

RISE-EDU

Software Design Specification

Phase 2

Revision History

Date	Revision	Description	Author
10/15/2025	1.0	Initial Version	Wail Mohammed
10/15/2025	1.1	Purpose, Scope, Definitions Updated	Wail Mohammed
10/15/2025	1.2	Updated class candidates 01,02,03,04,05	Wail Mohammed
10/17/2025	1.3	Updated class candidates 06,07,08,09,10	Wail, Yesenia, Emmanuel, Shichang
10/23/2025	1.4	Updated Class Diagram and Sequence diagrams	Wail Mohammed
10/25/2025	1.5	Updated all class candidates, added class 11	Wail, Yesenia, Emmanuel
10/25/2025	1.6	Updated Use Case 1 and 2 with descriptions	Yesenia Ruiz
	1.7		
	1.8		
	1.9		
	1.10		
	1.11		
	1.12		
	1.13		

Table of Contents

1. PURPOSE	5
1.1. SCOPE	5
1.2. DEFINITIONS, ACRONYMS, ABBREVIATIONS.....	5
1.3. REFERENCES	5
1.4. OVERVIEW	5
2. OVERALL DESCRIPTION	6
2.1. PRODUCT PERSPECTIVE	6
2.2. PRODUCT ARCHITECTURE.....	6
2.3. PRODUCT FUNCTIONALITY/FEATURES.....	ERROR! BOOKMARK NOT DEFINED.
2.4. CONSTRAINTS	ERROR! BOOKMARK NOT DEFINED.
2.4. ASSUMPTIONS AND DEPENDENCIES	ERROR! BOOKMARK NOT DEFINED.
3. SPECIFIC REQUIREMENTS	ERROR! BOOKMARK NOT DEFINED.
3.1. FUNCTIONAL REQUIREMENTS	ERROR! BOOKMARK NOT DEFINED.
3.1.1. <i>Common Requirements:</i>	<i>Error! Bookmark not defined.</i>
3.1.2. <i>The Student Module Requirements:</i>	<i>Error! Bookmark not defined.</i>
3.1.3. <i>The School Administrator Module Requirements:</i>	<i>Error! Bookmark not defined.</i>
3.1.4. <i>The Reporting Module Requirements:</i>	<i>Error! Bookmark not defined.</i>
3.2. EXTERNAL INTERFACE REQUIREMENTS.....	ERROR! BOOKMARK NOT DEFINED.
3.3. INTERNAL INTERFACE REQUIREMENTS.....	ERROR! BOOKMARK NOT DEFINED.
4. NON-FUNCTIONAL REQUIREMENTS.....	ERROR! BOOKMARK NOT DEFINED.
4.1. SECURITY AND PRIVACY REQUIREMENTS.....	ERROR! BOOKMARK NOT DEFINED.
4.2. ENVIRONMENTAL REQUIREMENTS	ERROR! BOOKMARK NOT DEFINED.
4.3. PERFORMANCE REQUIREMENTS.....	ERROR! BOOKMARK NOT DEFINED.
5. USE CASE SPECIFICATION	ERROR! BOOKMARK NOT DEFINED.
USE CASE 1: MANAGE USER LOGIN	ERROR! BOOKMARK NOT DEFINED.
USE CASE 2: COURSE ENROLLMENT.....	ERROR! BOOKMARK NOT DEFINED.
USE CASE 3: COURSE DROP	ERROR! BOOKMARK NOT DEFINED.
USE CASE 4: COURSE WAITLIST	ERROR! BOOKMARK NOT DEFINED.
USE CASE 5: COURSE PREREQUISITES CHECK	ERROR! BOOKMARK NOT DEFINED.
USE CASE 6: ACCESS STUDENT CLASS SCHEDULE	ERROR! BOOKMARK NOT DEFINED.
USE CASE 7: MANAGE STUDENT HOLD	ERROR! BOOKMARK NOT DEFINED.
USE CASE 8: ENROLLMENT REPORTING	ERROR! BOOKMARK NOT DEFINED.
USE CASE 9: UPDATE CHANGES REPORT.....	ERROR! BOOKMARK NOT DEFINED.
USE CASE 10: CREATE COURSES.....	ERROR! BOOKMARK NOT DEFINED.
USE CASE 11: EDITING COURSES	ERROR! BOOKMARK NOT DEFINED.
6. UML USE CASE DIAGRAMS.....	ERROR! BOOKMARK NOT DEFINED.
USE CASE 1: MANAGE USER LOGIN	7
USE CASE 2: COURSE ENROLLMENT.....	8
USE CASE 3: COURSE DROP	9
USE CASE 4: COURSE WAITLIST	10
USE CASE 5: COURSE PREREQUISITES CHECK	10
USE CASE 6: ACCESS STUDENT CLASS SCHEDULE	12
USE CASE 7: MANAGE STUDENT HOLD	13
USE CASE 8: ENROLLMENT REPORTING	14
USE CASE 9: UPDATE CHANGES REPORT.....	15
USE CASE 10: CREATE COURSES.....	17
USE CASE 11: EDITING COURSES	ERROR! BOOKMARK NOT DEFINED.
7. CLASS DIAGRAMS.....	18
8. SEQUENCE DIAGRAMS	30
USE CASE 1: MANAGE USER LOGIN	30

USE CASE 2: COURSE ENROLLMENT.....	31
USE CASE 3: COURSE DROP.....	32
USE CASE 4: COURSE WAITLIST.....	33
USE CASE 5: COURSE PREREQUISITES CHECK.....	34
USE CASE 6: ACCESS STUDENT CLASS SCHEDULE.....	35
USE CASE 7: MANAGE STUDENT HOLD.....	36
USE CASE 8: ENROLLMENT REPORTING.....	37
USE CASE 9: UPDATE CHANGES REPORT.....	38
USE CASE 10: CREATE COURSES.....	39
USE CASE 11: EDITING COURSES.....	40

1. Purpose

This document outlines the system design for the College Course Enrollment System Project. This expands on the software requirements specifications and outlines the system architecture, components, use case classes designs, communication models and the overall implementation of the system

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. It will allow school administrators to create and manage the school's course schedule, while also providing students with tools to enroll, drop, and withdraw from courses. The system will also manage prerequisites for courses and allow users to waitlist if class sizes are full.

1.2. Definitions, Acronyms, Abbreviations

- 1.2.1 CCES: College Course Enrollment System
- 1.2.2 Student User: User that will be able to enroll, drop, waitlist and withdraw from classes.
- 1.2.3 Administrator: School admin user that will be responsible for managing course listings.
- 1.2.4 TCP/IP: A piece of software suite that will allow communication between client and server.
- 1.2.5 Client: Users interact with the client entity to be able to send and receive information from the server.
- 1.2.6 Server: The server is responsible for listening to and interacting with multiple clients at the same time, and managing information being received from the clients.
- 1.2.7 Add/Drop period is the same as registration period.
- 1.2.8 Withdrawal period is another period when student will be allowed to withdraw from a course.

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. Design Description

2.1. Product Perspective

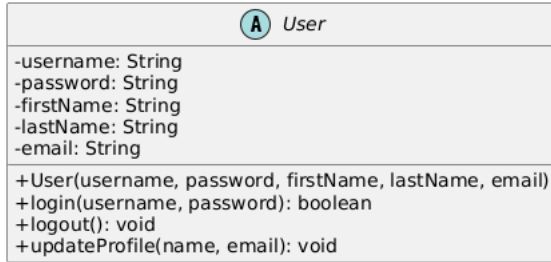
The CCES system is a platform designed for university students and administrators. Administrators can control the number of courses, class size, waiting list size, prerequisites list, and issue reports. Students can enroll in courses, drop courses, and withdraw from courses. The logging module enables administrators to issue reports and for students to display their class schedules.

2.2. System Architecture

2.2.1 (Talk about how the system is designed)


3. Class Design

Class 1: User




Note: may be updateprofile???

Class 2: Student

 Student
-studentID: String -major: String -schoolYear: String -hasPriorityRegistration: boolean
+Student(..., studentID, major, schoolYear) +viewCourseCatalog(): List<Course> +enrollInCourse(course): boolean +dropCourse(course): boolean +withdrawFromCourse(course): boolean +joinWaitlist(course): boolean +viewSchedule(): Schedule +checkPrerequisites(course): boolean +checkAvailability(course): int +viewHolds(): List<Hold>


Note: May remove hasPriority registration

Class 3: Administrator

 Administrator
-adminID: String
+Administrator(..., adminID) +createCourse(...): boolean +editCourse(course, details): boolean +deleteCourse(course): boolean +addStudentToCourse(student, course): boolean +dropStudentFromCourse(student, course): boolean +placeHoldOnAccount(student, holdReason): boolean +removeHoldFromAccount(student, hold): boolean +viewEnrollment(course): List<Student> +viewWaitlist(course): List<Student> +generateEnrollmentReport(course): Report +generateStudentReport(student): Report


Note: may include username/password/name and email as attributes.

Class 4: System Manager

 SystemManager
-addDropPeriod: DateRange -withdrawalPeriod: DateRange
+SystemManager() +addUniversity(universityName): University +getUniversity(universityID): University +authenticateUser(username, password, universityID): User +findCourse(courseID, universityID): Course +findStudent(studentID, universityID): Student +processEnrollment(student, course): boolean +processDrop(student, course): boolean +processWithdrawal(student, course): boolean

Note: please call date attribute as Date type and then look at including university id in the class methods.

Class 5: Course

 Course
<div><div>-courseID: String</div><div>-courseName: String</div><div>-instructorName: String</div><div>-time: String</div><div>-days: String</div><div>-location: String</div><div>-units: int</div><div>-classSize: int</div><div>-availableSeats: int</div></div>
<div><div>+ Course(courseID, courseName, ..., units, classSize)</div><div>+ addStudent(student): boolean</div><div>+ removeStudent(student): boolean</div><div>+ addToWaitlist(student): boolean</div><div>+ isFull(): boolean</div><div>+ getAvailableSeats(): int</div></div>


Note:

Also, look at creating courseList (list of courses)


Add CourseList or ListOfCourses or CourseCatalog

Class 6: Schedule (List of courses)


Note: Lets look at this later. (may be fine)

 Schedule
-semester: String
+Schedule(student, semester) +addCourse(course): boolean +removeCourse(course): boolean +displaySchedule(): String


Class 7: Hold

 Hold
-holdID: String -reason: String -datePlaced: Date
+Hold(reason, admin) +getHoldDetails(): String

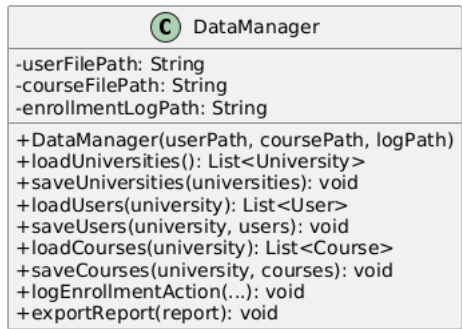
Class 8: Waitlist

 Waitlist
-maxWaitlistSize: int
+Waitlist(course, maxSize) +addStudent(student): boolean +promoteNextStudent(): Student +getPosition(student): int +getSize(): int


Class 9: Report

 Report
-reportID: String -reportType: String -generatedDate: Date -reportData: String
+Report(reportType) +generate(data): void +displayReport(): void +saveReport(): boolean

Class 10: Data Manager

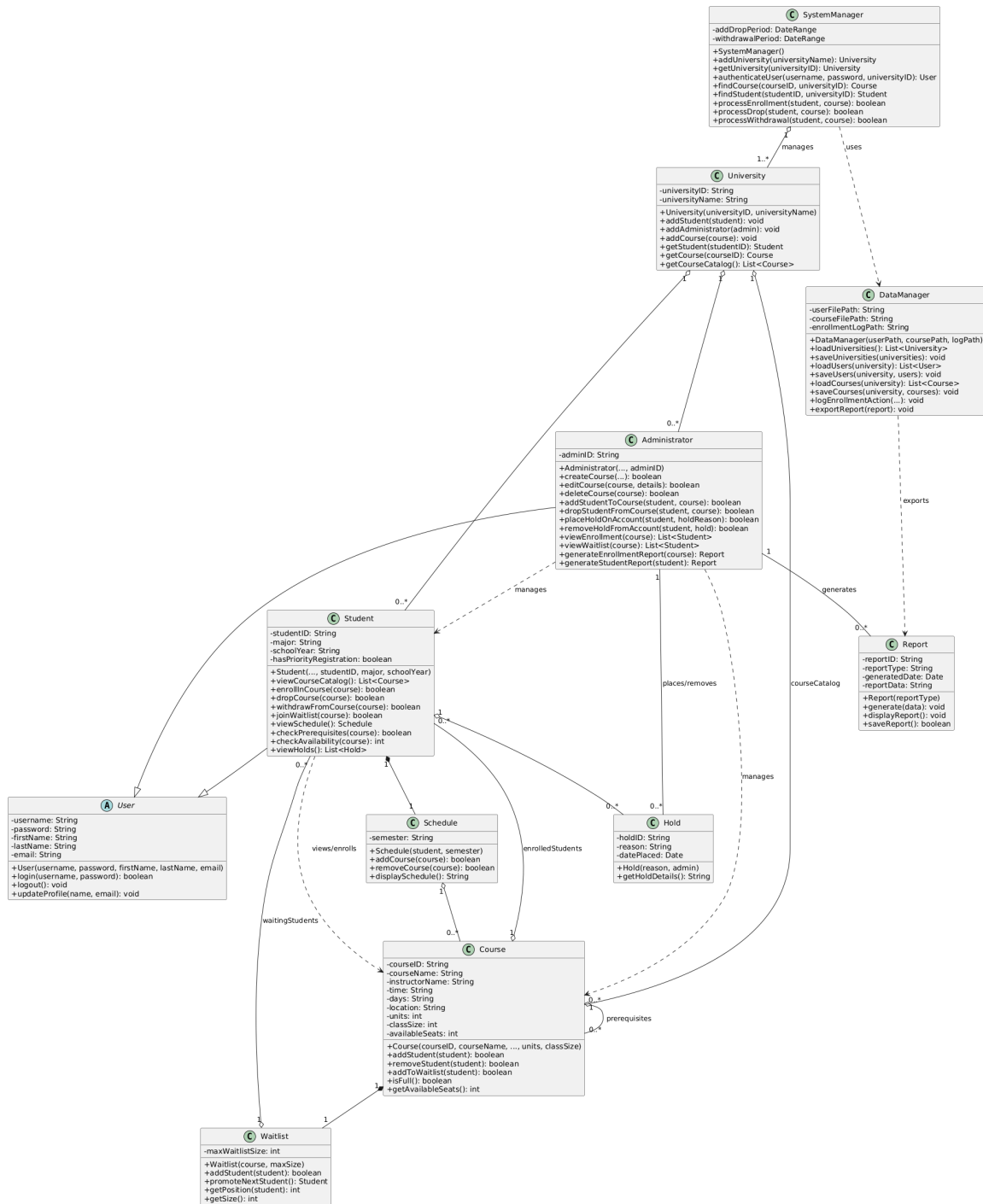


Class: 11: University

 University
-universityID: String -universityName: String
+University(universityID, universityName) +addStudent(student): void +addAdministrator(admin): void +addCourse(course): void +getStudent(studentID): Student +getCourse(courseID): Course +getCourseCatalog(): List<Course>

4. Class Diagram

RISE-EDU System Class Diagram (Multi-University)

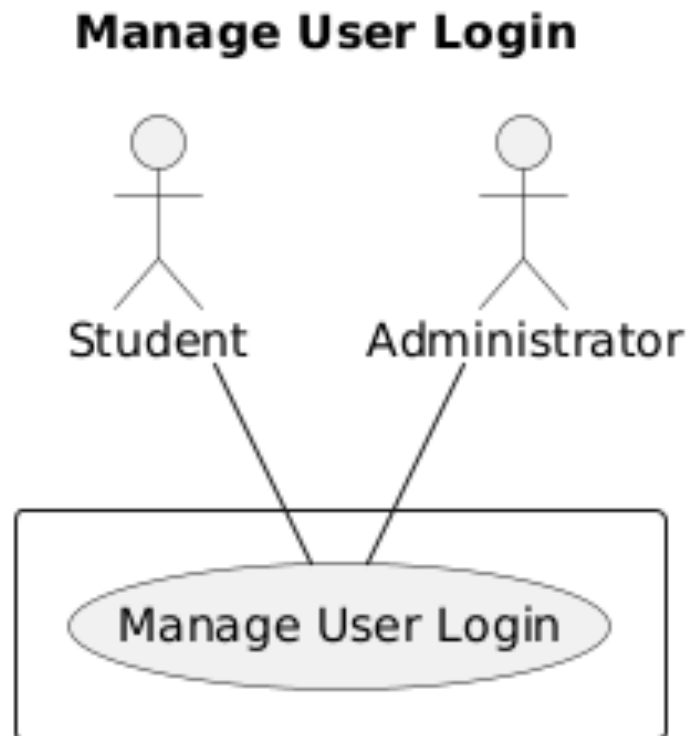


6. Use Cases

6.1 UC01 : Manage User Login

Actor : Student, Admin

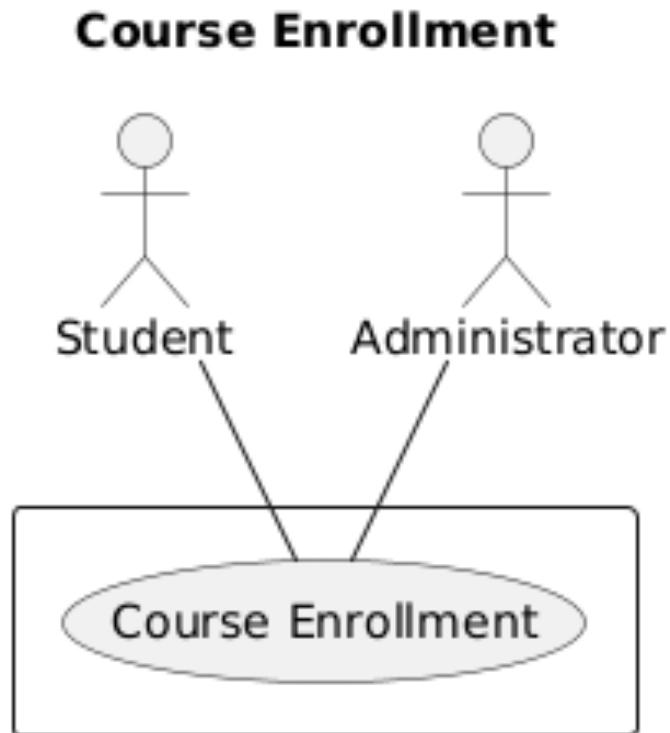
Description: The system will check to make sure the student and admin are logging in with the correct credentials.



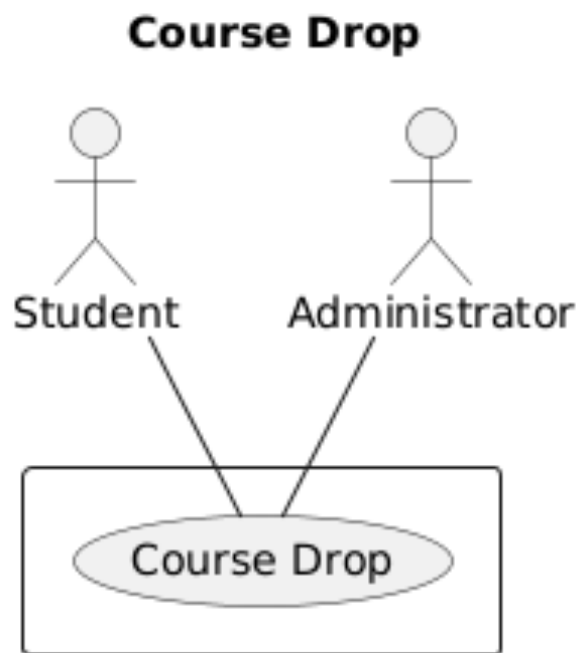
6.2 UC02: Course Enrollment

Actor: Student, Admin

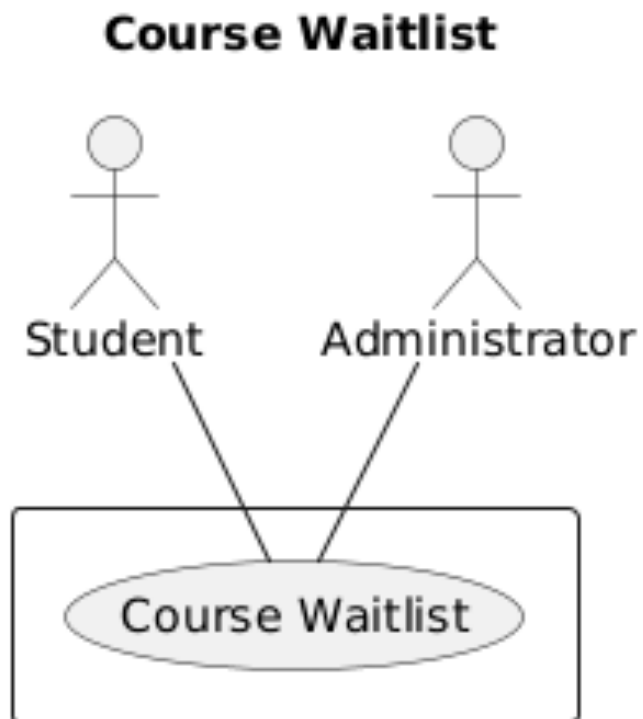
Description: The system will allow the student to enroll in a class, withdraw from a class, and drop a class. It will allow the admin to do the same for the student.



6.3 UC03: Course Drop



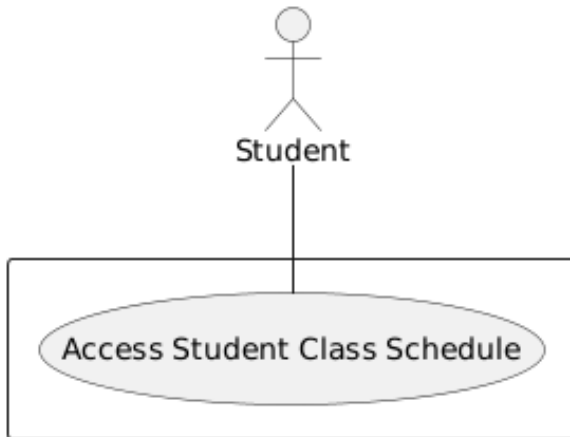
6.4 UC04: Course Waitlist



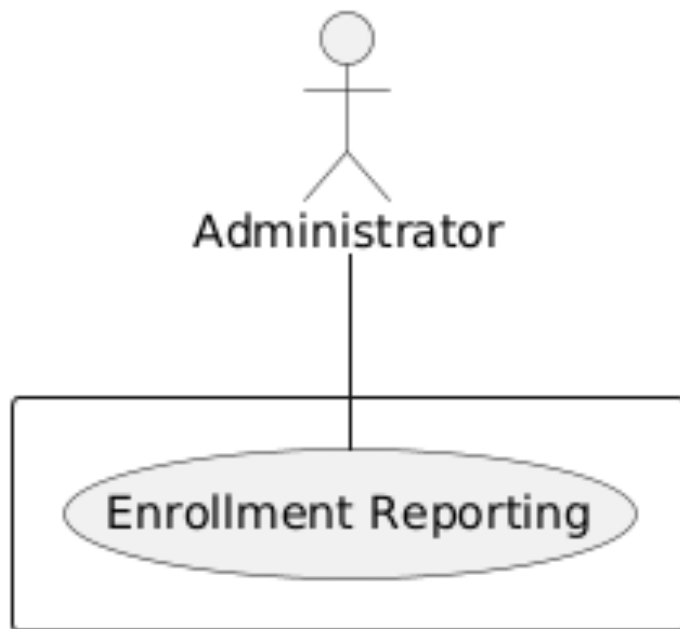
6.5 UC05: Course Prerequisites Check

6.6 UC06: Access Schedule

Access Student Class Schedule

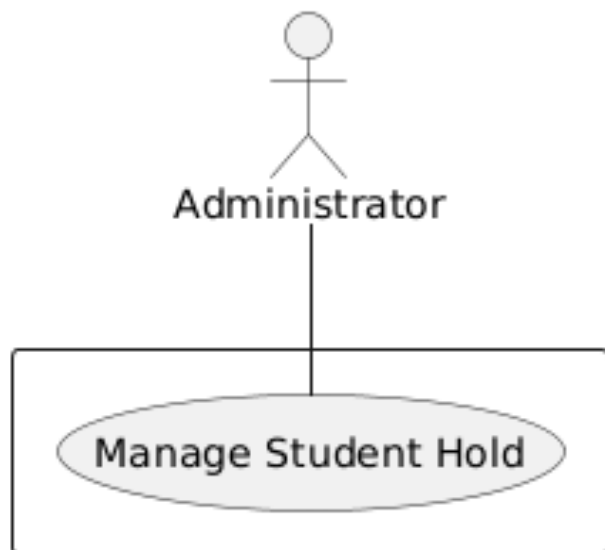


Enrollment Reporting



6.8 UC08: Manage Hold

Manage Student Hold

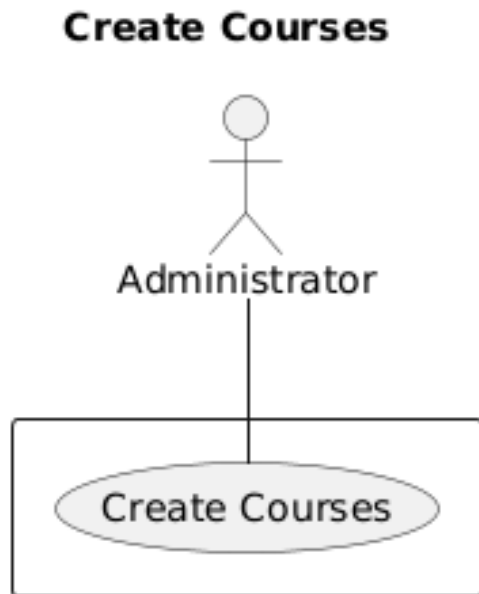


6.9 UC09: Update Changes Report

6.10UC10: Create Courses

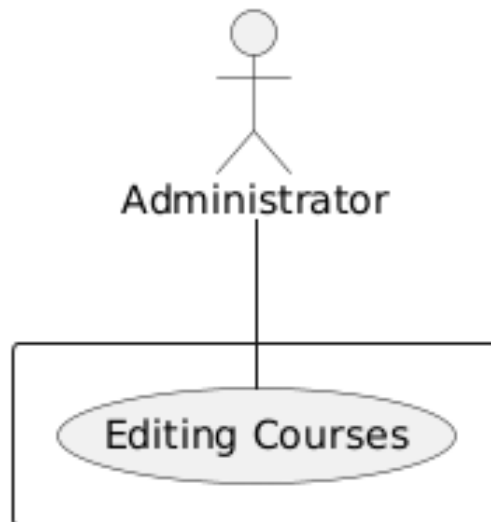
Actor :

Description:



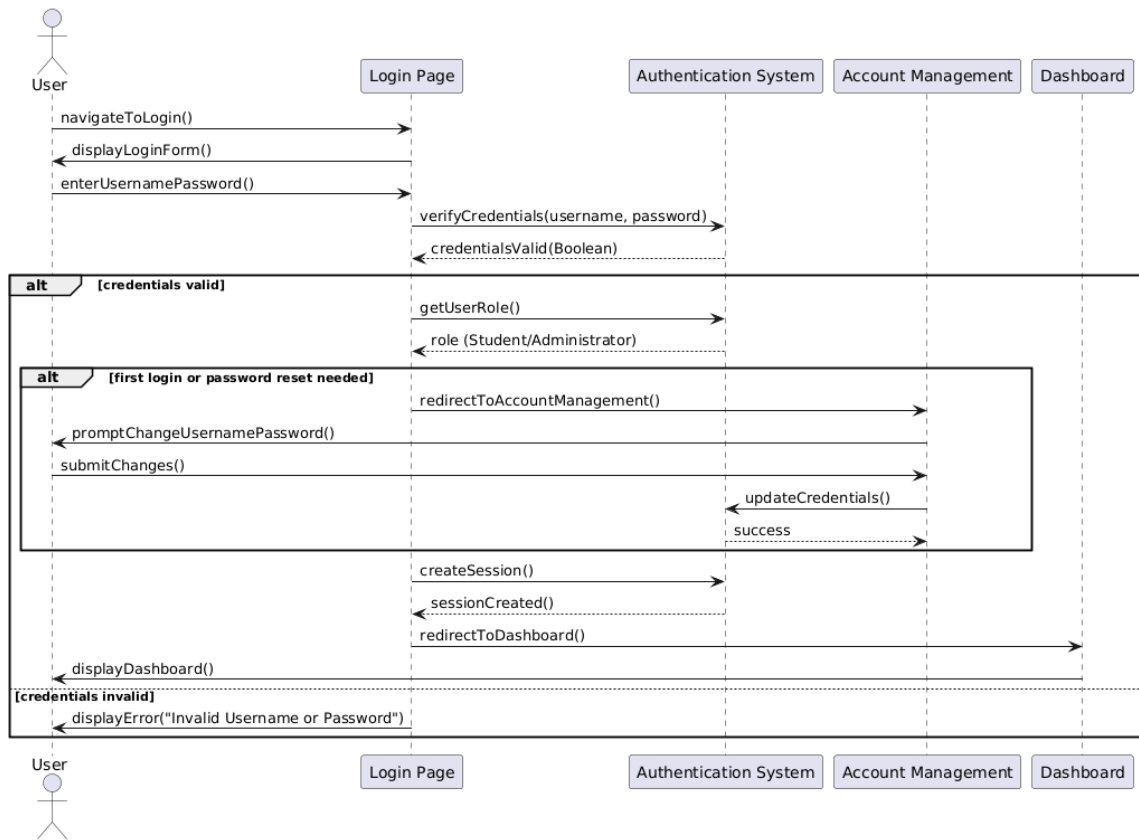
6.11UC11: Edit Courses

Editing Courses

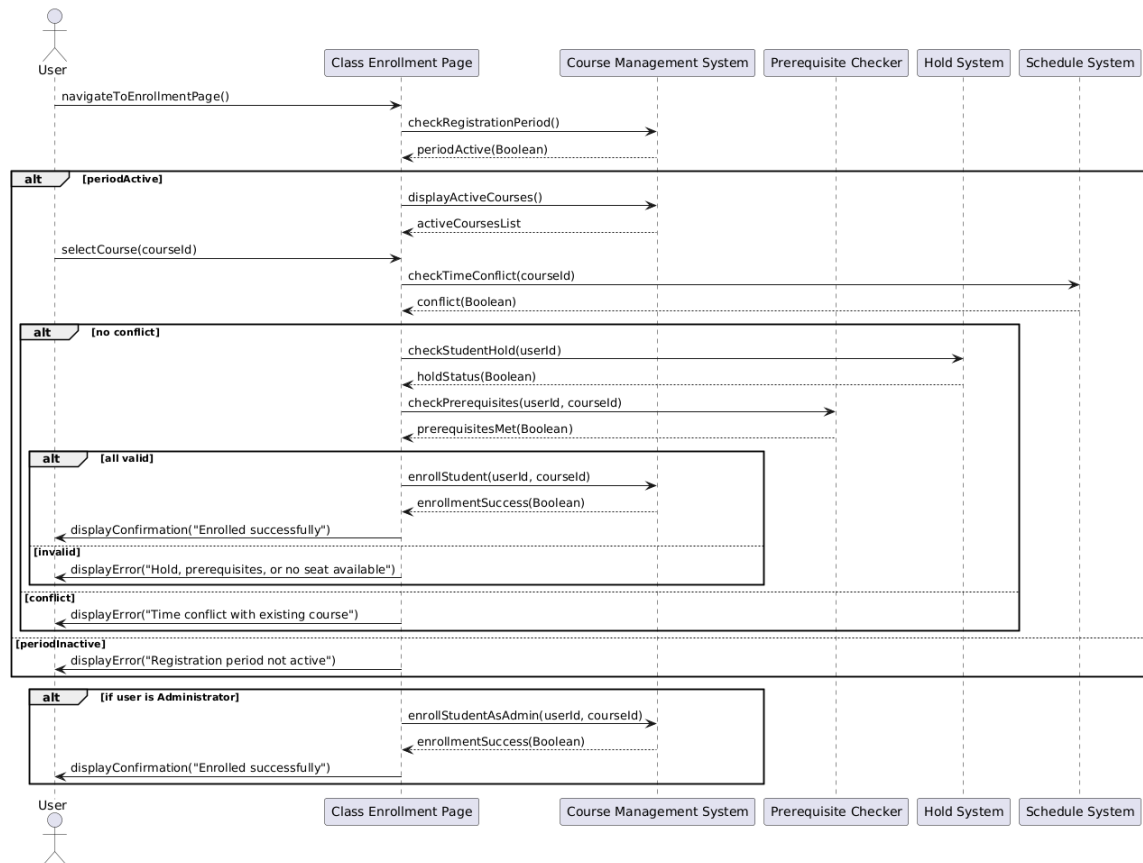


7. Sequence Diagrams

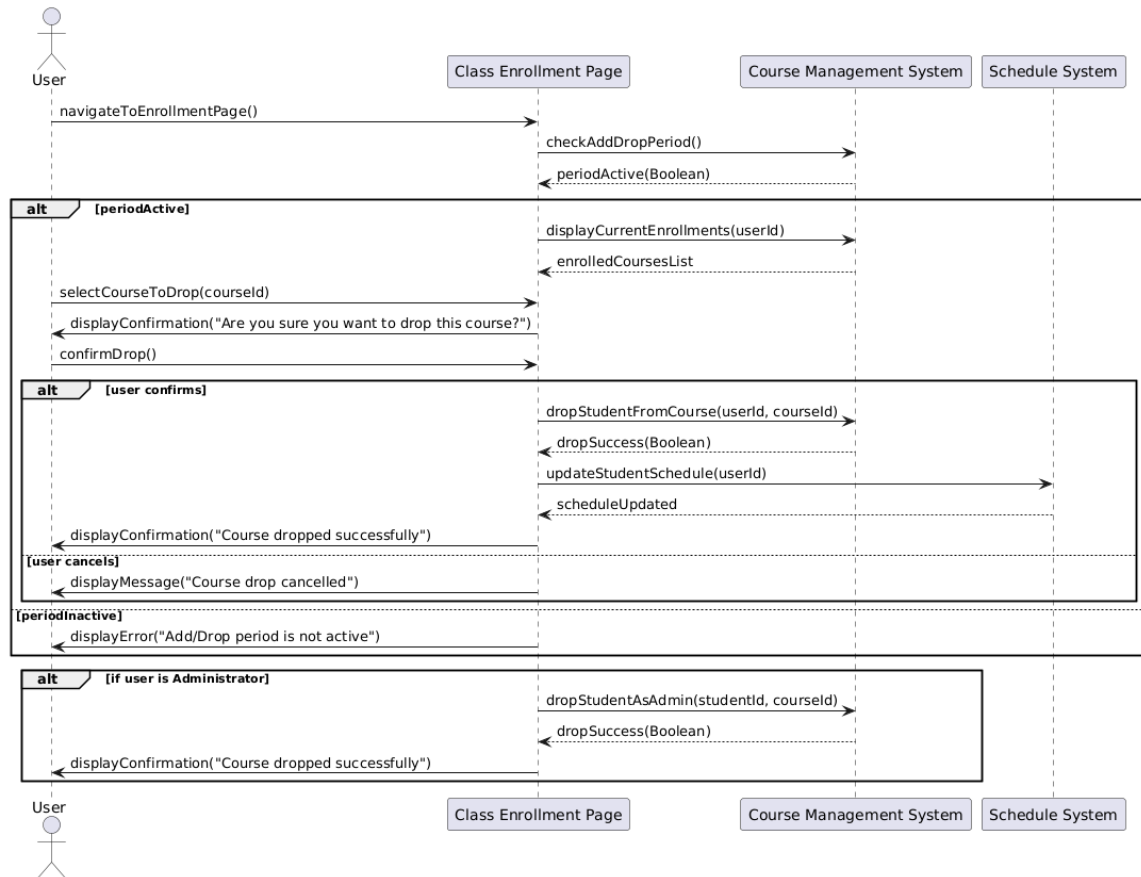
Use Case 1: Manage User Login



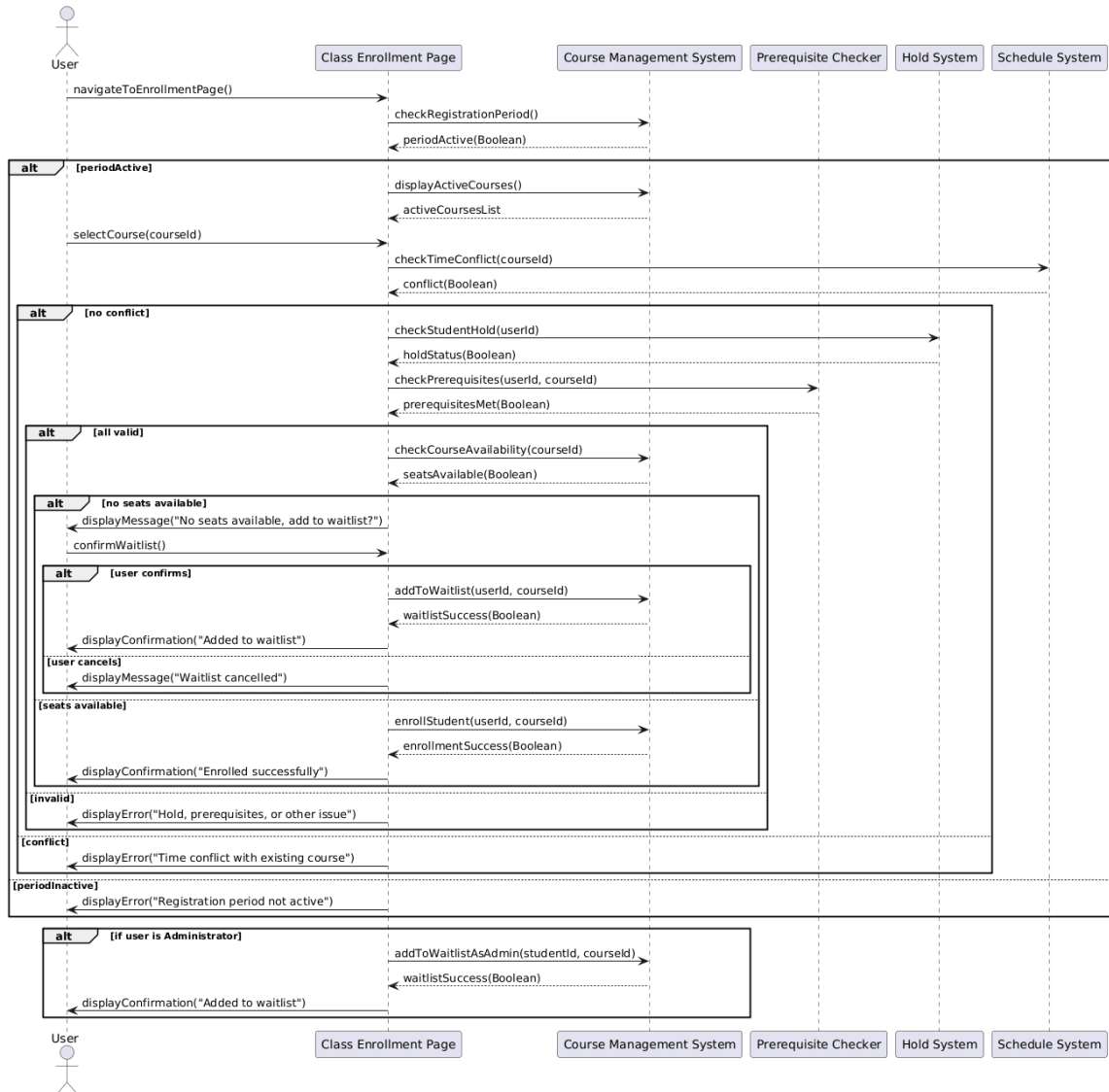
Use Case 2: Course Enrollment



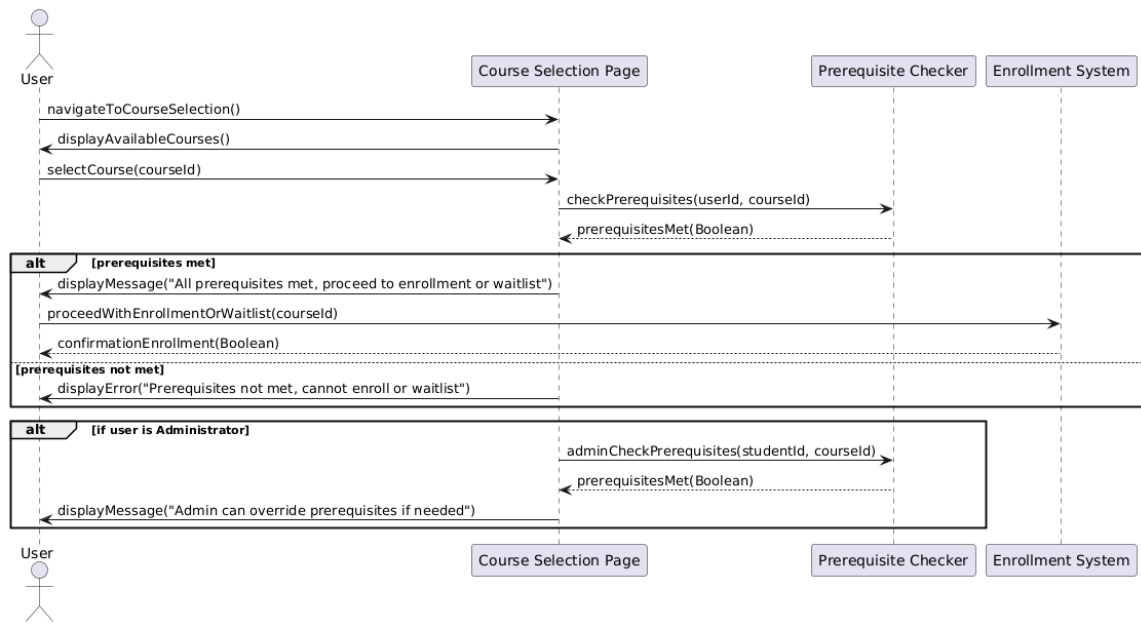
Use Case 3: Course Drop



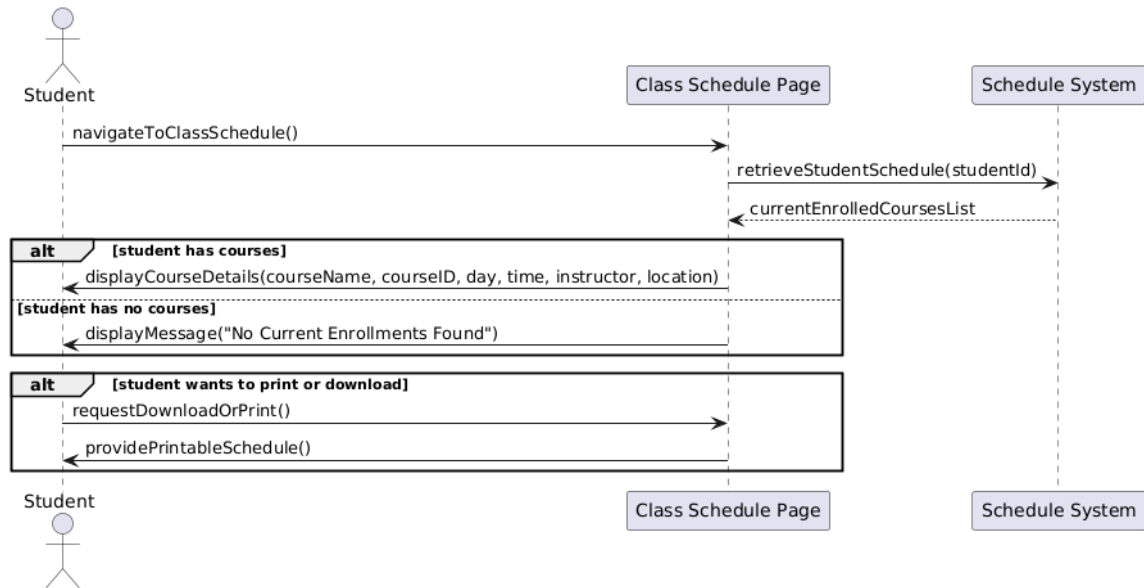
Use Case 4: Course Waitlist



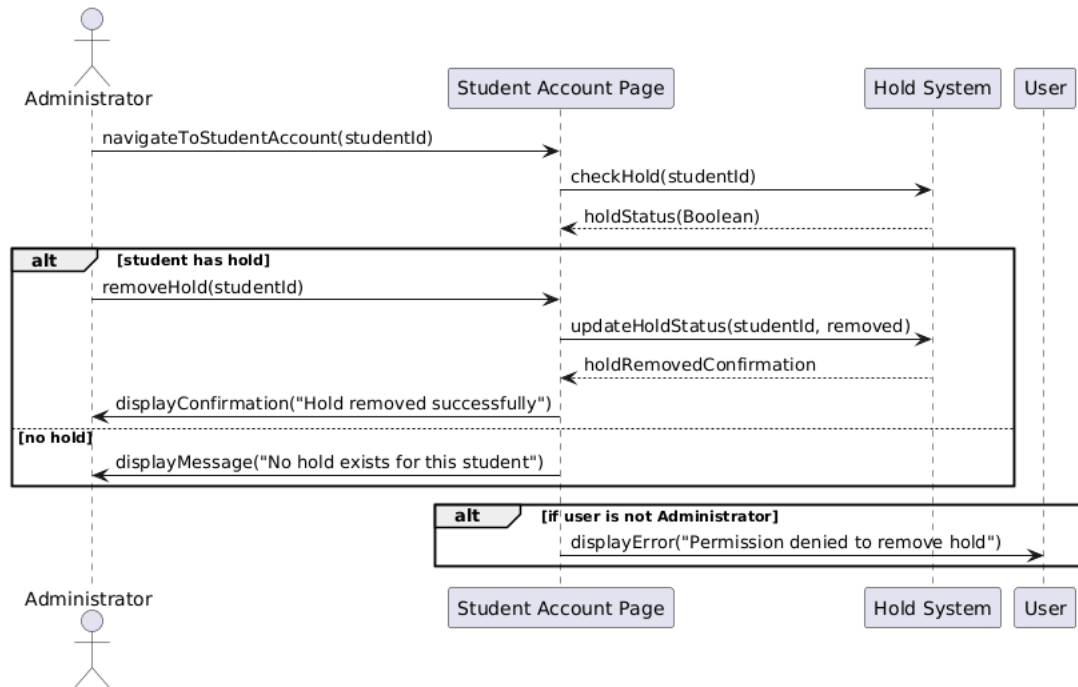
Use Case 5: Course Prerequisites Check



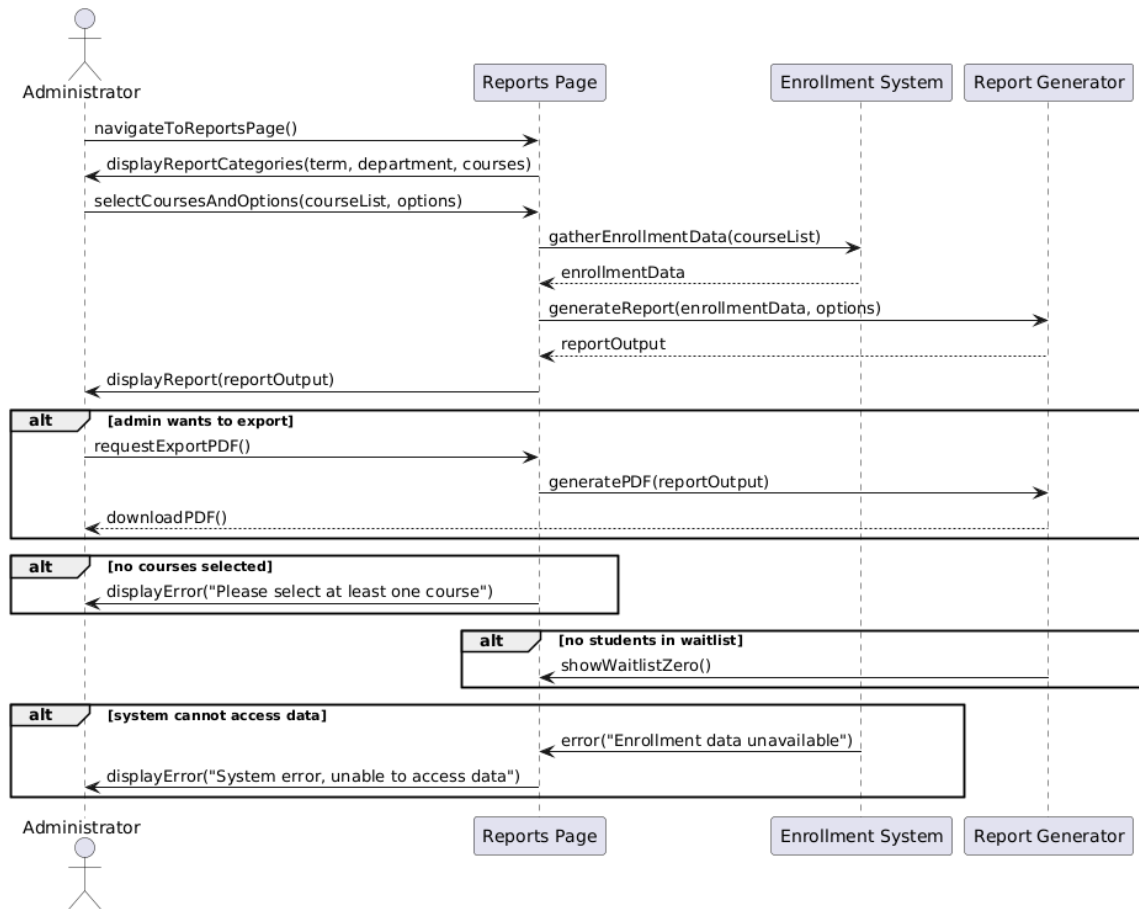
Use Case 6: Access Student Class Schedule



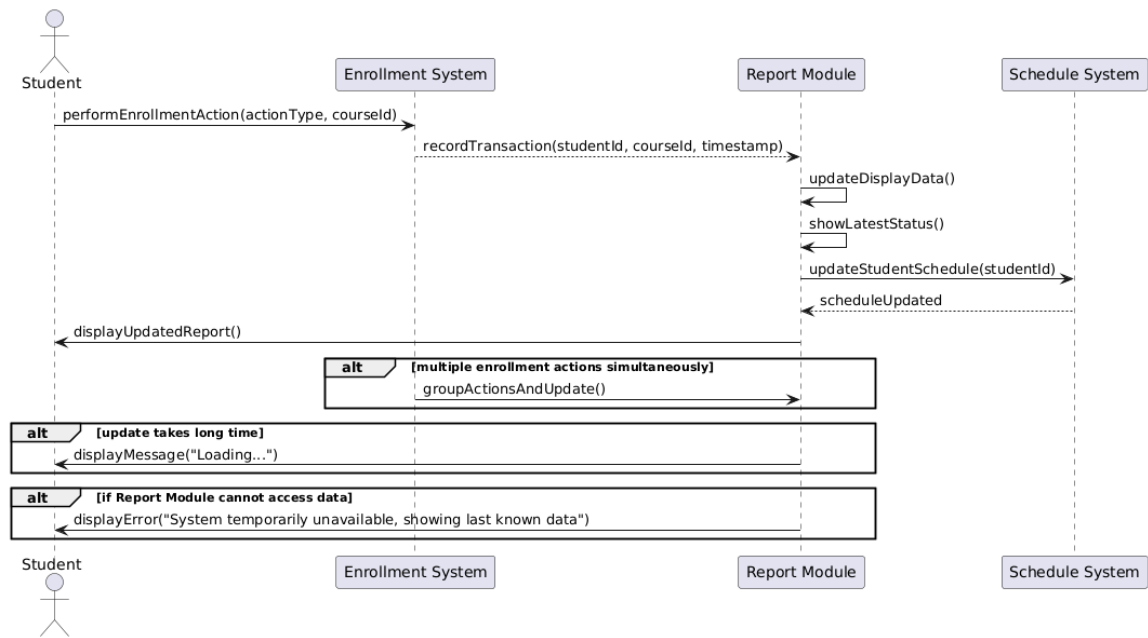
Use Case 7: Manage Student Hold



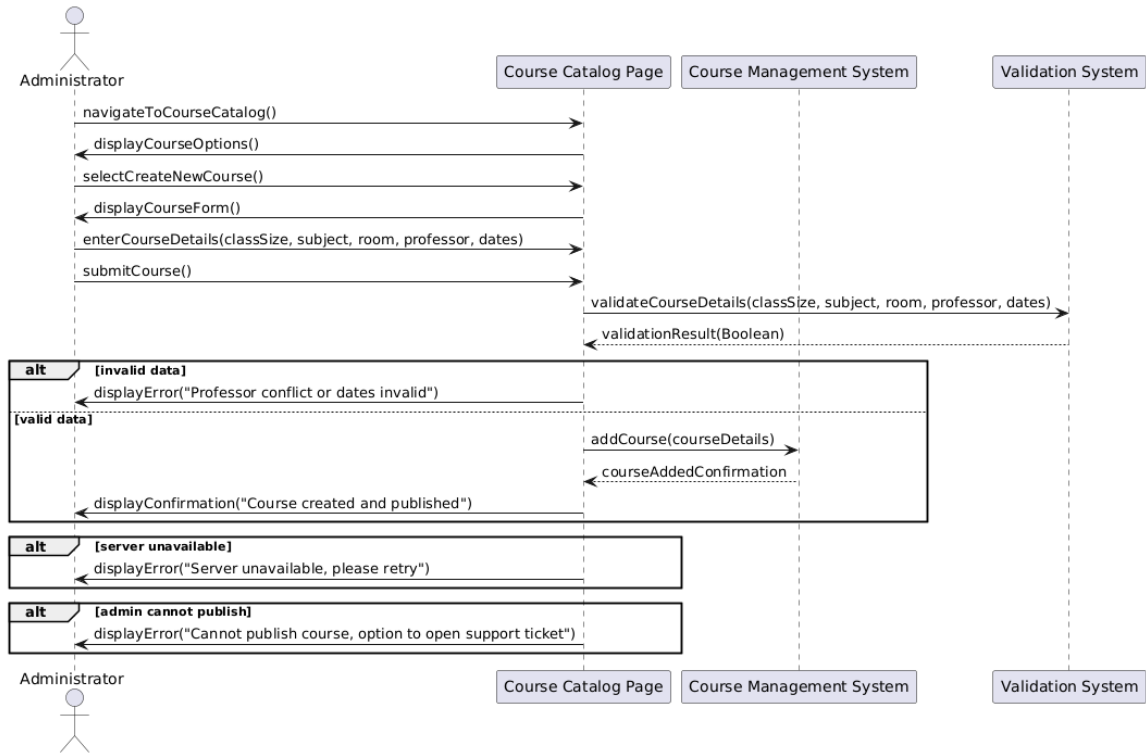
Use Case 8: Enrollment Reporting



Use Case 9: Update Changes Report



Use Case 10: Create courses



Use Case 11: Editing courses

