RISE-EDU

Software Requirements Specification

Revision History

Date	Revision	Description	Author
09/17/2025	1.0	Initial Version	Wail Mohammed
09/23/2025	1.1	Purpose, Scope, Definitions Updated	Wail, Emmanuel, Shichang, Yesenia
09/24/2025	1.2	Assumptions and Internal/External Requirements Update	Wail
09/25/2025	1.3	Assumptions and Course Planner Non-Functional Environmental Requirement Update	Shichang
09/25/2025	1.4	Functional Student Module Requirements Non-Functional Performance Requirements	Emmanuel
09/28/2025	1.5	Added a Use Case	Emmanuel
9/29/2025	1.6	Added Requirements	Yesenia Ruiz
09/29/2025	1.7	Updated Two Use Cases	Wail Mohammed
09/29/2025	1.8	Updated Two Use Cases-Course Planner/Reporting	Shichang Wang
09/29/2025	1.9	Updated UCs: 01,02,03,04,05,06,07,08,09	Wail Mohammed
9/30/2025	1.10	Added Use Cases 12 and 13	Yesenia Ruiz
09/30/2025	1.11	Added Use Case:10, 11-Report module	Shichang Wang

Table of Contents

1. Purpose

This document outlines the requirements for the College Course Enrollment System Project.

1.1. Scope

This document will catalog the user, system, and hardware requirements for the CCES (College Course Enrollment) system. It will not, however, document how these requirements will be implemented.

1.2. Definitions, Acronyms, Abbreviations

CCES: College Course Enrollment System

1.3. References

Use Case Specification Document

UML Use Case Diagrams Document

Class Diagrams

Sequence Diagrams

1.4. Overview

The CCES (College Course Enrollment System) allows for the creation of college course schedules by administrators and allows students to enroll in these courses. The system will support class sizes, waiting lists, prerequisites, and reports. This system supports a network of universities, students, and Administrators. This is a Java application with a GUI that operates over TCP/IP. This system requires a server application and a client application. There is no web or HTML component.

2. Overall Description

2.1. Product Perspective

The CCES system is a platform designed for university students and administrators. Administrators can control the number of courses, class size, waiting list size, prerequisites lists, and issue reports. Students can enroll in courses, drop courses, and view schedules,

2.2. Product Architecture

The system will be organized into three major modules: the User module, the Course Scheduler module, and the Reporting (log) module.

2.3. Product Functionality/Features

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

2.3.1 The enrollment system must have an authentication, authorization and account creation system that differentiates between school admins and students based on their student or admin existing IDs.

2.3.2 Students will be able to access their online accounts, and input their name, email address, major and school year. Similarly, admins will be able to access their online accounts, and input their name, email address.

2.3.3 Students must be able to view college courses attributes such as; class IDs, names, times, days, instructor names, number of units, credit/no credit, number of available seats, and available waitlists .

2.3.4 School admin users must be able to view all class info, class availability status and student info.

2.3.5 The system must be large enough to handle and be able to store student account data, the list of all active courses, and the list of all waitlisted courses.

2.3.6 The system must allow admins to store and update a running list of courses with all courses attributes.

2.3.7 The system must allow students to view courses, must check if students are eligible to enroll, drop, waitlist, and/or withdraw from courses. The system must not give any students access to other student's accounts.

2.3.8 The system must allow admins only to add/edit and delete courses and drop students if needed.

2.3.9 The system must be available to all users (students and admins) during course enrollment periods, and take into account earlier registration for students who have higher priority.

2.3.10 The system must be able to update course info as well as students records in real time. Admins must also be able to see the up-to-date students and classes records.

2.4. Constraints - Yesenia

List appropriate constraints.

Constraint example: Since users may use any web browser to access the system, no browser-specific code is to be used in the system.

2.5. Assumptions and Dependencies—shichang wang and Wail

Wail:

2.5.1. All users will have network access to connect to the system.

2.5.2. It is assumed that there will be two user roles within the system. The school administrator and the student role.

2.5.3. It is assumed that school administrators will maintain course addition, deletion, and maintain prerequisite requirements for courses that need it.

2.5.4. It is assumed that students are associated with one university.

2.5.5. It is assumed that each school has unique courses.

2.5.6 It is assumed that the maximum number of users at a given time is 15,000.

Shichang Wang:

2.5.7 student and administrators will

2.5.8 It is assumed that students are responsible for checking their enrollment results, including waitlist status.

2.5.9 It is assumed that students will enter correct personal information when creating their accounts.

2.5.10 It is assumed that students will check the system frequently during the enrollment period for updates.

2.5.11 It is assumed that students will complete their enrollment within the deadlines set by the university.

2.5.12 It is assumed that the reporting module will only generate reports based on data already stored in the system.

2.5.40 It is assumed that the reporting module does not require real-time analytics, only accurate record keeping.

3. Specific Requirements

3.1. Functional Requirements

3.1.1. Common Requirements:

3.1.1.1 Users should be allowed to log in using their issued id and pin, both of which are alphanumeric strings between 6 and 20 characters in length.

3.1.1.2 The system should recognize and differentiate between student users and supervisor users.

3.1.1.3

3.1.2. __The Student ___ Module Requirements:-Emmanuel

Provide module specific requirements as appropriate.

Example:

3.1.2.1 Users should be allowed to log in using their issued id and pin, both of which are alphanumeric strings between 6 and 20 characters in length.

3.1.2.1 Students shall be allowed access into the school's system by entering their username, Student ID detail, Password all of which must be alphanumeric strings between 5 and 25 characters long.

3.1.2.2 Students shall be allowed to look up courses by Course names, Course ID, time, which semester those courses would be offered, and course information.

3.1.2.3 The system shall enforce all course prerequisites are satisfied before successfully enrolling in a course.

3.1.2.4 The system shall check the availability of seats in the class before full registration of courses is carried out.

3.1.2.5 The system shall ask student if they would want to be placed on waitlist, add a class, or drop a class

3.1.2.6 The system shall place students on waitlist if there are no available seats in a course during registration.

3.1.3. _The School Administrator____ Module Requirements: - Yesenia

3.1.2.1 The Administrator will have a username and password to login into the system. Both the username and the password will be alphanumeric between 6 and 20 characters in length.

3.1.2.2 The Administrator will be allowed to create courses, edit courses, and remove courses.

3.1.2.3 The Administrator will be allowed to see which students are enrolled in the course and who is on a waiting list.

3.1.2.4

3.1.4. __Course Planner/Reporting___ Module Requirements: -Shichang Wang(Reporting (log))

3.1.4.1

The system should keep track of every enrollment action that students make, such as add, drop, or waitlist for each course. It needs to save the student ID, the course ID, and also the timestaps. This is important because otherwise there is no way to trace back what the student has done before.

3.1.4.2

Admins should be able to make reports that show how many people are in a class and also how many are on the waitlist. These numbers are useful so that admins can see which classes are full and maybe open another section. Without this feature, they won't know how to adjust class capacity.

3.1.4.3

The system will let administrators check a student's past enrollment history. For example, what courses the student added before or dropped before. This way if a student has questions or a problem, the admin can look it up and confirm.

3.1.4.4

After a student makes a change, the reporting module should update almost synchronously(avoid **commercial advertisement**) so students can see the result. For example, if a student adds a course, it should appear in their schedule when they switch the pages. The idea is that the update must be shown right after the action is completed.

3.1.4.5

The system also needs to display error messages whenever students make invalid action. Like if a student tries to register for a class but didn't meet the prerequisites, it should display "you need finished… to register this course".

3.1.4.6

The system will generate official documents such as transcripts or enrollment reports. These should come out as read-only files. Students can only view and print the document, while only authorized faculty or admins can edit. This makes sure the reports are secure and reliable.

3.2. External Interface Requirements-Wail

3.2.1 The system must provide an interface to the University billing system administered by the Bursar's office so that students can be automatically billed for the courses in which they have enrolled. The interface is to be in a comma-separated text file containing the following fields: student id, course id, term id, action. Where "action" is whether the student has added or dropped the course. The file will be exported nightly and will contain new transactions only.

3.2.2 The system interface must be clear, intuitive and provide a consistent user experience.

3.2.3 The system must provide error messages when invalid actions take place.

3.3. Internal Interface Requirements - Wail

3.3.1 The system must use a client-server communication via TCP/IP network.
3.3.2 The system must process a data-feed from the enrollment system such that the student updated course records are stored in the form of a comma-separated interface file that is exported from the enrollment system upon user's demand.
3.3.3 The data feed from the enrollment system must have the following fields included in the file ; student name, student id, major, school name, name and number of all courses enrolled and withdrawn.

4. Non-Functional Requirements - ALL

4.1. Security and Privacy Requirements - Yesenia

Example:
4.1.1 The System must encrypt data being transmitted over the Internet.
4.1.2 The System must not allow students to view each others data, meaning only the student should be able to view their own data.
4.1.2 The System must allow administrators to only view necessary student information, not where they live and other data like that.

4.2. Environmental Requirements- shichang wang

4.2.1 The system shall run on the university's existing Linux-based server infrastructure.
4.2.2 The system shall operate properly with the university's existing network and power infrastructure.
4.2.3 The system shall not require high-performance machines; any standard desktop used by students or administrators will be sufficient to run the client.
4.2.4 The system shall be deployable on existing university servers without requiring hardware upgrades.
4.2.5 The system shall remain compatible with future minor upgrades to the university's operating systems.

4.3. Performance Requirements - Emmanuel

4.3.1 The system shall accommodate 15,000 users during the peak usage time window of 8:00 am to 12 pm local time.
4.3.3 The system shall display the status of course enrollments, dropped courses, waitlisted positions after the operation is carried out.
4.3.4 The system shall show a list of available courses within 5 seconds when a student tries to search through.

5. Use Case Specifications Document

Use Case 1: Manage Student Sign up
Use Case ID: UC01
Use Case Name: Manage Student Sign up
Relevant Requirements:
· Requirements Document ID: Security and Privacy Document 4.1
Primary Actor:
· Student
Pre-conditions:
· The server is online, and the user client is connected to the server.
· The student must have access to the system.
Post-conditions:
· The user credentials are stored in a data file.
· The student is registered in the system.
· Appropriate user role is given to students.
Basic Flow or Main Scenario:
1) The user navigates to the sign-up page.
2) The user is prompted to enter the student's email address and a temporary default password.
3) The user is prompted to enter personal details, such as Name, Phone number, Mailing Address, and Billing Address.
4) The student submits the form.
5) The system stores the information, validates it, and creates a student account.
6) The system sends a success message to the user.
7) The system redirects the user to the main dashboard to log in.
Extensions or Alternate Flows:
· Invalid information at sign up
If the user enters invalid current student information, per the system’s security requirements, the system displays an error message, “Invalid Personal Information Provided”, and asks the user to retry signing up again.
Exceptions:
· If the authentication system or server is unavailable, an error message is displayed to the user, and is prompted to retry again.
Related Use Cases:
· UC02 Manage User Login

Use Case ID: UC02
Use Case Name: Manage User Login
Relevant Requirements:
· Requirements Document ID: Security and Privacy Document 4.1
Primary Actor:
· Student
· School Supervisor
Pre-conditions:
· The server is online and the user client is connected to the server.
· The user must have an existing account and be active.
· The user credentials are stored in a data file.
· The client has access to the user list.
Post-conditions:
· User credentials are checked.
· Session is created.
· Appropriate user roles and access are given to students and supervisors.
Basic Flow or Main Scenario:
1) The user navigates to the login page.
2) The user is prompted to enter the username and password .
3) The user enters the username and password, then hits submit.
4) The system verifies the credentials.
5) The system assigns a role (supervisor or student) based on the account type.
6) If this is the first time the user logs in or if a password reset is needed:
a) The system redirects to the account management page.
b) The user must either change their default username and/or password per system security requirements.
7) The system validates the changes and updates the user records.
8) The system creates an authenticated session.
9) The system redirects the user to the main dashboard.
Extensions or Alternate Flows:
· **Invalid Credentials at log in**:
If the user enters an invalid current credentials, per the system's security requirements, the system displays an error message, "Invalid Username or password", and asks the user to retry.
· **Invalid Credentials at update page**:
If the user enters an invalid username or password per the system's security requirements, the system displays an error message "Invalid Username or Password" and asks the user to retry.
Exceptions:
· If the authentication system or server is unavailable, an error message is displayed to the user.
Related Use Cases:
· UC03 Course Enrollment.

Use Case ID: UC03
Use Case Name: Course Enrollment
Relevant Requirements:

-

Primary Actor:

- Student.
- Supervisor

Pre-conditions:

- The server is online, and the user client is connected to the server.
- The user must be logged into an active session in the system.
- Course registration add/drop period is active.

Post-conditions:

- The student is enrolled in selected active courses.

Basic Flow or Main Scenario:

1) The user navigates to the “Class Enrollment” page.
2) The system checks if the course registration date is active.
3) The system displays the current active classes for the current semester.
4) The user selects the course to be enrolled in.
5) If the selected course day and time is not equal to another existing enrolled course day and time:
    a) The system allows the user to proceed to the next step.
    b) Otherwise, the system prompts the user with a time conflict message and redirects back to the previous step(3).
6) The system checks for students' holds, prerequisite courses, and available spots.
7) If system checks are all valid:
    a) The student is enrolled, and a confirmation message is shown to the user.
    b) Otherwise, the system prompts the user with an error message and redirects back to the previous step(3).

Extensions or Alternate Flows:

- If the user is a school supervisor, the system allows the same basic flow.

Exceptions:

· If the class management system or server fails, an error message is displayed to the user.

Related Use Cases:

- UC01 Manage Student Sign up
- UC02 User Login

Use Case ID: UC04
Use Case Name: Course Drop
Relevant Requirements:

-

Primary Actor:

- Student.
- Supervisor

Pre-conditions:

- The server is online, and the user client is connected to the server.
- The user must be logged into an active session in the system.
- Course registration add/drop period is active.
- The student must be enrolled in at least one class.

Post-conditions:

- The student is dropped from selected active courses.

Basic Flow or Main Scenario:

1) The user navigates to the “Class Enrollment” page.
2) The system checks if the course registration add/drop period is active.
3) The system displays the student’s current active enrolled classes for the current semester.
4) The user selects the course to be dropped.
5) The system displays a confirmation message before dropping the class.
6) If the user confirms, the student is dropped from selected class or classes, and a confirmation message is shown to the user.

Extensions or Alternate Flows:

- If the user is a school supervisor, the system allows the same basic flow.

Exceptions:

· If the class management system or server fails, an error message is displayed to the user.

Related Use Cases:

- UC01 Student Registration
- UC02 User Login
- UC03 Course Enrollement

Use Case ID: UC05
Use Case Name: Course Waitlist
Relevant Requirements:

-

Primary Actor:

- Student.
- Supervisor

Pre-conditions:

- The server is online, and the user client is connected to the server.
- The user must be logged into an active session in the system.
- Course registration add/drop period is active.
- Available spots in a selected course/courses exceed the maximum class size.

Post-conditions:

- The student is added to a waiting list for the selected active course/courses.

Basic Flow or Main Scenario:

1) The user navigates to the “Class Enrollment” page.
2) The system checks if the course registration add/drop period is active.
3) The user selects the course to be added.
4) The system checks for students' holds, prerequisite courses, and available spots.
5) The system detects that available slots exceed the maximum class size.
6) The system displays an error message and prompts the user to be waitlisted.
7) If the user confirms, the student is added to the waitlist for the selected class or classes, and a confirmation message is shown to the user.

Extensions or Alternate Flows:

- If the user is a school supervisor, the system allows the same basic flow.

Exceptions:

- If the class management system or server fails, an error message is displayed to the user.

Related Use Cases:

- UC01 Student Registration
- UC02 User Login
- UC03 Course Enrollement

Use Case ID: UC06
Use Case Name: Course Prerequisites Check
Relevant Requirements:

-

Primary Actor:

- Student.
- Supervisor

Pre-conditions:

- The server is online, and the user client is connected to the server.
- The user must be logged into an active session in the system.
- Course registration add/drop period is active.
- Selected course/courses have prerequisites.

Post-conditions:

- The user is allowed or denied to sign up or waitlist courses.

Basic Flow or Main Scenario:

1) The user selects the course to be added.
2) The system checks the student's completed course list.
3) The system checks the required prerequisites list for the selected course.
5) If prerequisites are found in the student's completed course list:
    a) The system displays "All prerequisites have been met for this course" and allows the user to proceed with UC03 and UC05.
    b) The system displays an error message and prompts the user to exit.

Extensions or Alternate Flows:

- If the user is a school supervisor, the system allows the same basic flow.

Exceptions:

- If the class management system or server fails, an error message is displayed to the user.

Related Use Cases:

- UC01 Student Registration
- UC02 User Login
- UC03 Course Enrollement
- UC05 Course Waitlist

Use Case ID: UC07
Use Case Name: Access Student Class Schedule
Relevant Requirements:

- Requirements Document ID: Security and Privacy Document 4.1

Primary Actor:

- Student.

Pre-conditions:

- The server is online and user client is connected to server.
- The user must have an existing account.
- The student must be logged into an active session in the system.

Post-conditions:

- The student can view current enrolled classes for the semester at any time.The student must be logged in to the system.
- The student can view course name, course day and time, instructor name, location.

Basic Flow or Main Scenario:

1) The student navigates to the “Class Schedule” page.
2) The system retrieves the student’s current enrolled classes if any, for the current semester.
3) The system displays the course name, course ID, course day and time, instructor name, class location.

Extensions or Alternate Flows:

- If the student is not enrolled in any classes, the system displays a message to the user “No Current Enrollments Found”.
- The student can download or print the course schedule.

Exceptions:

· If the class management system or server fails, an error message is displayed to the user.

Related Use Cases:

- UC01 User Login
- UC02 Student Registration
- Course Sign up
- Course Drop
- Course Withdraw
- Waitlist Management

Use Case ID: UC08
Use Case Name: Manage Student Payment
Relevant Requirements:

- Requirements Document ID: Security and Privacy Document 4.1

Primary Actor:

- Student
- Payment System

Pre-conditions:

- The server is online, and the user client is connected to the server.
- The student is logged into the system.
- The student is enrolled in at least one class.
- The payment system is active.

Post-conditions:

- The payment is successful.
- The payment log/receipt is generated.

Basic Flow or Main Scenario:

1) The user navigates to the payment section of the system.
2) The user is presented with the number of classes currently enrolled and a payment cost for each class.
3) The user is presented with the total for payment with the option to pay now / pay later.
4) If the student selects Pay Now:
    a. The system redirects to the payment portal.
    b. The student enters payment details.
    c. The system processes the payment.
    d. If payment is successful, the system generates a receipt and redirects to the main dashboard.
5) If the student selects pay later, the system redirects to the main dashboard.

Extensions or Alternate Flows:

- Invalid Payment:

    If the student's payment transaction fails, the system displays an error message, “Invalid Payment”, and prompts the user to retry.

  Missing Payment:

    If the student misses the payment deadline to pay for the classes, a hold is added to the student’s account, which can only be removed by a school administrator.

Exceptions:

- If the payment system or server is unavailable, an error message is displayed to the user.

Related Use Cases:

- UC05 Course Planner and Enrollment.
- UC08 Hold Management

Use Case ID: UC09
Use Case Name: Manage Student Hold
Relevant Requirements:

-

Primary Actor:

- Supervisor

Pre-conditions:

- The server is online, and the user client is connected to the server.
- The user must be logged into an active session in the system.
- The student account must exist and have a hold.

Post-conditions:

- The student's account hold is removed.

Basic Flow or Main Scenario:

1) The school administrator logs in to the system..
2) The administrator navigates to the student's account.
3) The administrator removes the hold from the student's account.
5) The system sends the user a confirmation message, and the student account hold is removed.

Extensions or Alternate Flows:

- If the user is a student and tries to remove an account hold, an error message is displayed to let the user know of the error.

Exceptions:

- If the class management system or server fails, an error message is displayed to the user.

Related Use Cases:

- UC01 Student Registration
- UC02 User Login
- UC03 Course Enrollement
- UC05 Course Waitlist
- UC08 Manage Student Payment

Use Case ID: UC10
Use Case Name: Enrollment & waitlist Report
Relevant Requirements:

- Course enrollment data

Primary Actor:

- Supervisor(Admin)

Pre-conditions:

- The Supervisor is already logged in to the system.
- Enrollment data exists.

Post-conditions:

- The report is displayed on the screen.
- The supervisor can export the report.

Basic Flow or Main Scenario:

1) The supervisor logged in to the system and went to the report module.
2) The target course has been selected.
3) The system gathers information.
4) Report display on the screen.
5) The supervisor can export the report.

Extensions or Alternate Flows:

- If the supervisor exports the report, the report will be in pdf format.
- If the supervisor didn't choose any course, the system will prompt the supervisor to choose at least one course.
- If there is no student in the waiting list, the report still shows a waiting list with 0.

Exceptions:

- If the system can't access the enrollment data, the system will display the error message and record the error.

Related Use Cases:

- UC02 Manage User Login
- UC03 Course Enrollment
- UC04 Course Course Drop
- UC05 Course Waitlist

Use Case ID: UC11
Use Case Name: Real Time Report Update
Relevant Requirements:

- The system should update immediately after students make an action.

Primary Actor:

- Report module

Pre-conditions:

- The server is online and the system is running normally.
- At least one student made an action.

Post-conditions:

- Report module update immediately to display the latest status.
- Student's schedule updates immediately.

Basic Flow or Main Scenario:

1) the student logging in to the system.
2) Students make an enrollment action(add, drop and waiting list).
3) System record the transaction(student ID, Course number and timestamp)
4) The reporting module updates the enrollment and waitlist status.

Extensions or Alternate Flows:

- If more than one enrollment action made at same time, the system will group them and update together.
- If the update takes a long time, we will display a "loading…" message until the update is finished.

Exceptions:

- If the Report module can't reach enrollment data, the page keeps the last known information and displays an error message "System is temporarily unavailable".

Related Use Cases:

- UC02 User Login
- UC03 Course Enrollment
- UC04 Course Drop
- UC05 Course Waitlist
- UC10 Enrollment & Waitlist Report

- UC08 Hold Management

Use Case ID: UC12

Use Case Name: Creating courses

Relevant Requirements:

· Requirements Document ID: Security and Privacy Document 4.1

Primary Actor:

· Administrator

Pre-conditions:

· The server is online, and the user client is connected to the server.

· The administrator must have access to the system.

· The administrator is registered in the system as an administrator.

Post-conditions:

· A new course is added into the course catalog.

Basic Flow or Main Scenario:

1. The administrator navigates to the catalog of courses page.
2. The administrator selects to create a new course.
3. The administrator enters in class size, subject, room number, professor's name, and the date and times the class will take place.
4. The administrator submits the new course information.
5. The administrator publishes the course after revision.
6. The administrator is redirected by the system to view the published course.

Extensions or Alternate Flows:

· Invalid information for new course - professor already teaching class

If the administrator enters a professor's name but the professor is already teaching a diff course at the same time then the system will highlight the professor's name and the time the course is in red.

· Invalid information for new course - dates don't align with semester schedule.

If the administrator enters a date that does not match with the semester dates, then the system will highlight those dates in red.

Exceptions:

· If the server is unavailable, an error message is displayed to the user, and is prompted to retry again.

· If the system doesn't allow the administrator to publish the course then the administrator will be given the option to open a ticket for support.

Related Use Cases:

·

Use Case ID: UC13
Use Case Name: Editing courses
Relevant Requirements:
· Requirements Document ID: Security and Privacy Document 4.1
Primary Actor:
· Administrator
Pre-conditions:
· The server is online, and the user client is connected to the server.
· The administrator must have access to the system.
· The administrator is registered in the system as an administrator.
Post-conditions:
· A course has been edited in the course catalog.
Basic Flow or Main Scenario:
1. The administrator navigates to the catalog of courses page.
2. The administrator selects to edit a course.
3. The administrator can edit any of the following class size, subject, room number, professor's name, and the date and times the class will take place.
4. After the administrator is done editing the course they will submit the new course.
5. The administrator then will review the information changed and publish the course.
6. The system will redirect the administrator to the course that was just edited and is now republished in the catalog.

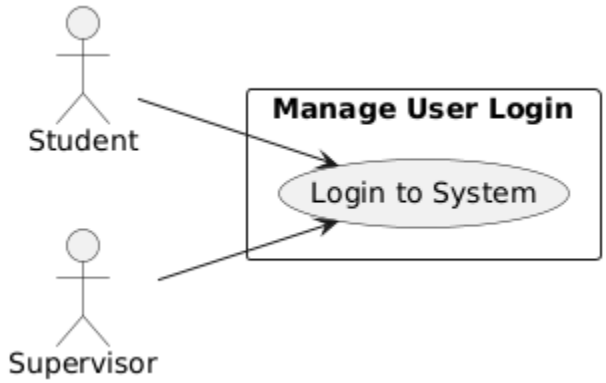
Extensions or Alternate Flows:
· Invalid information for editing a course - professor already teaching class
If the administrator enters a professor's name but the professor is already teaching a diff course at the same time then the system will highlight the professor's name and the time the course is in red.
· Invalid information for editing a course - dates don't align with the semester schedule.
If the administrator enters a date that does not match with the semester dates, then the system will highlight those dates in red.

Exceptions:
· If the server is unavailable, an error message is displayed to the user, and is prompted to retry again.
· If the system doesn't allow the administrator to publish the course then the administrator will be given the option to open a ticket for support.
Related Use Cases:

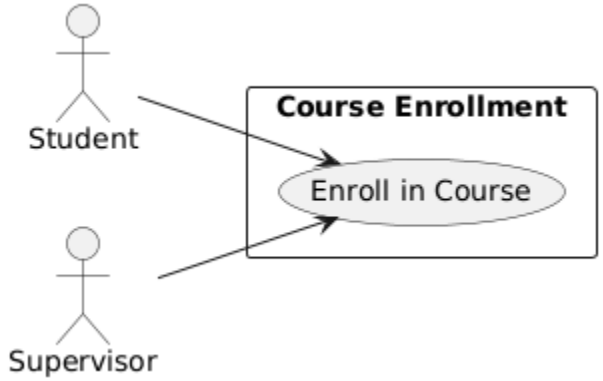
6. UML Use Case Diagrams

Use Case 1:

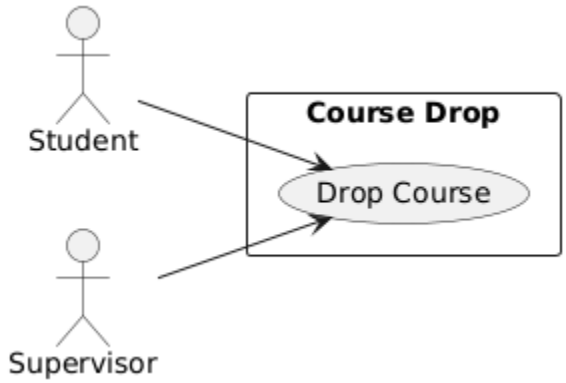
Use Case 2:



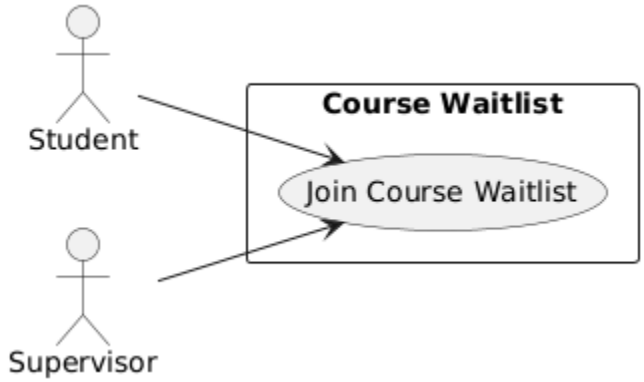
Use Case 3:



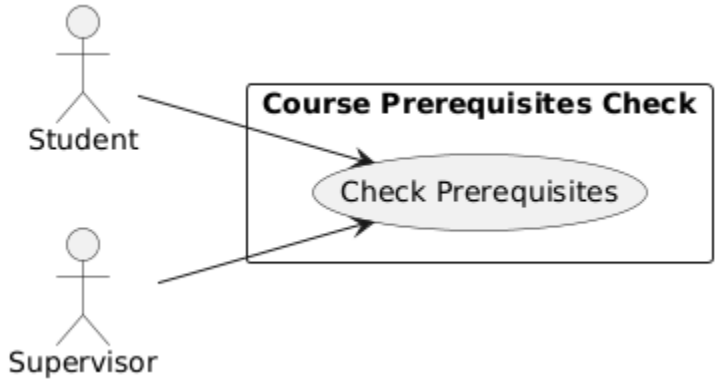
Use Case 4:



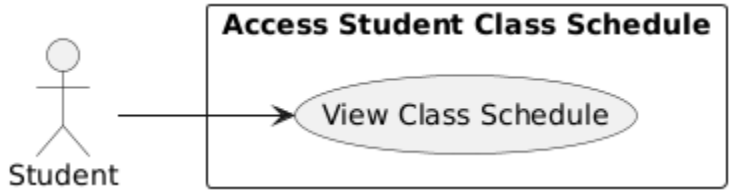
Use Case 5:



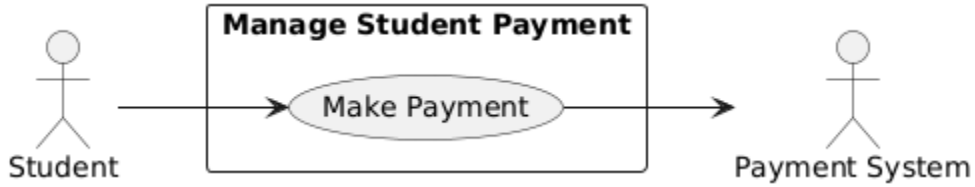
Use Case 6:



Use Case 7:



Use Case 8:



Use Case 9: