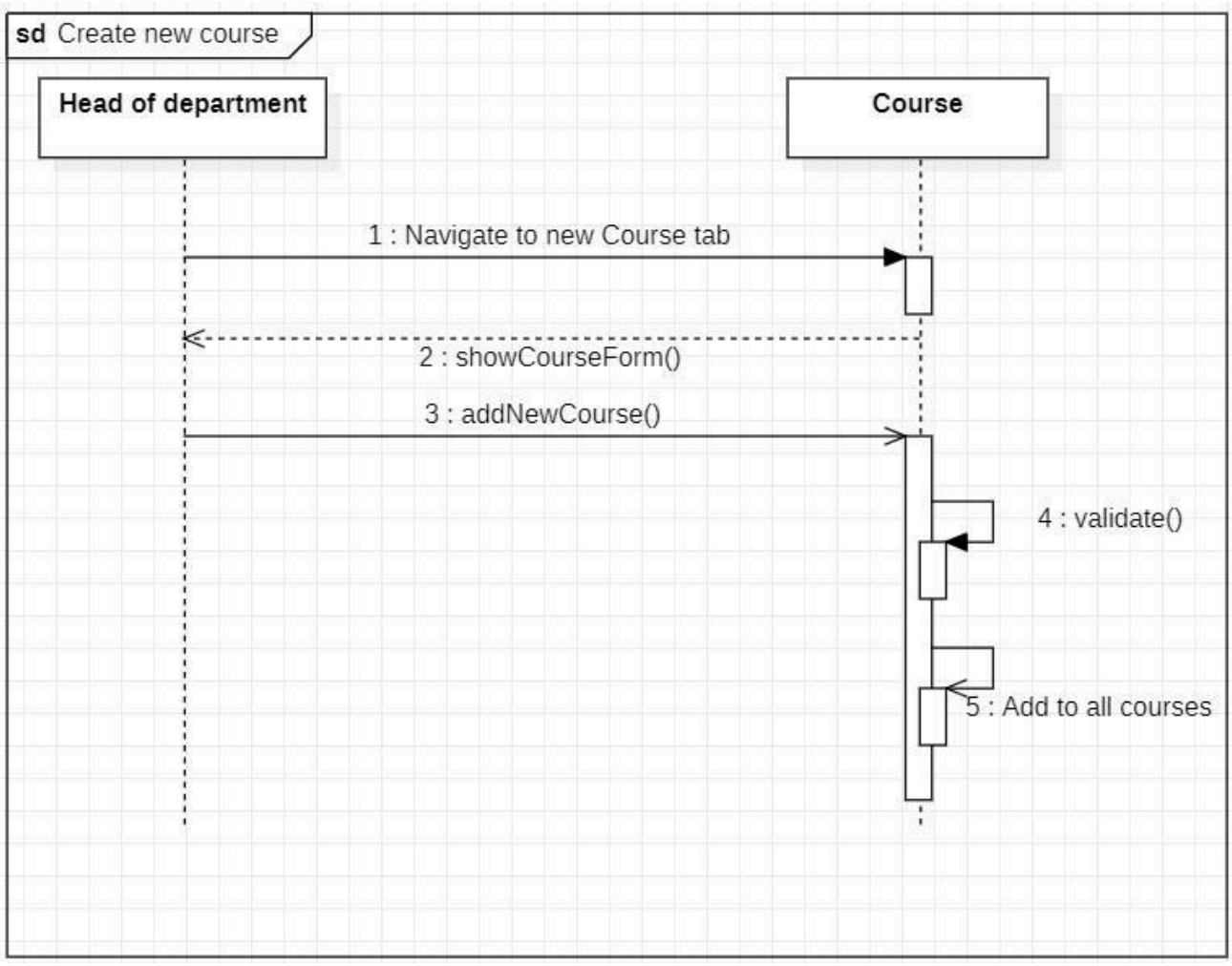
UC18 - Access Survey Responses



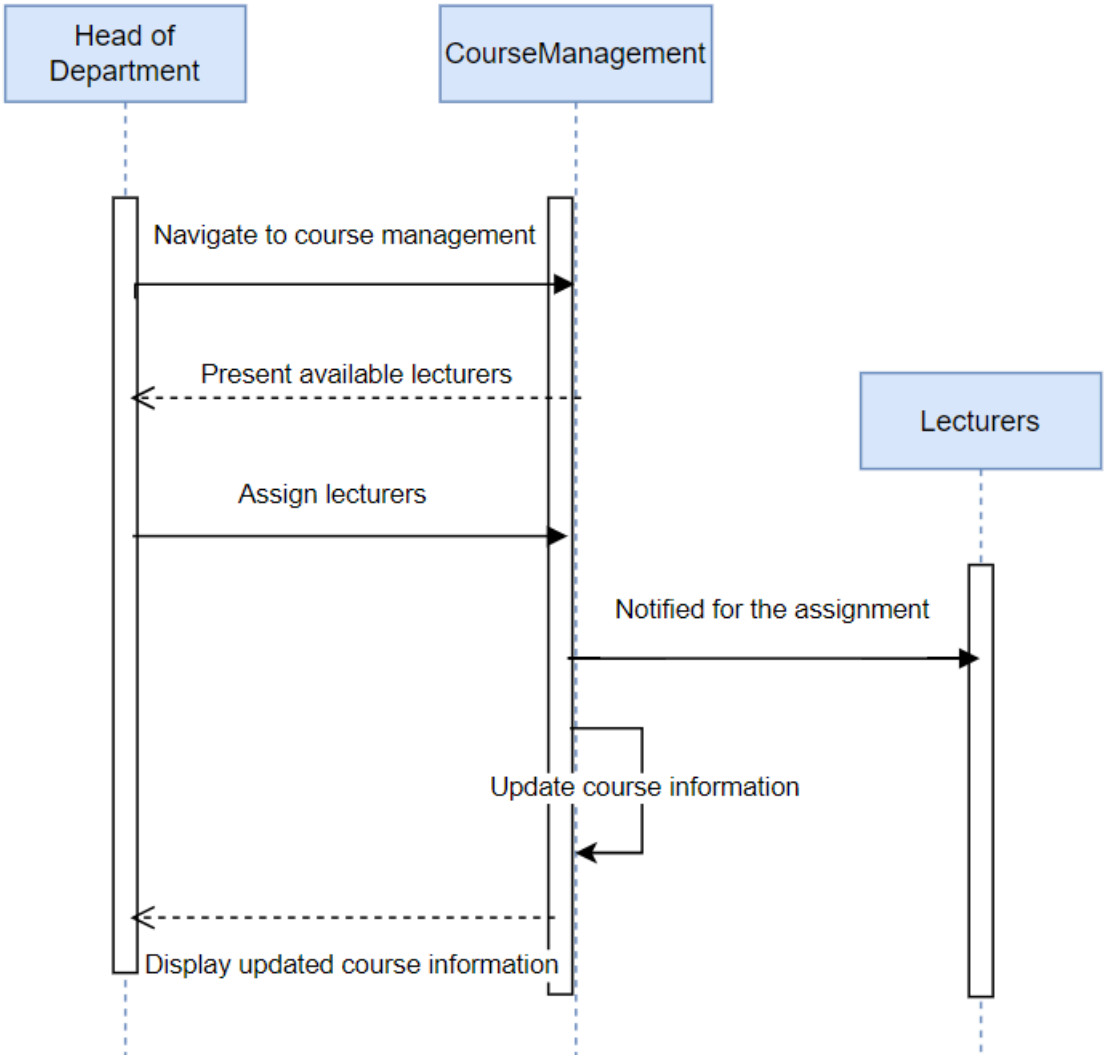
UC19, UC21- Syllabus Publishing



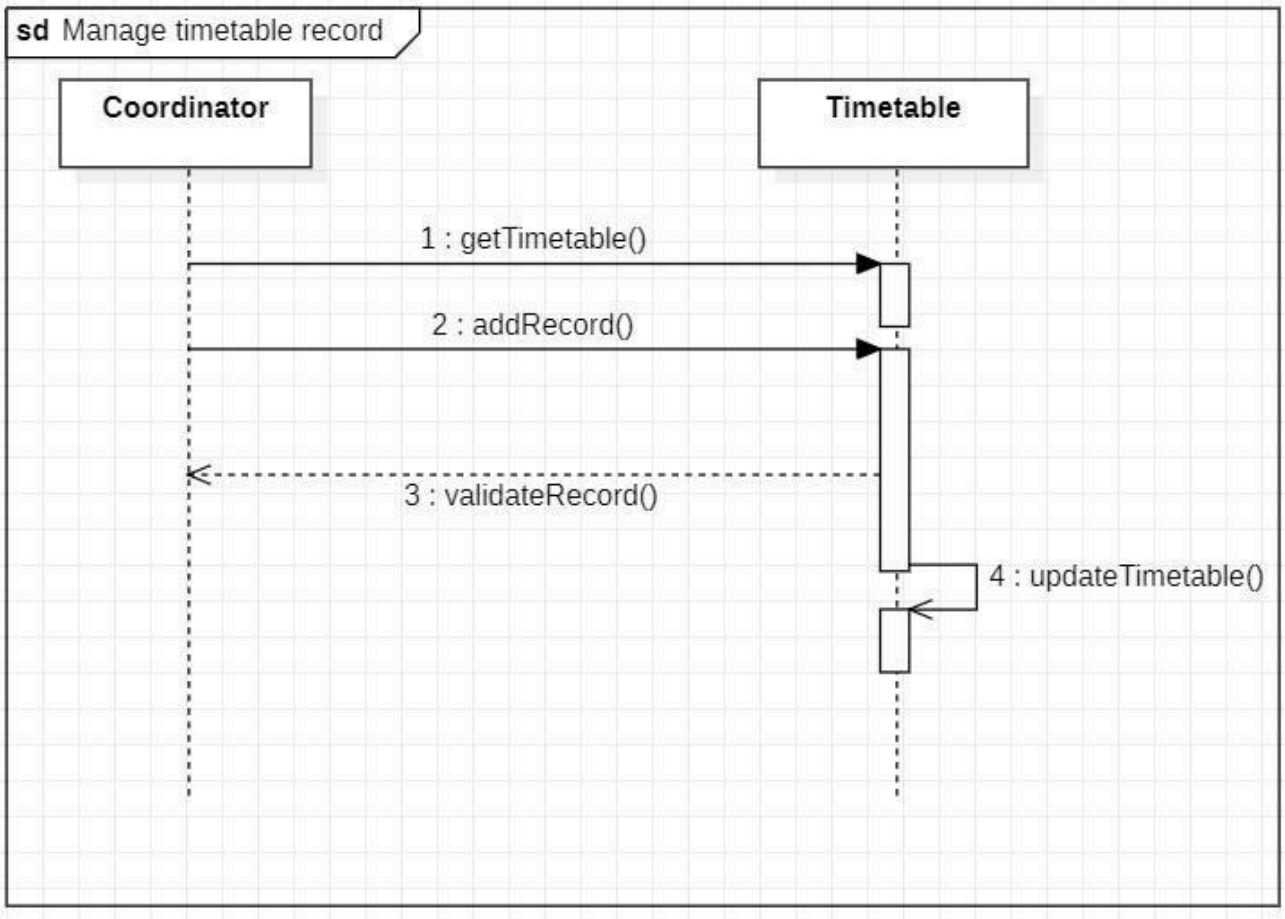
UC22 - Create Course



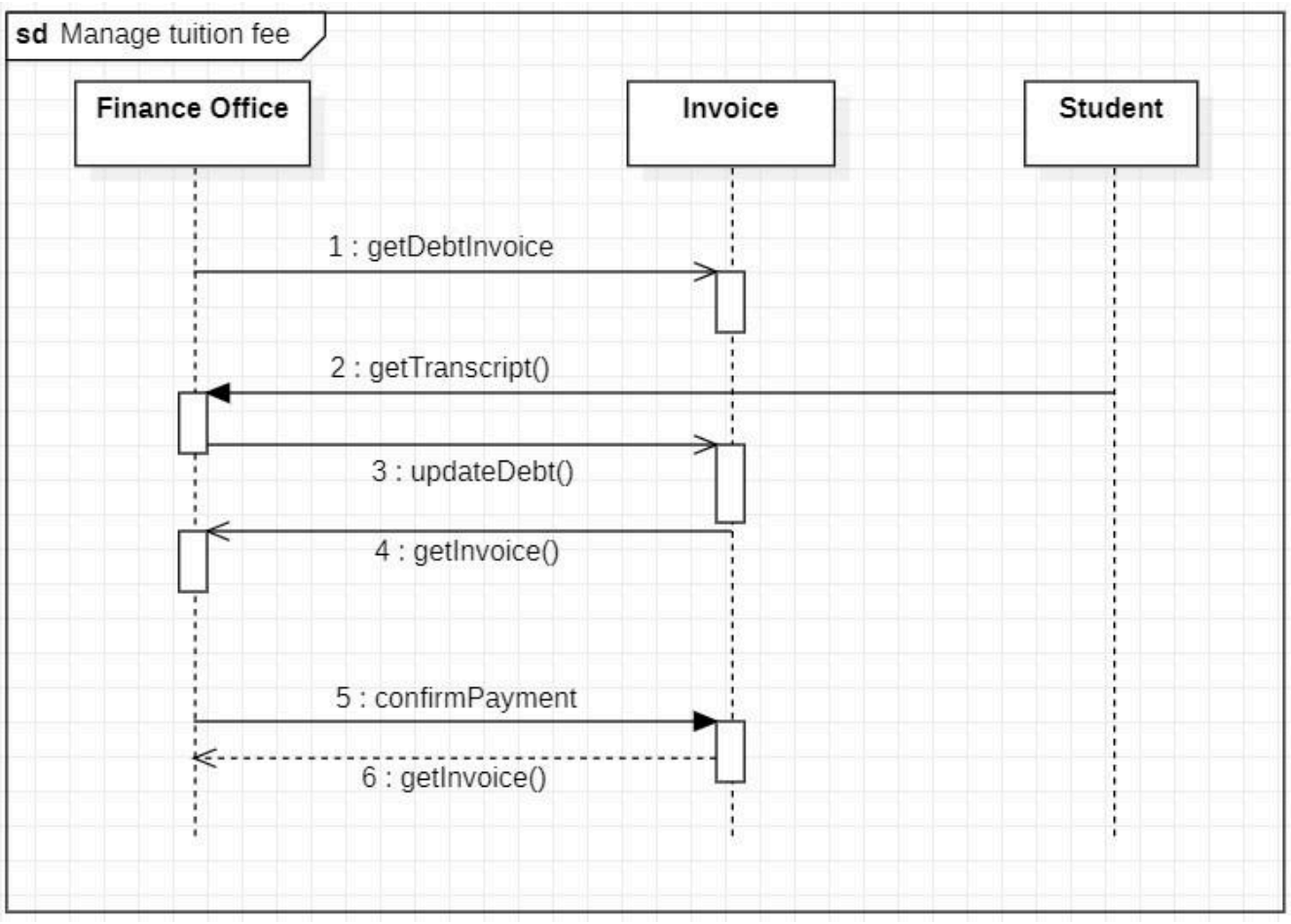
UC23 - Assigning lecturers to courses



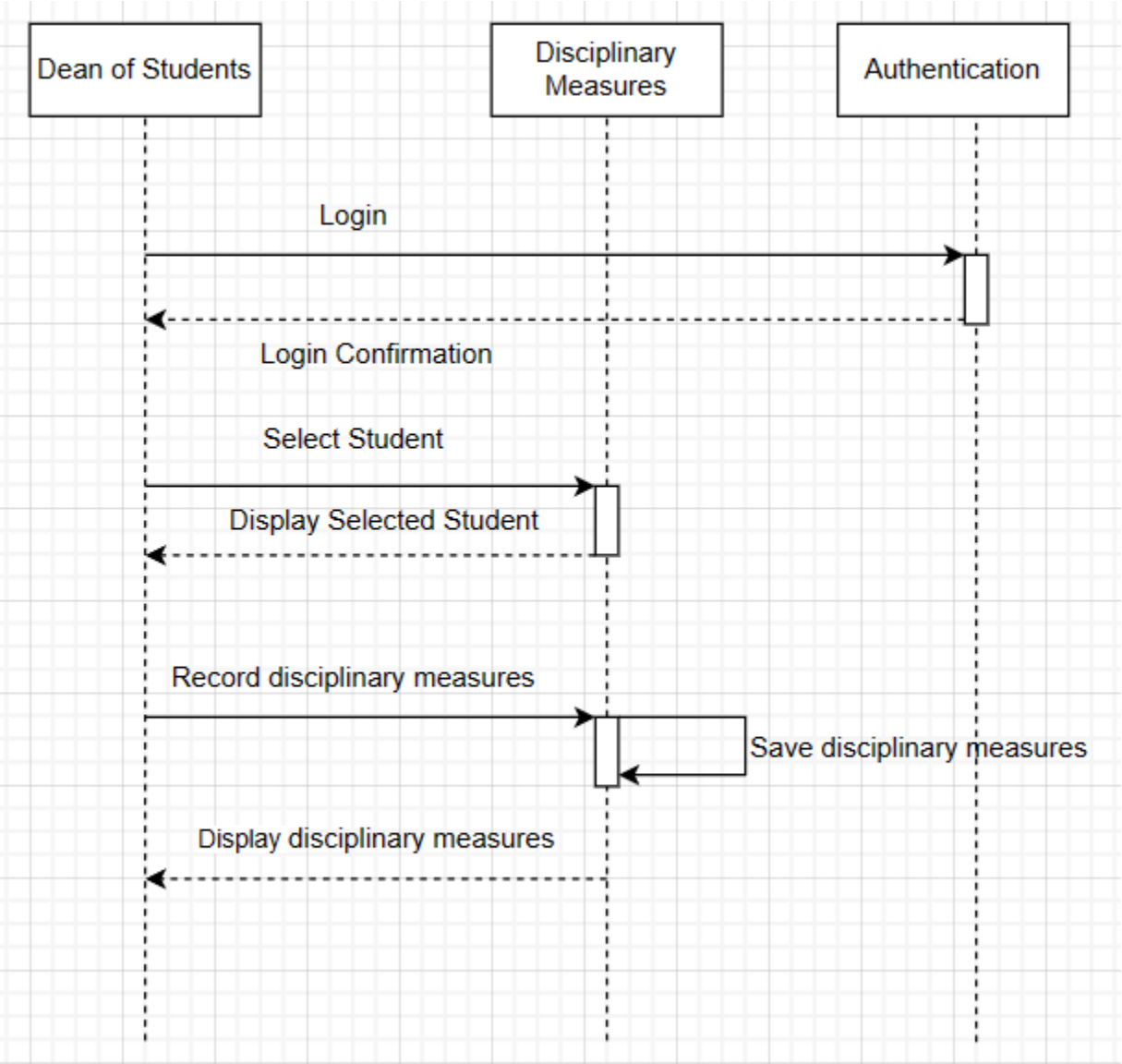
UC24 - Upload timetable



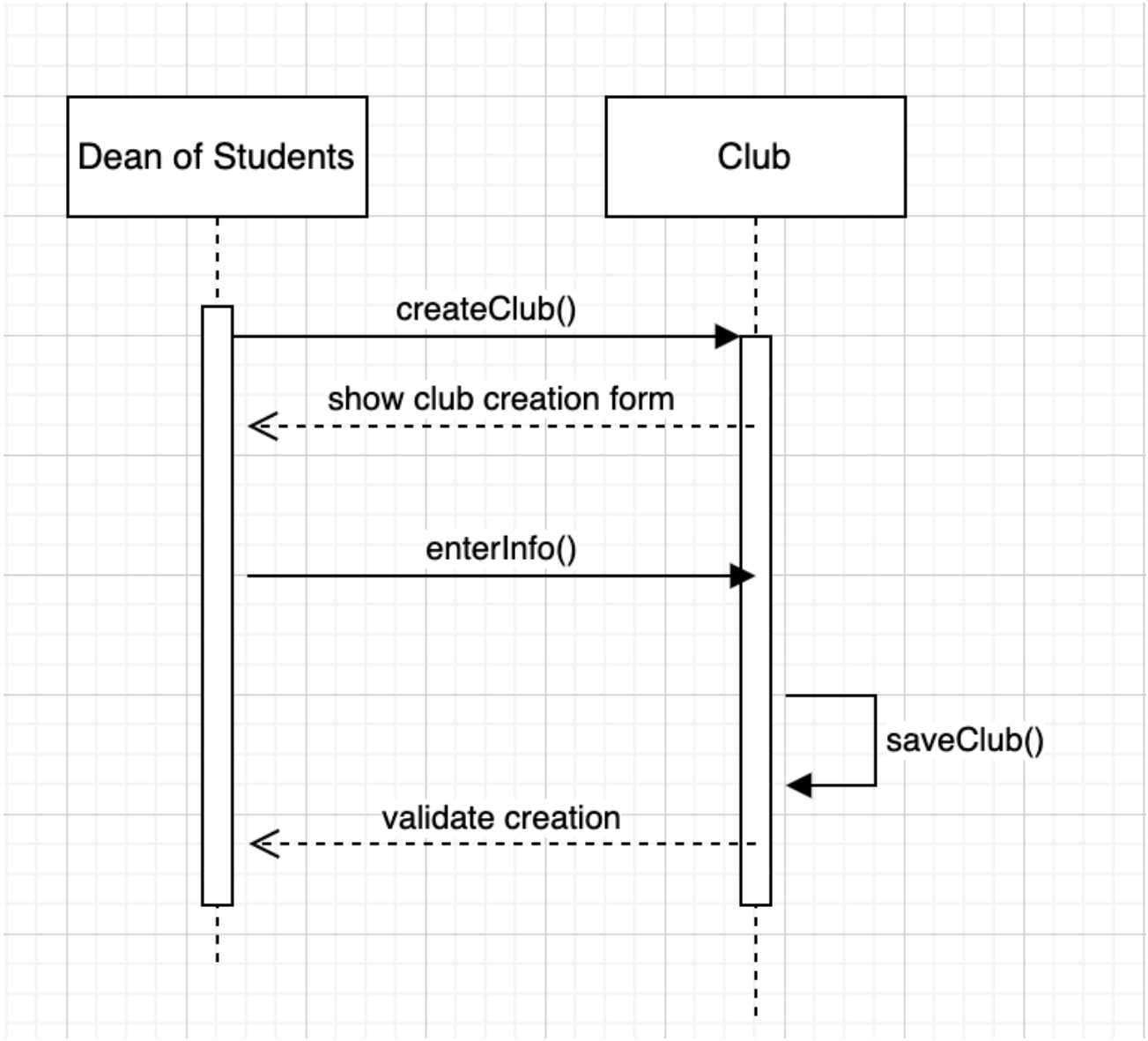
UC25 - Manage tuition fee



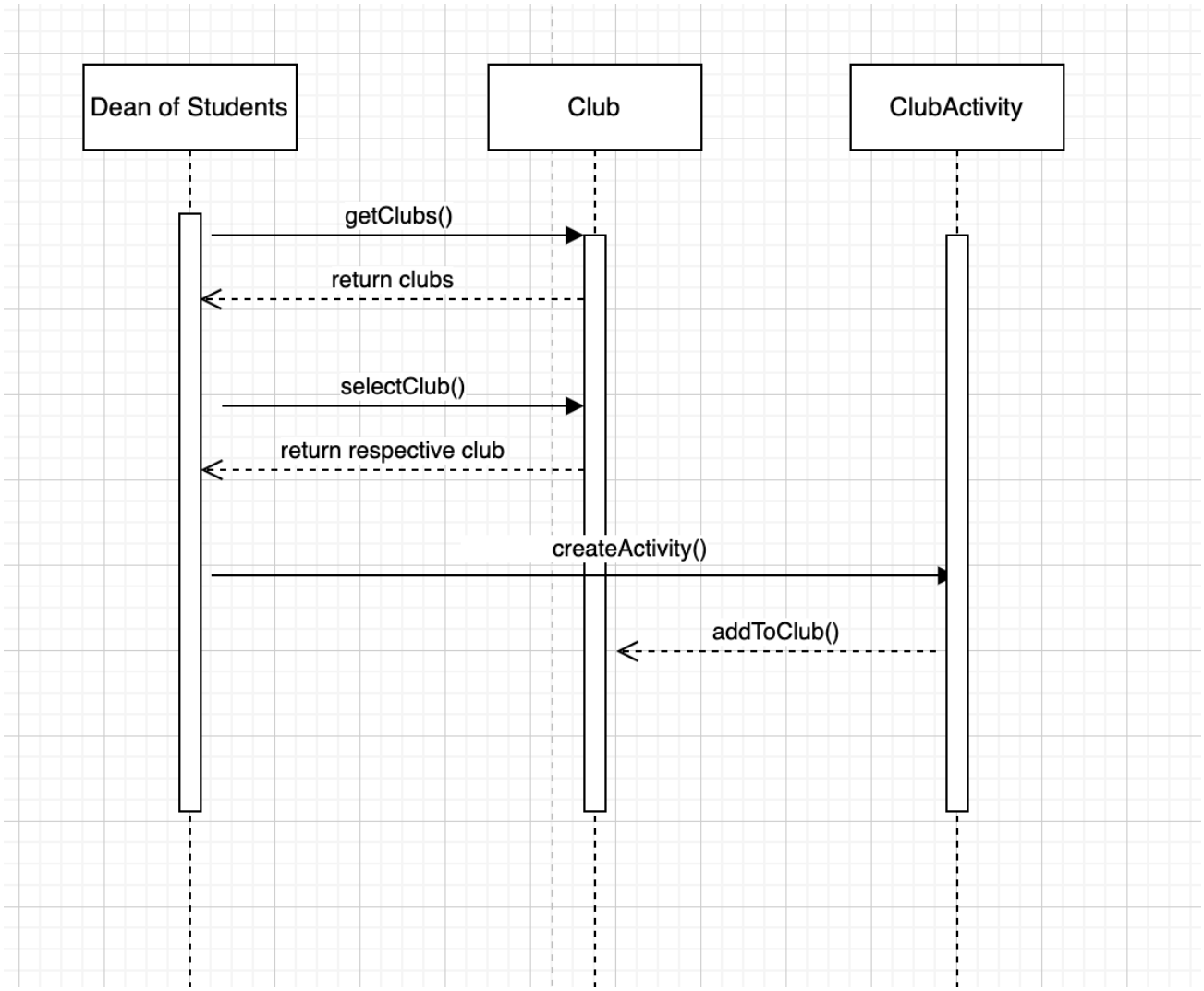
UC27- Record disciplinary cases



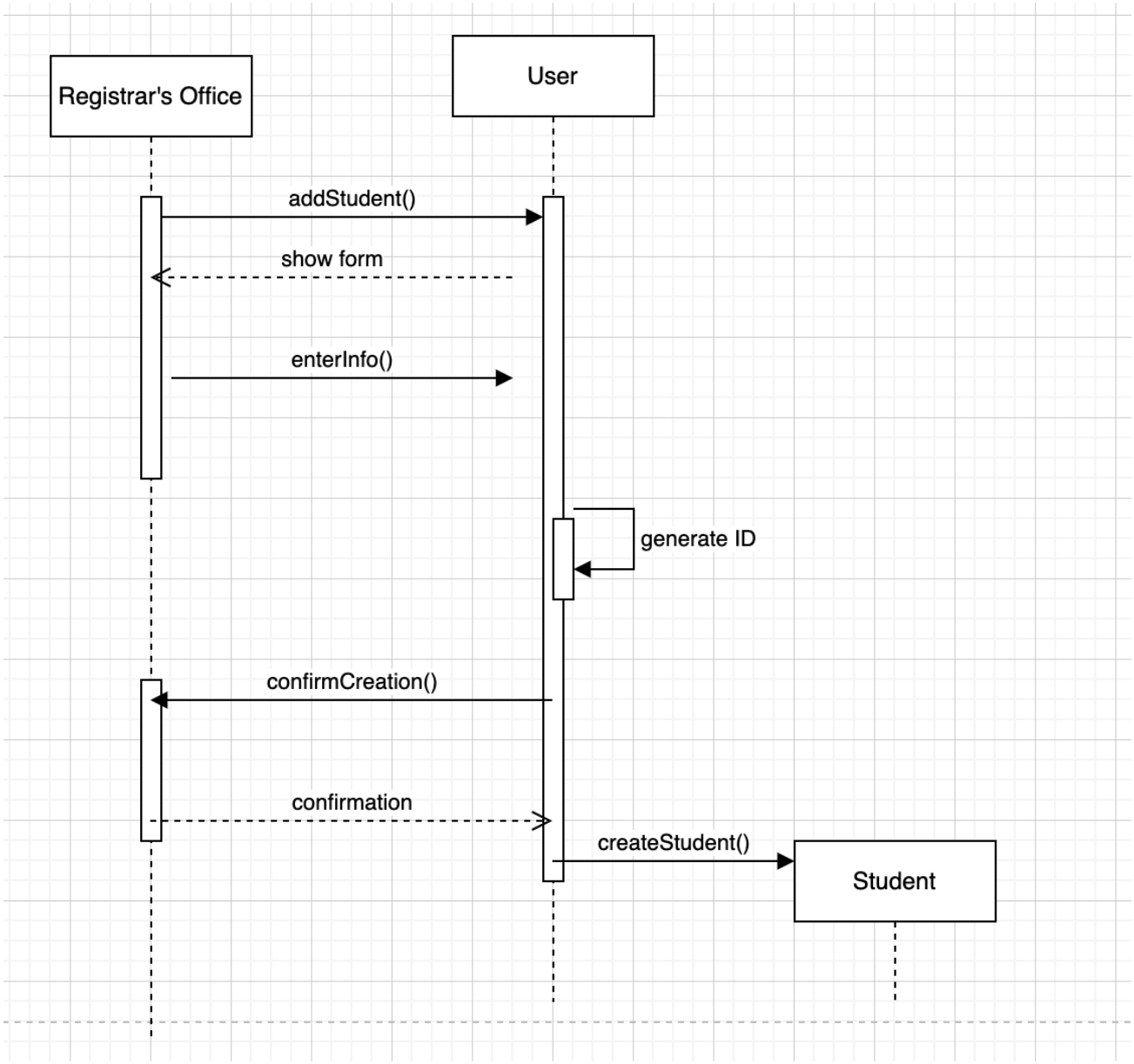
UC28 - Create Student Clubs



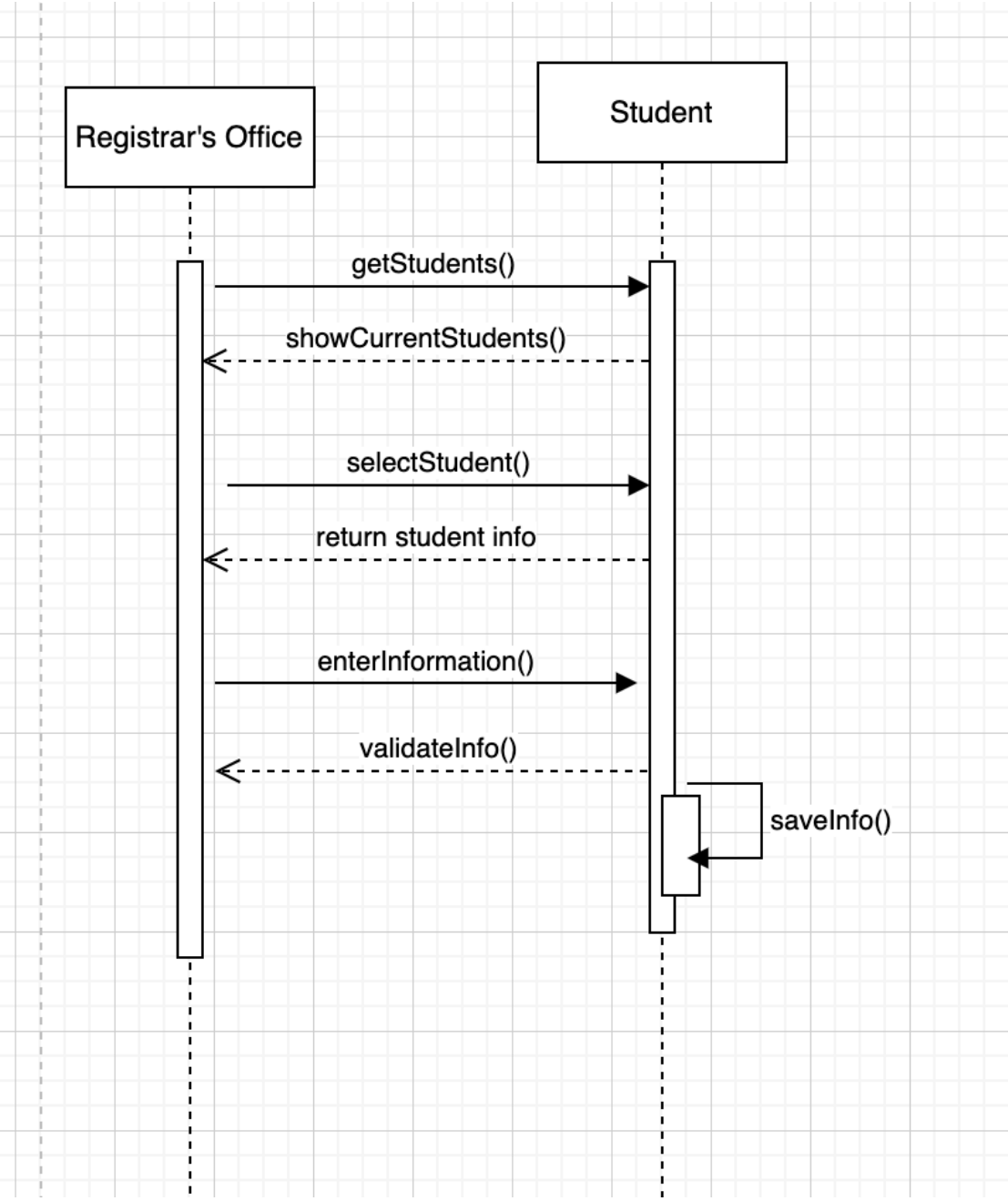
UC29 - Post Student Club Activities



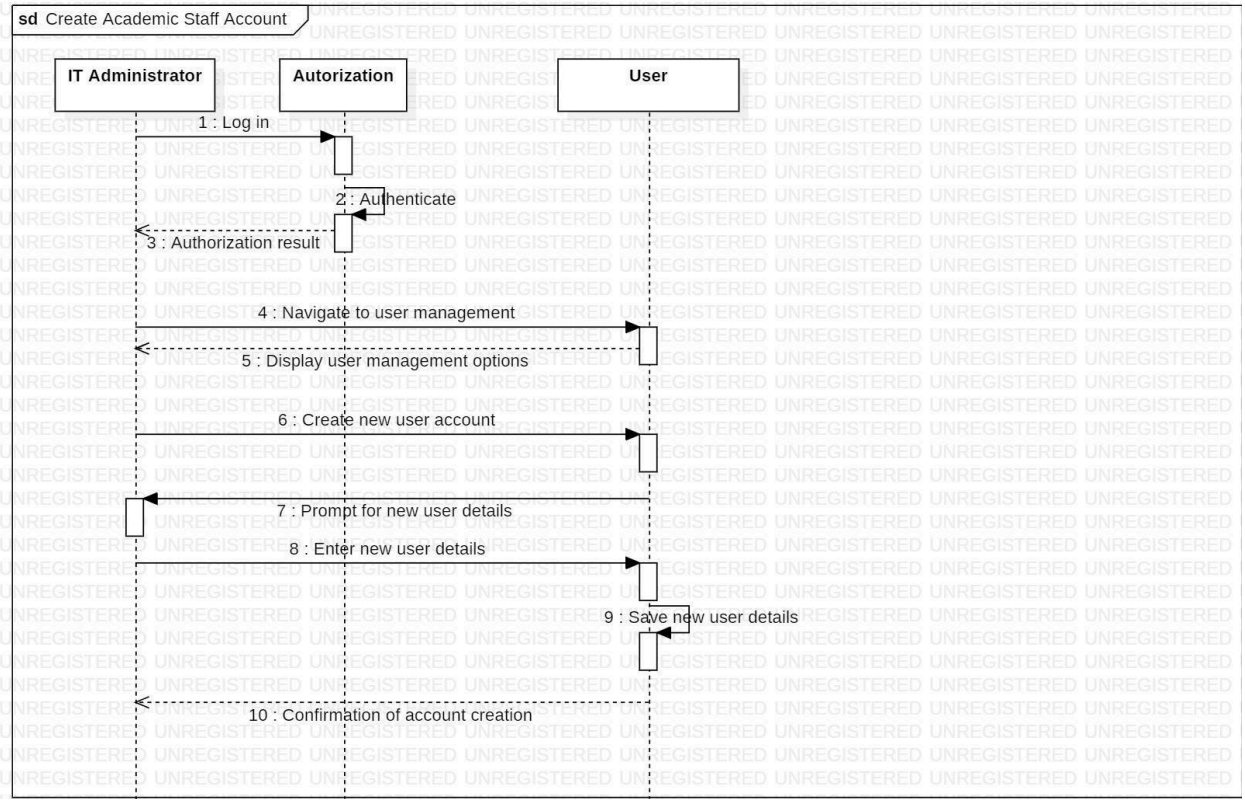
UC30 - Create new Student Account



UC31 - Update/Edit student information

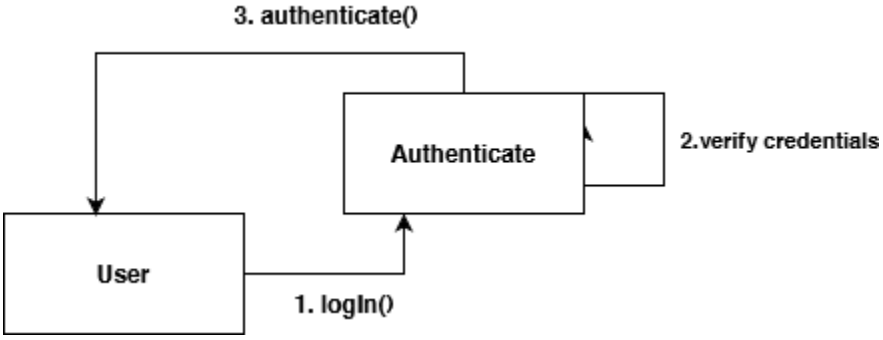


UC33 - Create Academic Staff Account

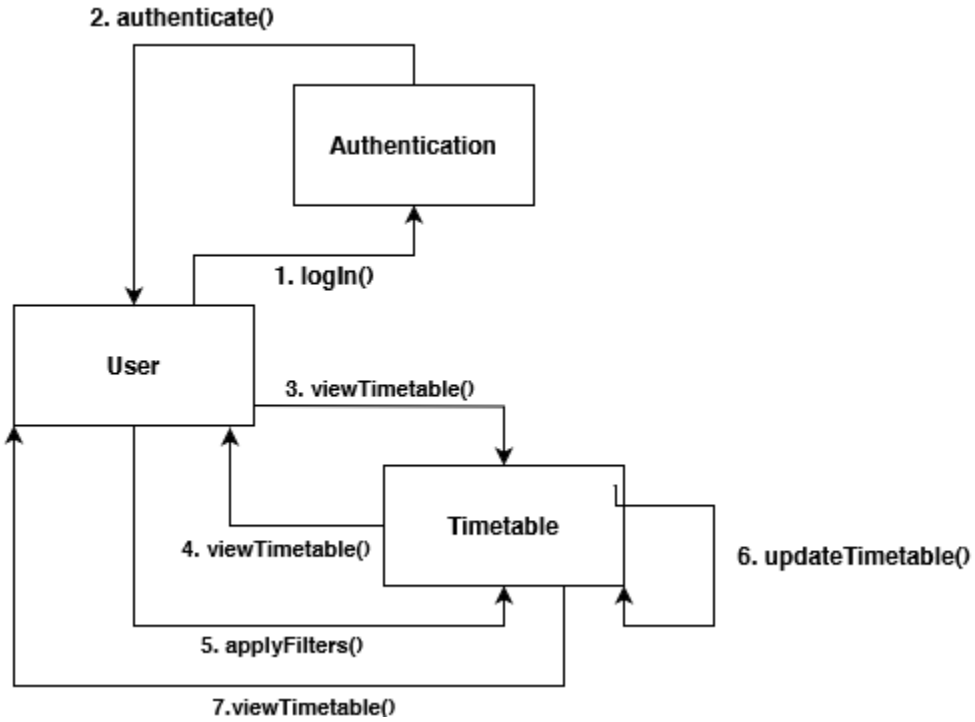


Collaboration diagrams

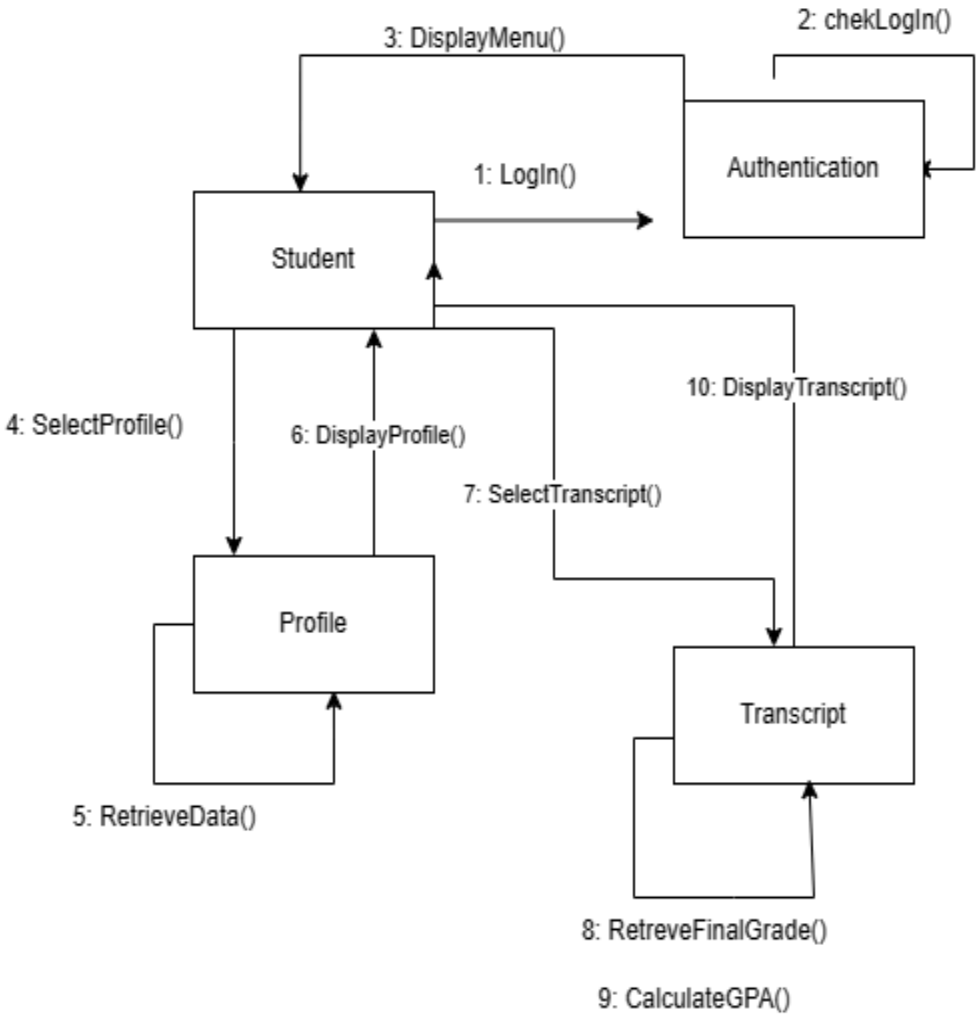
UC01 - Log in



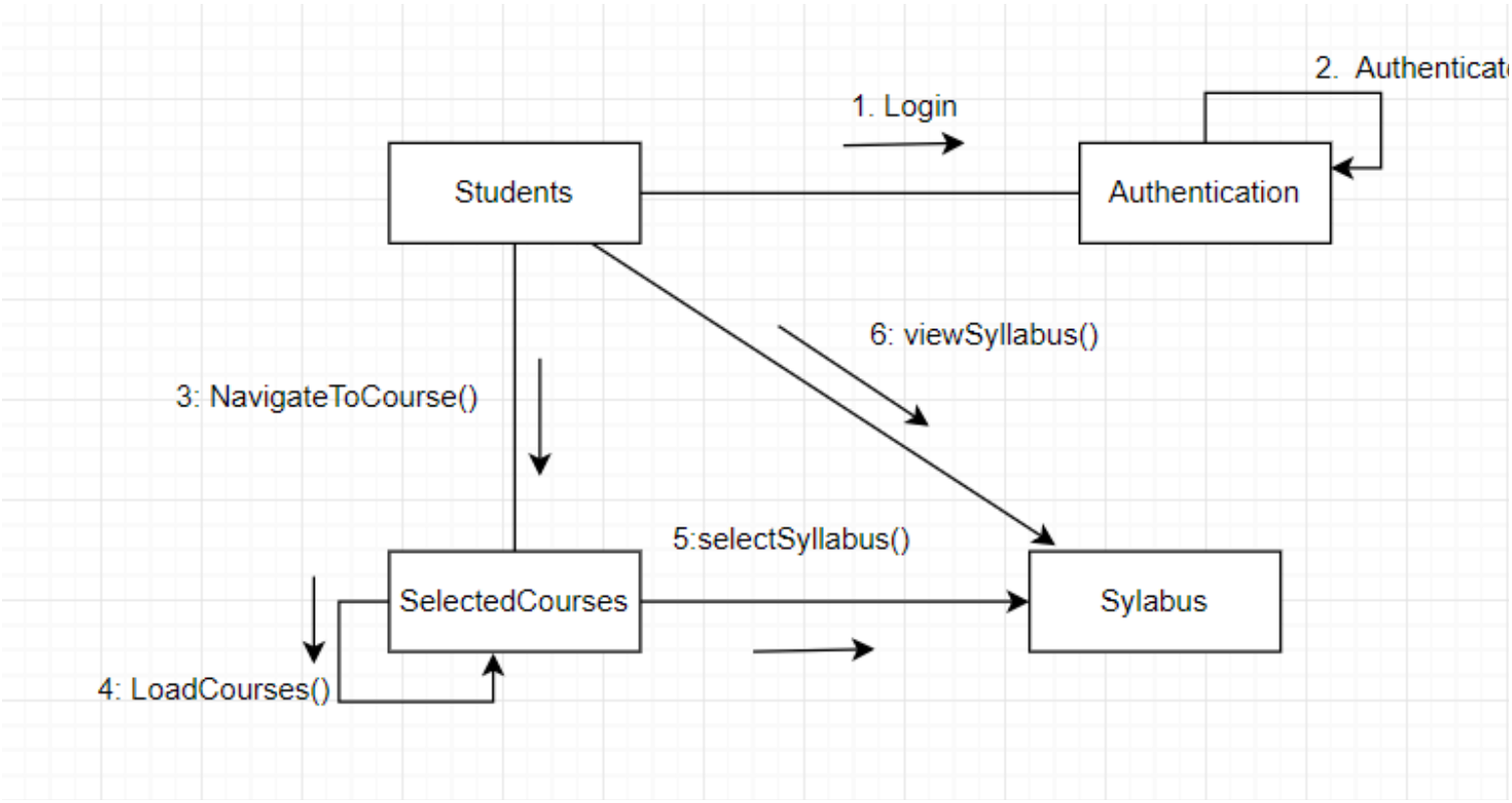
UC02 - View Timetable



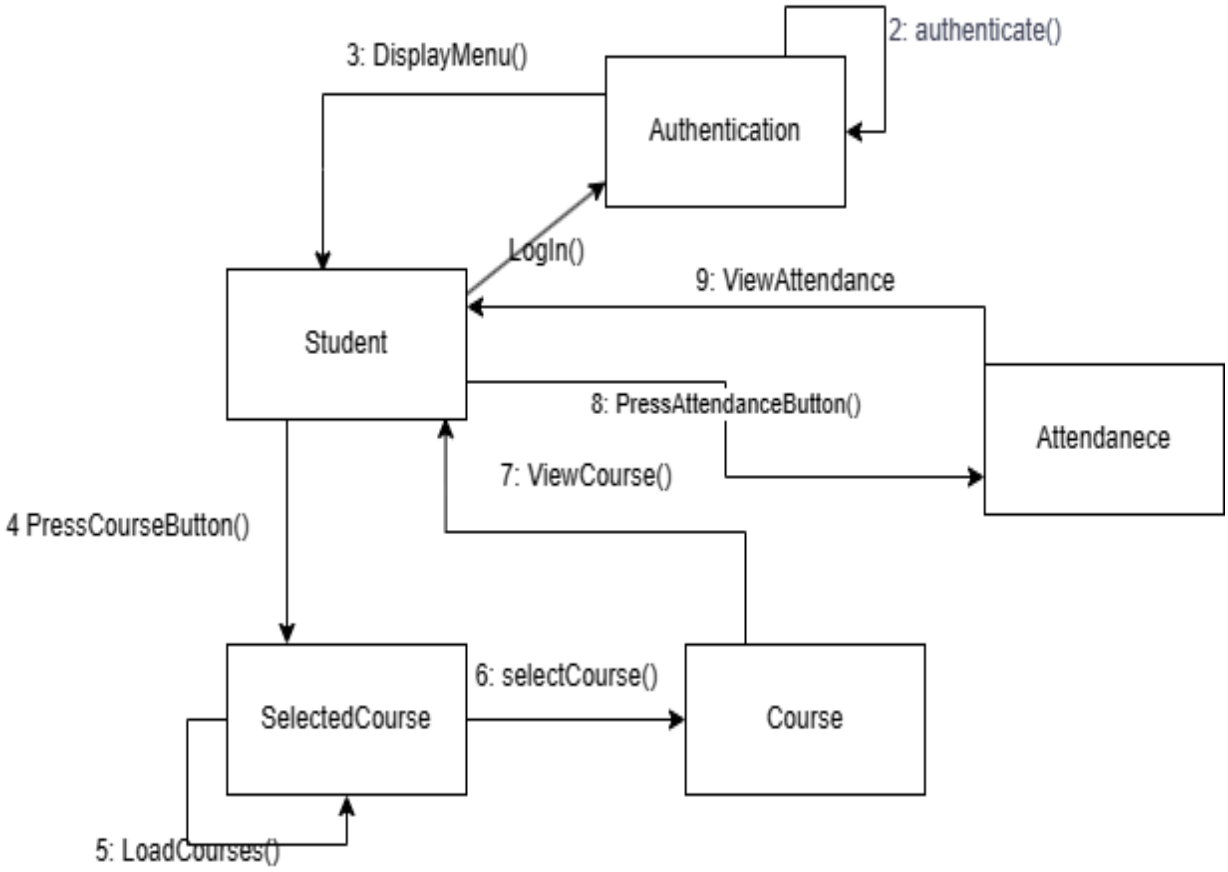
UC03 - ViewTranscript



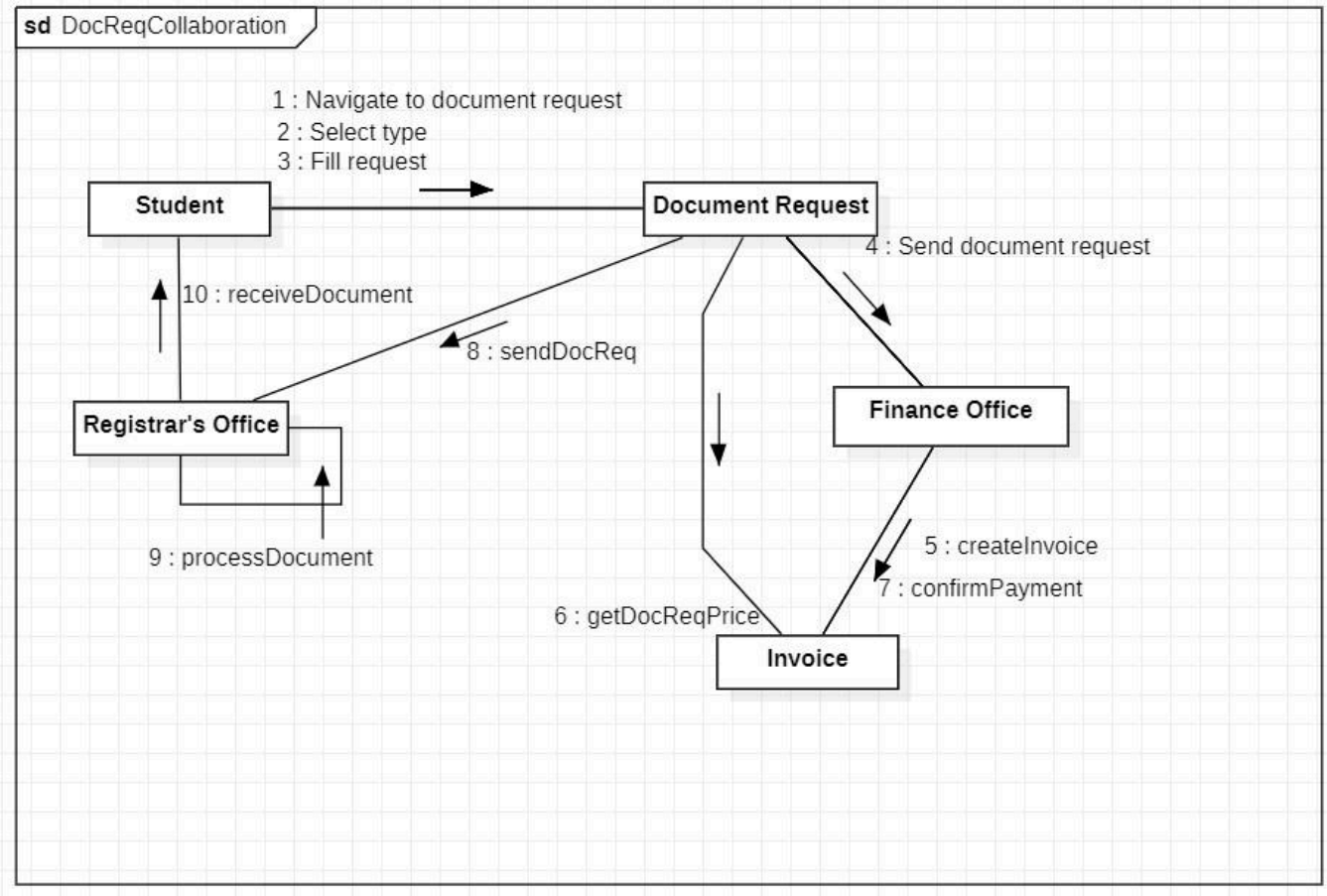
UC04 - ViewGrades



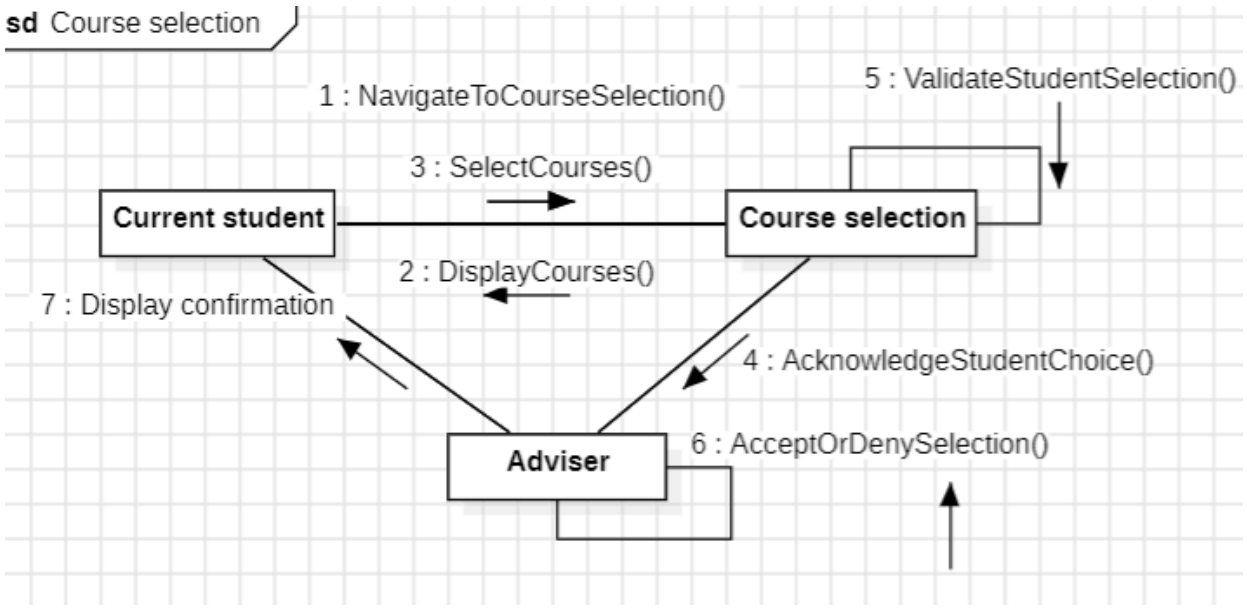
UC05 - View Attendance



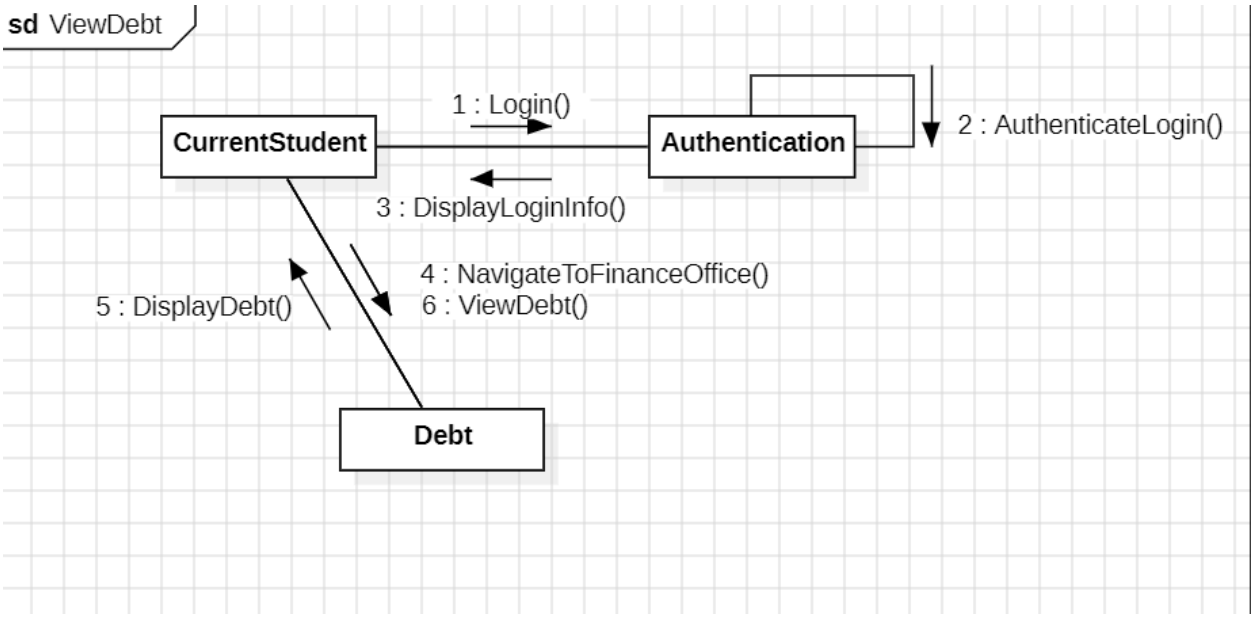
UC06, UC26, UC32 - Document Request



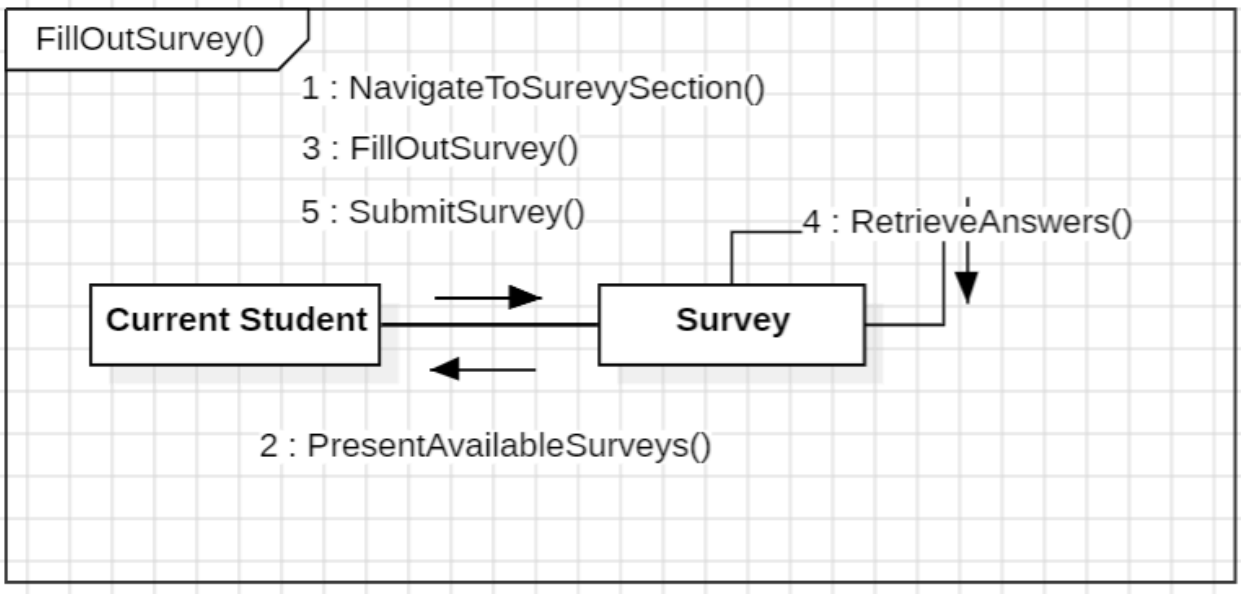
UC07 ,UC20- Course selection



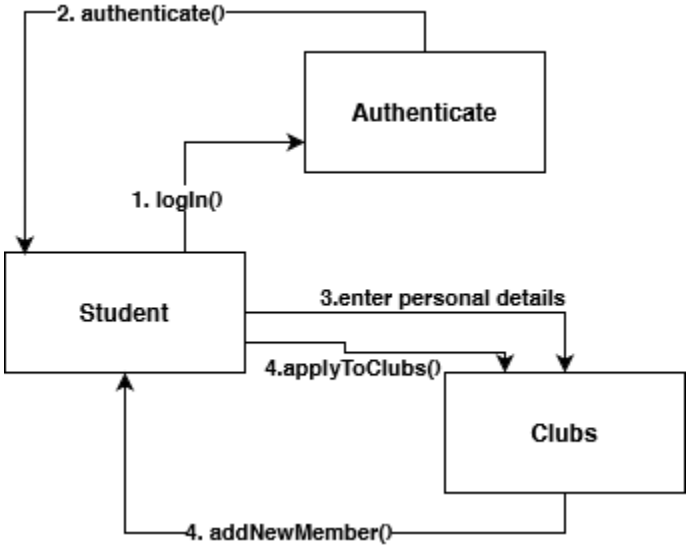
UC08 - View Debt



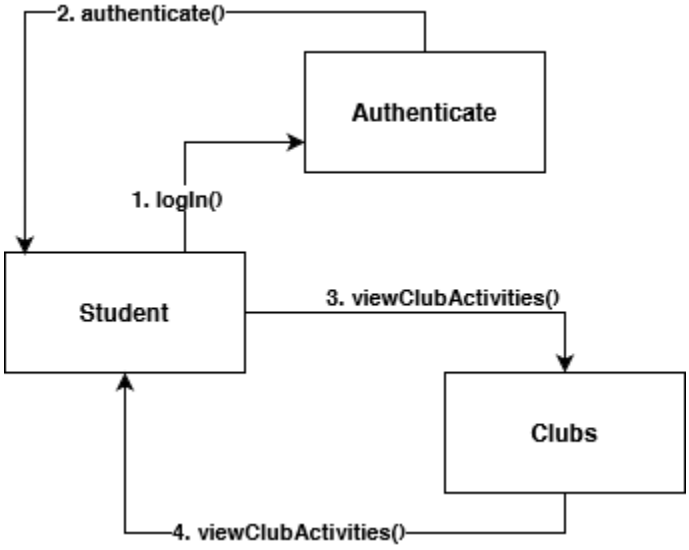
UC09 - Fill out Survey



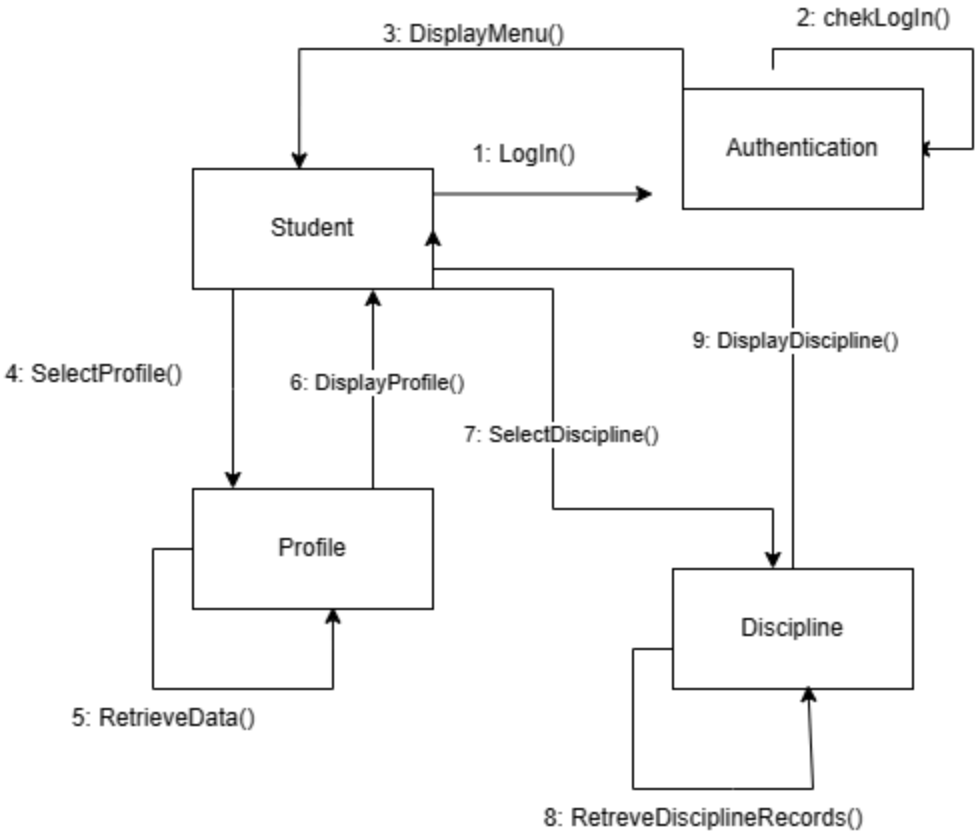
UC10 - Apply to Student Clubs



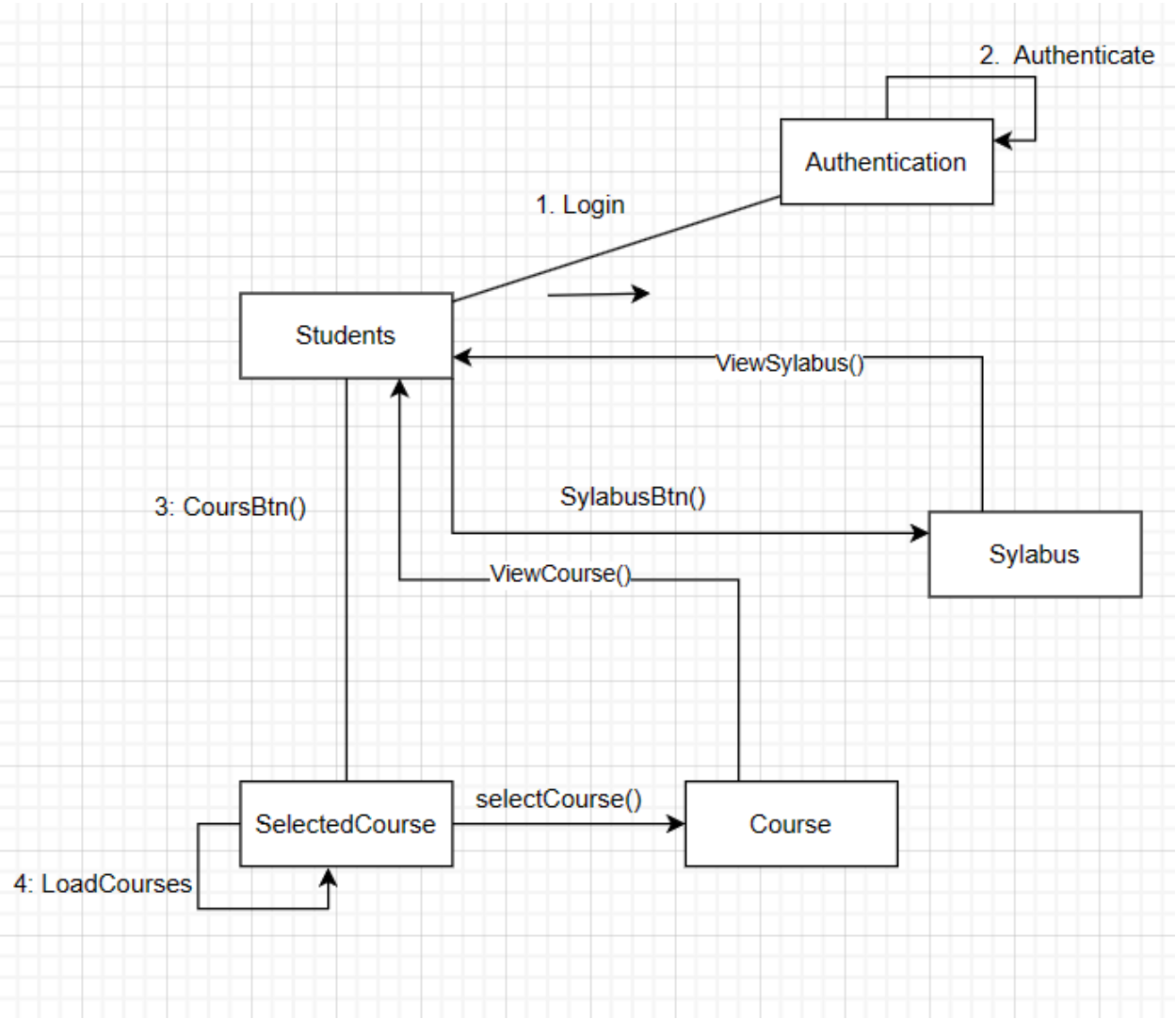
UC11 - View Club Activities



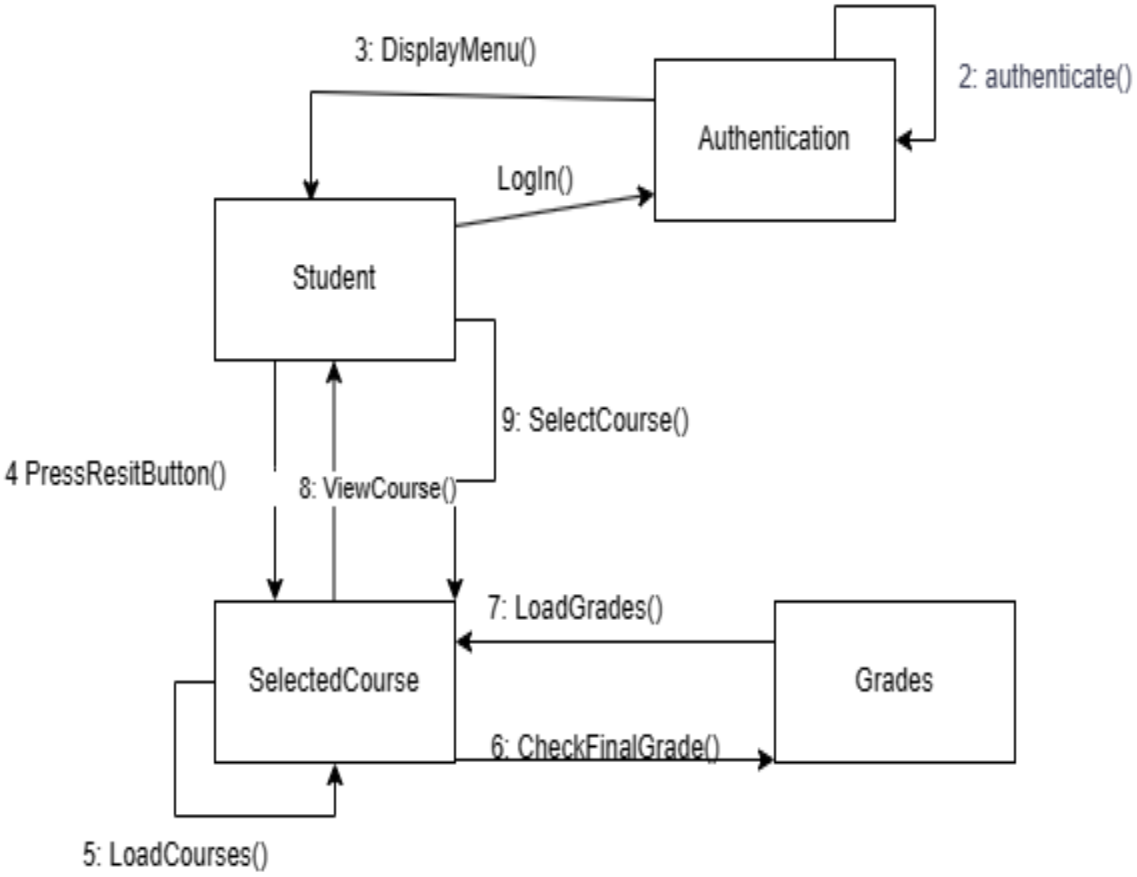
UC12 - View Disciplinary



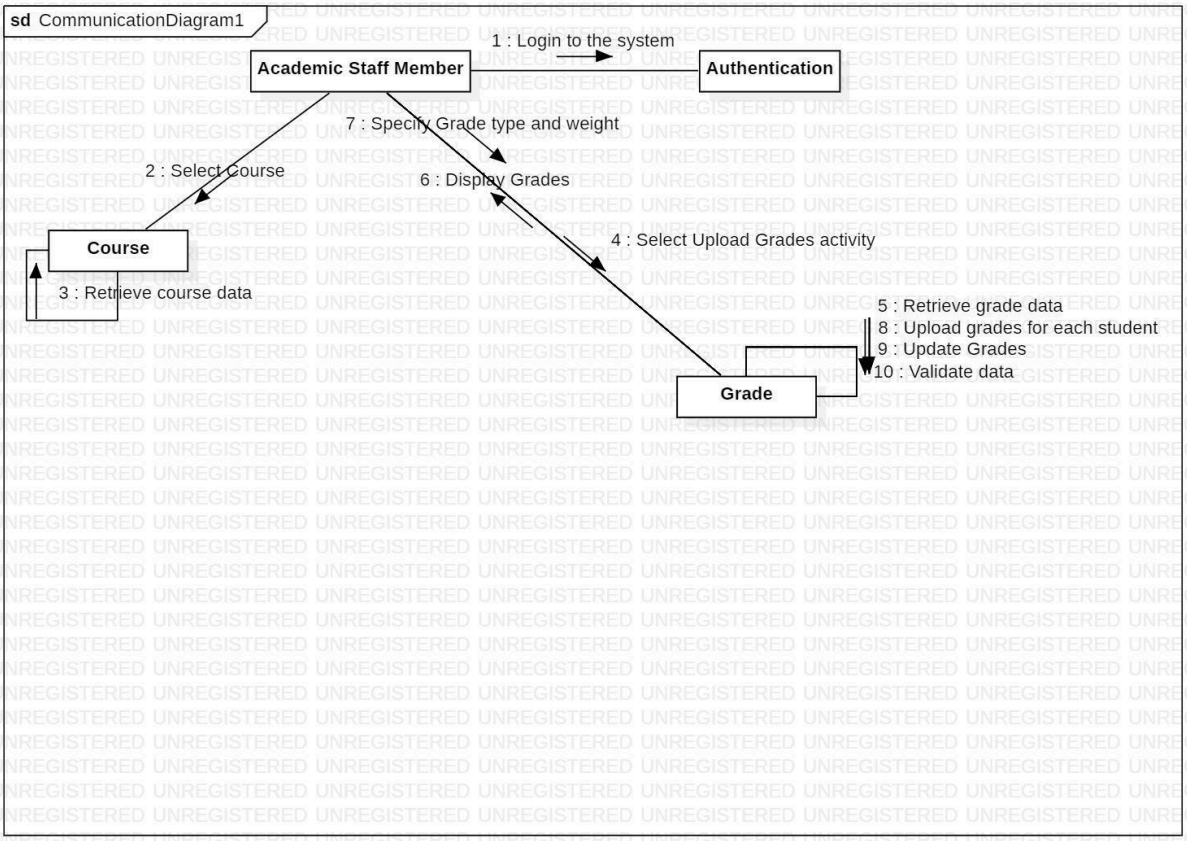
UC13 - View Course Syllabus



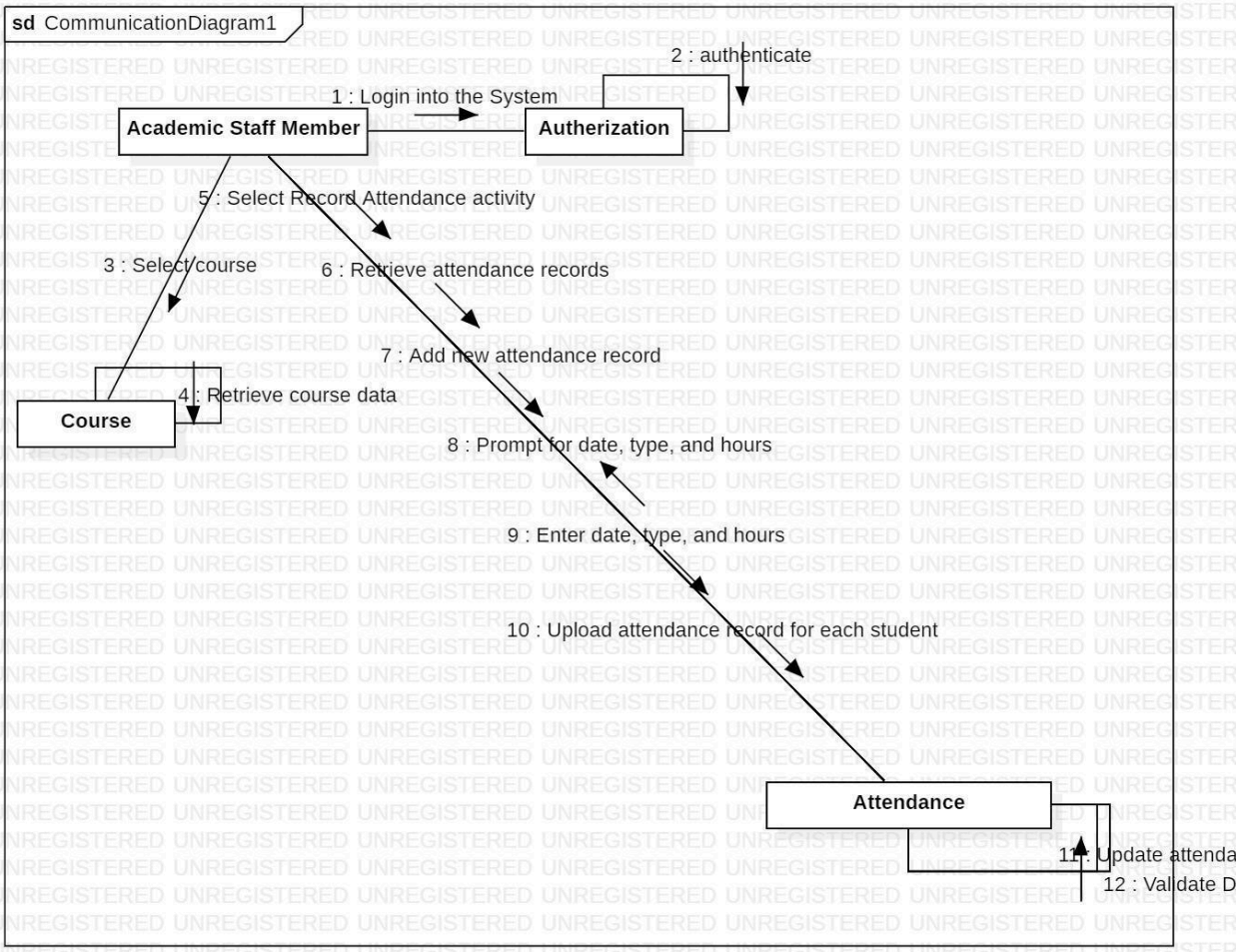
UC14 - Apply for Resit Exam


3: DisplayMenu()
2: authenticate()
Authentication
LogIn()
Student
9: SelectCourse()
4 PressResitButton()
8: ViewCourse()
7: LoadGrades()
SelectedCourse
Grades
6: CheckFinalGrade()
5: LoadCourses()

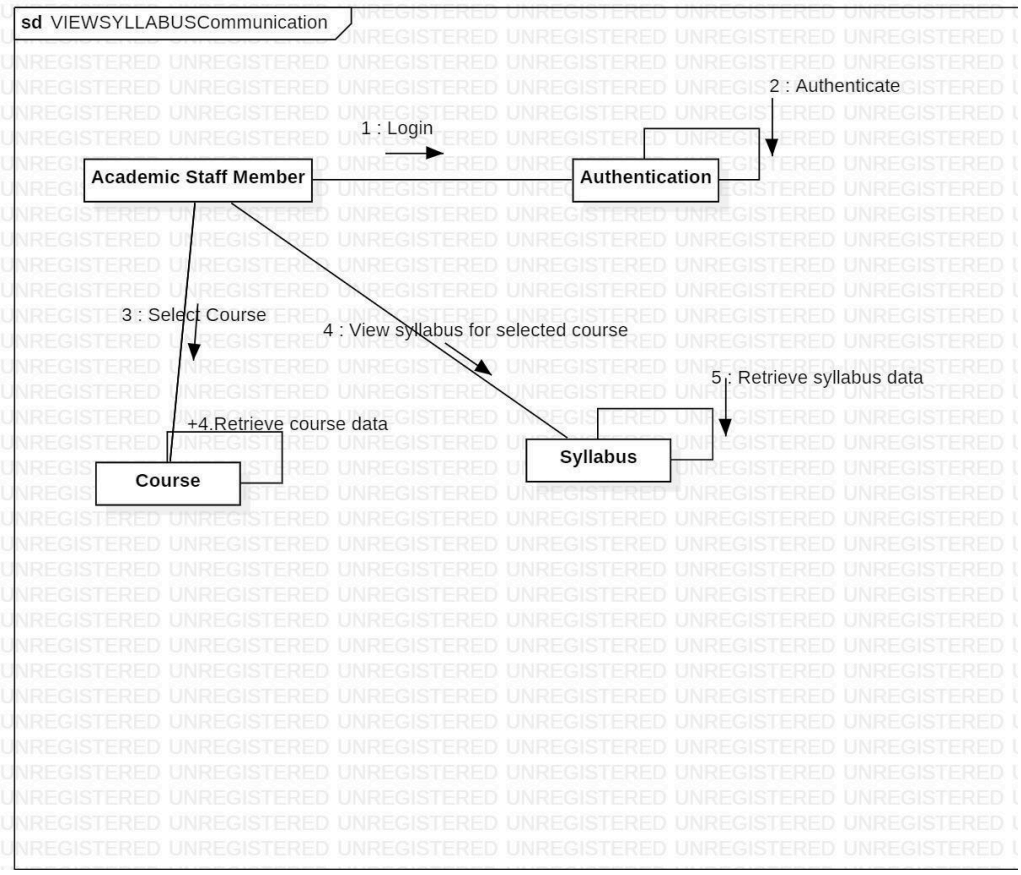
UC15 - Load grades into the system



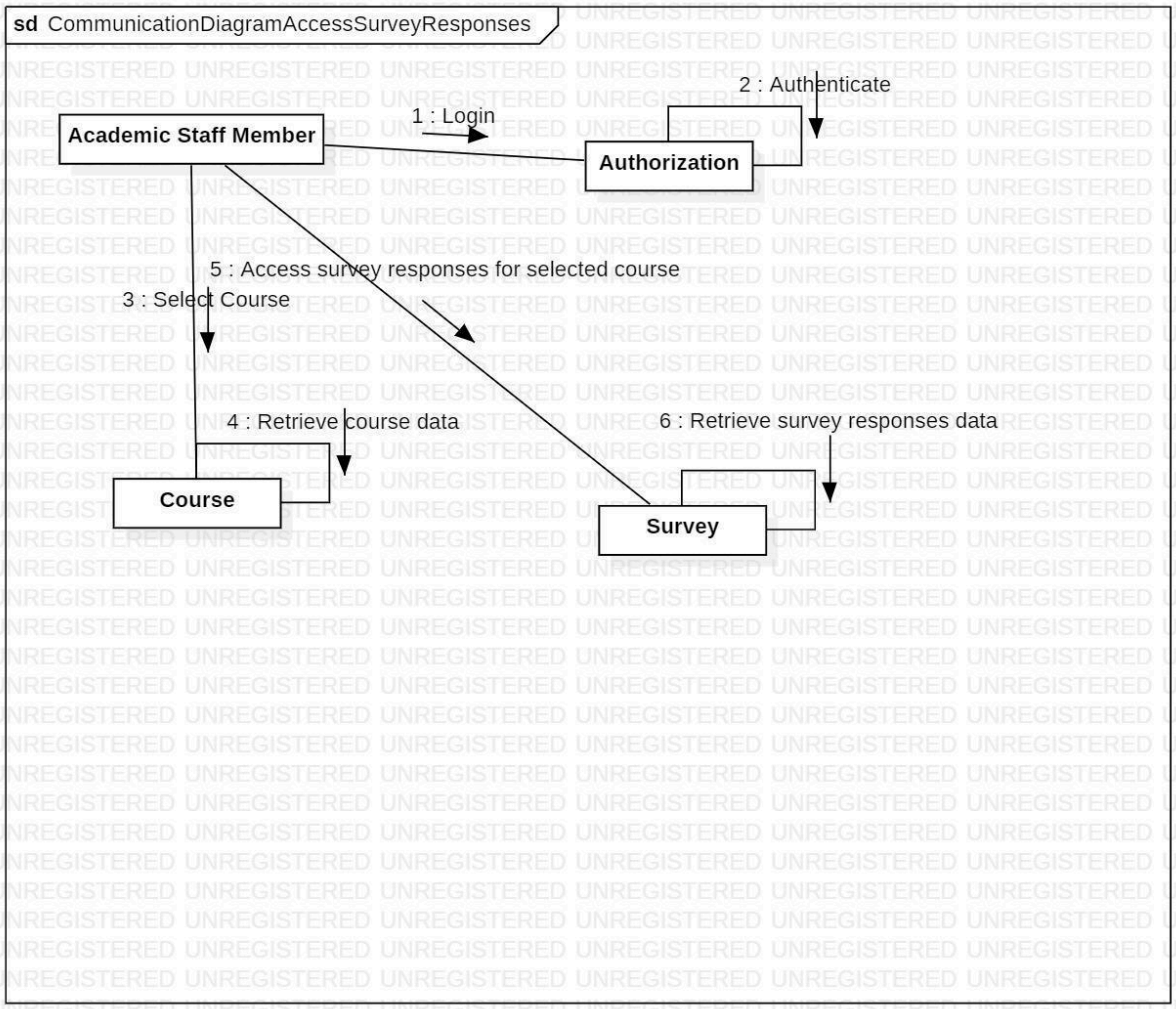
UC16 - Record Attendance of Students



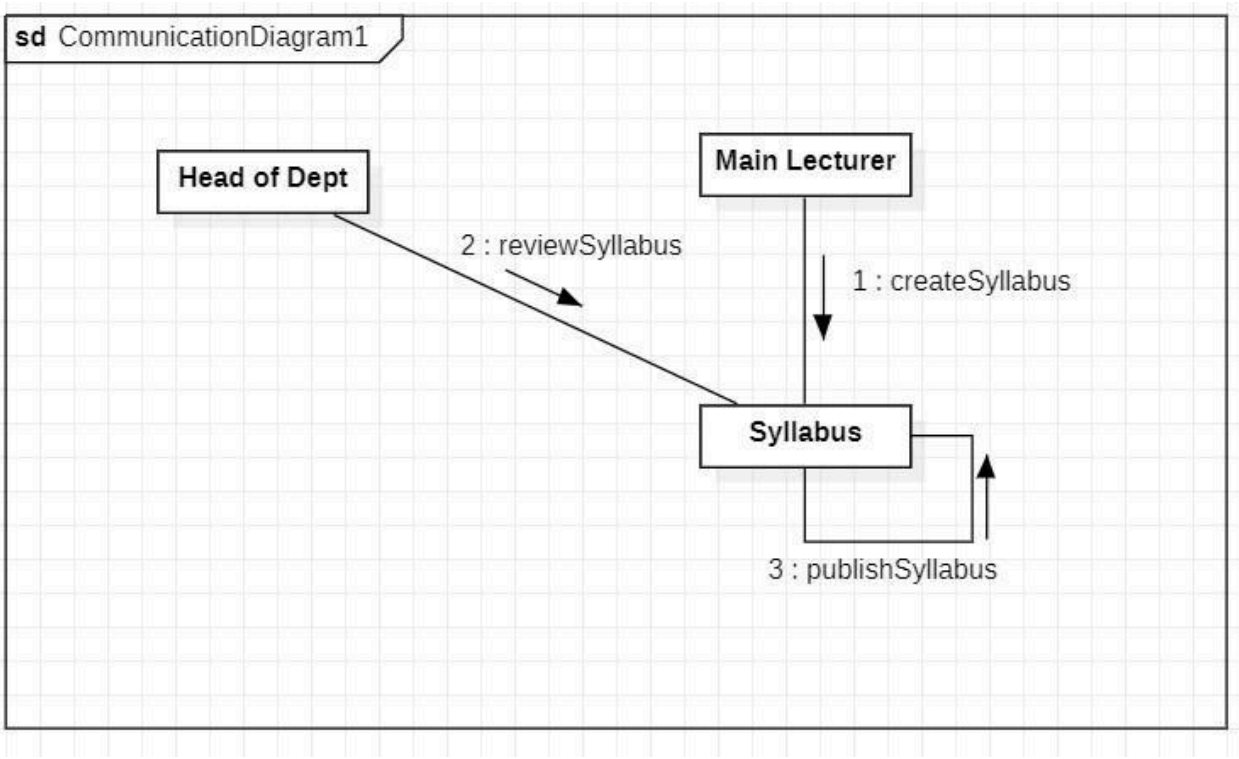
UC17 - View Course Syllabus



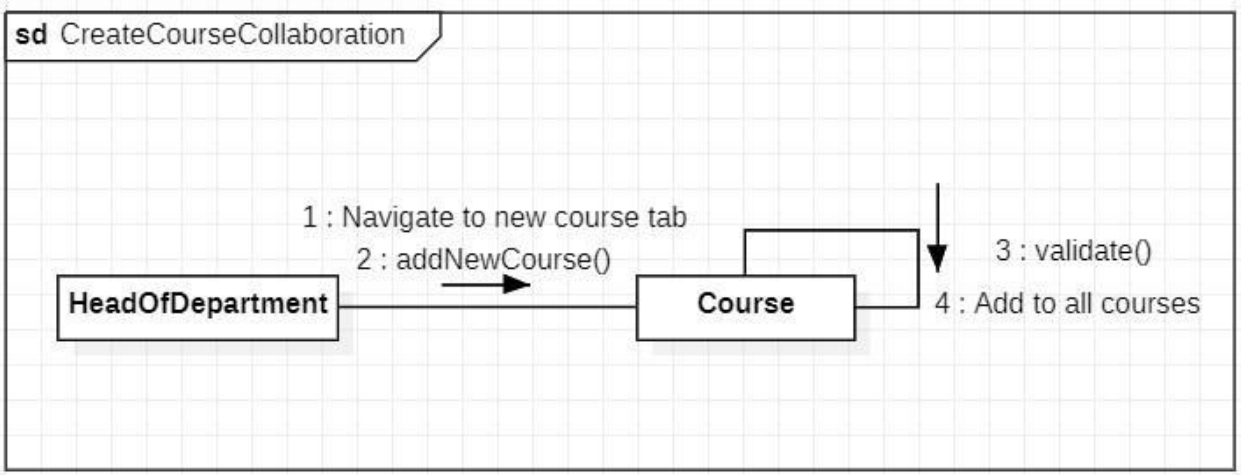
UC18 - Access Survey Responses


sd CommunicationDiagramAccessSurveyResponses
Academic Staff Member
Authorization
Course
Survey
1 : Login
2 : Authenticate
3 : Select Course
4 : Retrieve course data
5 : Access survey responses for selected course
6 : Retrieve survey responses data

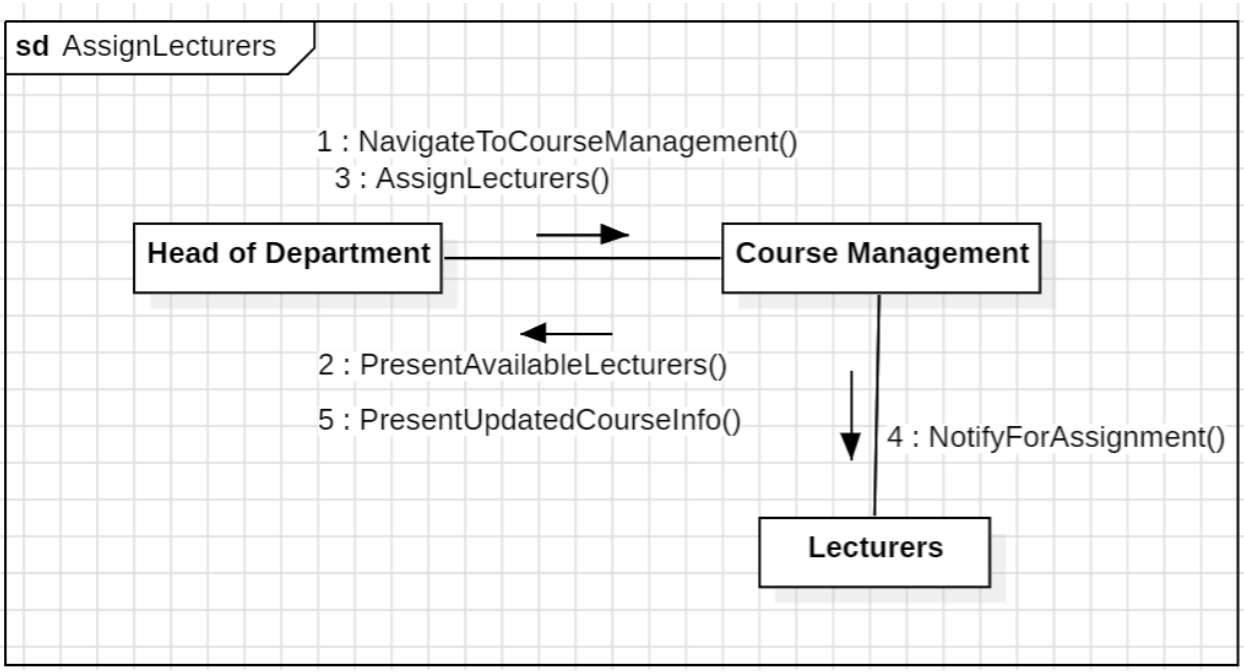
UC19,UC21 - Syllabus publishing



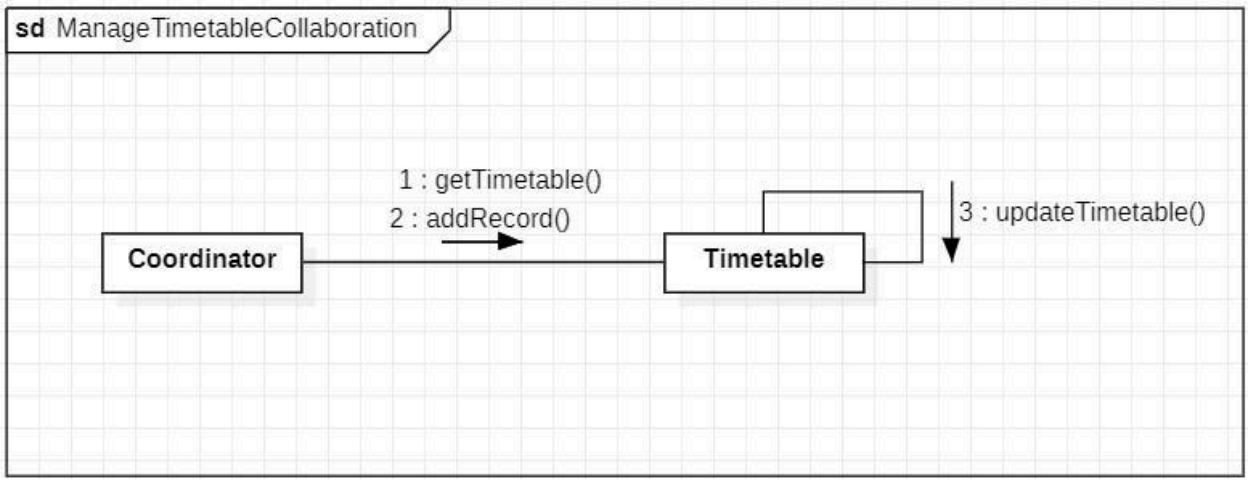
UC22 - Create Course



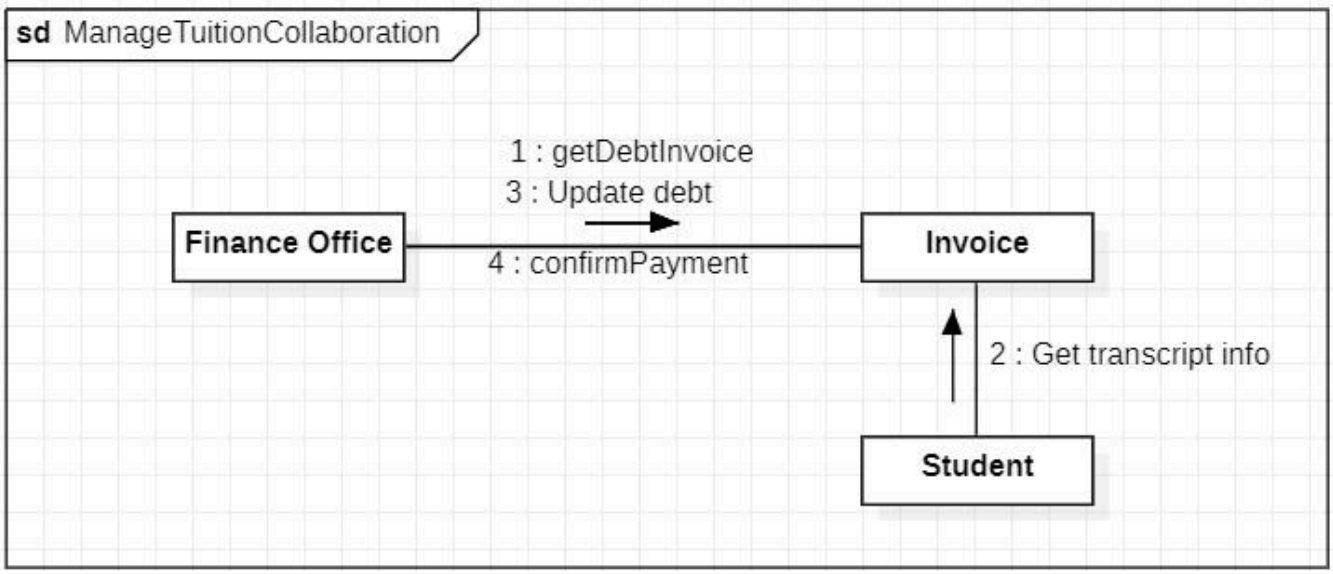
UC23 - Assigning lecturers to courses



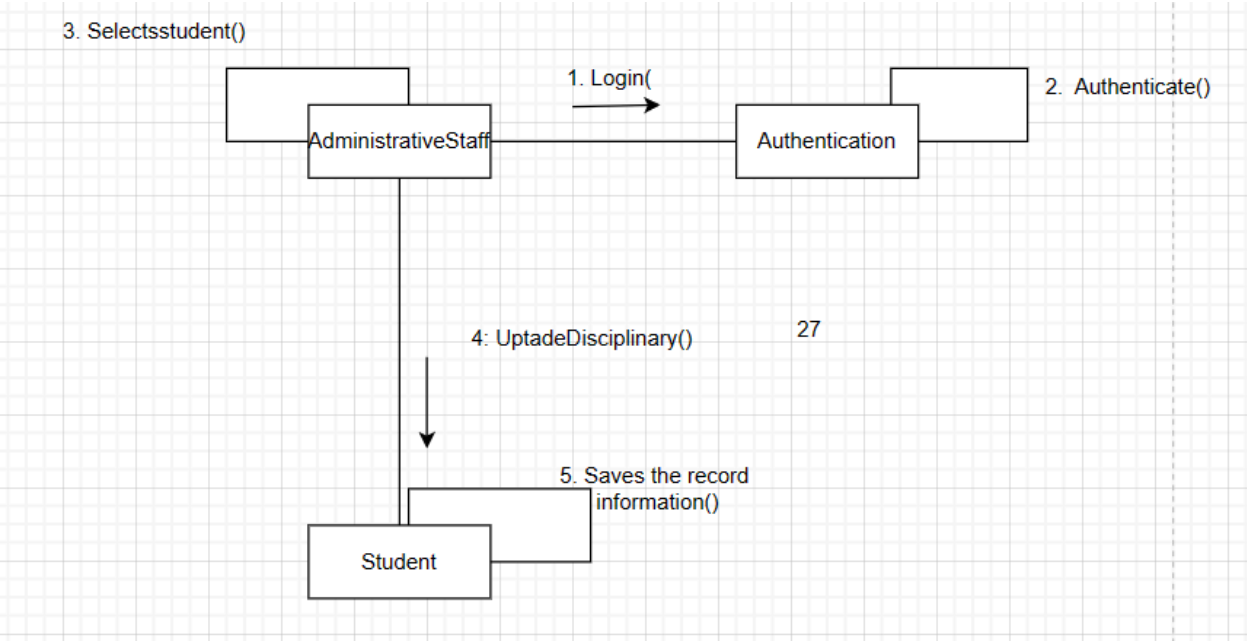
UC24 - Upload timetable



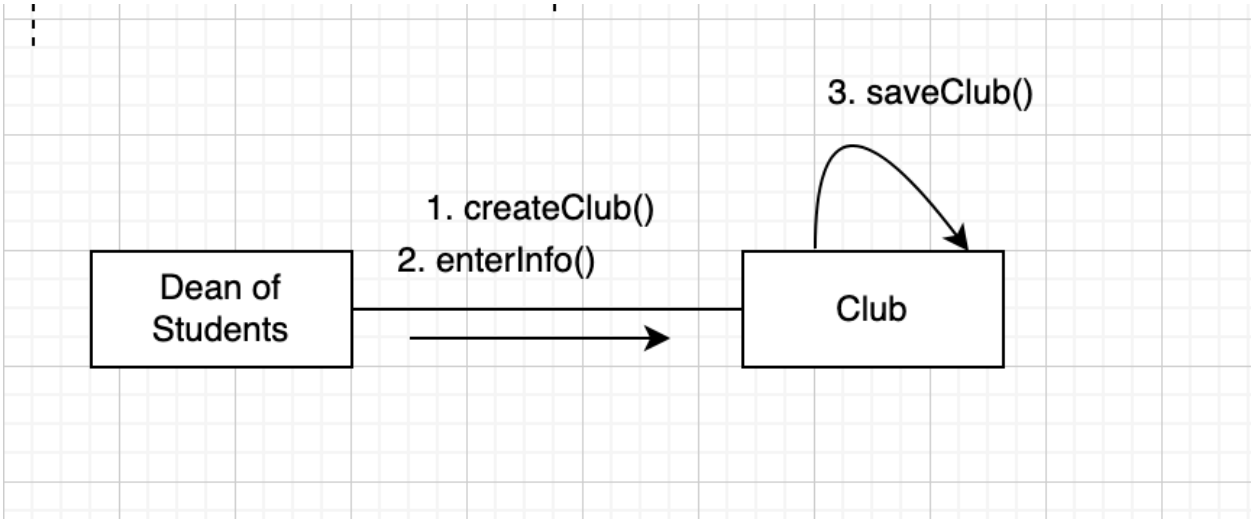
UC25 - Manage tuition fee



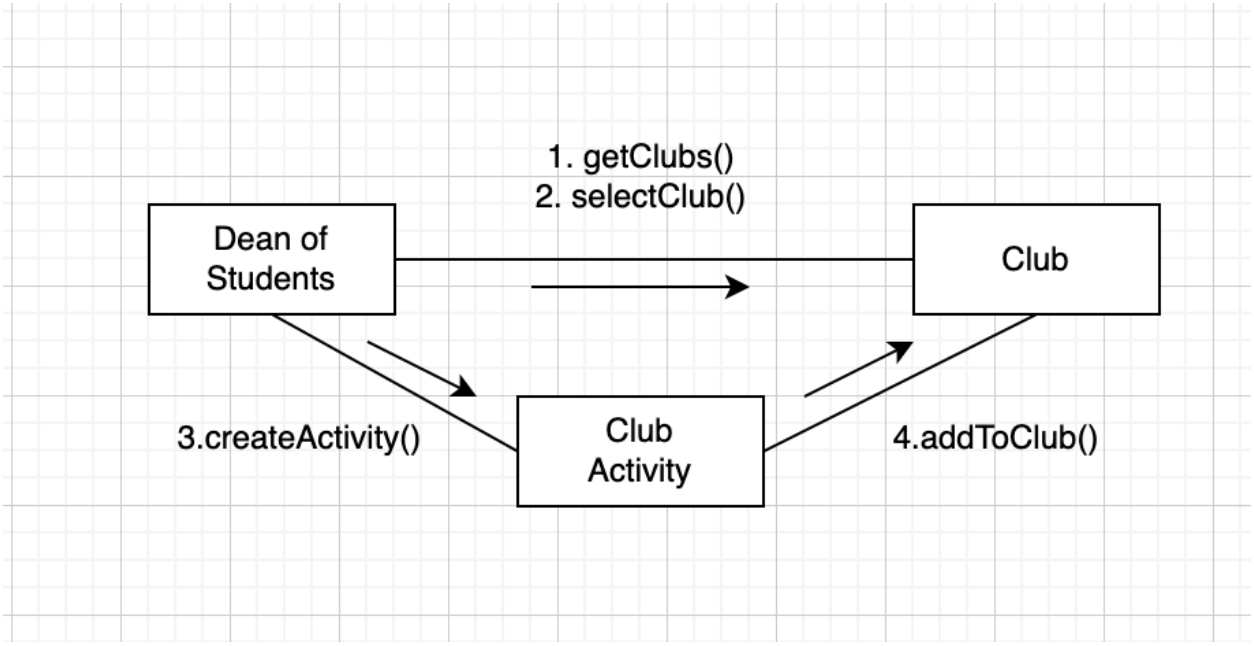
UC27- Record disciplinary cases



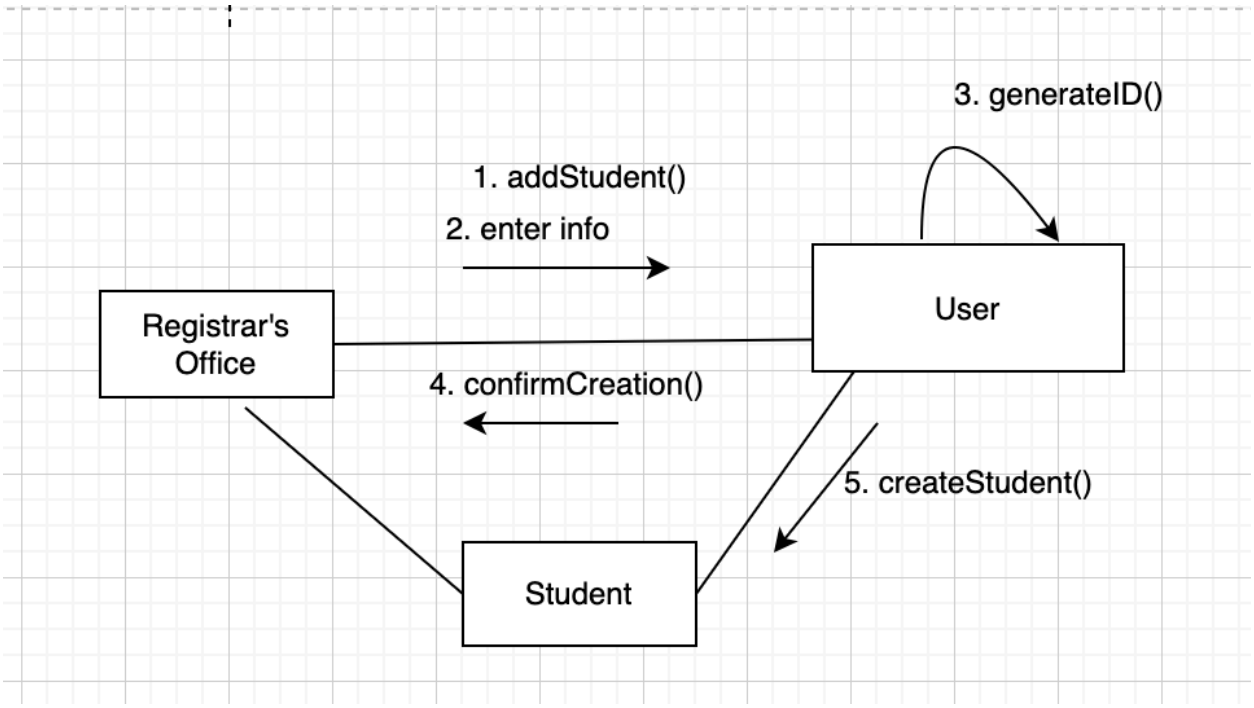
UC28 - Create Student Clubs



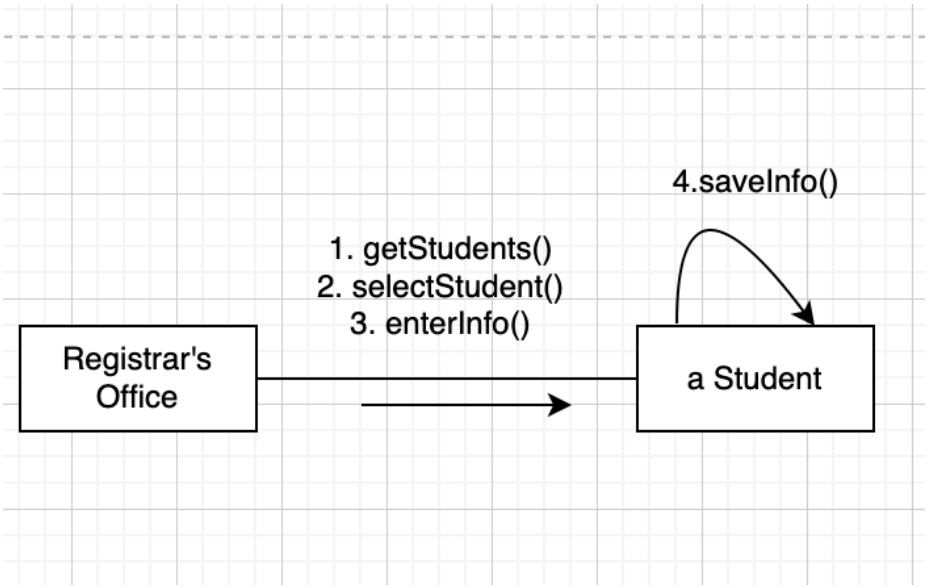
UC29 - Post Student Club Activities



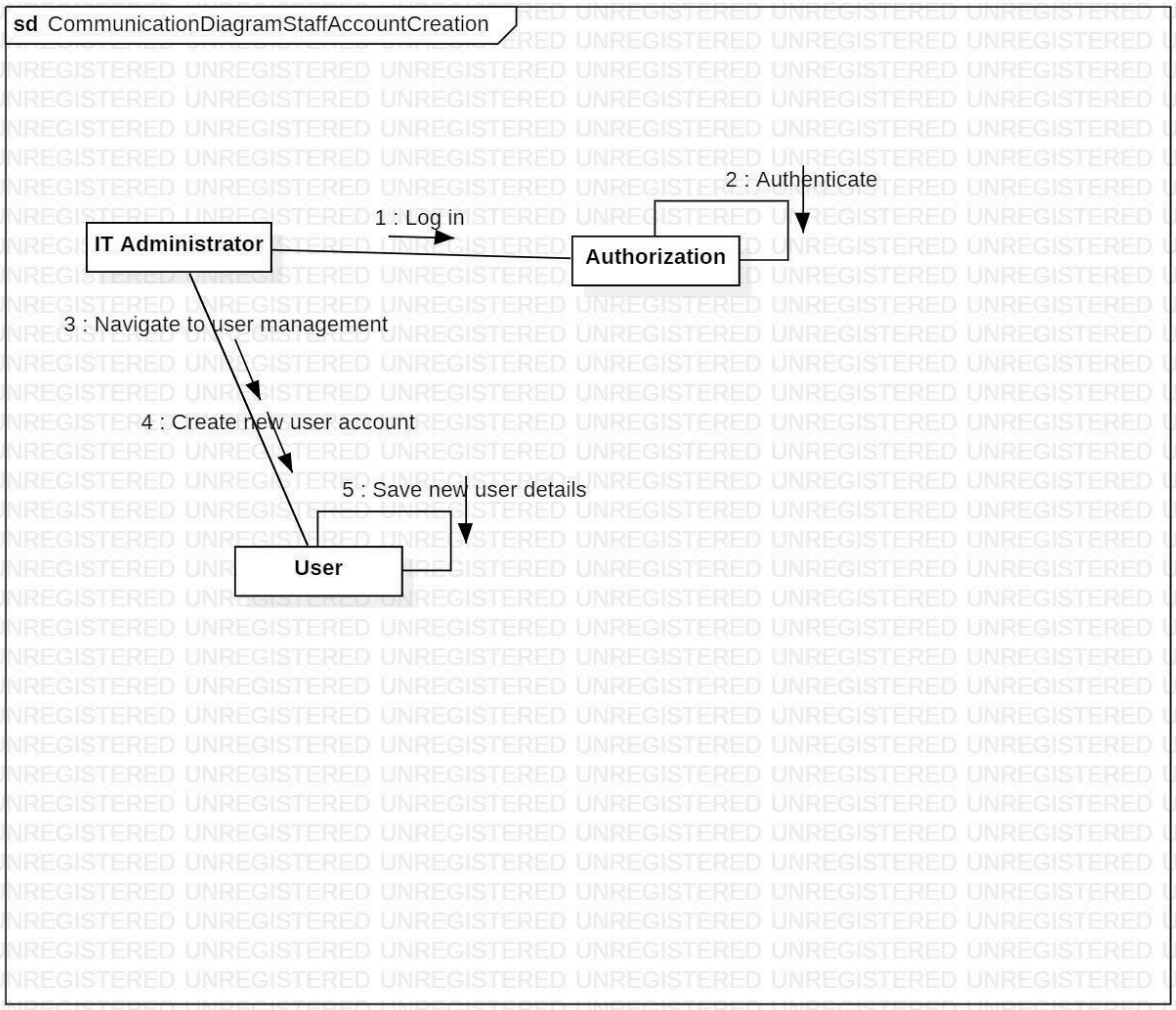
UC30 - Create new Student Account



UC31 - Update/Edit student information



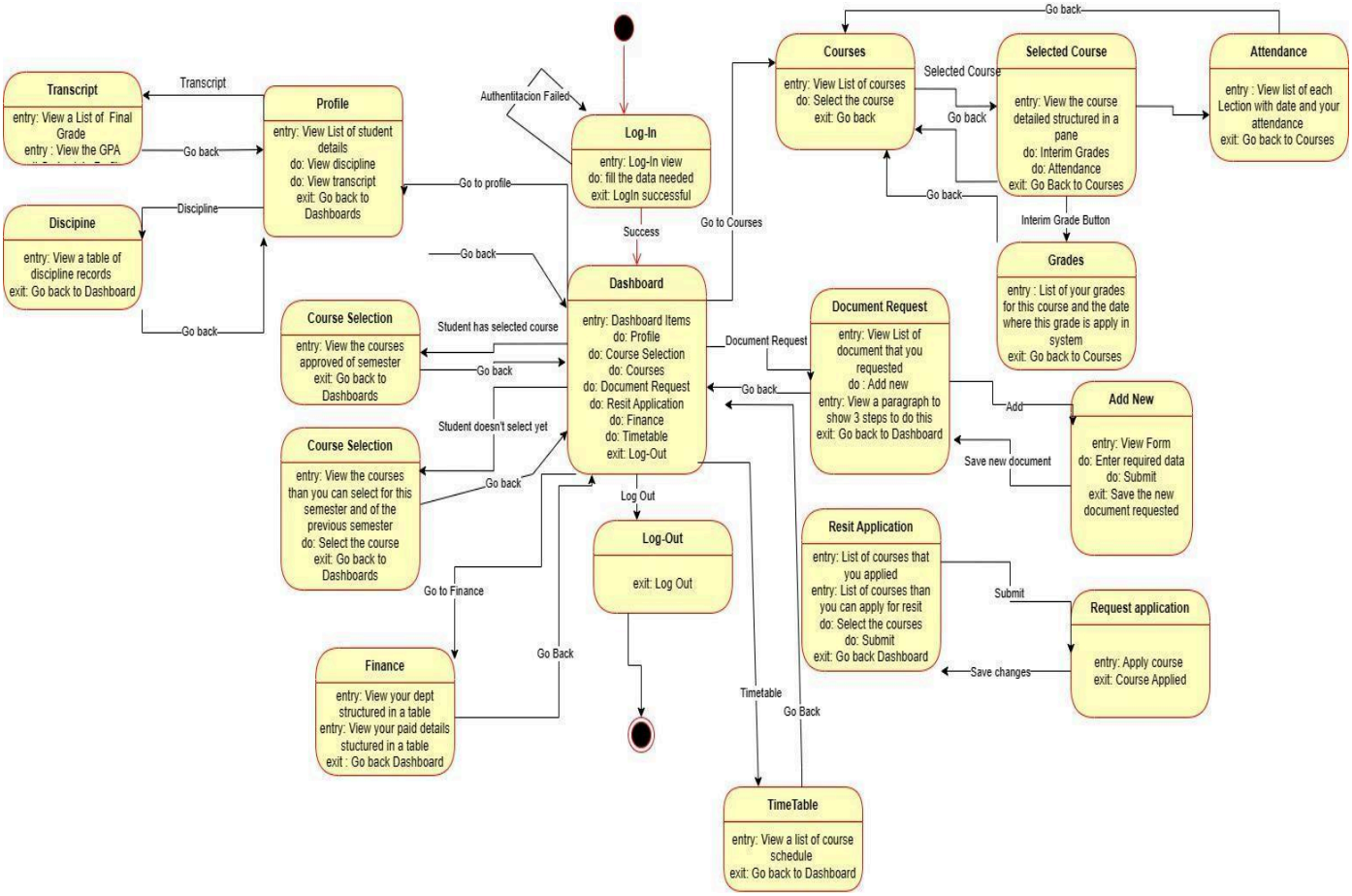
UC33 - Create Academic Staff Account



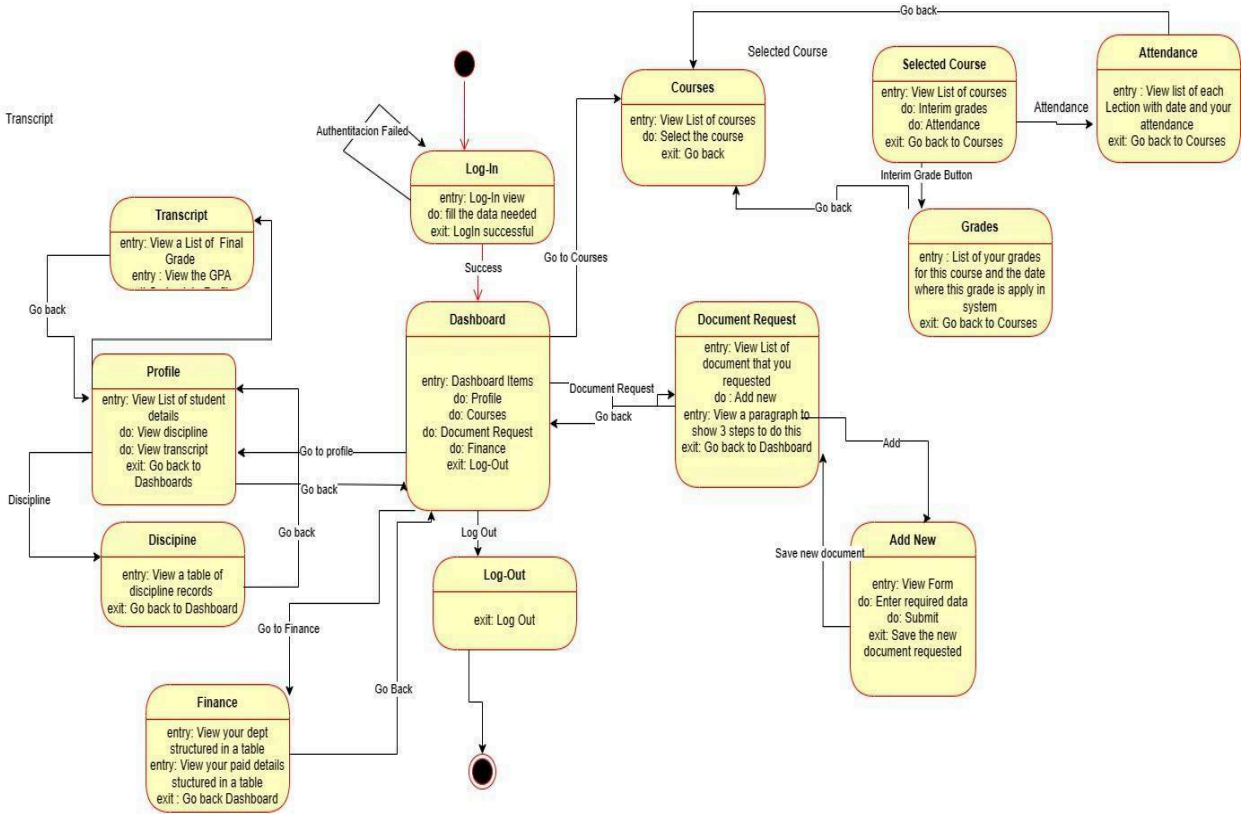
State diagrams

State Diagrams by Levels of Users

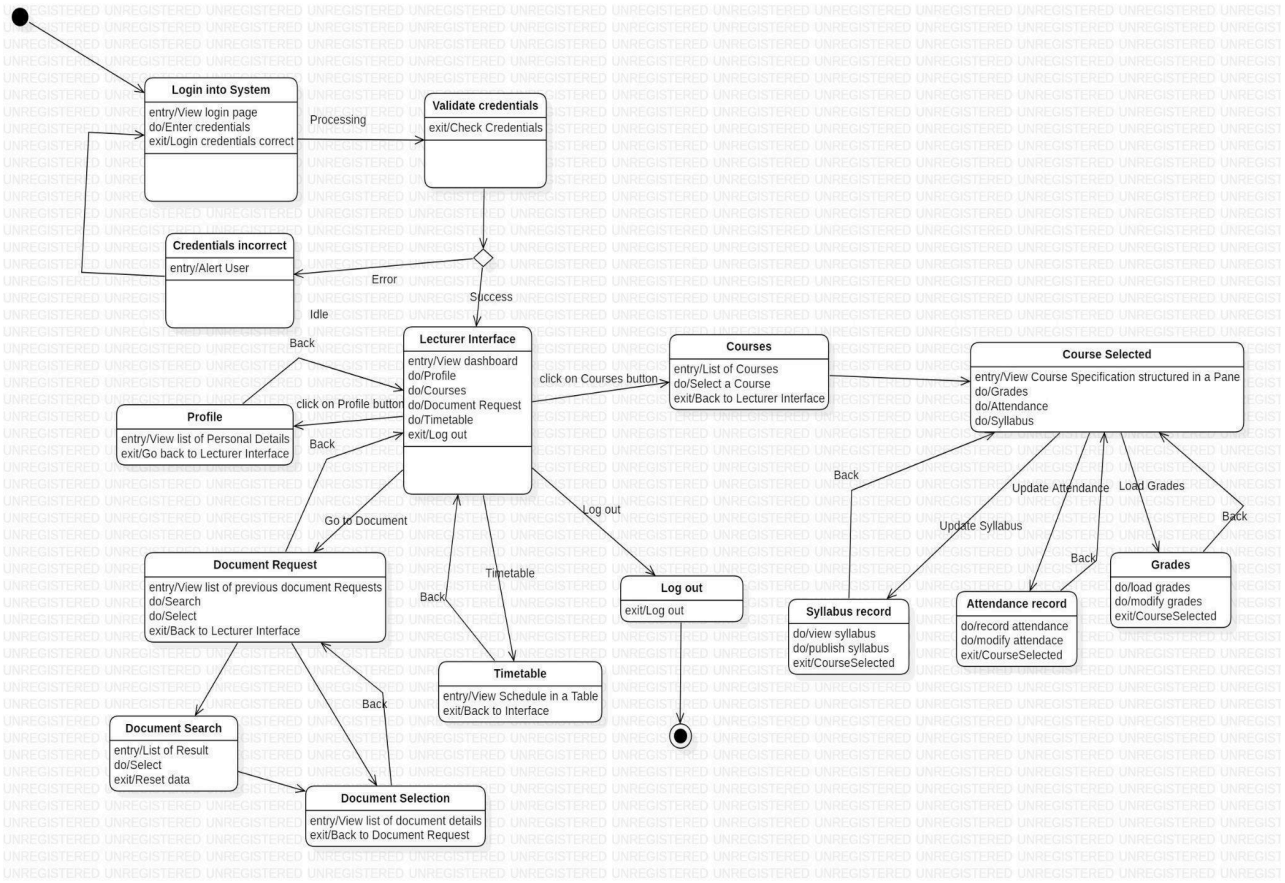
Students



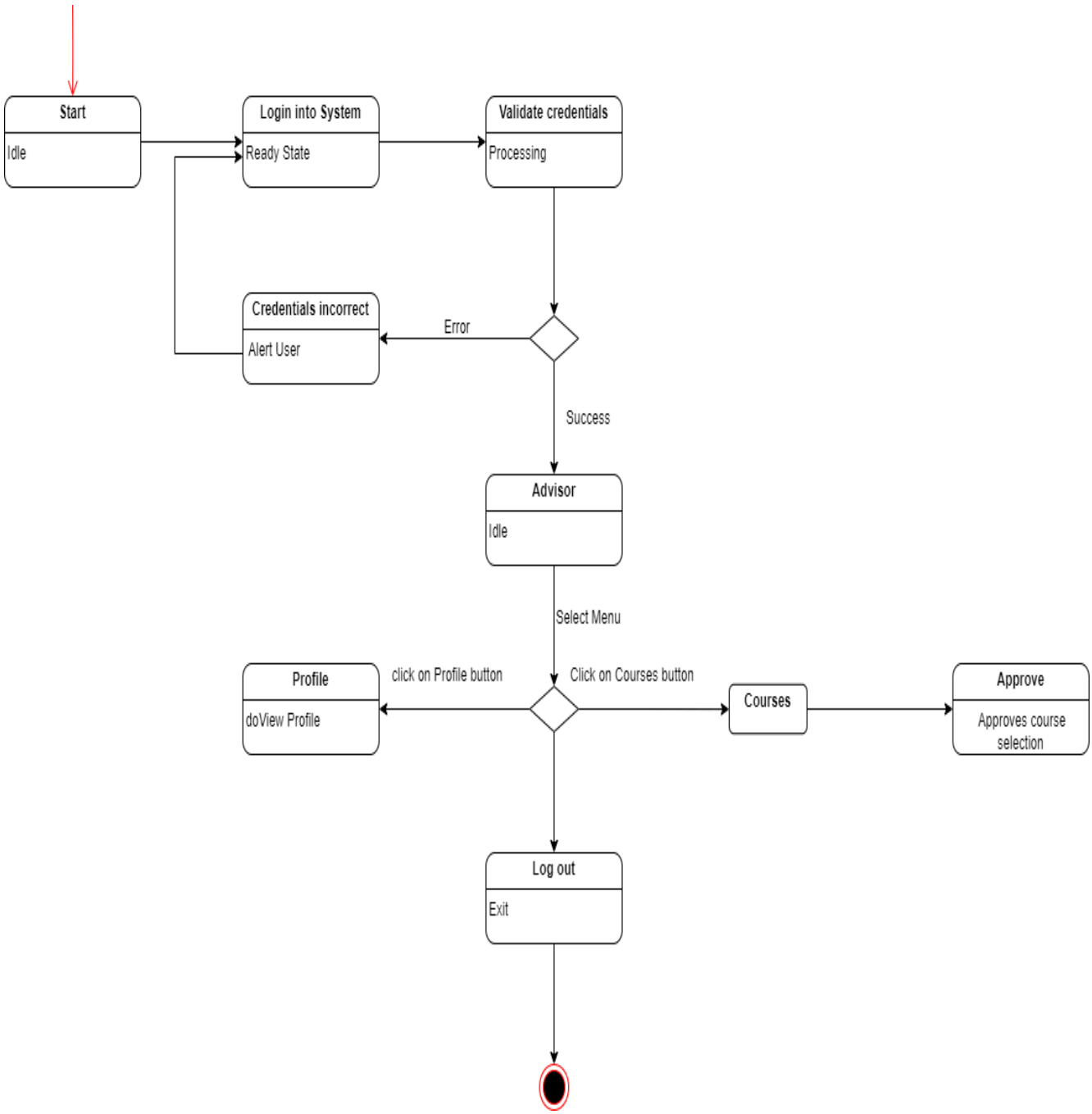
Alumni


Transcript
Authentitacion Failed
Log-In
entry: Log-In view
do: fill the data needed
exit: LogIn successful
Success
Courses
entry: View List of courses
do: Select the course
exit: Go back
Go back
Selected Course
Selected Course
entry: View List of courses
do: Interim grades
do: Attendance
exit: Go back to Courses
Attendance
Attendance
entry : View list of each Lection with date and your attendance
exit: Go back to Courses
Interim Grade Button
Go back
Grades
entry : List of your grades for this course and the date where this grade is apply in system
exit: Go back to Courses
Go to Courses
Transcript
entry: View a List of Final Grade
entry : View the GPA
Go back
Dashboard
entry: Dashboard Items
do: Profile
do: Courses
do: Document Request
do: Finance
exit: Log-Out
Document Request
Go back
Document Request
entry: View List of document that you requested
do : Add new
entry: View a paragraph to show 3 steps to do this
exit: Go back to Dashboard
Add
Profile
entry: View List of student details
do: View discipline
do: View transcript
exit: Go back to Dashboards
Go to profile
Go back
Discipline
Discipine
entry: View a table of discipline records
exit: Go back to Dashboard
Go back
Log Out
Log-Out
exit: Log Out
Save new document
Add New
entry: View Form
do: Enter required data
do: Submit
exit: Save the new document requested
Go to Finance
Go Back
Finance
entry: View your dept structured in a table
entry: View your paid details stuctured in a table
exit : Go back Dashboard

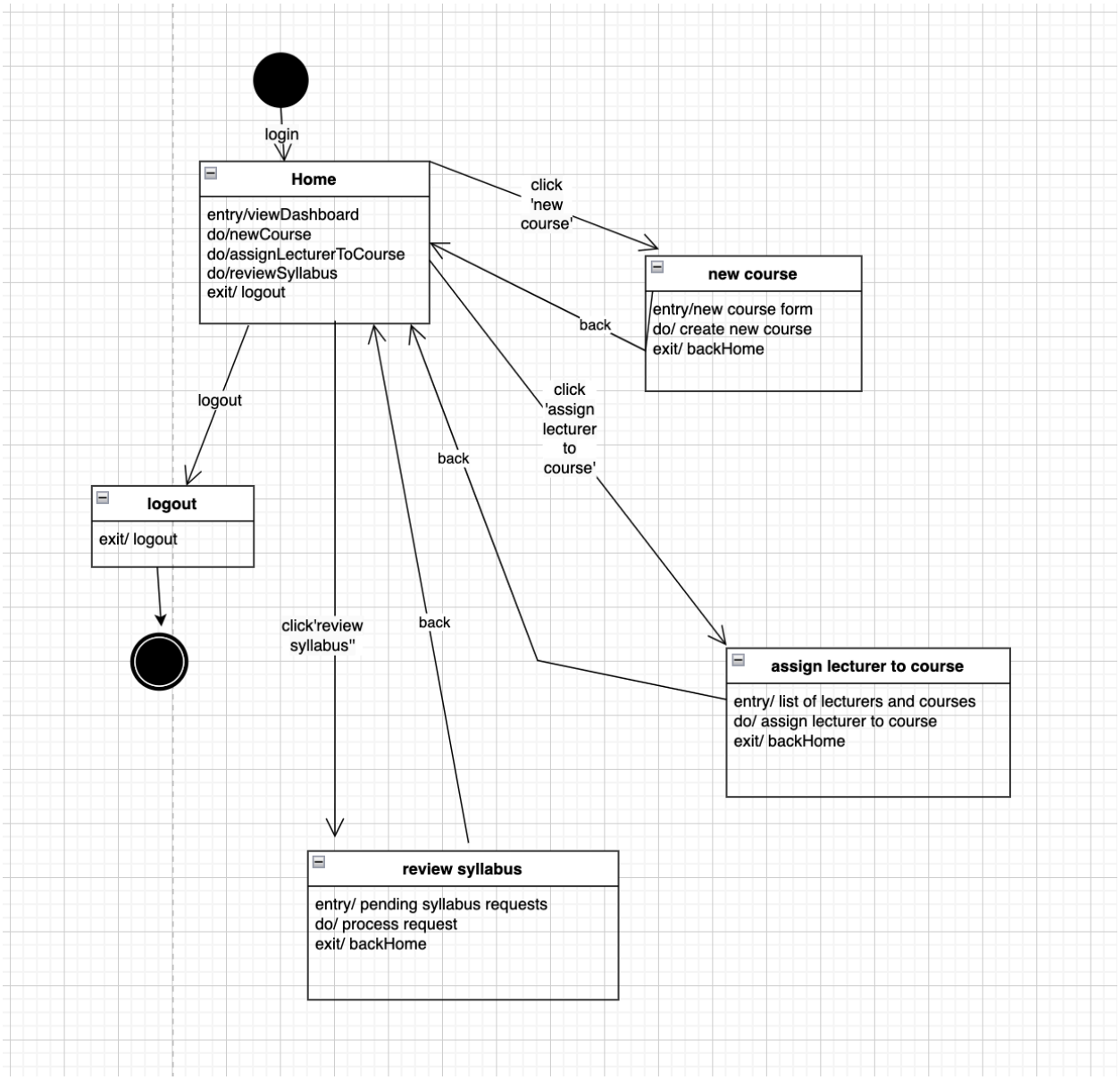
Main Lecturer/Assistant Lecturer



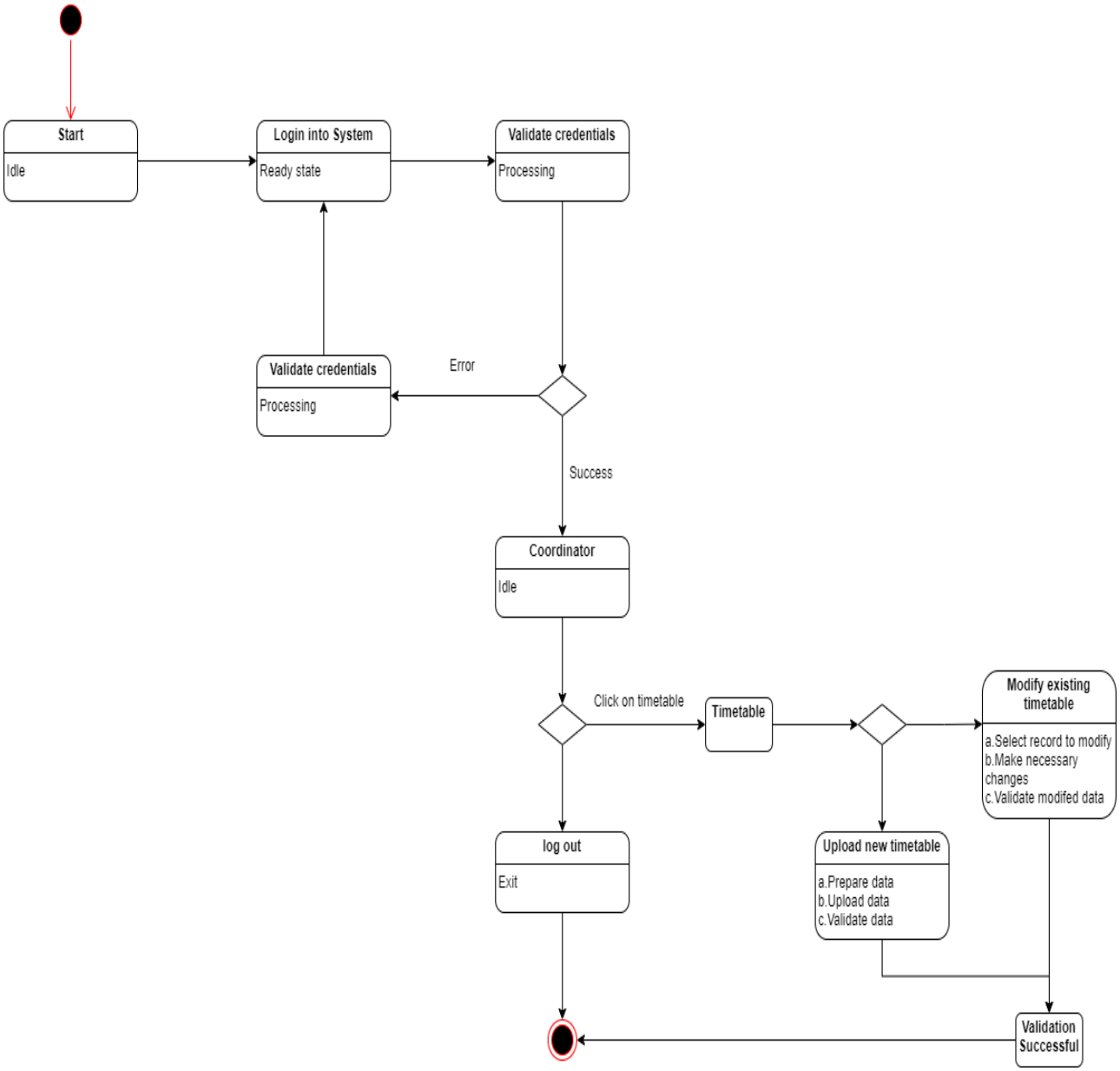
Advisor


Start
Idle
Login into System
Ready State
Validate credentials
Processing
Credentials incorrect
Alert User
Error
Success
Advisor
Idle
Select Menu
Profile
doView Profile
click on Profile button
Click on Courses button
Courses
Approve
Approves course selection
Log out
Exit

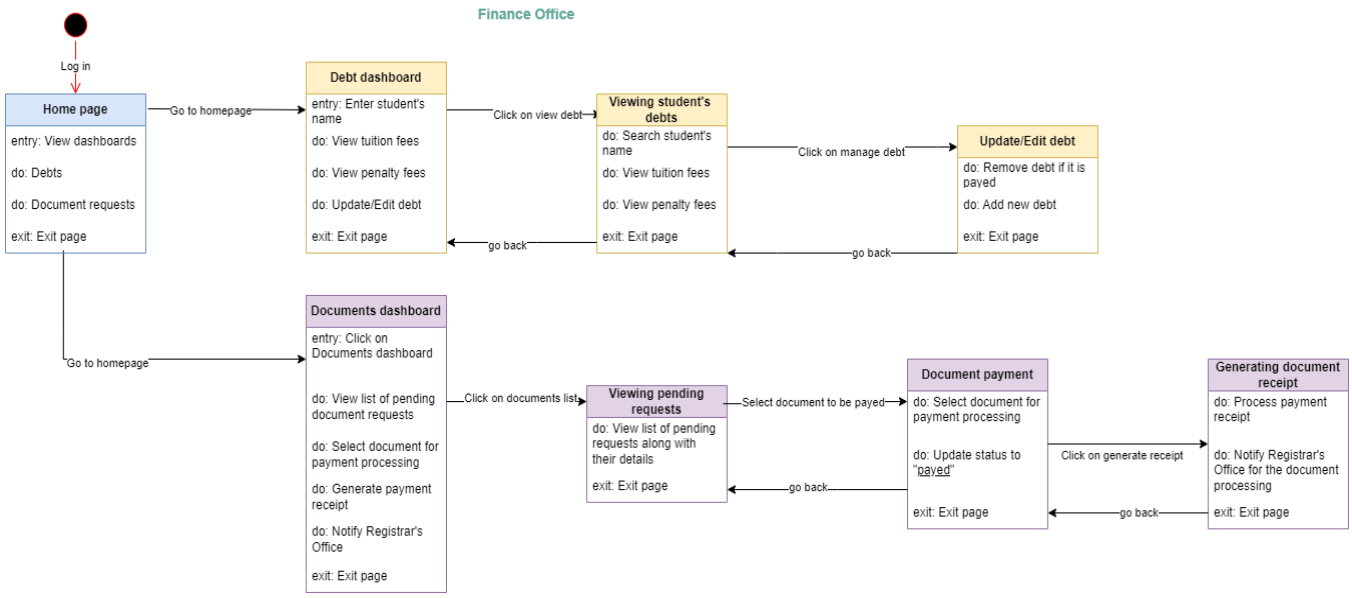
Head of Department:



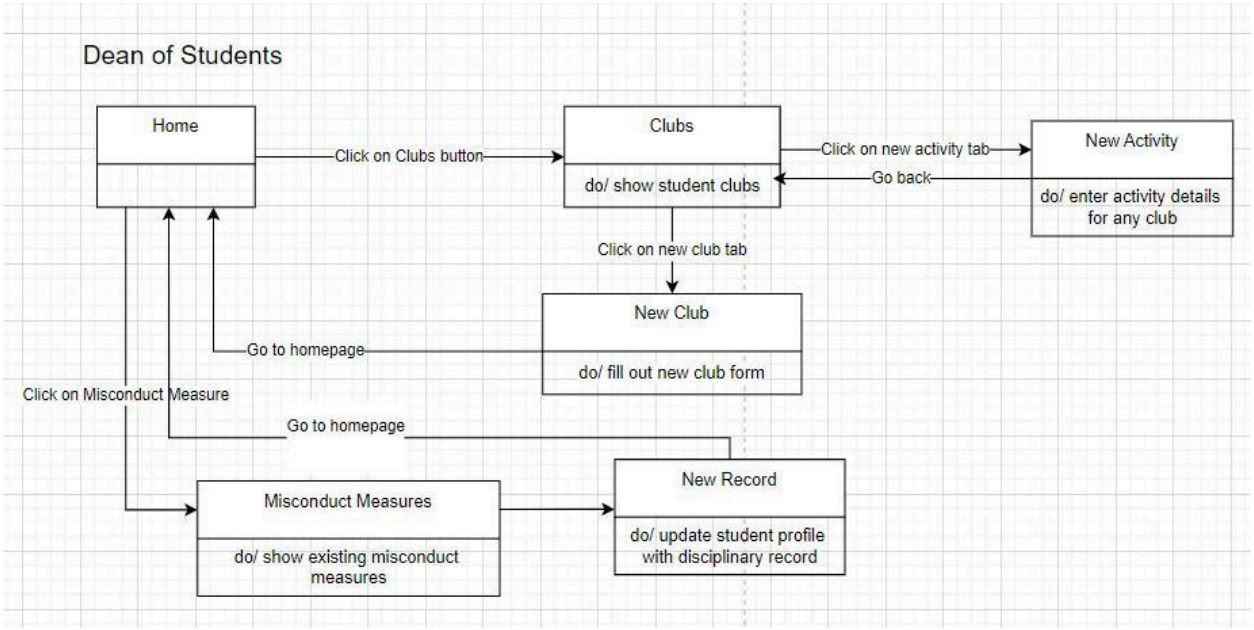
Coordinators:



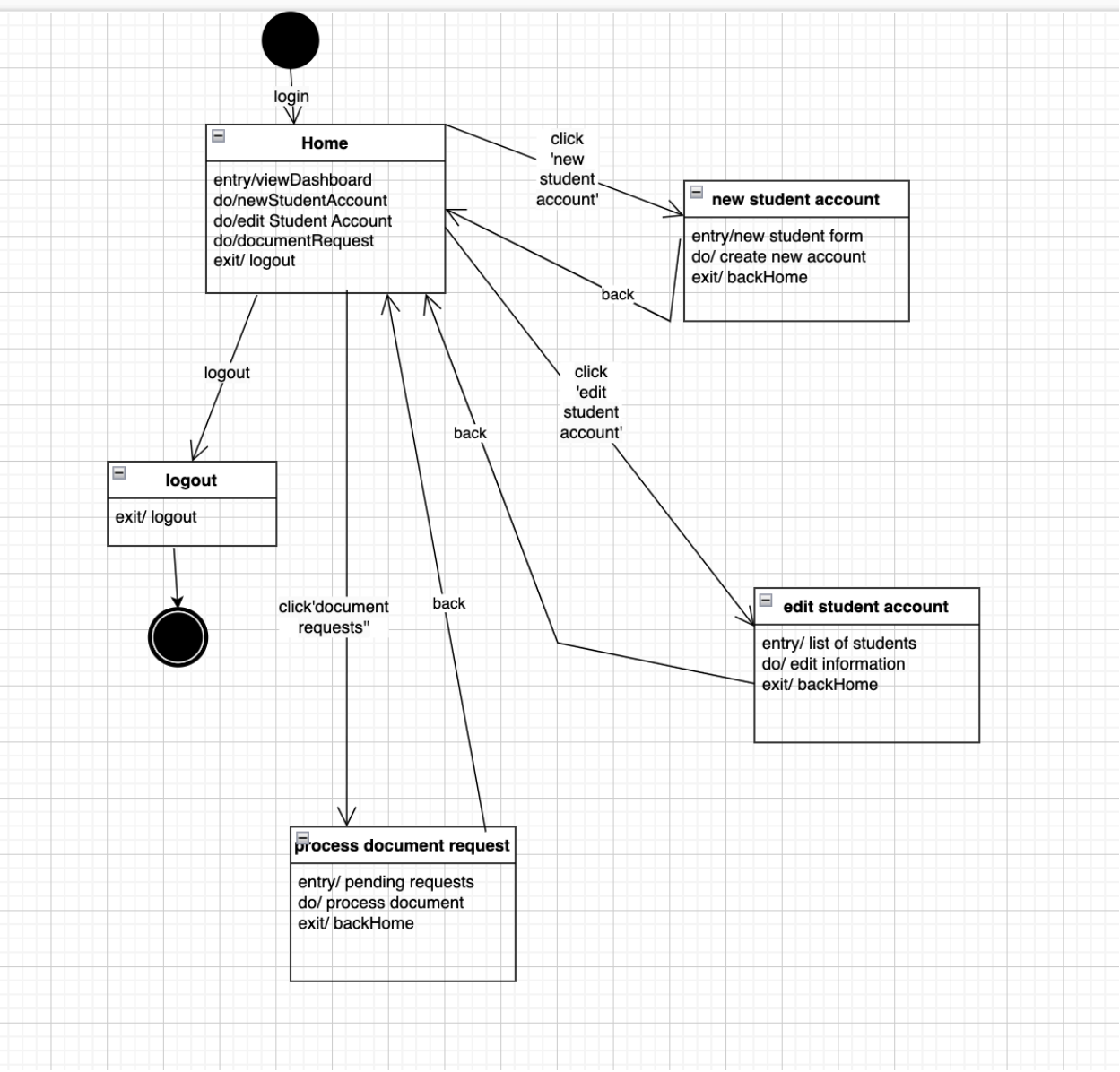
Finance Office:



Dean of Students

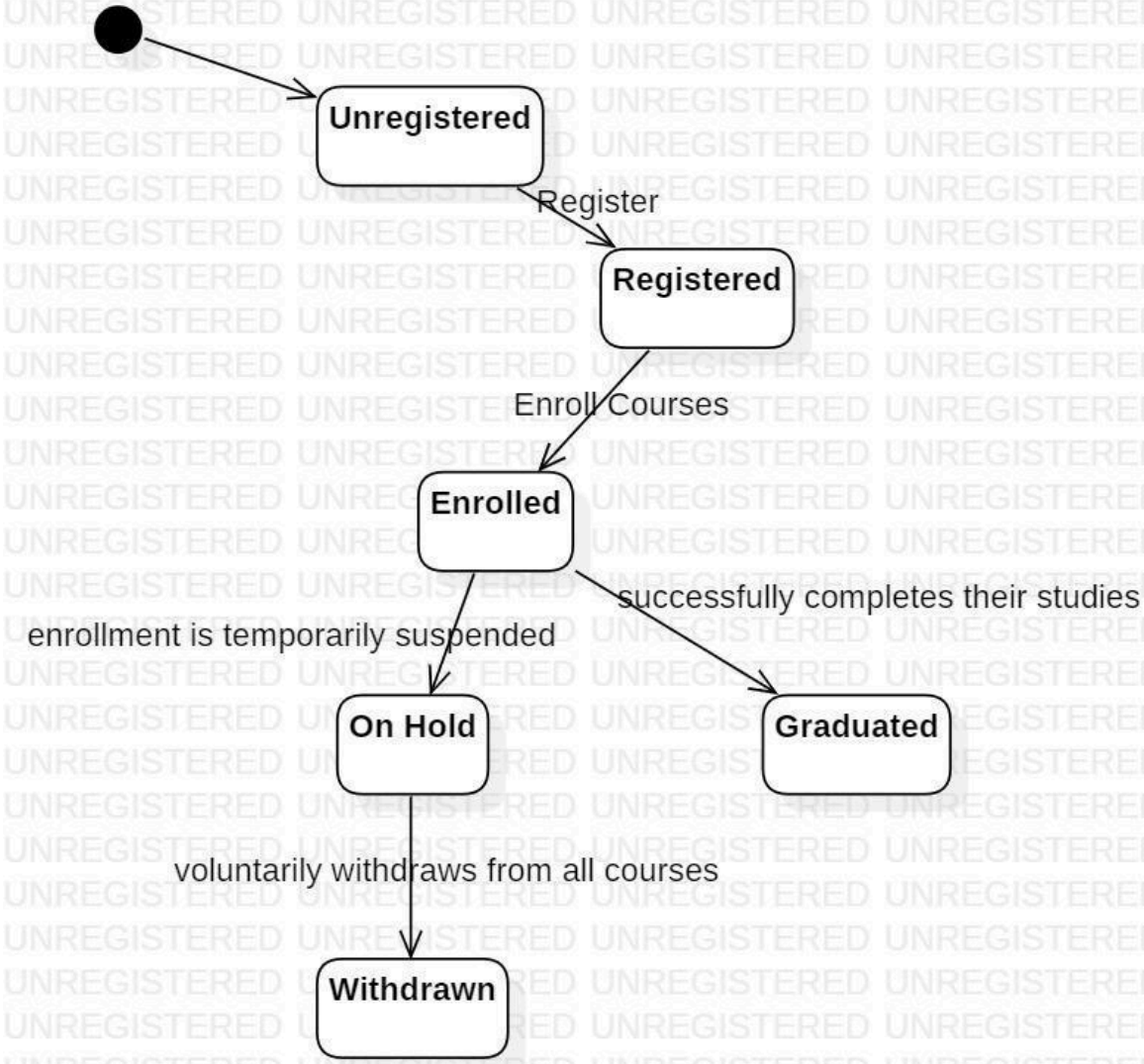


Registrar’s Office

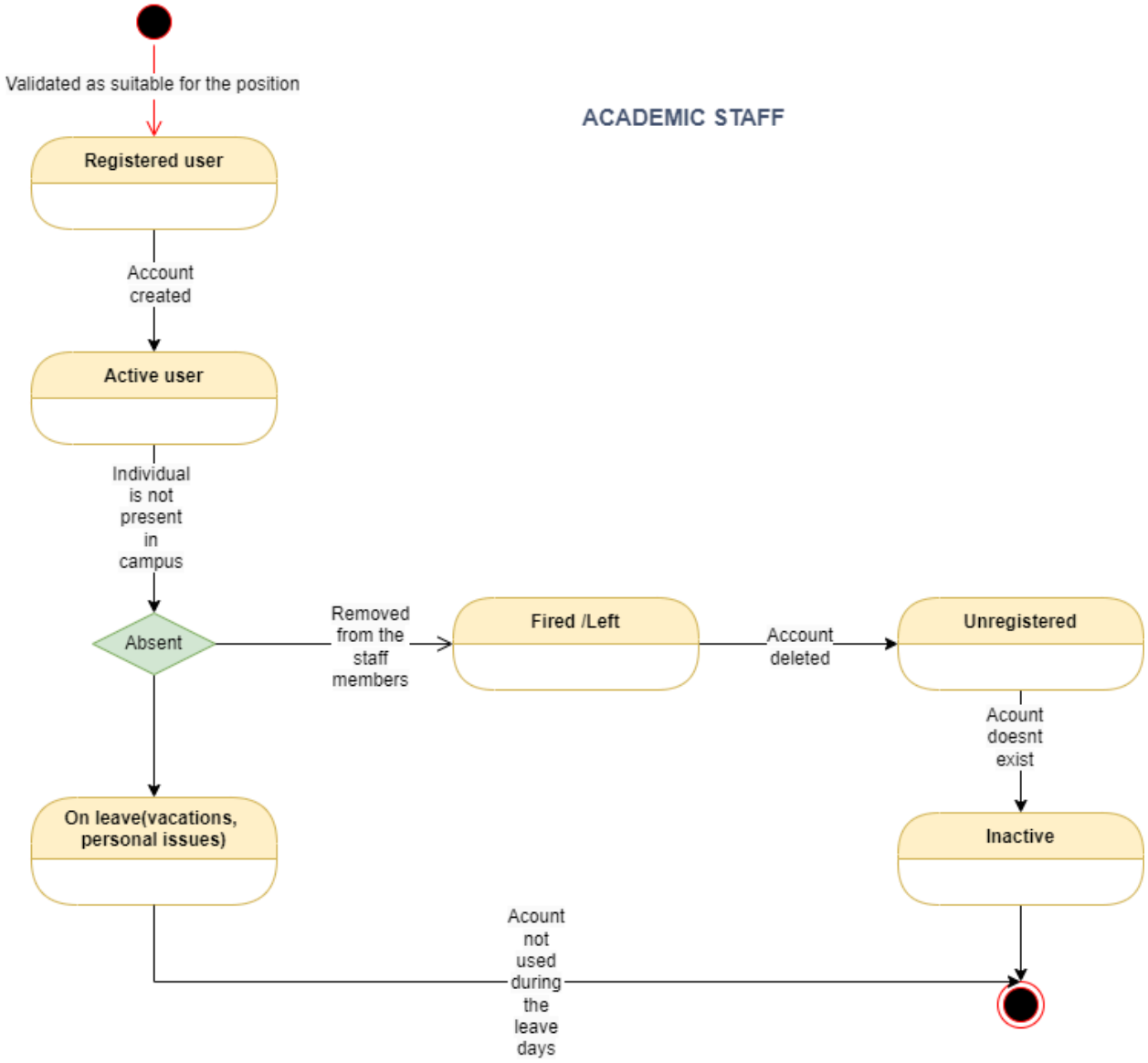


State Diagrams by Interfaces/Forms:

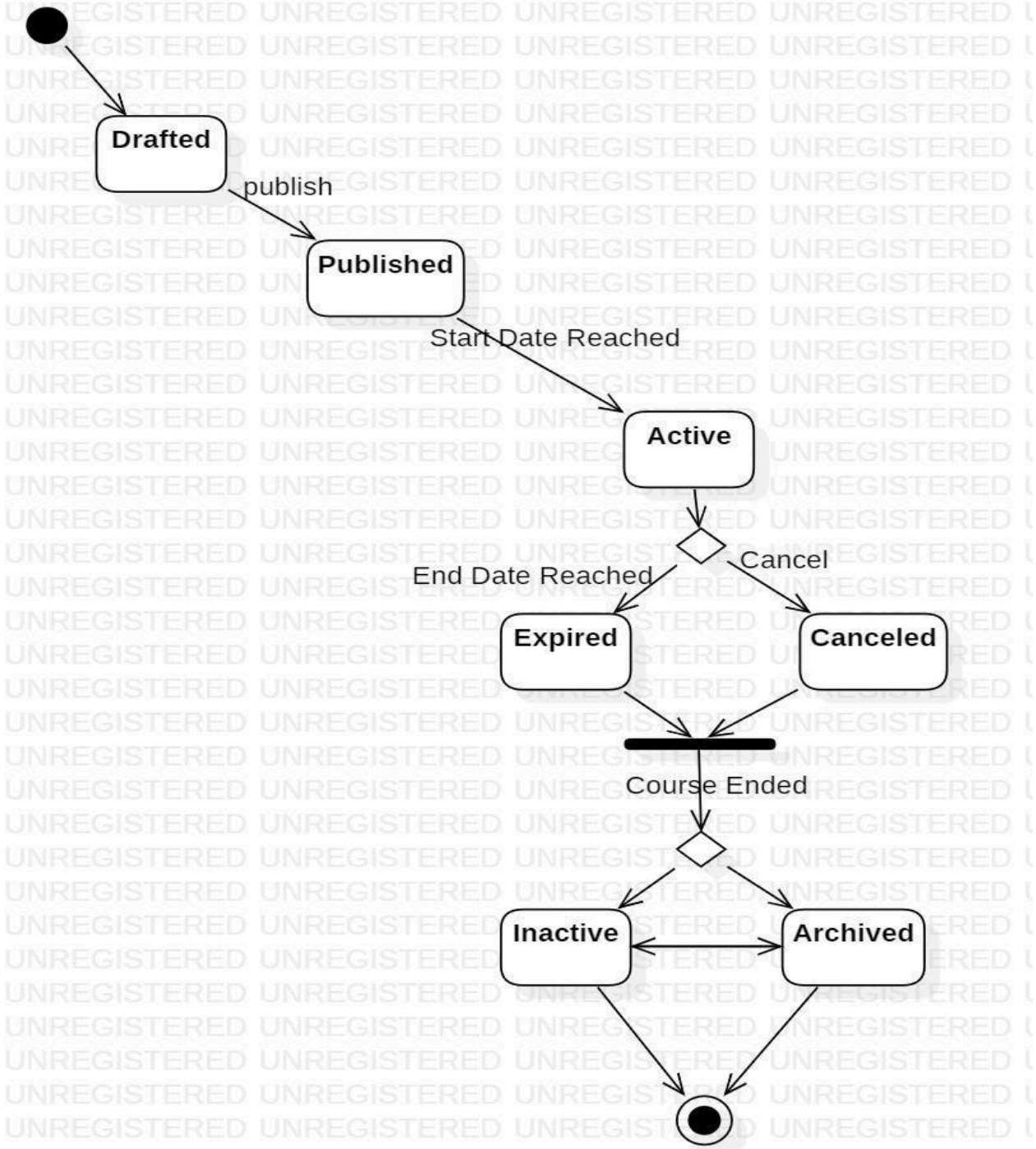
Students



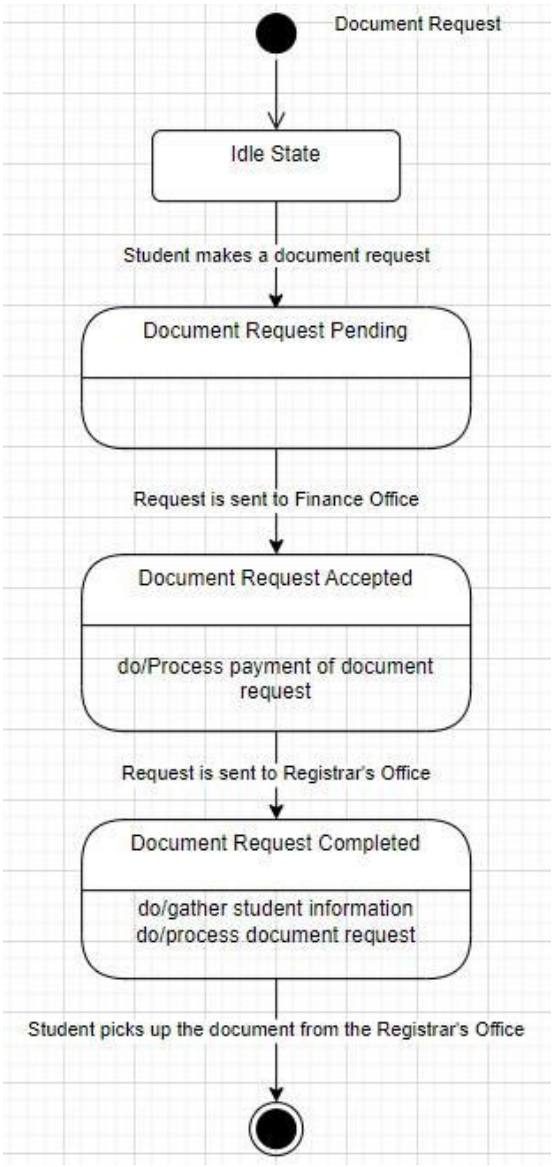
Academic Staff



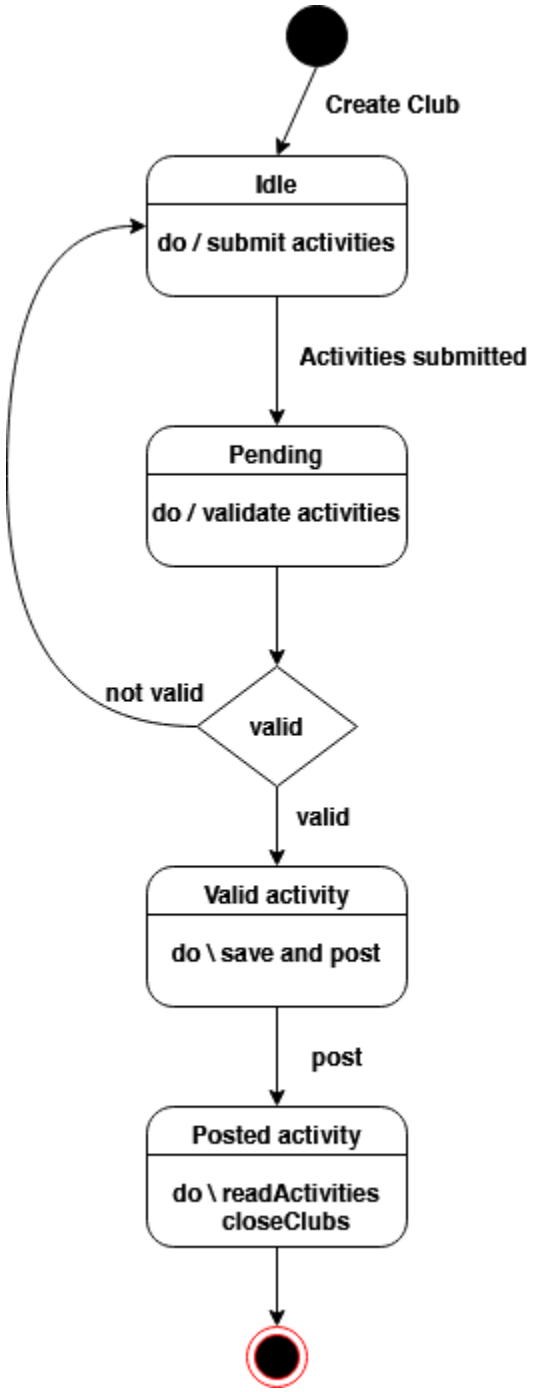
Courses


Drafted
publish
Published
Start Date Reached
Active
End Date Reached
Cancel
Expired
Canceled
Course Ended
Inactive
Archived

Document Requests(or Document)



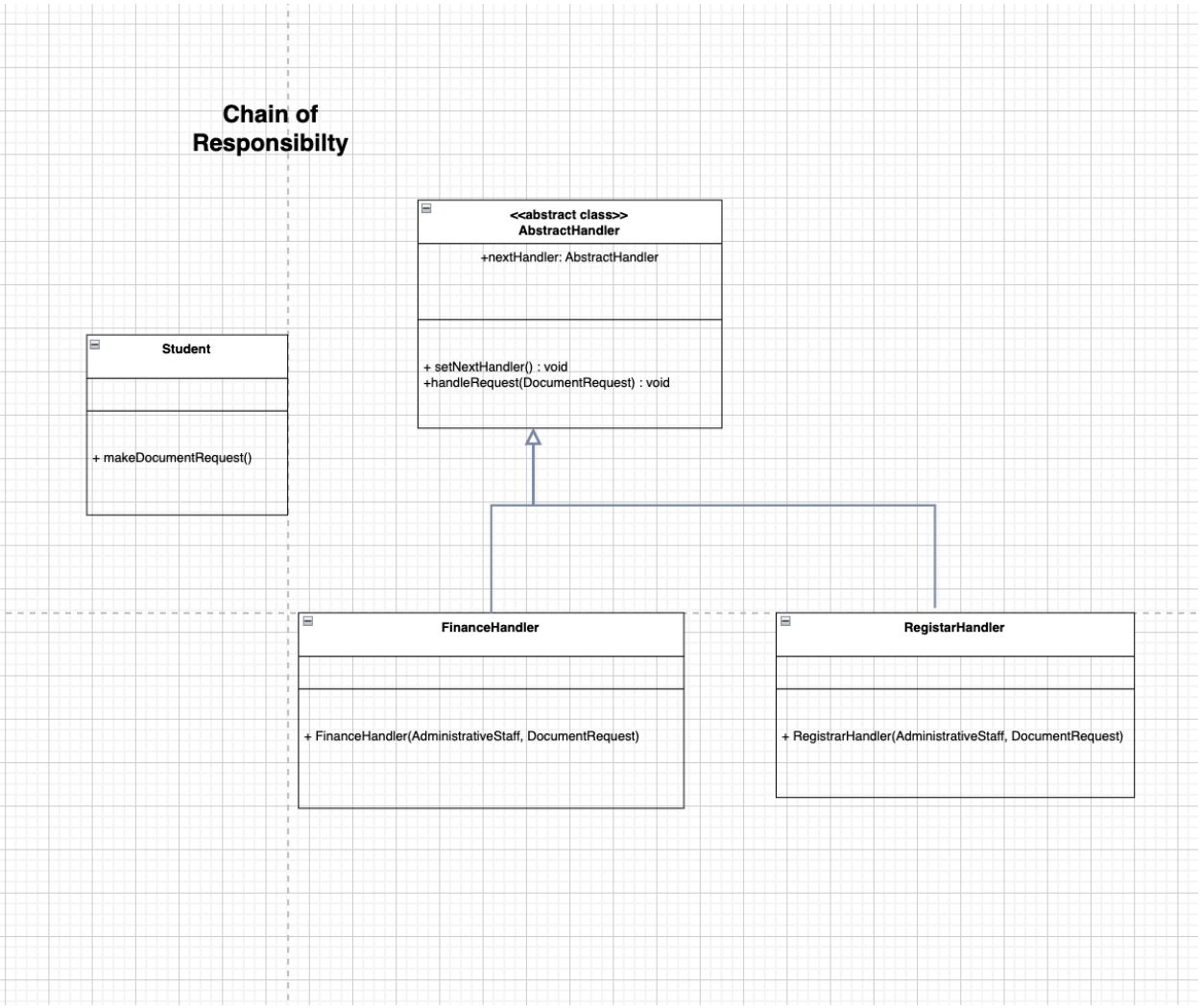
Student's Clubs



DESIGN PATTERNS

- **Chain of responsibility**

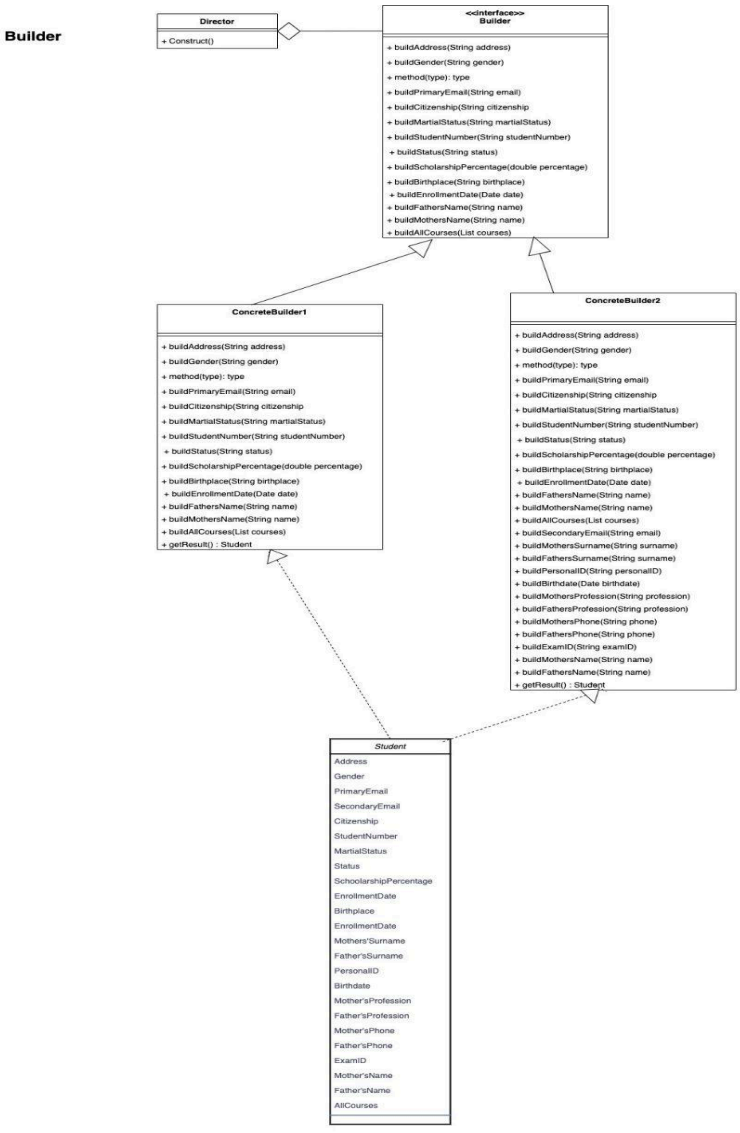
Since the document request use case involves a sequential chain of events involving various actors, a chain of responsibility design pattern might simplify the code organization. A student makes a document request, starting a chain reaction of events: the request first goes to the Finance Office and after finance approval it goes to the Registrar's Office to get handled. So we have an AbstractHandler interface and the concrete FinanceHandler and RegistrarHandler.



- **Builder**

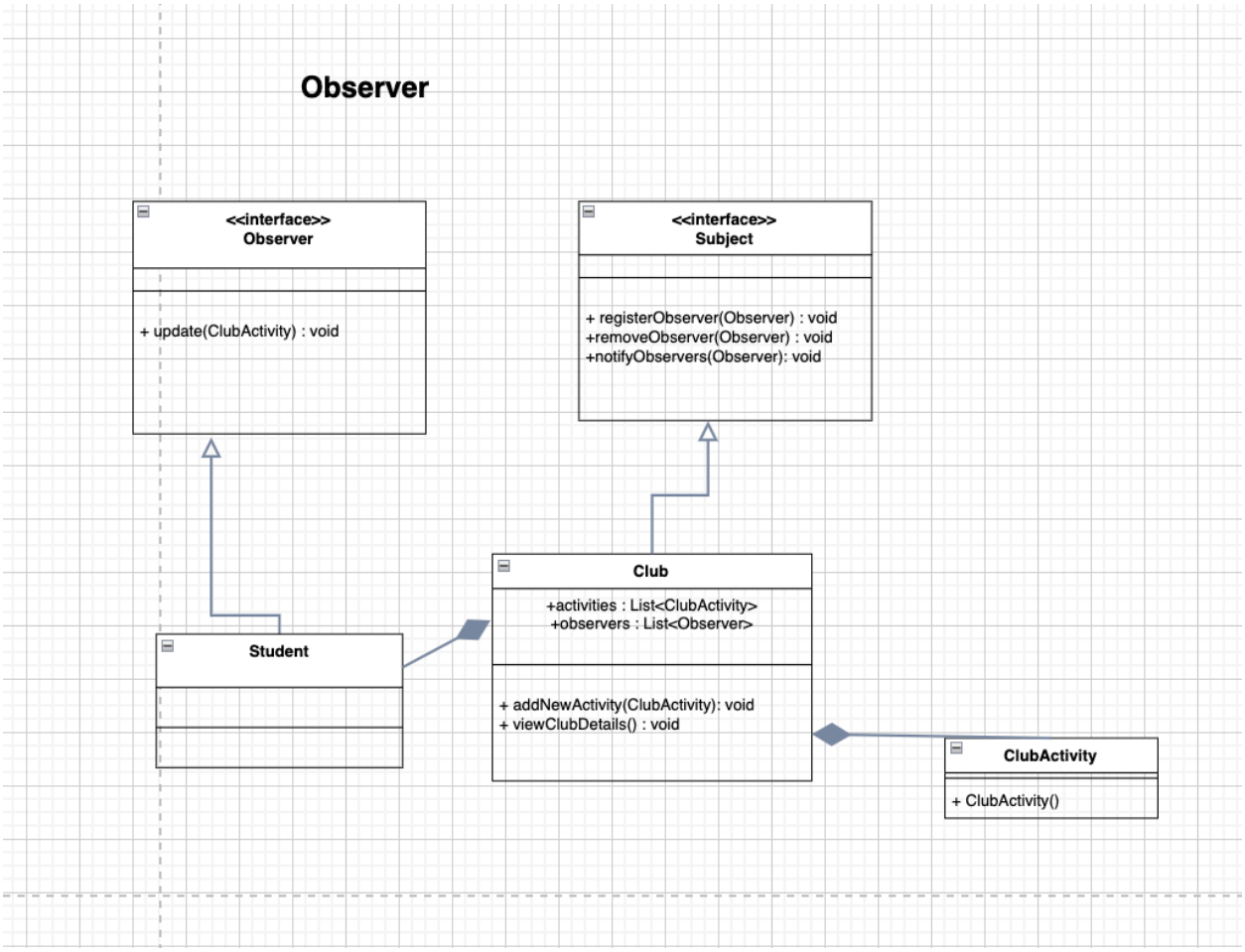
We have chosen to use the Builder design pattern for the Student class due to its extensive set of attributes. This design pattern is particularly advantageous in this context for several reasons:

1. Readability and Maintainability: The Student class has a large number of attributes, which can make its constructor cumbersome and difficult to read. By using the Builder pattern, we can construct Student objects in a more readable manner, where each attribute can be set in a step-by-step fashion. This enhances the code's readability and maintainability.
2. Scalability: As the Student class evolves, additional attributes may be added. The Builder pattern provides a scalable approach to object construction, as new attributes can be incorporated into the builder without modifying the existing constructor, thus adhering to the open/closed principle.



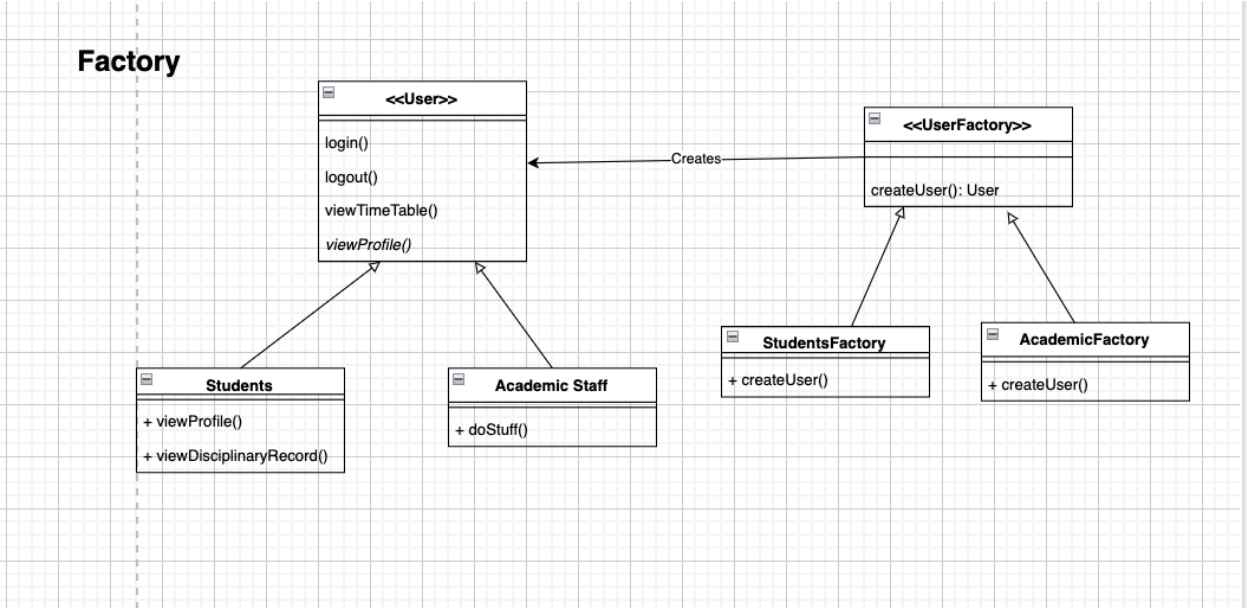
- **Observer**

Students could get notified about new club activities from the clubs they are members of. We have the Observer interface and the Subject. In our case the concrete observer is Student and concrete Subject is Club. The club will hold a list of observers (student members of that club) and notify them when new activities are posted. We could also make AcademicStaff an Observer, making the club hold a list of student members and also staff members related to the club (club advisors for example)



- **Factory**

The factory design pattern is used to provide an interface for creating objects in a superclass, allowing subclasses to alter the type of objects that will be created. This is useful in a school management system where different types of users (e.g., students, academic staff) may need to be created, each with its specific attributes and behaviors.



- **Proxy**

We have implemented the Proxy design pattern because multiple users need to access and perform various operations on the same database. The Proxy pattern offers several key benefits in this context:

1. Controlled Access: By using a proxy, we can control and manage access to the database. This is essential when multiple users are involved, as it ensures that only authorized users can perform specific operations. The proxy can handle authentication and authorization checks before allowing any actions on the database.
2. Resource Management: The proxy can manage database connections efficiently. Instead of each user opening a new connection, the proxy can pool connections and reuse them, which improves performance and reduces the overhead associated with opening and closing database connections frequently