College Enrollment System

Software Requirements Specification

Revision History

Date	Revision	Description	Author
09/11/2024	1.0	Initial Version Creation	Janelle Kwofie
09/19/2024	2.0	Edited to fit Google Docs	Janelle Kwofie
09/23/2024	3.0	Completed Section One and Four	Janelle Kwofie
9/27/2024	3.1	Completed Section Two, Excluding 2.1	Roy Alkhoury
9/27/2024	3.2	Completed Section 2.1	Janelle Kwofie
9/27/2024	3.3	First revision of Section 3 with minor edits for Section 4	Bach Ngo
9/27/2024	3.4	Updated Section 3.2 & 3.3	Roy Alkhoury
9/27/2024	3.5	Updated Section 2 & 4	Bach Ngo
9/27/24	3.6	UML Class Diagram	Anthony Josh Legrama
9/27/24	3.7	Updated Section 4.1,4.2,4.3	Anthony Josh Legrama
9/28/24	3.8	Updated Class Diagram	Bach Ngo
9/28/24	3.9	Added Use Cases for Student Actor	Janelle Kwofie
9/29/24	3.10	UML Sequence Diagram for Student Enrollment	Anthony Josh Legrama
9/29/24	3.11	UML Sequence Diagram for View Enrollment	Anthony Josh Legrama
9/29/24	3.12	Updated Use Case Specification (WIP)	Bach Ngo
9/30/24	4.0	Fixed Formatting and Updated 1.3	Janelle Kwofie
9/30/24	4.1	Updated Section 6.2 (WIP)	Bach Ngo
10/01/2024	4.2	Updated Some Formatting in Section 6	Roy Alkhoury
10/01/2024	4.3	Completed Section 6.3	Roy Alkhoury
10/01/2024	4.4	Minor edits on Section 6.3	Bach Ngo

10/01/2024	4.5	Finish Section 6.2	Bach Ngo
10/01/2024	4.6	Fix Section 1.3 links (it now self-references within the PDF instead of linking to GDocs)	Bach Ngo
10/01/24	4.7	UML Sequence Diagrams for Case ID 0x8, 0x9, 0xA, 0xB	Anthony Josh Legrama
10/02/24	4.8	Meeting Minutes added as Section Seven	Janelle Kwofie
10/02/24	4.9	Add project timeline	Bach Ngo

Table of Contents

1. Purpose

This document will outline the requirements for the College Enrollment System (CES).

1.1. Scope

This document will catalog the user, system, and hardware requirements for the CES. This document will include written material and diagrams that depict the functionality of the system, but will not show any implementation.

1.2. Definitions, Acronyms, Abbreviations

CES - College Enrollment System

1.3. References

- Use Case Specification
- UML Class Diagram
- Use Case Diagram
- UML Sequence Diagram

1.4. Overview

The College Enrollment System, CES, is designed to perform the enrollment functions necessary for college enrollment. Specifically, it will look at the needs of both Students and Administrators. Students will need to be able to interact with prospective classes and register for ones they qualify for. Administrators will need to be able to set requirements for these courses such as capacity and setting prerequisites.

2. Overall Description

2.1. Product Perspective

The product will allow students to access the system through their username and password. Once they are granted access they will be able to perform tasks related to class enrollment like enrolling, dropping, and searching for classes. On the administrative side, administrators will be able to ensure that students meet the criteria for enrollment and can revise any class information and class enrollment that they need to.

2.2. Product Architecture

The system will be organized into 3 major modules: the Course module, the User module, and the Network module.

2.3. Product Functionalities/Features

2.3.1. The high-level features of the system are as follows (see section 3 of this document for more detailed requirements that address these features):

2.3.2. Allow admins to create, edit, and delete courses and sections.

2.3.3. Students can enroll and drop a section.

2.3.4. Courses can be searched by multiple criteria such as course prefixes and course number.

2.4. Constraints

2.4.1. Administrators, students, and instructors must be logged in to have their permissions.

2.4.2. Students must meet prerequisites and vacancies before enrolling or dropping a section.

2.4.3. System displays a warning when admin edits a section's schedule.

2.4.4. If a section is full and then later edited to have a smaller class size, students will roll over to the waitlist, exceeded waitlist will get dropped.

2.4.5. The system prevents the admin or student from completing a transaction if the data is outdated.

2.5. Assumptions and Dependencies

2.5.1. It is assumed that the login for the administrator, student, and instructor are all different.

2.5.2. It is assumed that a student belongs to one respective university.

2.5.3. It is assumed that all users have an Internet connection and have the required JRE installed.

2.5.4. It is assumed that the instructor meets the credentials to teach their section.

2.5.5. It is assumed that there is one instructor for each section.

2.5.6. It is assumed that the system has a university's course catalog information.

3. Specific Requirements

3.1. Functional Requirements

3.1.1. Common Requirements:

3.1.1.1. The system is used by many universities.

3.1.2. Course Module Requirements:

3.1.2.1. A course may contain multiple or no sections.

3.1.2.2. A section must have information about the class size, waitlist size, instructor, and the meeting time and place (schedule).

3.1.2.3. A section's schedule may be asynchronous (no specific time).

3.1.2.4. A section may be hybrid (for example: online on Monday but on-campus on Wednesday).

3.1.2.5. If a section is full and is later changed to have a smaller capacity, exceeded students will be put on the waitlist. Existing students on the waitlist may get dropped if the waitlist ends up over capacity.

3.1.2.6. If a section is full and the waitlist size is not met, future students will be put on the waitlist. Once a spot is available, the waitlist will roll over on a FIFO basis.

3.1.2.7. Courses can be searched via course prefix, course number, name/subject, date of week, and instructor.

3.1.2.8. A course's prerequisites may contain multiple or no courses.

3.1.2.9. The system must check for any prerequisite cycles before adding or editing a course's prerequisites.

3.1.3. User Module Requirements:

3.1.3.1. The user-facing interface must be a GUI.

3.1.3.2. A user can log out of their account.

3.1.3.3. A university must have at least one admin account.

3.1.3.4. The admin can add and delete courses.

3.1.3.5. The admin can edit course metadata (subject, description, unit).

3.1.3.6. The admin can add, edit, or delete a section.

3.1.3.7. The admin can enroll or drop a student from a section.

3.1.3.8. The system should issue a warning when an admin performs any actions that may affect a student's schedule. This includes editing a section's schedule, editing a section's capacity, enrolling or dropping a student, and deleting an existing course or section.

3.1.3.9. The system must be able to generate reports on enrollment information and sections with low and high enrollment.

3.1.3.10. The admin can create a student's account.
3.1.3.11. All students must have an account.
3.1.3.12. Students can drop or enroll in a section but must meet course prerequisites, section vacancies, and schedule availability before that.
3.1.3.13. Students can view their previously enrolled courses and currently enrolled and waitlisted courses.
3.1.3.14. An instructor may or may not have an account.
3.1.3.15. The instructor account can view information about the sections they teach in, including which students are enrolling.

3.1.4. Network Module Requirements:

3.1.4.1. The server must handle multiple concurrent connections.
3.1.4.2. The server must prevent the client from completing a transaction if the client data is stale.
3.1.4.3. The client must display an error message if it doesn't receive any responses to its request on time.

3.2. External Interface Requirements

3.2.1. The system must provide a way to export enrollment data. The format is in comma-separated text files.
3.2.2. The system must provide an interface to connect to the university billing system to automatically bill for enrolled sections. The interface will be a text file containing a student ID, section ID, term of enrollment such as "Fall 2024", and action indicating if they enrolled or dropped a course. The file will be exported after the enrollment period is over.
3.2.3. The system may have an interface to accept course information from an external source (such as from the university's catalog). The interface will be a text file containing the course number, course subject, and course description, along with other metadata.

3.3. Internal Interface Requirements

3.3.1. The server must provide an interface to communicate with the client.

4. Non-Functional Requirements

4.1. Security and Privacy Requirements

4.1.1. Students will need unique student identification (ID) numbers.

4.1.2. The system must authenticate users before they can perform any other activities.

4.1.3. An Administration Account has a higher level of security, clearance, and priority than a Student Account.

4.1.4. Logins to the system are done by entering the Student ID or Username and the Password.

4.1.5. Students can’t view each other's enrollment information.

4.1.6. The system doesn’t save most information aside from data used to initially launch.

4.2. Environmental Requirements

4.2.1. The system must be able to function across various operating systems such as Windows, macOS, and Linux with the appropriate JRE installed.

4.3. Performance Requirements

4.3.1. The system will respond to queries within three seconds.

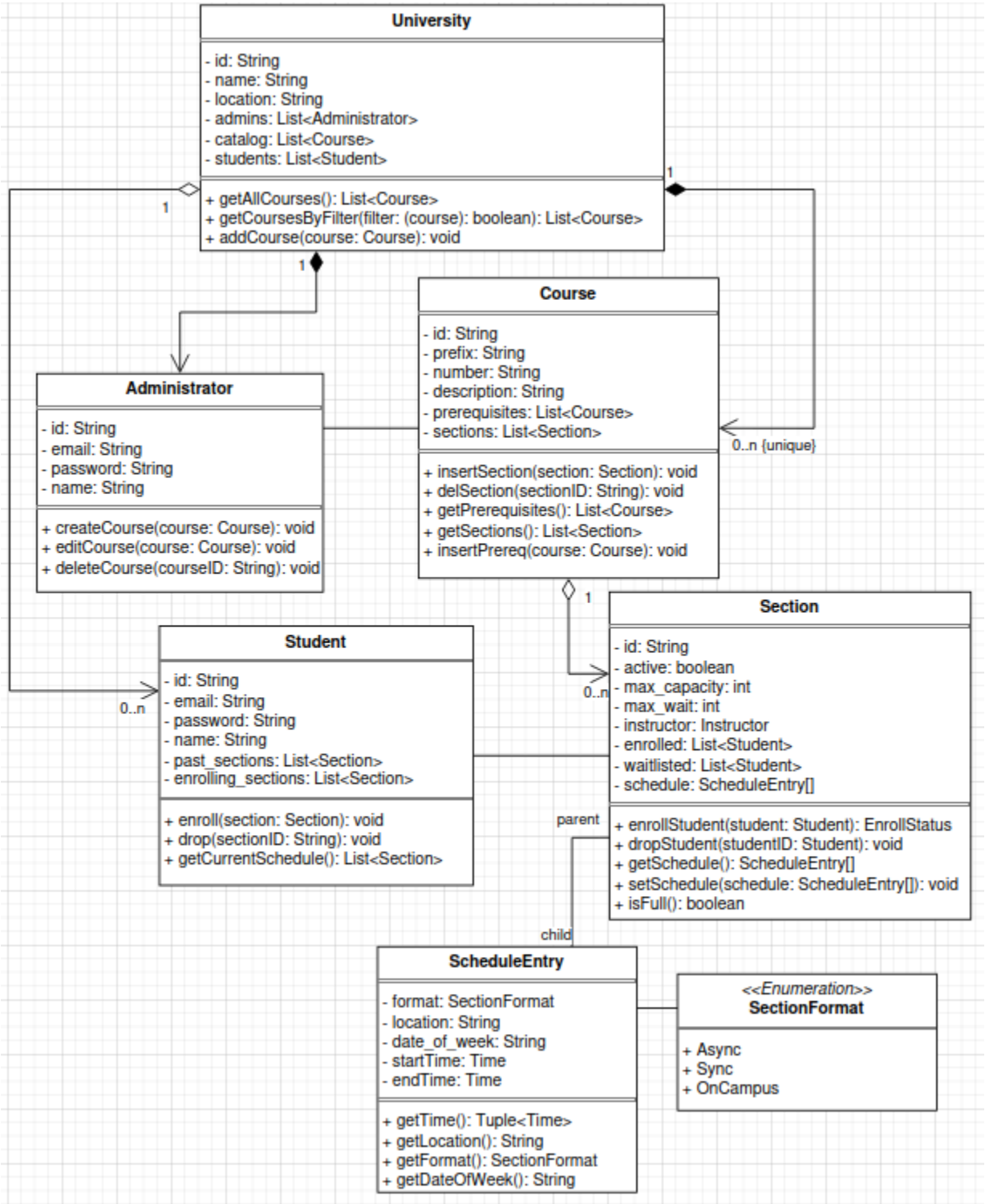
4.3.2. The system should launch in two seconds.

4.3.3. Data backup operations should be handled smoothly without affecting the current operations that the user is going through.

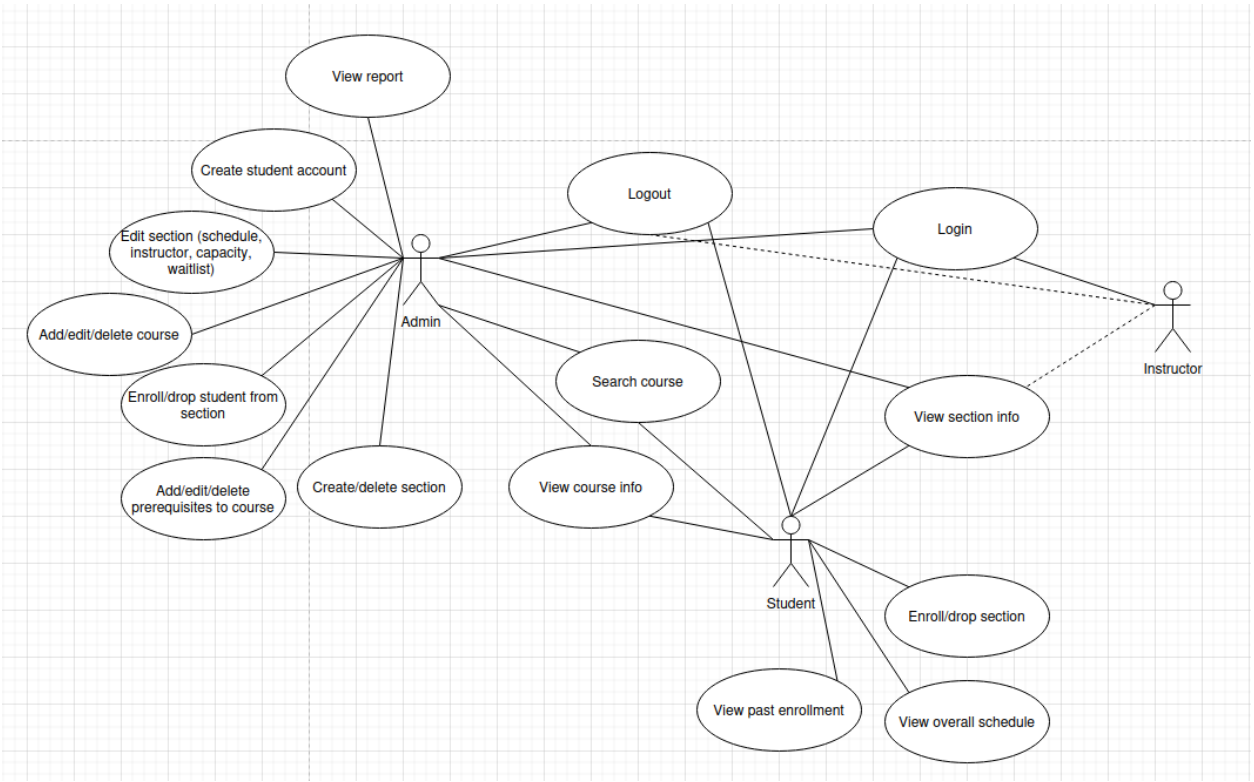
4.3.4. The system must clean up resources when a connection terminates.

5. Diagrams

5.1. UML Class Diagram

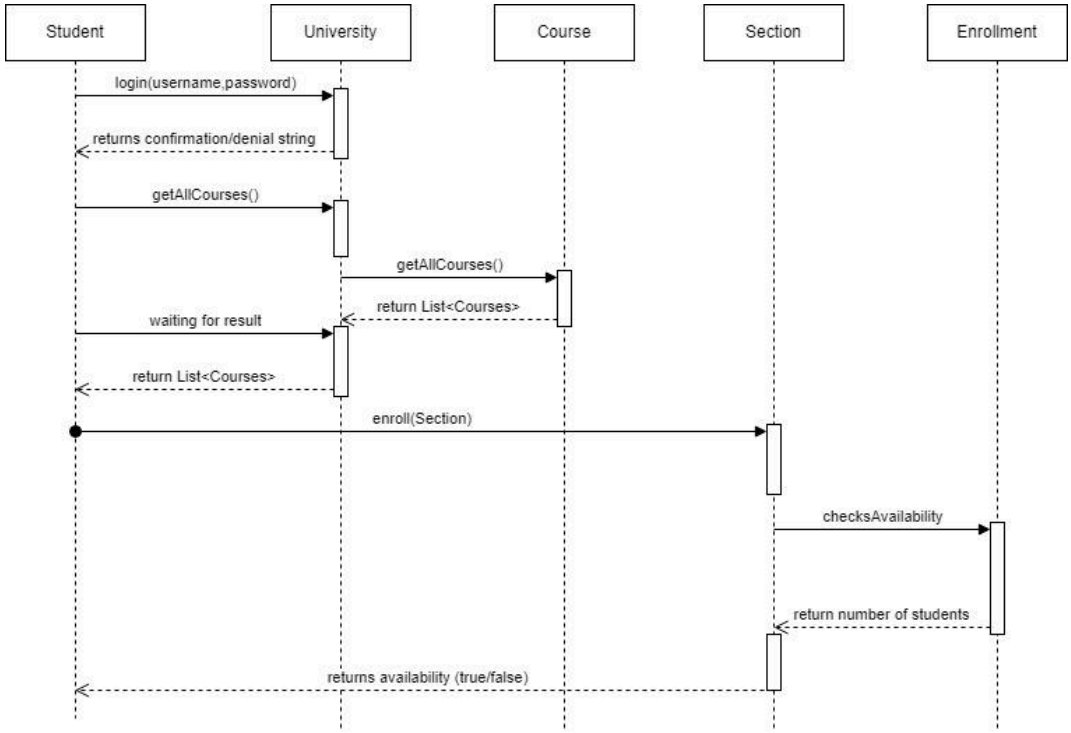


5.2. Use Case Diagram

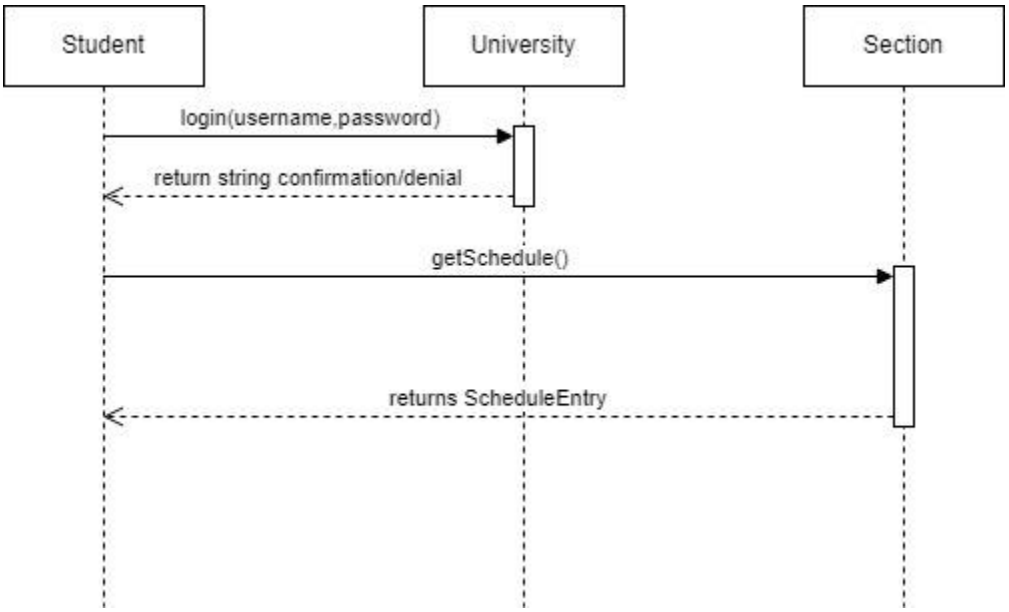


5.3. UML Sequence Diagram

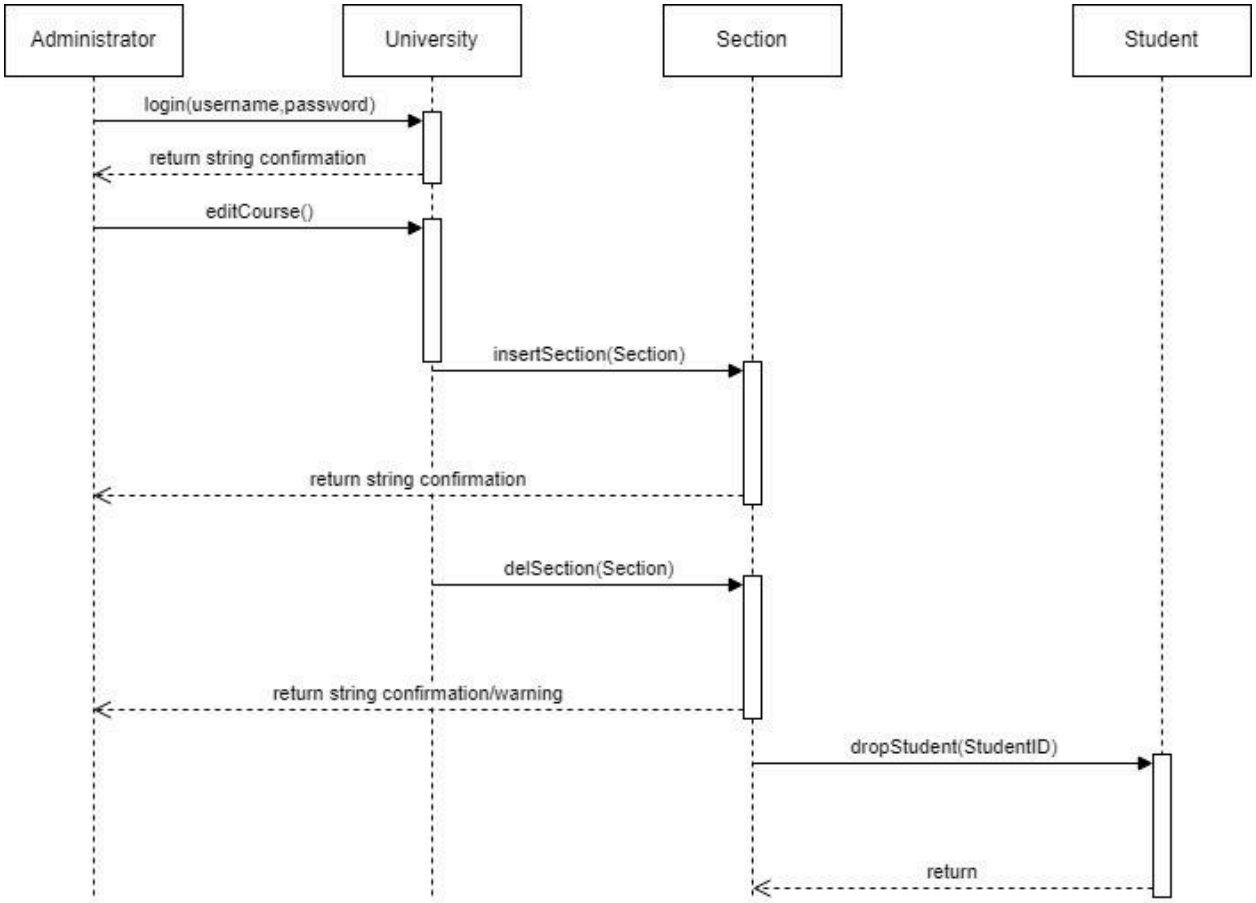
Case ID 0x0 Sequence Diagram:



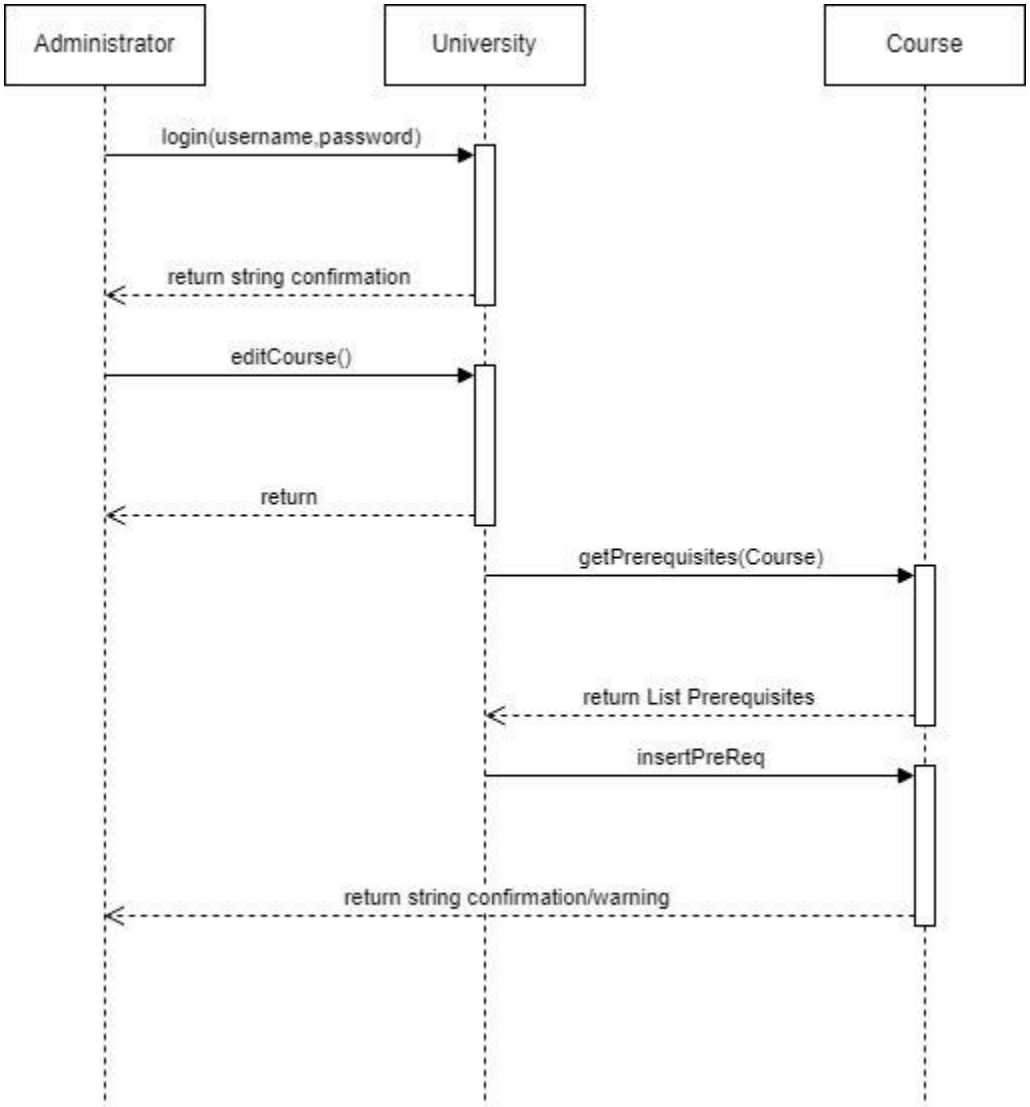
Case ID 0x6 Sequence Diagram:



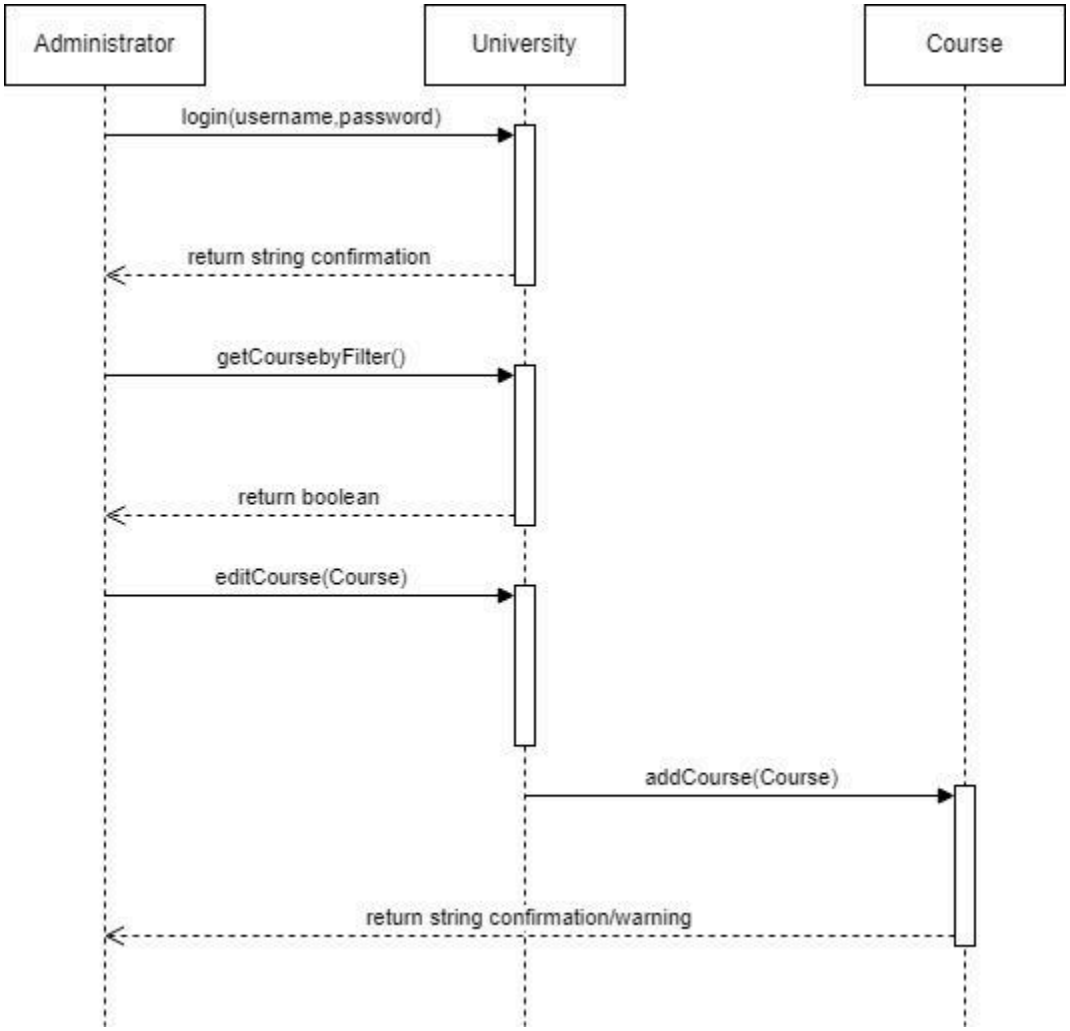
Case ID 0x8 Sequence Diagram:



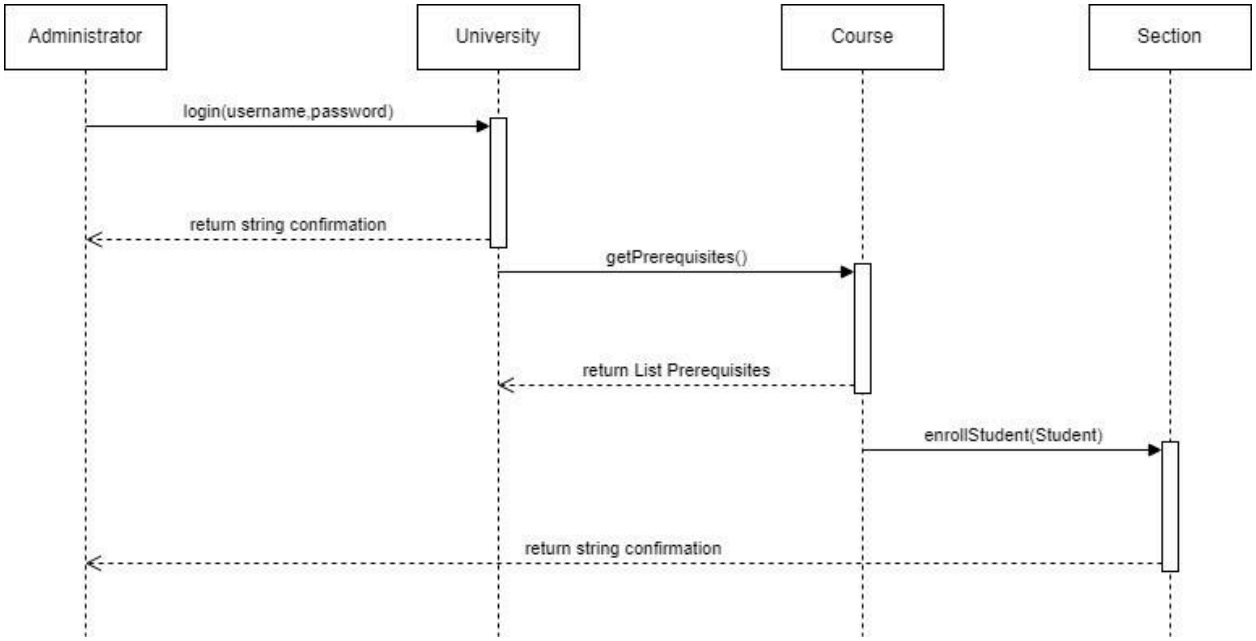
Case ID 0x9 Sequence Diagram:



Case ID 0xB Sequence Diagram:



Case ID 0xA Sequence Diagram:



6. Use Case Specification

6.1. Use Case: Enrollment

Use Case ID: 0x0

Use Case Name: Enroll Section

Relevant Requirements:

- Students can drop or enroll in a section but must meet course prerequisites, section vacancies, and schedule availability before that. (section 3.1.3.12)

Primary Actor: Student

Pre-conditions:

- The student is authenticated.
- The student has prerequisites for enrollment.
- The section exists and is not full.
- The section's schedule is compatible with the student's schedule.

Post-conditions: The student is either enrolled in the section, waitlisted, or not able to enroll.

Basic Flow or Main Scenario:

1. Student logs into the system with the correct credentials.
2. They search for the classes they want.
3. They have availability to add to the schedule and the course is not full, thus the system successfully enrolls them into the chosen section.

Extensions or Alternate Flows:

1. The student logs into the system with the correct credentials. They search for the classes they want. They have availability to add to the schedule, but the section capacity is full, thus the system tries to put them on the waitlist.

Exceptions:

1. Username and password are incorrect.
2. They don't have the availability to add to the schedule.
3. The section is full (both in capacity and waitlist).
4. The student is already in the section, either fully enrolled or waitlisted.

Related Use Cases: Drop Section (0x1)

Use Case ID: 0x1

Use Case Name: Drop Section

Relevant Requirements:

- Students can drop or enroll in a section but must meet course prerequisites, section vacancies, and schedule availability before that. (section 3.1.3.12)

Primary Actor: Student
Pre-conditions:

- The student is authenticated.
- The student is enrolled or waitlisted in a section.

Post-conditions:

- The student is dropped from the section.
- The section waitlist may roll over.

Basic Flow or Main Scenario:

1. Student logs into the system with the correct credentials.
2. They view their current schedule and select the section they want to drop.
3. The system responds by removing them from the chosen section. The system will update its capacity list and may also update its waitlist.

Extensions or Alternate Flows: None
Exceptions:

1. Username and password are incorrect.

Related Use Cases: Enroll Section (0x0)

Use Case ID: 0x2
Use Case Name: Login
Relevant Requirements:

- The system must authenticate users before they can perform any other activities. (section 4.1.2)
- Logins to the system are done by entering the Student ID or Username and the Password. (section 4.1.4)

Primary Actor: Student
Pre-conditions:

- The user has an account in the system.
- Username and password are entered correctly.

Post-conditions:

- The user is authenticated.

Basic Flow or Main Scenario:

1. The user selects the university they're in and provides their username and password.
2. The system responds with either success or failure.

Extensions or Alternate Flows:

1. An instructor selects the university they're in and provides their login credentials. The system responds with either success or failure.

Exceptions:

1. Username and password are incorrect.
2. The account doesn't exist in the system.

Related Use Cases: Logout (0x3), Enroll Section (0x0)

Use Case ID: 0x3
Use Case Name: Logout
Relevant Requirements:
- A user can log out of their account. (section 3.1.3.2)

Primary Actor: Student
Pre-conditions:
- The user is authenticated.

Post-conditions:
- The user is no longer authenticated and therefore not able to perform most other use cases.

Basic Flow or Main Scenario:
1. The user selects the option to log out of the account.
2. The system responds by prompting a confirmation, then logs the user out and terminates the connection between the user and the server.

Extensions or Alternate Flows: None.
Exceptions:
1. The user is already not authenticated.

Related Use Cases: Login (0x2)

6.2. Use Case: Scheduling

Use Case ID: 0x4
Use Case Name: View Schedule
Relevant Requirements:
- Students can view their previously enrolled courses and currently enrolled and waitlisted courses. (section 3.1.3.13)
- Students can’t view each other's enrollment information. (section 4.1.5)

Primary Actor: Student
Pre-conditions:
- The user is logged in.

Post-conditions:
- Student can see their class schedule.

Basic Flow or Main Scenario:
1. The student logs into their account
2. They click on a button to view the schedule.
3. The system responds by displaying the schedule, including enrolled and waitlisted sections.

Extensions or Alternate Flows: None.

Exceptions:

1. The user is not authenticated.
2. A student tries to request a schedule of another student.

Related Use Cases: View Past Enrollment (0x5)

Use Case ID: 0x5
Use Case Name: View Past Enrollment
Relevant Requirements:

- Students can view their previously enrolled courses and currently enrolled and waitlisted courses. (section 3.1.3.13)
- Students can't view each other's enrollment information. (section 4.1.5)

Primary Actor: Student
Pre-conditions:

- The user is logged in.

Post-conditions:

- Student can see their past enrollment.

Basic Flow or Main Scenario:

1. The student logs into their account
2. They click on a button to view their enrollment.
3. The system responds by displaying the student's past enrollment.

Extensions or Alternate Flows: None.
Exceptions:

1. The user is not authenticated.
2. A student tries to request the enrollment of another student.

Related Use Cases: View Schedule (0x4)

Use Case ID: 0x6
Use Case Name: Search Course Information
Relevant Requirements:

- Courses can be searched via course prefix, course number, name/subject, date of week, and instructor. (section 3.1.2.7)

Primary Actor: Student
Pre-conditions:

- The user is logged in.

Post-conditions:

- Student can search course information.

Basic Flow or Main Scenario:

1. The user logs into their account.
2. They use search functionality to define their query and submit it.
3. The system returns a list of courses that match the specified filters.

Extensions or Alternate Flows: None.
Exceptions:

1. The user is not logged in.
2. There are no courses that match the filters.
3. The filters are invalid/not defined by the system.

Related Use Cases: View Schedule (0x4)

Use Case ID: 0x7
Use Case Name: View Section Information
Relevant Requirements:

- A section must have information about the class size, waitlist size, instructor, and the meeting time and place (schedule). (section 3.1.2.2)
- A section's schedule may be asynchronous (no specific time). (section 3.1.2.3)
- A section may be hybrid (for example: online on Monday but on-campus on Wednesday). (section 3.1.2.4)

Primary Actor: Student
Pre-conditions:

- The user is logged in.

Post-conditions:

- Student can see the section information.

Basic Flow or Main Scenario:

1. The system responds to Use Case 0x7 with a list of courses (or no courses).
2. The user selects one of the courses to view more details.
3. The system responds by returning the information of the course, which includes each section's information.

Extensions or Alternate Flows: None.
Exceptions:

1. Username and password are incorrect.
2. The section or course no longer exists.

Related Use Cases: View Past Enrollment (0x5)

6.3. Use Case: Administrating

Use Case ID: 0x8

Use Case Name: Create/Delete Section

Relevant Requirements:

- A course may contain multiple or no sections (section 3.1.2.1)
- The admin can add, edit, or delete a section. (section 3.1.3.6)
- The system should issue a warning when an admin performs any actions that may affect a student's schedule. This includes editing a section's schedule, editing a section's capacity, enrolling or dropping a student, and deleting an existing course or section. (section 3.1.3.8)

Primary Actor: Administrator

Pre-conditions: Administrator must be logged in successfully to have admin privileges.

Post-conditions: Section is added or deleted.

Basic Flow or Main Scenario:

1. Administrator inputs his information to the system until a successful login
2. Admin creates the section.

Extensions or Alternate Flows: Admin deletes the section instead.

Exceptions:

- Incorrect login and needs to input the information again.

Related Use Cases: Edit Section (0xC)

Use Case ID: 0x9

Use Case Name: Add/Edit/Delete Prerequisites to Course

Relevant Requirements:

- The system must check for any prerequisite cycles before adding or editing a course's prerequisites. (section 3.1.2.9)
- The system should issue a warning when an admin performs any actions that may affect a student's schedule. This includes editing a section's schedule, editing a section's capacity, enrolling or dropping a student, and deleting an existing course or section. (section 3.1.3.8)

Primary Actor: Administrator

Pre-conditions: Administrator must be logged in successfully to have admin privileges.

Post-conditions: Admin adds, edits, or deletes prerequisites to course.

Basic Flow or Main Scenario:

1. Administrator inputs his information to the system until a successful login.
2. Admin checks prerequisites through the course catalog

3. Admin adds or edits the prerequisites to the course.

Extensions or Alternate Flows: 1. Deletes the prerequisite to the course instead.
Exceptions:

- Incorrect login and needs to input the information again.
- A prerequisite causes a prerequisite cycle.

Related Use Cases: Add/Edit/Remove course (0xB)

Use Case ID: 0xA
Use Case Name: Enroll/Drop Student from a Section
Relevant Requirements:

- The admin can enroll or drop a student from a section. (section 3.1.3.7)
- The system should issue a warning when an admin performs any actions that may affect a student's schedule. This includes editing a section's schedule, editing a section's capacity, enrolling or dropping a student, and deleting an existing course or section. (section 3.1.3.8)

Primary Actor: Administrator
Pre-conditions:

- Student must be a part of the university.
- Administrators must be logged in successfully to have admin privileges.
- Section must exist.

Post-conditions:

- Admin enrolls or drops student from a section

Basic Flow or Main Scenario:

1. Administrator inputs his information to the system until a successful login.
2. Admin checks if the student is still with the university.
3. Admin enrolls the student to the section.

Extensions or Alternate Flows: 1. Admin drops student from a section.
Exceptions:

- Incorrect login and needs to input the information again.
- Student is not part of the university anymore.
- Section does not exist anymore.

Related Use Cases: Create Student Account (0xD)

Use Case ID: 0xB
Use Case Name: Add/Edit/Remove Course
Relevant Requirements:

- A course may contain multiple or no sections (section 3.1.2.1)
- The system should issue a warning when an admin performs any actions that may affect a student's schedule. This includes editing a section's

schedule, editing a section's capacity, enrolling or dropping a student, and deleting an existing course or section. (section 3.1.3.8)

Primary Actor: Administrator
Pre-conditions: Administrators must be logged in successfully to have admin privileges.
Post-conditions: Admin adds, edits, or removes the course.
Basic Flow or Main Scenario:

1. Administrator inputs his information to the system until a successful login.
2. Checks to see if the course exists through the course catalog.
3. Admin adds or edits the course.

Extensions or Alternate Flows: 1. Admin deletes the course instead.
Exceptions:

- Incorrect login and needs to input the information again.
- Course does not exist through the course catalog.

Related Use Cases: Add/Edit/Delete Prerequisites to Course (0x9)

Use Case ID: 0xC
Use Case Name: Edit Section
Relevant Requirements:

- The admin can add, edit, or delete a section. (section 3.1.3.6)
- The system should issue a warning when an admin performs any actions that may affect a student's schedule. This includes editing a section's schedule, editing a section's capacity, enrolling or dropping a student, and deleting an existing course or section. (section 3.1.3.8)

Primary Actor: Administrator
Pre-conditions: Administrator must be logged in successfully to have admin privileges. Section must exist to be edited.
Post-conditions: Section is edited as the admin sees fit.
Basic Flow or Main Scenario:

1. Administrator inputs his information to the system until a successful login.
2. Admin edits the section.

Extensions or Alternate Flows: None.
Exceptions:

- Incorrect login and needs to input the information again.
- The section does not exist.

Related Use Cases: Create/Delete Section (0x8)

Use Case ID: 0xD
Use Case Name: Create Student Account

Relevant Requirements:

- The admin can create a student's account. (section 3.1.3.10)
- All students must have an account. (section 3.1.3.11)

Primary Actor: Administrator
Pre-conditions: Administrator must be logged in successfully to have admin privileges.
Post-conditions: Admin creates the student's account.
Basic Flow or Main Scenario:

1. Administrator inputs his information to the system until a successful login.
2. Admin creates the student's account.

Extensions or Alternate Flows: None.
Exceptions:

- Incorrect login and needs to input the information again.

Related Use Cases: Enroll/Drop Student from a Section (0xA)

Use Case ID: 0xE
Use Case Name: View Report
Relevant Requirements:

- The system must be able to generate reports on enrollment information and sections with low and high enrollment. (section 3.1.3.9)
- An Administration Account has a higher level of security, clearance, and priority than a Student Account. (section 4.1.3)

Primary Actor: Administrator
Pre-conditions: Administrator must be logged in successfully to have admin privileges.
Post-conditions: Admin views the system's generated reports on enrollment information and sections with low and high enrollment.
Basic Flow or Main Scenario:

1. Administrator inputs his information to the system until a successful login.
2. Admin is able to view reports on enrollment information.

Extensions or Alternate Flows: None.
Exceptions:

- Incorrect login and needs to input the information again.
- System does not have enough information to generate reports.

Related Use Cases: None.

7. Project Organization Content

7.1. Calendar Overview

Meeting	Dates	Attendees	Agenda
1.	September 11th, 2024 (Discord)	Roy Alkhoury Anthony Josh Legrama Janelle Kwofie Bach Ngo	● Discuss SRS and how we want to divide work. ● Officially welcome Janelle to the team
2.	September 18th, 2024 (Discord)	Roy Alkhoury Anthony Josh Legrama Janelle Kwofie Bach Ngo	● Go over current progress in each Section.
3.	September 23rd, 2024 (Discord)	Roy Alkhoury Anthony Josh Legrama Janelle Kwofie Bach Ngo	● Go over current progress in each Section.
4.	September 25th, 2024 (Discord)	Roy Alkhoury Anthony Josh Legrama Janelle Kwofie Bach Ngo	● Go over current progress in each Section. ● Officially approve work to add to SRS. ● Create Presentation for 9/30.
5.	September 30th, 2024 (Discord)	Roy Alkhoury Anthony Josh Legrama Janelle Kwofie Bach Ngo	● Go over any presentation and any comments from the professor. ● Prepare SRS for submission.

7.2. Meeting Minutes

Project Meeting: September 11th, 2024

NOTES:
- Janelle did a brief introduction to the team and explained her current ideas for the project.
- The team goes over their current outline for the project and what Janelle could take on.

TO DO's:
- Janelle will take Section One and Four, and serve as a fill in for work that needs to be completed.
- Roy will take Section Two, and serve as a fill in for work that needs to be completed.
- Bach will take Section Three and assist on the diagrams.
- Anthony will take the lead on the diagrams.

USAGE IDEA:
- College Enrollment System with main actors Student and Administrator.

RESOURCES:
- Google Drive: CS 401 Project
- Check email for GitHub link.

Project Meeting: September 18th, 2024

NOTES:
- Everyone went over their current ideas for each Section.
- SRS Document was created but needs to be formatted before use.

TO DO's:
- Janelle needs to format the SRS. Everyone will work on separate documents until the formatting issues are fixed

USAGE IDEA:
- See previous meeting.

RESOURCES:
- See previous meeting.

Project Meeting: September 23rd, 2024

NOTES:
- Officially approved everyone's ideas for the SRS.
- SRS is now fully operational for edits.

TO DO's:
- Finish assigned Section for SRS.
- Create Presentation.

USAGE IDEA:
- See previous meeting.

RESOURCES:
- See previous meeting.

Project Meeting: September 27th, 2024

NOTES:

- Bach volunteers to routinely check the document and add revisions and comments.
- Goes over current work done on the SRS.
- During meetings people switch sub-sections that they are confused on or that they believe they can improve upon.

TO DO's:

- Janelle takes Section 2.1.
- Bach takes over Section Four to add more information.
- Anthony also takes on Section Four to add more information.
- Roy takes on Section Three to add more information.
- Reformat Presentation and finalize the content on the slides.

USAGE IDEA:

- See previous meeting.

RESOURCES:

- See previous meeting.

Project Meeting: September 30th, 2024

NOTES:

- Went over UML and decided to add a couple more Use Cases and Use Case Diagrams.
- SRS is almost completed, grammar and formatting edits will be done until Tuesday.

TO DO's:

- Complete Section 6 and the Sequence Diagram.

USAGE IDEA:

- See previous meeting.

RESOURCES:

- See previous meeting.

Project Meeting: October 1, 2024

NOTES:

- Finished the Sequence Diagram.
- SRS is finalized. Grammar and formatting edits were all fixed.

TO DO's:

- Turn in the Phase One Slides and SRS to Canvas.

USAGE IDEA:

- See previous meeting.

RESOURCES:

- See previous meeting.

7.3. Project Timeline

WBS NUMBER	TASK TITLE	TASK OWNER	START DATE	DUE DATE	PCT OF TASK COMPLETE
1	**Phase 1 - SRS Document**				
1.1	Section 1	Janelle	9/5/24	9/30/24	100%
1.2	Section 2	Roy	9/5/24	9/30/24	100%
1.3	Section 3 - Functional Requirements	Bach & Roy	9/5/24	9/30/24	100%
1.4	Section 4 - Non-Functional Requirements	Anthony	9/5/24	9/30/24	100%
1.5	Section 5 - Diagrams	Anthony & Bach	9/5/24	9/30/24	100%
1.6	Section 6 - Use Case Specifications	Janelle & Roy	9/5/24	10/2/24	100%
1.7	Section 7 & 8 - Meeting Mins & Timeline	Janelle & Bach	9/5/24	10/2/24	100%
2	**Phase 2 - ???**				
2.1	???	???	10/3/24	10/30/24	0%
2.2	???	???	10/3/24	10/30/24	0%
2.3	???	???	10/3/24	10/30/24	0%
2.4	???	???	10/3/24	10/30/24	0%
3	**Final Phase - ???**				
3.1	???	???	10/31/24	12/2/24	0%
3.2	???	???	10/31/24	12/2/24	0%
3.3	???	???	10/31/24	12/2/24	0%
3.4	???	???	10/31/24	12/2/24	0%