Campus Online++

Requirements Specification and Analysis

Version 1.0

26.10.2015

Can Kavaloğlu

Kadir Tuğberk Arköse

Aslı Kübra Çamlıca

Nazlı Türk

Prepared for

SE301 Software Engineering



Table of Contents

[1. Introduction 1](#_Toc431983052)

[1.1. Purpose of the System 1](#_Toc431983053)

[1.2. Scope of the System 1](#_Toc431983054)

[1.3. Objectives and Success Criteria of the Project 1](#_Toc431983055)

[1.4. Definitions, Acronyms, and Abbreviations 1](#_Toc431983056)

[1.5. Overview 1](#_Toc431983057)

[2. Current System 1](#_Toc431983058)

[3. Proposed System 1](#_Toc431983059)

[3.1. Overview 1](#_Toc431983060)

[3.2. Functional Requirements 2](#_Toc431983061)

[3.3. Nonfunctional Requirements 2](#_Toc431983062)

[Usability 2](#_Toc431983063)

[Reliability 2](#_Toc431983064)

[Performance 2](#_Toc431983065)

[Supportability 2](#_Toc431983066)

[Implementation 2](#_Toc431983067)

[Interface 2](#_Toc431983068)

[Packaging 2](#_Toc431983069)

[Legal 2](#_Toc431983070)

[3.4. System Models 2](#_Toc431983071)

[Scenarios 2](#_Toc431983072)

[Use case model 2](#_Toc431983073)

[Object model 2](#_Toc431983074)

[Dynamic model 2](#_Toc431983075)

[User interface—navigational paths and screen mock-ups 3](#_Toc431983076)

[4. Glossary 3](#_Toc431983077)

[5. References 3](#_Toc431983078)

REQUIREMENTS ANALYSIS DOCUMENT [1]

# Introduction

## Purpose of the System

Purpose of the system is to provide an efficient and online way to students so that they can enroll their courses easily. Also, purpose of the system includes providing some functionalities to instructors so that they can manage their offered courses.

## Scope of the System

They system provides service to students, instructors, and quota managers. Thus, there are some common functionalities for both students and instructors, also some different functionalities for the both too.

For the services that the system provides to students is; the students can check out their curriculum that contains courses they are supposed to take, they can register to those courses online and after they are done registering, students can view their weekly schedule, and they can also evaluate the courses and the instructor who offers the courses at the end of a semester. In addition, they can view their transcript, and lastly, they can update their personal information registered on the system.

For the services that the system provides to instructors is; the instructors can view their weekly schedule and the courses that they offer. If the instructor has the role of being an advisor, then s/he can approve or disapprove the schedules of the students who made their schedule and want to register the courses that they have selected. In addition, instructors can give the final grades of the students who enrolled any course that the instructor offer.

For the services that the system provides to quota managers is; since the quota managers are only responsible for adding and managing quotas of the courses which are offered, they can only add quota to courses which they are responsible for.

## Objectives and Success Criteria of the Project

Campus Online++ is a system which will replace the Campus Online system. Thus, the objectives and success criteria of Campus Online++ would be the weakness points of the older version of it, which is the Campus Online system. Therefore, objectives of the Campus Online++ system are to have more performance by providing faster responses and services so that the user of the system can complete his/her task faster. Another objective of the project would be usability, by providing usability means the user can complete the task with less moves.

## Definitions, Acronyms, and Abbreviations

CCR: Co-Curricular Record.

RAD: Requirements Analysis Document

## Overview

Our RAD documentation contains Current System section, Proposed System section, Overview of Campus Online++ section, Functional Requirements section, Nonfunctional Requirements section, System Models section, Object Model section, Dynamic Model section and Glossary.

In Current System section of our RAD documentation, we talked about functions and features of current Campus Online. We mentioned disadvantages of current system such as too long response time and unnecessary functions, and also we explained that the current system does not work easy and properly especially to register to courses for students. In addition, we represented that current Campus Online interface is not user friendly.

In Proposed System section, we talked about our new Campus Online++ system’s advantages and modifications. For example, we defined that response time of Campus Online++ will be less than previous system or users can access whatever they want with less click. In addition, we explained that our new Campus Online++ will be more efficient by comparison current Campus Online.

In Overview section, we described overview of features, functions and details of Campus Online++. In addition, we explained functions from users’ perspective.

In Functional Requirements section, we talked about functions and features of Campus Online++. And also, we described functions of Campus Online++ from students’ and instructors’ perspective. Moreover, we mentioned interaction between Campus Online++ and user and their environments.

In Nonfunctional Requirements section, we mentioned nonfunctional parts of our new system such as usability, performance, reliability or availability and so on. We described all of nonfunctional requirements of Campus Online++.

In System Models section, we described scenarios and use cases of Campus Online++. We defined scenarios, actors, and use cases’ flow events and so on in this part.

In Object Model section, we explained class diagrams of our system and we defined relationships between classes of Campus Online++.

In Dynamic Model section, we mentioned sequence diagram that is to say we explained methods and functions and their operations with actors of Campus Online++.

In Glossary section, we described all things of Campus Online++ system, it can call Campus Online++ dictionary. We explained all words in RAD documentation.

# Current System

Campus Online is an online student information and registration system, it was developed for usage purpose of the students and the instructors of Işık University. On students’ perspective, students can display their courses, schedules, transcript, curriculum, CCR through Campus Online. And also, they can update and get their personal and educational information, they can evaluate their courses and instructor, and register to courses. In addition, students can communicate with instructors and their advisors by using Campus Online. On instructors’ perspective, instructors can display their schedules, offered courses, and also they can see schedules of students, transcript of students and free hours of students. In addition, they can communicate with students and they can approve students programs by using Campus Online. However the system does not work properly and the system is not user friendly. Some functions of Campus Online do not work properly and again some of them do not work as intended. For example in registration time, when students use Campus Online to register their courses, they face many difficulties on campus online; because, Campus Online does not respond to students or response time of Campus Online may get too long. Besides, students do not see quotas of the courses to register except their registration time, Campus Online does not support it. Moreover, Campus Online has some functions and features unnecessary for the users. Finally, the current system has some flaws due to previous technologies and lack of new technologies advantages.

# Proposed System

First of all, new system will be developed by using new technologies, thus it will provide advantages of new technologies. The new system Campus Online++ will be more efficient and friendly for the users than current Campus Online. For example, students will be able to see quotas of courses or prerequisites of courses any time they want with the new version. In other words, they won’t have to be in the process of their registrations to see the quotas of the courses. This way will be more efficient to register for students. And also, response time of the system will not be long, so the users can use Campus Online++ more efficiently .Therefore, the users will be able to accomplish their tasks with less clicks and moves such as registration by using Campus Online++. In addition, interaction between Campus Online++ and the users will be better than previous one. The interface will be developed with new technologies, thus it will be impressive for users. In addition, performance of Campus Online++ will be much better by providing faster services and response time than previous version.

## Overview

Campus Online++ is an online registration and course system for students and instructors. The students use Campus Online++ to reach courses’ information such as schedule, transcript and to update their personal and educational information. In addition, students register to courses by using Campus Online++. Instructors use Campus Online++ to reach offered courses’ information, their schedules, and schedules of students and so on. The students and the instructors can use Campus Online++ from outside the campus. Campus Online has common and different functionalities for users. Users of the system are students, instructors, advisors, admin, and quota managers. They have some common and different functions. For example, students and instructor can display their schedules, it is common for both of them. However, instructors can see students’ schedules and transcript, so it is not common for students and instructors. Otherwise, instructors cannot register to courses only students register to courses, on the other hand; only instructors approve their students programs. In Campus Online++ students display their weekly schedule, transcript, curriculum, quotas of courses, and prerequisites of courses. In addition, students can register to courses and update their personal information. Instructors can display their schedules, offered courses, schedule of students, and transcript of students. In addition instructors approve students’ program, reserve a room for course, view free hours of students. Campus Online++ allows control and manage courses, registration and information. Campus Online++ is more efficient and user friendly than the old version. For example, students can see quotas of courses before they start to register to courses. In addition, performance of Campus Online++ will be much better by providing faster services and response time than previous version.

## Functional Requirements

Campus Online++ is an online registration and course information platform for students and instructors. On student’s side, Campus Online++ has several functions. For example, students can see their courses, transcript, schedules, quotas of courses, prerequisites of courses by using Campus Online++. In addition they can change password and evaluate instructors and courses by using Campus Online++.

On instructor’s side, Campus Online++ has different functions for instructors. For example, instructors can display their schedules, offered courses, available rooms, moreover; they can see schedule of students and transcript of students by using Campus Online++.

The main and important function of Campus Online++ is registration. Students use Campus Online++ to register and instructors can approve students program.

## Nonfunctional Requirements

### Usability: For a student who has three courses (only one of them is elective) should make thirteen clicks to register (including 4 clicks made for login and start registration).

### Reliability: The system must be running 90% of the time.

### Performance: The system must allow 1500 parallel users.

### Supportability: The system must be able to be maintained easily.

### Implementation: The system will be implemented on Visual Studio platform. In addition, C# will be used as the programming language and ASP.NET will be used as a framework in the process. User Interface should be web-based (accessible via WWW Browser).

### Interface: There is no external systems or legacy systems to work with. Thus, there is no interface requirements.

### Packaging: No constraints on the actual delivery of the system are determined. Thus, packaging requirements will be decided in the future.

### Legal: The software is provided "as is", without warranty of any kind, express or implied, including but not limited to the warranties of merchantability, fitness for a particular purpose and no infringement. In no event shall the authors or copyright holders be liable for any claim, damages or other liability, whether in an action of contract, tort or otherwise, arising from, out of or in connection with the software or the use or other dealings in the software.

## System Models

### Scenarios

**Scenario 1:**

|  |
| --- |
| *Scenario name:* letterGradesAreGiven |
| *Participant actor instances:* emine:Instructor |
| *Flow of events:* 1. The final exam is over of a course that Emine offers in the current semester and she has read the papers of students who took the exam. Thereby, Emine wants to give the letter grades of the students. Emine first logs in to Campus Online++.    2. Emine chooses “Display Class List” option which is located on the screen. Then, she chooses one of the courses listed which she offers in the current semester. Finally, she chooses submit grades option on the screen to see the list of students.  3. After the list of students is provided and displayed by Campus Online++, Emine enters the letter grades of each student.  4. Emine sends the list of the student after she gives the letter grades of each student.  5. Campus Online++ receives the list and saves it into the “Course Grades Of Students” table which contains information of students, courses taken by the students and students’ final grades.  6. Emine receives an acknowledgement indicates that the process has been completed successfully and chooses the “Print” option to get a hard copy of the final grades list. |

**Scenario 2**

|  |
| --- |
| *Scenario name:* instructorAndCourseEvaluated |
| *Participant actor instances:* can:Student |
| *Flow of events:* 1. Can’s all final exams are over, wants to check his letter grades out. Can first logs in to Campus Online++.  2. Can clicks on the CCR option on the screen to see what his letter grades are. However, he gets confronted with an error message which indicates that he must complete the evaluation form of instructors and courses that he has taken in the current semester.  3. Then, he clicks on the Evaluate Instructor And Course option on the screen to complete the evaluation form.  4. Campus Online++ presents the form to Can. The form contains the courses that Can is taking in the current semester.  5. Can chooses a course by choosing the select option and sends a request to Campus Online++ to get to the evaluation form which contains questions about both the course and the instructor who is giving that selected course.  6. Campus Online++ receives the request, then responds by presenting the form contains the evaluation questions.  7. Can evaluates the course and the instructor of the course by selecting the evaluation point for each question. Then, Can submits the filled form to Campus Online++ by clicking the submit option.  8. In the end, Campus Online++ receives the form and answer information that Can gave, then saves in into “Survey Answers” table. Finally, Campus Online++ displays an acknowledgement message to Can, which indicates that the evaluation process of the course and the instructor has been completed successfully. |

**Scenario 3:**

|  |
| --- |
| *Scenario name:* messagingBetweenStudentAndInstructor |
| *Participant actor instances:* Tugberk : Student  Emine : Instructor |
| *Flow of events:* 1. Tugberk needs to ask a book name to Emine. Tugberk enters to Campus Online++ then logs in with his password and student id. After successful login, he selects option to send message. When send message screen loads, he selects Emine from the list which contains instructor names Tugberk can send message and types the message to text field.  2. Emine enters to Campus Online++. Then she types her password and id. She logs in successfully. After she sees a notification which saying she has a message from Tugberk. She reads the message then selects option send message. She types name of the book and chooses option to send the message.  3. Tugberk enters to Campus Online++. Then he logs in using his password and student id. He sees the message from Emine. |

**Scenario 4:**

|  |
| --- |
| *Use case name:* updatingAddressInfo |
| *Participant actor instances:* Nazli:Student |
| *Flow of events:* 1. Nazli and her family have moved to another city because of Nazli’s father’s job. So that Nazli’s home address changed.  2. Nazli wants to update her home address on Campus Online++. Nazli enters the Campus Online++ with her student id and password. After successful login, she selects option to change the information.  3. Campus Online++ prepares the information form which is stated above. Then, displays it on Nazli’s screen.  4. Nazli edits address information through the form. And then, Nazli clicks button to update information.  5. Campus Online++ receives the request and updates the “Student” table and displays the acknowledgement message to Nazli.  6. Nazli sees “Your Personal Information Updated!” message on her screen. |

**Scenario 5:**

|  |
| --- |
| *Use case name:* takingAttendanceList |
| *Participant actor instances:* Emine:Instructor |
| *Flow of events:* 1. Emine is the Instructor of SE301 and CSE202 courses. During classes Emine takes attendance list. When Emine finishes lessons, Emine goes to office and opens computer to update attendance list.  2. Emine wants to see her courses’ attendance list and wants to update it. Emine enters to the Campus Online++ with her instructor id and password. After successful login Emine chooses a course among the list that contains the courses given by her and she waits for acknowledgement.  3. Campus Online++ fetches the list of student and their attendance data of the selected course from a table. Then, Campus Online++ displays the students and their attendance information on the screen.  4. Emine edits the attendance list and she clicks option to update attendance list.  5. Campus Online++ receives the request and updates attendance list.  6. Emine sees “Your attendance list updated!” message on her screen. |

**Scenario 6:**

|  |
| --- |
| *Use case name:* changingPassword |
| *Participant actor instances:* Tugberk:Student |
| *Flow of events:* 1. Tugberk thinks his password is not safe enough so he thinks about changing it.  2. Tugberk enters to Campus Online++ and logins with his credentials.  3. Campus Online++ checks his password and id in Student Table after confirming displays Home page.  4. Tugberk selects Change Password option.  5. Campus Online++ fetches request and displays a form with four fields, which are 'old password', 'new password', 'confirm new password' and 'submit'.  6. Tugberk types his old password and types his new password, he confirms new password by typing again. Then submits the form through the submit option.  7. Campus Online++ receives request and changes Tugberk's password field on Student Table.  8. Tugberk sees 'Your password has been changed' message on the screen. |

**Scenario 7:**

|  |
| --- |
| *Scenario name:* settingSpecialQuota |
| *Participant actor instances:* Nazli:Student  Esra: QuotaManager |
| *Flow of events:* 1. In registration time, all students select courses according to their departments and curriculum for that semester, so Nazlı wants to register for 'SE301', but she cannot enroll because there is no quota for 'SE301'.  2. Nazlı talks to Esra who is the Quota Manager of the department which is department of the course 'SE301'.  3. Esra first logs in with her credentials to Campus Online++ to set special quota for Nazlı.  4. Esra chooses “Set Quota” option which is located on the screen. Then, she chooses the course. Then, she enters the Nazlı’s student id to set quota for Nazlı. Finally, she chooses submit option to submit the form on the screen.  5. Campus Online++ receives the form and saves it into the “Course” table which contains quota information of the course. |

**Scenario 8:**

|  |
| --- |
| *Scenario name:* studentRegistersToMultipleCourses |
| *Participant actor instances:* Tugberk:Student |
| *Flow of events:* 1. Tugberk's registration date has come and Tugberk wants to register to multiple courses for that semester. Tugberk goes to Campus Online++ and logins with his credentials.  2. After a successful login he sees there are no registration holds for Tugberk. So Tugberk chooses Register For Courses option.  3. Campus Online ++ displays possible error types along with their explanation. Tugberk reads them and proceeds.  4. Campus Online++ displays a page with Course List and Selected Courses and fetches filtered Course List from the Course Table.  5. Tugberk chooses 'CSE466' and 'CSE487' from Course List section and chooses option to enroll to those courses.  6. Campus Online++ retrieves the request and displays possible ways to enroll 'CSE466' and 'CSE487' in a list.  7. Tugberk chooses 'Departmental Elective' option from the list for both of those courses.  8. Campus Online++ displays a message to Tugberk saying your program has send to your Advisor for approval. |

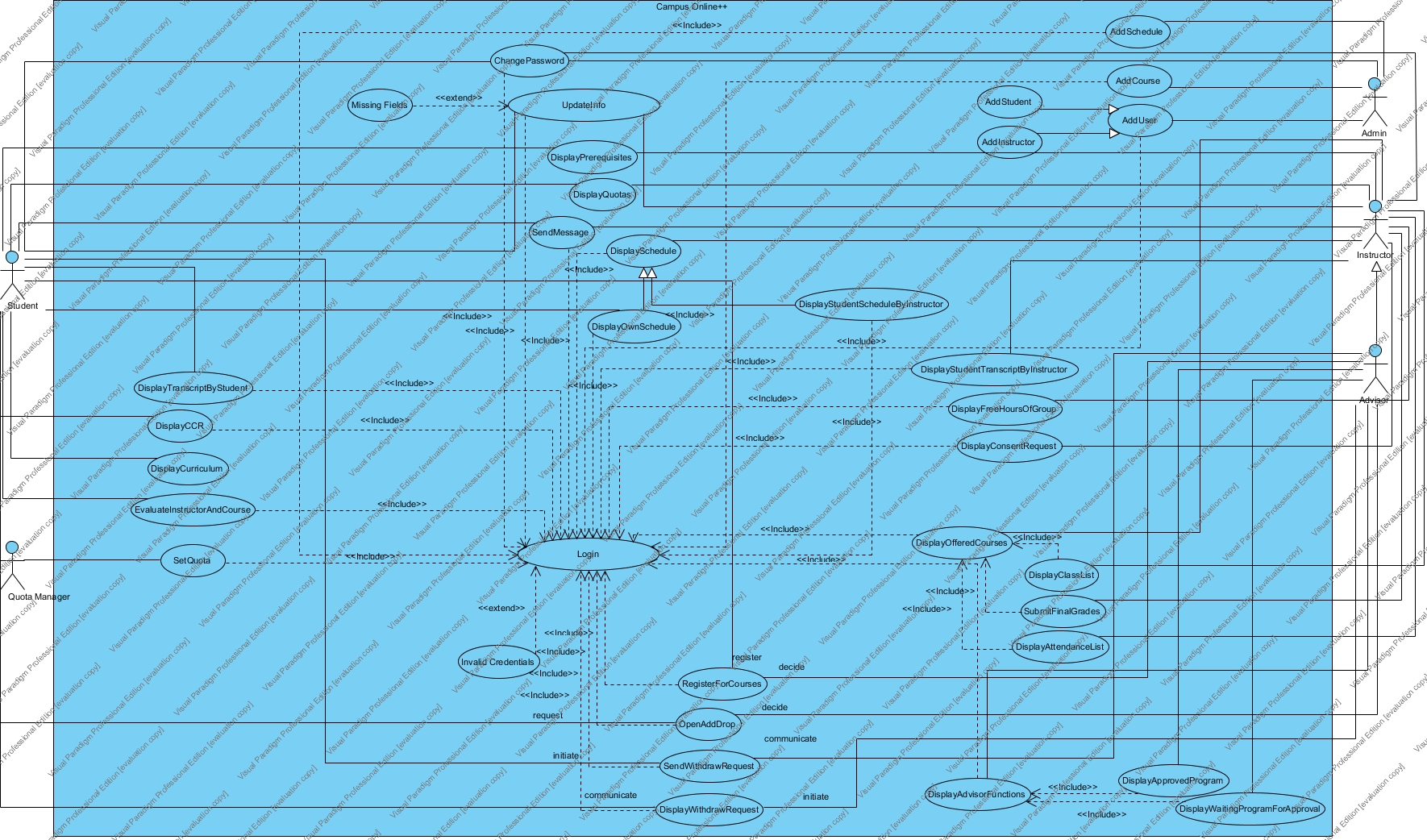
**Scenario 9:**

|  |
| --- |
| *Scenario name:* withdrawFromACourse |
| *Participant actor instances:* Tugberk:Student  Emine:Advisor |
| *Flow of events:* 1. The first midterm of the course 'SE301' has passed. And Tugberk has got a bad score and thinks about withdrawing from that course.  2. Tugberk enters to Campus Online++ and logins with his credentials.  3. Campus Online++ checks his password and id in Student Table after confirming displays Home page.  4. Tugberk selects Send Withdraw Request option.  5. Campus Online++ fetches request and displays the “Withdraw Form” using “Schedule” table with current course schedule of Tugberk and option to select course(s) for withdraw and a submit option.  6. Tugberk sees his current schedule and selects “SE301” then submits the “Withdraw Form” using submit option.  7. Emine logs in with her credentials. Then sees a notification on her home page saying “Tugberk wants to withdraw from SE301”.  8. Emine selects Display Withdraw Request option.  9. Campus Online++ fetches request and displays the withdraw request form using “Schedule table” and “AdvisorOfStudent” table with withdraw request(s)'s send to her.  10. Emine selects Tugberk's request. And displays courses Tugberk wants to withdraw.  11. Emine by selecting to accept Tugberk's withdraw request submits the withdraw request from.  12. Campus Online++ receives the request, by using “the Schedule table” it updates Tugberk's schedules and changes “SE301”'s status into “Withdrawed”.  13. Tugberk receives a request saying “Your advisor's approved your withdraw request for SE301” |

**Scenario 10:**

|  |
| --- |
| *Scenario name:* openingAddDrop |
| *Participant actor instances:* Emine:Advisor  Tugberk:Student |
| *Flow of events:* 1. Tugberk does not like CSE466 course’s teacher because of that Tugberk decides to drop CSE466 course.  2. Tugberk enters Campus Online++ with her student id and password. After successful login, he clicks “Open Add Drop” button and he starts to wait for Emine’s approval.  3. Emine enters to Campus Online++. Then she types her password and id. She logs in successfully. After she sees a notification which send by Tugberk. She opens notification and she approves Tugberk’s add-drop request after that, she submits the form.  4. Tugberk enters Campus Online++ to drop CSE466 course. Tugberk selects Cse466 course from registered courses table and Tugberk deletes Cse466 course. After that, Tugberk clicks “Send For Approval” button to send to Emine.  5. Emine enters to Campus Online++ and sees Tugberk’s approval request. Emine approves Tugberk’s schedule and submits it.  6. Tugberk enters Campus Online ++ and sees notification “Your Program is Approved By Advisor”. |

### Use case model



**Use Case 1:**

|  |
| --- |
| *Use case name:* Login (High Priority) |
| *Participant actors:* Initiated by Student, OR Instructor, OR Admin |
| *Flow of events:* 1. The user first enters to Campus Online++ system.    2. Campus Online++ presents the “Login Form” to the user.  3. The user enters his/her username into username text field on the screen, also enters his/her password into password text field on the screen. Lastly, the user sends a request to Campus Online++ by using login button on the screen to be logged in.  4. Campus Online++ checks the username and password from the “User” table so that Campus Online++ allows the user to login. Then it determines role of the user by retrieving the role data from the “User” and redirects to proper screen. |
| *Entry Condition:* The user enters to login screen. |
| *Exit Condition:* The user is logged in, OR,  The user has received an explanation indicating why he/she could not login. |

**Use Case 2:**

|  |
| --- |
| *Use case name:* UpdateInfo (Low Priority) |
| *Participant actors:* Initiated by Student, OR Instructor |
| *Flow of events:* 1. The user sends a request to Campus Online++ to update his/her personal information which are his/her Identification, Address, Education and Job, Health and Medical or Previous University Information.  2. Campus Online++ prepares the “Personal Information Form” by using the “Student” table or the “Instructor” table. Then, displays it on the screen so that the user can update fields he/she wishes to.  3. The user edits the fields he/she wants, then he/she submits the form by using the update button to be updated.  4. Campus Online++ updates the information into the “Student” or the “Instructor” table. Finally, it displays the “Acknowledgement Notice” to the user. |
| *Entry Condition:* The user is logged into Campus Online++.  The user chooses Update Info option on the screen. |
| *Exit Condition:* Information of the user has been updated and the user has received an acknowledgement message. OR,  The user has received an explanation indicating why he/she could not login. |

**Use Case 3:**

|  |
| --- |
| *Use case name:* DisplayCurriculum (Low Priority) |
| *Participant actors:* Initiated by Student |
| *Flow of events:* 1. The Student sends a request to Campus Online++ to see his/her curriculum on the screen.  2. Campus Online++ first finds the department of the student which s/he belongs by using the “Student” table, then prepares the “Curriculum Notice” according to the department of the student. The “Curriculum Notice” is divided into semesters and each semester part contains course code, course full name and credit of the courses that the student supposed to take during that semester. Then, displays it on the screen so that the user can see his/her curriculum on the screen. |
| *Entry Condition:* The Student is logged into Campus Online++.  The user chooses Display Curriculum option on the screen. |
| *Exit Condition:* The Student has viewed his/her curriculum content. OR,  The Student has received an explanation indicating why the transaction could not be processed. |

**Use Case 4:**

|  |
| --- |
| *Use case name:* DisplayCCR (Low Priority) |
| *Participant actors:* Initiated by Student |
| *Flow of events:* 1. The Student sends a request to Campus Online++ to see his/her CCR on the screen.  2. Campus Online++ first finds the department of the student which s/he belongs by using “Student” table, then also using the “Student Course” prepares the CCR according to the department of the student to display. The CCR contains not only courses’ codes, full names and courses’ credits of user according to every semester, but also, contains letter grades of courses which were taken by the student. Then, displays it on the screen so that the user can see his/her CCR on the screen. |
| *Entry Condition:* The Student is logged into Campus Online++.  The Student chooses Display CCR option on the screen. |
| *Exit Condition:* The Student has viewed his/her CCR content. OR,  The Student has received an explanation indicating why the transaction could not be processed. |

**Use Case 5:**

|  |
| --- |
| *Use case name:* DisplaySchedule (Low Priority) |
| *Participant actors:* Initiated by Instructor |
| *Flow of events:* |
| *Entry Condition:* The user is logged into Campus Online++.  The user chooses Schedule option on the screen. |
| *Exit Condition:* The user has viewed his/her weekly schedule content. OR,  The user has received an explanation indicating why the transaction could not be processed. |

**Use Case 6:**

|  |
| --- |
| *Use case name:* DisplayOwnSchedule (Low Priority) |
| *Participant actors:* Initiated by Student, OR Instructor |
| *Flow of events:* 1. The user first sends a request to Campus Online++ to view his/her weekly schedule on the screen.  2. Campus Online++ retrieves data which are relevant to days and hours of the classes that the user has by using the “Student Schedule” table or the “Instructor Schedule” table (according to the user’s role). Then, Campus Online++ prepares the “Schedule Notice” which contains rows for weekdays and hours columns of each day row, and fills columns of the rows with the course name and course classroom that the user has that hour on that day. Finally, Campus Online++ displays his/her weekly schedule. |
| *Entry Condition:* Inherited from DisplaySchedule use case. |
| *Exit Condition:* Inherited from DisplaySchedule use case. |

**Use Case 7:**

|  |
| --- |
| *Use case name:* DisplayStudentScheduleByInstructor (Low Priority) |
| *Participant actors:* Inherited from DisplaySchedule use case. |
| *Flow of events:* 1. The Instructor first chooses Display Student Schedule option on the screen, then sends a request to Campus Online++ to view the weekly schedule of a student.  2. Campus Online++ responds by presenting the “Student Schedule Form” to the Instructor. The form contains a text field and a search button so that the Instructor can submit the form.  3. The Instructor enters student's id to text field. Once the form is completed, the Instructor chooses search option to submit the form and see student's weekly schedule on the screen.  4. Campus Online++ receives the form and retrieves data which are relevant to days and hours of the classes that the given student has by using the “Student Schedule” table. Then, Campus Online++ displays the weekly schedule content of the student according to the student’s id that given by the Instructor. |
| *Entry Condition:* Inherited from DisplaySchedule use case. |
| *Exit Condition:* The Instructor has viewed the student’s weekly schedule content. OR,  The Instructor has received an explanation indicating that there is no such student matches with the given id. |

**Use Case 8:**

|  |
| --- |
| *Use case name:* EvaluateInstructorAndCourse (Medium Priority) |
| *Participant actors:* Initiated by Student |
| *Flow of events:* 1. The Student sends a request to Campus Online++ to evaluate the instructors and courses which are taken in the current semester.  2. Campus Online++ receives the request then, using the Student id, it fetches the courses taken by the Student from the “Student Course” table, then it responses by presenting the “Course Form” to the Student on the screen. The form contains the list of courses that the Student is taking in the current semester.  3. The Student selects a course which he/she wants to evaluate by choosing the select option. Once a course is selected, the Student submits the form by choosing submit button.  4. Campus Online++ receives the form and the selected course information, then it presents “Evaluation Form” on the screen.  5. The Student fills out the form by selecting the evaluation point for each question. Once the form is completed, the Student submits the form by clicking on submit button on the screen.  6. Campus Online++ receives the form and saves the answer information into the “Survey Answers” table. In addition, Campus Online++ displays the “Acknowledgement Notice” to the Student. |
| *Entry Condition:* The Student is logged into Campus Online++.  The Student chooses Evaluate Instructor And Course option on the screen. |
| *Exit Condition:* The Student has evaluated the course and the instructor related to that course and the Student has received an acknowledgement message. OR,  The Student has received an explanation indicating why the transaction could not be processed. |

**Use Case 9:**

|  |
| --- |
| *Use case name:* DisplayTranscriptByStudent (Low Priority) |
| *Participant actors:* Initiated by Student |
| *Flow of events:* 1. The Student sends a request to Campus Online++ to view his/her transcript on the screen.  2. Campus Online++ prepares transcript according to student’s id by fetching the necessary data from the “Student Course” table. Then Campus Online++ displays transcript with the fields, student name, student id, course names and grades of the courses on the screen. |
| *Entry Condition:* The Student is logged into Campus Online++.  The Student chooses Transcript option on the screen. |
| *Exit Condition:* The Student has viewed transcript content on the screen. OR,  The Student has received an explanation indicating why the transaction could not be processed. |

**Use Case 10:**

|  |
| --- |
| *Use case name:* DisplayStudentTranscriptByInstructor (Low Priority) |
| *Participant actors:* Initiated by Instructor |
| *Flow of events:* 1. Instructor enters a student number into the text field that is located in the home screen. Then, s/he sends a request to Campus Online++ to view the transcript of the student by using the search option.  2. Campus Online++ fetches the data from the “Student Course” table by using the student id given by the Instructor. Then, it displays the transcript content of the student searched by the Instructor. |
| *Entry Condition:* The Instructor is logged into Campus Online++.  The Instructor gives the student’s id which he/she wants to view. |
| *Exit Condition:* The instructor views the student’s transcript content. OR,  The instructor has received an explanation indicating that there is no such student matches with the given id. |

**Use Case 11:**

|  |
| --- |
| *Use case name:* ChangePassword (Low Priority) |
| *Participant actors:* Initiated by Student, OR Instructor, OR Admin |
| *Flow of events:* 1. The user sends a request to Campus Online++ to change his/her current password.  2. Campus Online++ receives the request and it responses by presenting the “Password Change Form” to the user on the screen.  3. The user enters his/her current password into password text field on the screen. Also, enters new password (the new one to be prevail) into new password text field on the screen and the same new password into confirm new password text field to match the two so that the new password will be constructed unambiguously. Lastly, the user sends a request to Campus Online++ through the send option on the screen to change his/her current password.  4. Campus Online++ receives the request from the logged user and checks current password to be matched with the password which is stored in the “User” table. It also checks the new password and the confirm new password to be matched each other. Then, Campus Online++ saves the new password into the table stated above and displays the “Acknowledgement Notice” to the user. |
| *Entry Condition:* The user is logged into Campus Online++.  The user chooses Change Password option on the screen. |
| *Exit Condition:* The user’s password has been changed, and the user has been received an acknowledgement message. OR,  The user has received an error message which informs the user that why the request could not be completed. |

**Use Case 12:**

|  |
| --- |
| *Use case name:* DisplayAdvisorFunctions (Low Priority) |
| *Participant actors:* Initiated by Advisor |
| *Flow of events:* 1. The Advisor sends a request to Campus Online++ to see the Advisor Functions on the screen.  2. Campus Online++ receives the request and responses by presenting the “Advisor Functions Form” the screen so that the Advisor can see the functions of an advisor on the screen. |
| *Entry Condition:* The Advisor should be logged in.  The Advisor chooses Advisor Functions option. |
| *Exit Condition:* The Advisor views functionalities of advisors. OR,  The Advisor has received an explanation indicating why the transaction could not be processed. |

**Use Case 13:**

|  |
| --- |
| *Use case name:* DisplayFreeHoursOfGroup (Medium Priority) |
| *Participant actors:* Initiated by Instructor |
| *Flow of events:* 1. The Instructor sends a request to Campus Online++ to see free hours of selected groups of students on the screen.  2. Campus Online++ receives the request and using the “Instructor Course” table, it responses by presenting the “Free Hours Form” so that the Instructor can select a group of courses that he/she offers and submit the form.  3. The Instructor selects multiple courses among the list of the courses which s/he offers. Once the courses are selected, the Instructor submits the form.  4. Campus Online++ fetches data from the “Schedule” table by using course’ id given by the Instructor. The form shows free hours of the students by cross matching hours of courses given by the Instructor and hours of courses fetched from “Schedule” table. Then, Campus Online++ displays it on the screen so that the Instructor can see the number of students who have common free hours. |
| *Entry Condition:* The Instructor should be logged in.  The Instructor chooses Report Free Hours option. |
| *Exit Condition:* The instructor views free hours of selected students. OR,  The instructor has received an explanation indicating why the transaction could not be processed. |

**Use Case 14:**

|  |
| --- |
| *Use case name:* DisplayQuotas (Medium Priority) |
| *Participant actors:* Initiated by Student, OR Instructor |
| *Flow of events:* 1. The user sends a request to Campus Online++ to see quotas of all courses on the screen.  2. Campus Online++ first retrieves all courses and their current quotas from the “Course” table, then it responds by presenting the “Course Quotas Notice” on the screen so that the user can see quotas of the all courses on the screen. |
| *Entry Condition:* The user chooses Display Quotas option. |
| *Exit Condition:* The user views quotas of all courses. OR,  The user has received an explanation indicating why the transaction could not be processed. |

**Use Case 15:**

|  |
| --- |
| *Use case name:* DisplayOfferedCourses (Medium Priority) |
| *Participant actors:* Initiated by Instructor |
| *Flow of events:* 1. The Instructor sends a request to Campus Online++ to see his/her offered courses along with their sections on the screen.  2. Campus Online++ receives the request from the logged user and it retrieves courses that the Instructor’s offered from the “Instructor Course” table, then it responses by presenting the “Offered Courses Form” to the Instructor on the screen. |
| *Entry Condition:* The Instructor is logged into Campus Online++.  The Instructor chooses Offered Courses option on the screen. |
| *Exit Condition:* Campus Online++ displays the Instructor’s offered courses list along with their sections. OR,  The Instructor has received an error message which informs the Instructor that why the request could not be completed. |

**Use Case 16:**

|  |
| --- |
| *Use case name:* DisplayPrerequisites (Low Priority) |
| *Participant actors:* Initiated by Instructor, Student |
| *Flow of events:* 1. The user sends a request to Campus Online++ to see prerequisite of any course.  2. Campus Online++ receives the request then, it retrieves courses from the “Course” table, then responses by presenting the “Prerequisites Course Form” to the user on the screen. The form contains the list of all courses.  3. The Instructor selects a course which he/she wants to see prerequisites by using select option for that course.  4. Campus Online++ responses by presenting the “Prerequisites Notice” from by using the “Prerequisites” table which contains prerequisites for the course which user wants to see. And displays prerequisites for that course on the screen. |
| *Entry Condition:* The user chooses Display Prerequisites option on the screen. |
| *Exit Condition:* The prerequisites are displayed for that selected course OR,  The user has received an explanation indicating why the prerequisites for that course could not be processed. |

**Use Case 17:**

|  |
| --- |
| *Use case name:* DisplayClassList (Medium Priority) |
| *Participant actors:* Initiated by Instructor |
| *Flow of events:* 1. The Instructor sends a request to Campus Online++ to see list of students who take any course(s) that the Instructor offers in the current semester.  2. Campus Online++ receives the request then, by using the “Instructor Course” table, it fetches the courses the Instructor offers and responses by presenting the “Class List Form” to the Instructor on the screen. The form contains the list of all courses which are given by the Instructor and a button named “Display”.  3. The Instructor selects a course which he/she wants to see student list of the course by clicking on the course name. Once the course is selected, the Instructor submits the form by using the “Display” option.  4. Campus Online++ first, finds the students who take the selected course by using the “Student Course” table, then responses by presenting the “Student List Notice” which contains students’ list along with their personal information and pictures by using the “Student” table. Then, Campus Online++ displays the students as a list along with their personal information and pictures on the screen. |
| *Entry Condition:* The Instructor is logged into Campus Online++.  The Instructor chooses Class List option on the screen. |
| *Exit Condition:* The Instructor has received the list of the students who take the offered class by given the Instructor.  The Instructor has received an error message which informs the Instructor that why the request could not be completed. |

**Use Case 18:**

|  |
| --- |
| *Use case name:* DisplayAttendanceList (Medium Priority) |
| *Participant actors:* Initiated by Instructor |
| *Flow of events:* 1. First, the Instructor chooses a course among the list that contains the courses he/she offers in the current semester. Then, the Instructor chooses Display Attendance List option on the screen to see and edit the attendance list of the selected course given by the Instructor.  2. Campus Online++ fetches the list of students’ id and their attendance data of the selected class from the “Attendance” table. Also, it fetches students’ name and photo from the “Student” table by using the students’ id which obtained from the “Attendance” table. Then, Campus Online++ displays the “Attendance Form” on the screen.  3. The Instructor edits the “Attendance Form” by choosing the select option for each present student, then sends a request to Campus Online++ to update the attendance list of the chosen course which is currently given by the Instructor.  4. Campus Online++ receives the request and update the attendance list. |
| *Entry Condition:* The Instructor is logged into Campus Online++.  The Instructor selects Offered Courses option on the screen. |
| *Exit Condition:* The Instructor has received the list of the students and their attendance data who take the offered class by the Instructor. In addition, the Instructor edits the attendance list. Finally, the attendance list of the selected course is updated by Campus Online++ after the Instructor sends a request to.  The Instructor has received an error message which informs the Instructor that why the request could not be completed. |

**Use Case 19:**

|  |
| --- |
| *Use case name:* SubmitFinalGrades (High Priority) |
| *Participant actors:* Initiated by Instructor |
| *Flow of events:* 1. The Instructor, first, chooses a course among the list that contains the courses he/she offers in the current semester. Then, the Instructor chooses “Enter Grades” option on the screen to give grades of the students who take the selected course given by the Instructor.  2. Campus Online++ retrieves the list of student of the selected course from the “Student Course” table. Then, Campus Online++ responds by presenting the “Final Grades Form”.  3. The Instructor receives the form and gives letter grades of each student by selecting a grade value from the choice option represented by Campus Online++, then sends a request to submit the given letter grades through submit option on the screen to Campus Online++ to be saved.  4. Campus Online++ receives the request and saves the given final letter grades into the “Student Course” table. Then, Campus Online++ displays the “Acknowledgement Notice” to the Instructor along with “Save into Local” and “Print” buttons on the screen. The Instructor could save the submitted list into his/her local, or s/he could print out the submitted list at the end. |
| *Entry Condition:* The Instructor is logged into Campus Online++.  The Instructor selects Offered Courses option on the screen. |
| *Exit Condition:* The Instructor has given the final grades of the students who take the selected course by given the Instructor. In addition, the Instructor has saved the final grades into his/her local or the Instructor has printed out the final grades list.  The Instructor has received an error message which informs the Instructor that why the request could not be completed. |

**Use Case 20:**

|  |
| --- |
| *Use case name:* DisplayConsentRequest (Medium Priority) |
| *Participant actors:* Initiated by Instructor |
| *Flow of events:* 1. The Instructor sends a request to Campus Online++ to see his/her waiting consent request on the screen.  2. Campus Online++ receives the request and fetches all consent request from the “Consent Requests” table. Then Campus Online++ responses by presenting the “Consent Request Form”.  3. The Instructor answers each consent request by first, choosing the consent request and then choosing “Accept” or “Decline” option. Finally, once an option is chosen, the Instructor submits the form to Campus Online++.  4. Campus Online++ receives the form and the information that the Instructor sent along with the form, then Campus Online++ saves the answers given by the Instructor into the “Consent Requests” table. |
| *Entry Condition:* The Instructor is logged into Campus Online++.  The Instructor chooses Consent Requests option on the screen. |
| *Exit Condition:* The Instructor has given answers to each consent requests that are waiting for him/her to accept or decline. OR,  The Instructor has received an explanation indicating why the consent request could not be processed. |

**Use Case 21:**

|  |
| --- |
| *Use case name:* SendMessage (Medium Priority) |
| *Participant actors:* Initiated by Instructor or Student Communicates with Instructor or Student |
| *Flow of events:* 1. The user sends a request to Campus Online++ to send a message to an instructor or a student.    2. Campus Online++ receives the request and displays the “Message Form” that contains a text field to enter receiver's e-mail address and another two text fields for typing message's title and body. The form also contains a send button.  3. The user fills the form, then sends it to Campus Online++ by clicking on the send button.  4. Campus Online++ receives the form and the message information, then using the “Student” or “Instructor” table, it fetches the user id to send the message. Finally, it updates the “Message” table and displays the “Acknowledgement Notice” to the user. |
| *Entry Condition:* The user is logged into Campus Online++.  The user chooses Message Online option on the screen. |
| *Exit Condition:* The user has sent the message OR,  The user has received an explanation indicating why the message could not be send. |

**Use Case 22:**

|  |
| --- |
| *Use case name:* RegisterForCourses (High Priority) |
| *Participant actors:* Initiated by Student  Communicates with Advisor |
| *Flow of events:* 1. The Student sends a request to Campus Online++ to register for courses.  2. Campus Online++ receives the request then, using the “Type Errors” table, it responses by presenting “Type Error Form” to the Student on the screen. The form contains some information about the possible error types and their explanations which the Student must pay attention while s/he is choosing his/her courses. The form also contains a accept option so that the Student can proceed.  3. The Student sends another request to proceed by choosing accept option on the screen.  4. Campus Online++ receives the request and by using student’s department information from the “Student” table, then it responses by presenting another “Register Form” which is composed of two different form parts namely “Selected Course Forms” and “Course List Form”. Course List part contains the courses that filtered according to the Student’s information and quota. In other words, Course List part does not contain any unnecessary course.  5. The Student selects multiple courses or just one course that s/he wants to register. In addition, the Student deletes multiple selected courses, just one selected course or all selected courses at once as well. Once the Student finishes selecting courses, s/he submits the form by using submit option  6. Campus Online++ receives the request and by using the “Course For Electives” table it checks possible ways to enroll to the selected courses. Then it displays a list to show possible enrollment ways to that course on the screen. The possible ways are: 'Non-Credit', 'Complementary Elective', 'Departmental Elective', 'Science Elective', 'Hss Elective', 'Free Elective'.  7. The Student selects the way to take that course from the list, then Campus Online++ saves the list of the courses in the “Schedule” table with the tag named “Incomplete” even the Student does not finish the registration process, once the Student finishes registering, s/he completes registration.  8 Campus Online++ receives the form and saves it into “Student Schedule” table with a tag 'Waiting for approval'. Also Campus Online++ sends a notification to the Student's advisor so that the advisor can approve the Student's schedule. Besides, Campus Online++ sends a notification to the instructor of any course that the Student has chosen and requires consent.    9. The use case named “DisplayWaitingProgramForApproval” is used.  10. Campus Online++ displays the “Acknowledgement Notice” to the Student on the screen. |
| *Entry Condition:* The Student is logged into Campus Online++.  The Student chooses Register For Courses option on the screen.  There shouldn't be any holds for registration for that Student.  The date must be the Student’s registration date. |
| *Exit Condition:* The Student has registered for the courses that s/he wants to. OR,  The Student has received an explanation indicating why the transaction could not be processed. |

**Use Case 23:**

|  |
| --- |
| *Use case name:* DisplayWaitingProgramForApproval (High Priority) |
| *Participant actors:* Advisor |
| *Flow of events:* 1.The Advisor sends a request to Campus Online++ to display the students’ program which are waiting for approval.  2. Campus Online++ receives the request then obtains student ids using the “Student” table (the students who supervised by the Advisor). After obtaining students’ ids, it queries the “Student Schedule” table to retrieve students with the obtained students’ ids and program which is tagged with 'Waiting For Approval'. Then Campus Online++ responds by presenting the “Waiting Students Form”.  3. The Advisor sees the form on the screen. Then, the Advisor chooses a student and submits the form by choosing the View Schedule option.  4. Campus Online++ receives the request and using the selected student’s id, it fetches the schedule of the student from “Student Schedule” table. Then, Campus Online++ responds by presenting the “Waiting Schedules Form”.  5. The Advisor receives the form and chooses either 'Accept Schedule' or 'Decline Schedule' option by doing so the Advisor submits the form.  4. Campus Online++ receives the request and based on the Advisor's choice updates the status field of the “Student Schedule” table. |
| *Entry Condition:* The Advisor is logged into Campus Online++.  The Advisor chooses Waiting Program For Approval option on the screen. |
| *Exit Condition:* The Advisor has listed all unapproved programs on the screen. OR,  The Advisor has received an explanation indicating that there is no such student program to be listed. |

**Use Case 24:**

|  |
| --- |
| *Use case name:* DisplayApprovedProgram (Medium Priority) |
| *Participant actors:* Advisor |
| *Flow of events:* 1. The Advisor sends a request to Campus Online++ to see all approved programs which were confirmed by himself/herself.  2. Campus Online++ receives the request then obtains student ids using the “Student” table (the students who supervised by the Advisor). After obtaining students’ ids, it queries the “Student Schedule” table to retrieve students with the obtained students’ ids and program which is tagged with 'Approved'. Then Campus Online++ responds by presenting the “Approved Students Form”.  3. The Advisor sees the form on the screen. Then, the Advisor chooses a student and submits the form by choosing the View Schedule option.  4. Campus Online++ receives the request and using the selected student’s id, it fetches the schedule of the student from “Student Schedule” table. Then, Campus Online++ responds by presenting the “Approved Schedules Form”. |
| *Entry Condition:* The Advisor is logged into Campus Online++.  The Advisor chooses Approved Program option on the screen. |
| *Exit Condition:* The Advisor has received the list of the approved student programs. OR,  The Advisor has received an explanation indicating that there is no such student program to approve. |

**Use Case 25:**

|  |
| --- |
| *Use case name:* SetQuota (Medium Priority) |
| *Participant actors:* QuotaManager |
| *Flow of events:* 1. The QuotaManager sends a request to Campus Online++ to display and set quota of courses which are available in the current semester on the screen.  2. Campus Online++ receives the request then, using QuotaManager's department information which obtained from “QuotaManager” table, it filters the courses which are obtained from the “Course” table and displays them on the screen.  3. The QuotaManager selects a course by using select button which he/she wants to set quota for it. Once a course is selected, the QuotaManager submits the “Course Quota Form”.  4. Campus Online++ receives the form and the selected course information and by using the Course table, then it presents the “Set Quota Form” which contains the course’s name and the course’s quota. Also the form contains two different options named Special Quota for an Individual Student by using the student’ id, and Common Quota for all students.  5. The QuotaManager sets quota through form for Individual Student or all Students. Once the form is completed, the Quota Manager submits the form.  6. Campus Online++ receives the form and updates the Course Table with newly set quota values. In addition, Campus Online++ displays an acknowledgment message to the QuotaManager on the screen. |
| *Entry Condition:* The QuotaManager is logged into Campus Online++.  The QuotaManager chooses Set Quotas option on the screen. |
| *Exit Condition:* The QuotaManager has set the quota of a course, and the Quota Men has received an acknowledgement message. OR,  The QuotaManager has received an explanation indicating why the transaction could not be processed. |

**Use Case 26:**

|  |
| --- |
| *Use case name:* OpenAddDrop (High Priority) |
| *Participant actors:* Initiated by Student  Communicates with Advisor |
| *Flow of events:* 1. The Student sends a request to Campus Online++ to open add drop for taking more courses and dropping some of courses.  2. Campus Online++ receives the request then it displays the “Send Add Drop Request” form.  3. The Students sees the “Send Add Drop Request Form” and the Student selects send for add-drop option.  4. Campus Online++ receives the request then, Campus Online++ sends a notification to the Advisor of the student by using the “Student” table and updates the “Add Drop Request” table.  5. The Advisor sees this notification which is sent by the Student on the screen and clicks on the Display Add-Drop Request option and with the “Add Drop Request Form” then, the Instructor approves or disapproves the Student’s request by sending the form.  6. Campus Online++ updates the “Add Drop Request” and “Schedule” tables based on the Advisor's choice.  7. The use case named “RegisterForCourses” is used. |
| *Entry Condition:* The Student is logged into Campus Online++.  The Student chooses Open Add Drop option on the screen.  The add drop period should not be passed. |
| *Exit Condition:* The Student has added-dropped a course. OR,  The Student has received an explanation indicating why the transaction could not be processed. |

**Use Case 27:**

|  |
| --- |
| *Use case name:* SendWithdrawRequest (Medium Priority) |
| *Participant actors:* Initiated by Student  Communicates with Advisor |
| *Flow of events:* 1. The Student sends a request to Campus Online++ to withdraw for course/s.    2. Campus Online++ receives the request and by using the Schedule table, it responses by presenting the “Withdraw” form to the Student on the screen. The form contains courses which the Student enrolled.  5. The Student chooses multiple courses or just one course that s/he wants to withdraw by using the select option. When the Student finishes choosing courses it submits the withdraw form by selecting submit option.  6. Campus Online++ receives the request and by using the “Student” table it checks the advisor of the Student. Then it sends a notification to the Advisor of the Student. Also Campus Online++ updates the “Withdraw Request” table and adds courses which the Student wanted to withdraw with a tag “Withdraw Request Sent”. |
| *Entry Condition:* The Student is logged into Campus Online++.  The Student chooses Send Withdraw Request option on the screen.  The Student must be enrolled for course/s. |
| *Exit Condition:* The Student has send withdraw request to his/her Advisor for the courses that s/he wants to withdraw. OR,  The Student has received an explanation indicating why the transaction could not be processed. |

**Use Case 28:**

|  |
| --- |
| *Use case name:* DisplayWithdrawRequest (Medium Priority) |
| *Participant actors:* Initiated by Advisor  Communicates with Student |
| *Flow of events:* 1. The Advisor sends a request to Campus Online++ to display withdraw requests.    2. Campus Online++ receives the request and by using the “Withdraw” table and “Student” table, it fetches the withdraw request which sent to the Advisor by student(s). Then, Campus Online++ responds by presenting the “Withdraw Request Form”.  3. The Advisor receives the form and selects a student to see the courses which the student wants to withdraw. By clicking the student’s id, the Advisor submits the form.  4.Campus Online++ receives the request and by using the student’s id which the Advisor chose, Campus Online++ uses the “Withdraw Request” table to fetch the courses which student wants to withdraw, then it responses by presenting the “Withdraw Courses Form“ form to the Advisor on the screen.  5. The Advisor selects multiple courses or just one course which the student wants to withdraw by using the select option. Then, the Advisor submits the form by choosing accept or decline option.  6. Campus Online++ receives the request and by using the “Student Course” table, it updates the course(s) grade fields with “W” (Withdrew) or “E” (Enrolled) based on the submit option the Advisor has chosen. Also, Campus Online++ updates the “Withdraw” table and changes course's status with “A” (Withdraw Accepted) or “D” (Withdraw Declined). Campus Online++ displays the “Acknowledgement Notice” to the Student. |
| *Entry Condition:* The Advisor is logged into Campus Online++.  The Advisor chooses DisplayWithdraw Request option on the screen. |
| *Exit Condition:* The Advisor has displayed withdraw request/s. OR,  The Advisor has received an explanation indicating why the transaction could not be processed. |

**Use Case 29:**

|  |
| --- |
| *Use case name:* AddUser (High Priority) |
| *Participant actors:* Initiated by Admin |
| *Flow of events:* |
| *Entry Condition:* The Admin is logged into Campus Online++.  The Admin chooses Add User option on the screen. |
| *Exit Condition:* The Admin has registered the user. OR,  The Admin has received an explanation indicating why the transaction could not be processed. |

**Use Case 30:**

|  |
| --- |
| *Use case name:* AddStudent (High Priority) |
| *Participant actors:* Inherited from AddUser use case. |
| *Flow of events:* 1. The Admin sends a request to Campus Online++ to add a new student to the Campus Online++ system.    2. Campus Online++ receives the request and responses by presenting the “User Enrolment Form”.  3. The Admin receives the form and fills out the form by selecting the user type and a department from the department option and entering the personal information parts. Once the Admin completes filling the form, s/he submits the form.  4. Campus Online++ receives the form and the information given by the Admin, then it saves the information into the “Student” table. Campus Online++ also registers the student’s username and password into the “User” table. Finally, Campus Online++ displays the “Acknowledgement Notice” to the Admin. |
| *Entry Condition:* Inherited from AddUser use case. |
| *Exit Condition:* The Admin has registered the student. OR,  The Admin has received an explanation indicating why the transaction could not be processed. |

**Use Case 31:**

|  |
| --- |
| *Use case name:* AddInstructor (High Priority) |
| *Participant actors:* Inherited from AddUser use case. |
| *Flow of events:* 1. The Admin sends a request to Campus Online++ to add a new instructor to the Campus Online++ system.    2. Campus Online++ receives the request and responses by presenting the “User Enrolment Form”.  3. The Admin receives the form and fills out the form by selecting the user type and a department from the department option and entering the personal information parts. Once the Admin completes filling the form, s/he submits the form.  4. Campus Online++ receives the form and the information given by the Admin, then it saves the information into the “Instructor” table. Campus Online++ also registers the instructor’s username and password into the “User” table. Finally, Campus Online++ displays the “Acknowledgement Notice” to the Admin. |
| *Entry Condition:* Inherited from AddUser use case. |
| *Exit Condition:* The Admin has registered the instructor. OR,  The Admin has received an explanation indicating why the transaction could not be processed. |

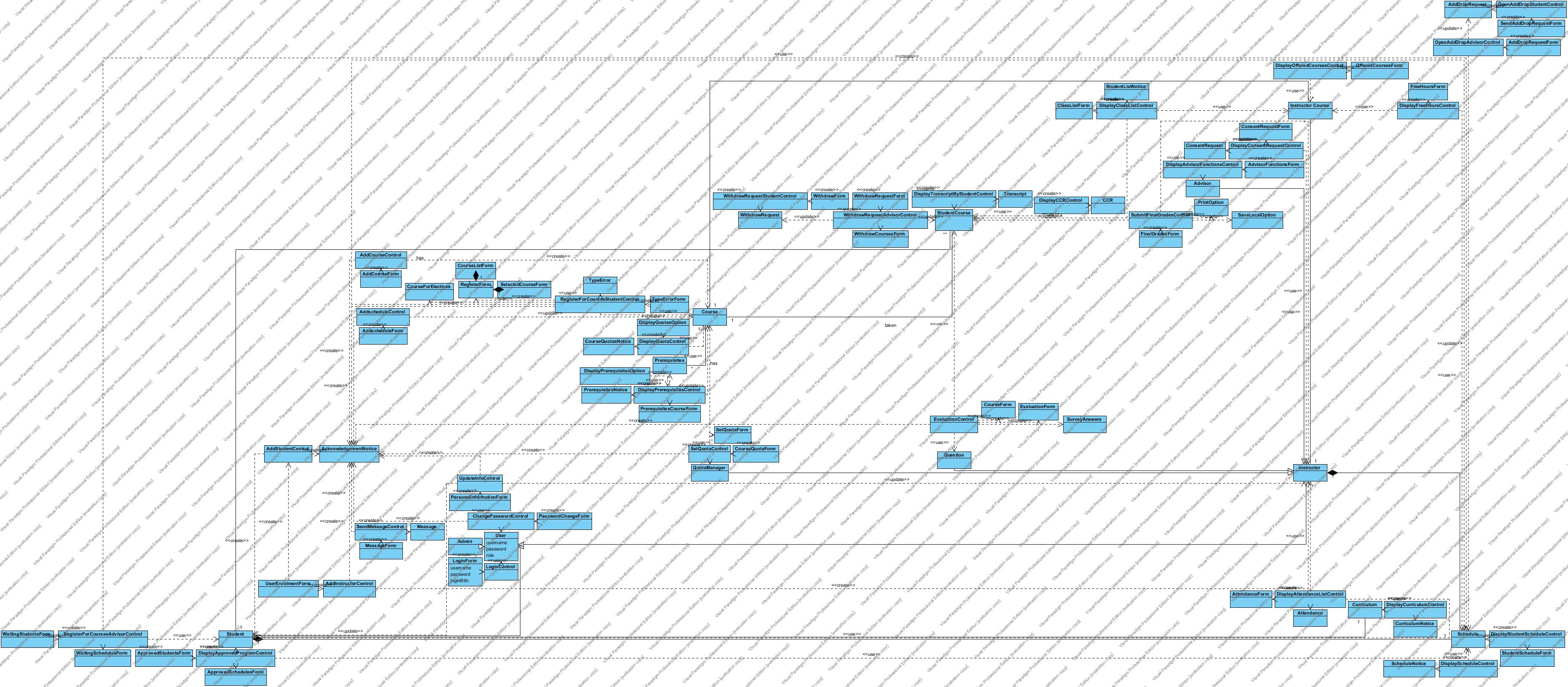
**Use Case 32:**

|  |
| --- |
| *Use case name:* AddCourse (High Priority) |
| *Participant actors:* Initiated by Admin |
| *Flow of events:* 1. The Admin sends a request to Campus Online++ to add a new course to the Campus Online++ system.  2. Campus Online++ receives the request and responses by presenting the “Add Course Form”.  3. The Admin receives the form and fills out the form by typing the course id and a department from the department option and entering the course information parts. Once the Admin completes filling the form, s/he submits the form.  4. Campus Online++ receives the form and the information given by the Admin, then it saves the information into the “Course” table. Finally, Campus Online++ displays the “Acknowledgment Notice” to the Admin. |
| *Entry Condition:* The Admin enters to login screen. |
| *Exit Condition:* The Admin has added the course. OR,  The Admin has received an explanation indicating why the transaction could not be processed. |

**Use Case 33:**

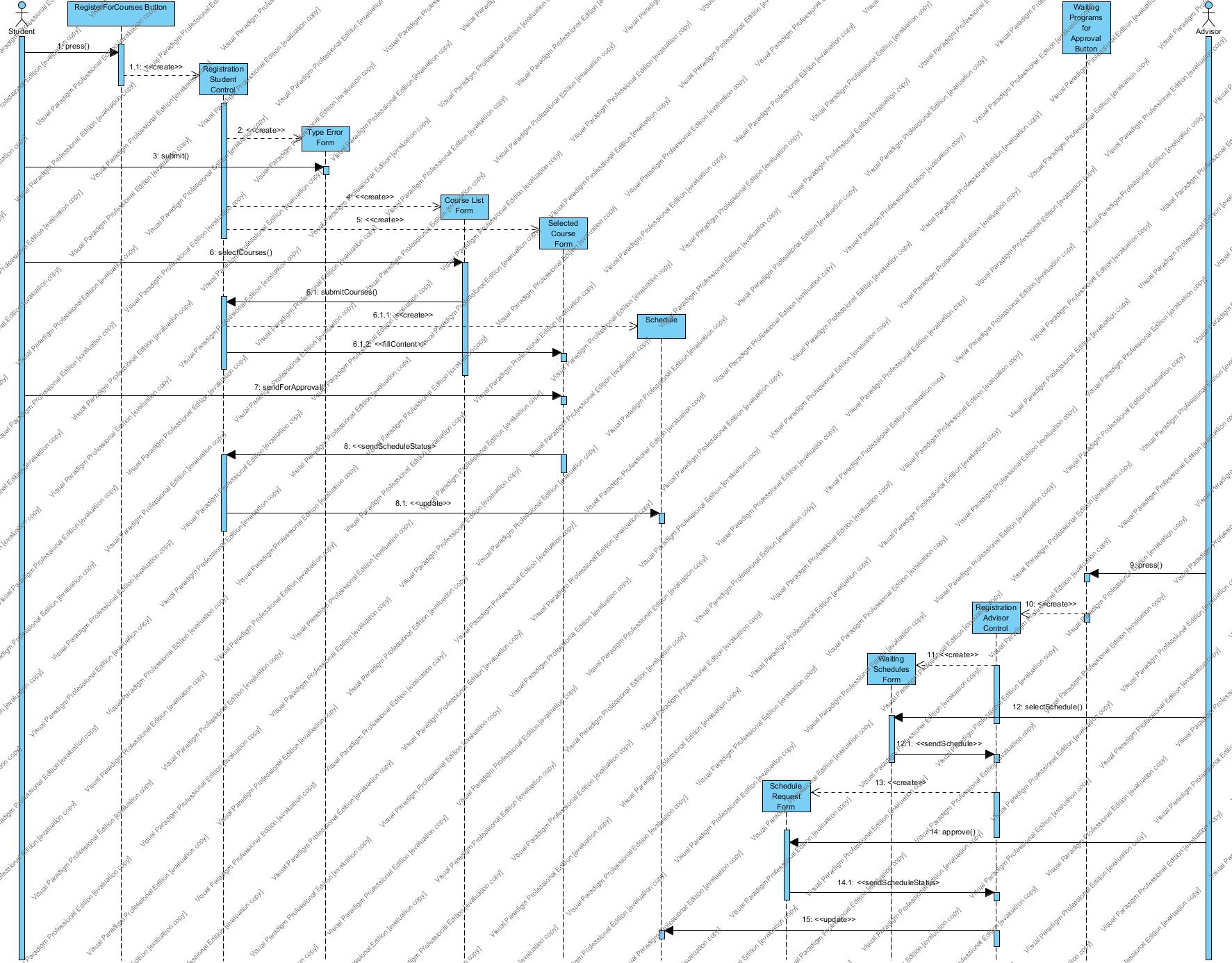
|  |
| --- |
| *Use case name:* AddSchedule (High Priority) |
| *Participant actors:* Initiated by Admin |
| *Flow of events:* 1. The Admin sends a request to Campus Online++ to add a new schedule to the Campus Online++ system.  2. Campus Online++ receives the request and responses by presenting the “Add Schedule Form” using the course id from “Course Table”.  3. The Admin receives the form then selects the course id and selects lecture hours and classrooms for the selected course. Once the Admin completes filling the form, s/he submits the form.  4. Campus Online++ receives the form and the information given by the Admin, then it saves the information into the “Course” table. Finally, Campus Online++ displays the “Acknowledgment Notice” to the Admin. |
| *Entry Condition:* The admin enters to login screen. |
| *Exit Condition:* The Admin has added the schedule. OR,  The Admin has received an explanation indicating why the transaction could not be processed. |

### Object model

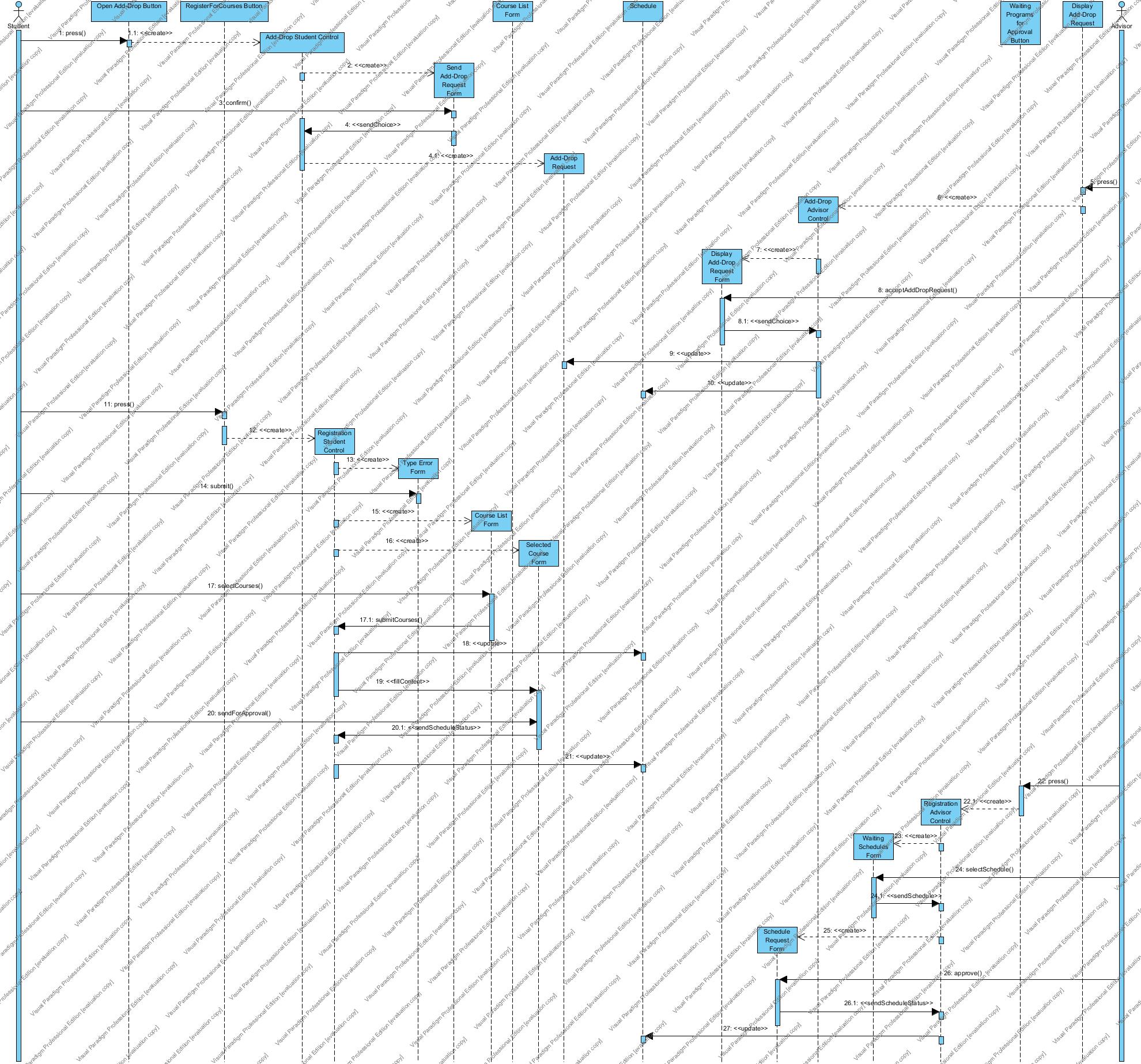


### Dynamic model

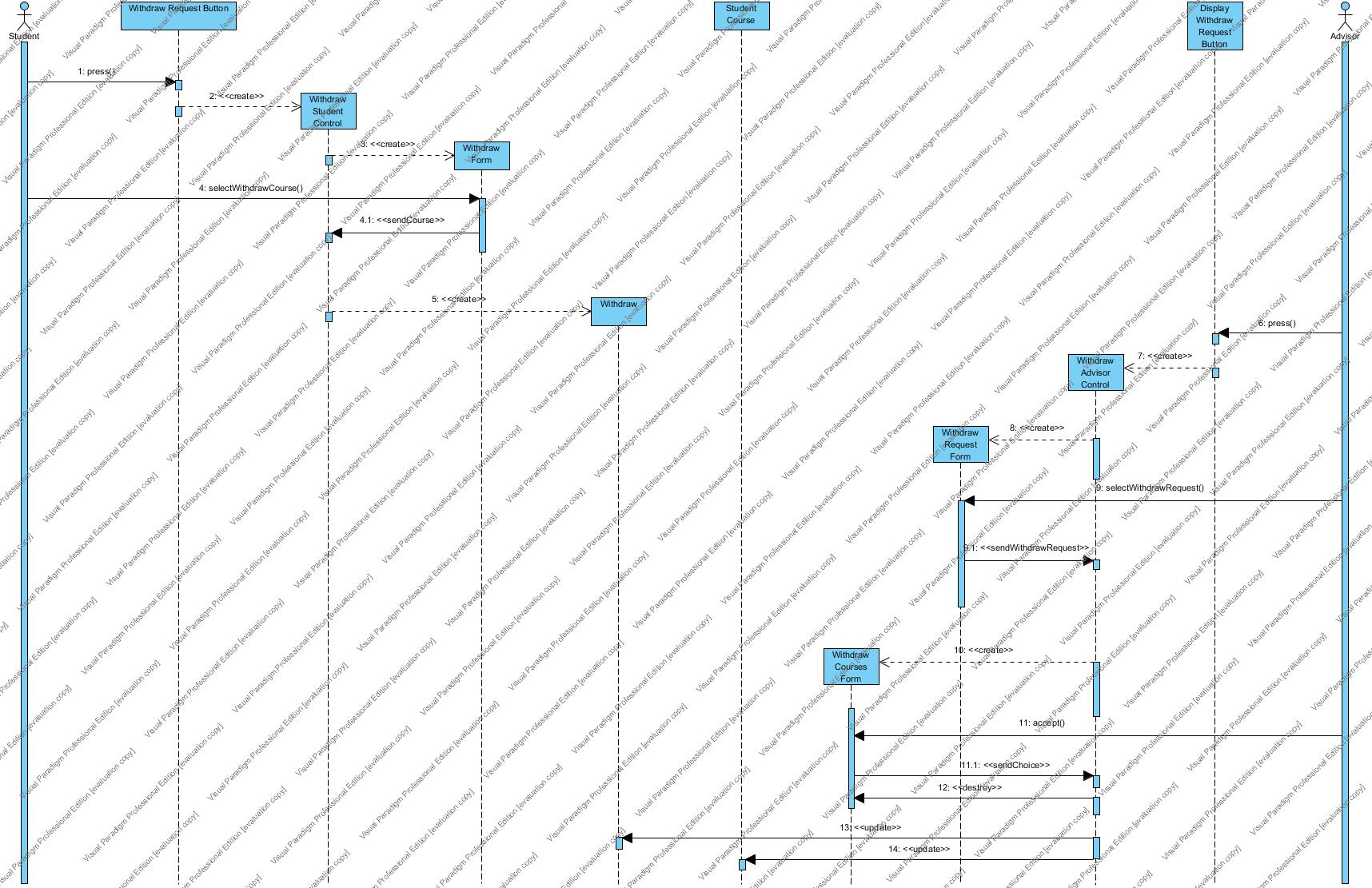
Register



AddDrop

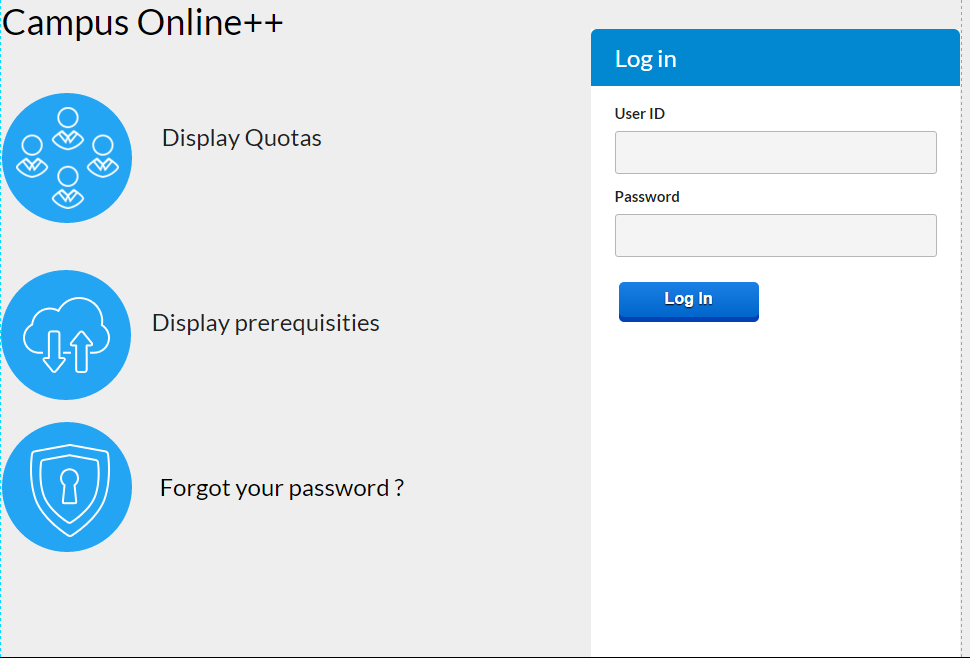


WithDraw

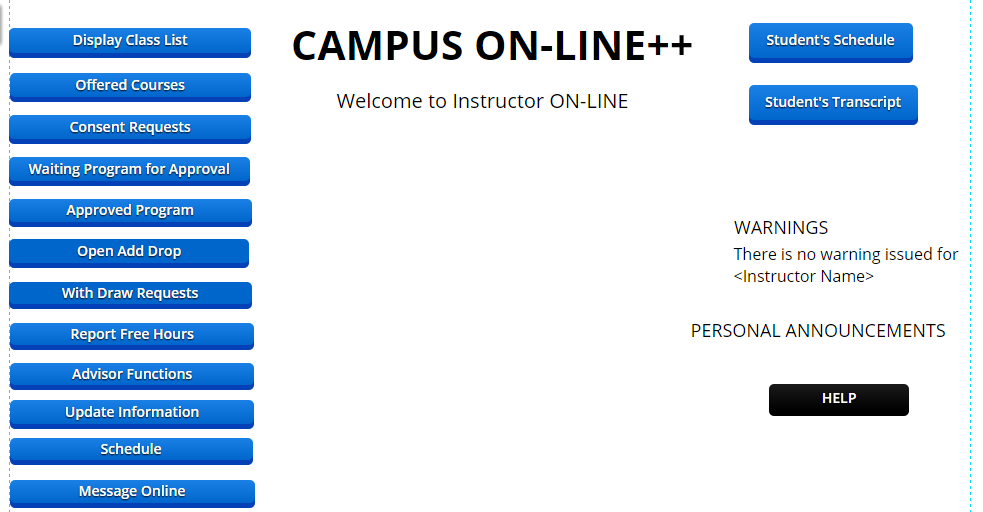


### User interface

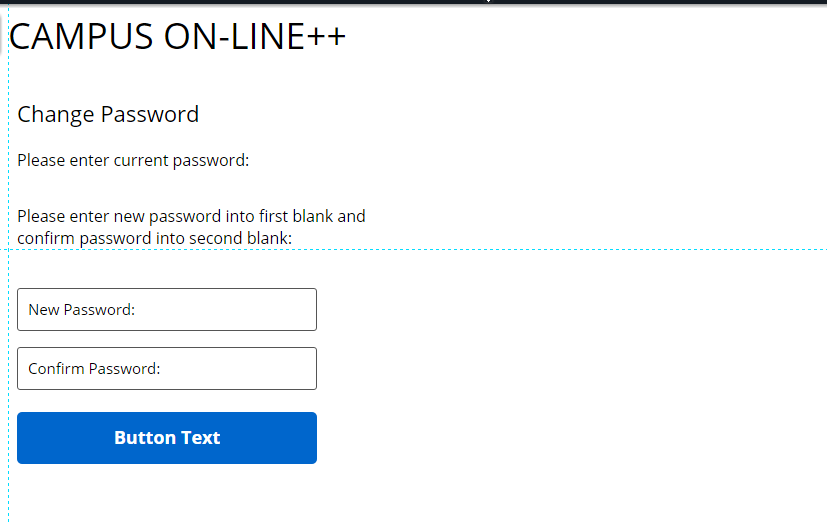
LOGIN



INSTRUCTOR LANDING PAGE



PASSWORD CHANGE



# Glossary

*User table:*Contains username, password, and user role.

*Student table:* Contains the Students’ username, department, personal information, and advisor id.

*Instructor table:* Contains the Instructors' username, department, and personal information.

*Admin table:* Contains the Admins’ username, and personal information.

*Course table:* Contains courses offered that semester with their course id, section, title, lecture hours, department, rooms, credit, maximum quota and number of student enrolled to the course.

*Instructor Course table*: Contains instructor id, course id, course section, year and semester the course given.

*Student Course table:* Contains student id, course id, course section, course elective type (e.g. Mandatory, Complementary Elective, Non-Credit), year and semester the course enrolled.

*Student Schedule table***:** Contains students’ id, course name, classroom of the courses to be lectured and status of the schedule which may be “Waiting for Approval”, “Approved”, “Incomplete”.

*Instructor Schedule table:* Contains instructors’ id, course name and classroom of the course to be lectured.

*Survey Answers table:* Contains courses’ id, instructors’ id, evaluation answers and time stamps which indicates the time of evaluation.

*Course For Electives table:* Contains courses which can be enrolled as electives and the possible ways to enroll such as 'Non-Credit', 'Complementary Elective', 'Departmental Elective', 'Science Elective', 'Hss Elective', 'Free Elective'. For example, CSE466 can be taken as a Departmental Elective or as a Complementary Elective.

*Prerequisites table:* Contains prerequisites for courses.

*Attendance table:* Contains student id, course id, time stamp which indicates the date of the taken attendance, and presence status which indicates whether the student was present or not.

*Quota Manager table:* Contains the Quota Managers' name, department, title.

*Consent Requests table:* Contains the list of consent requests which sent to the Instructor.

Type Errors table: Contains error type names and their explanations.

*Curriculum table:* Contains course id, course title, course credit, semester information.

*Schedule table:* Contains students’ id, course name, classroom of the courses to be lectured and status of the schedule which may be “Waiting for Approval”, “Approved”, “Incomplete”, or “Instructor”.

*Question table:* Contains question id, and question definition.

*Message* *table:* Contains sender user id, receiver user id, message title, message body, time stamp information.

*Type Error table:* Contains type error id, and type error definition.

*Add Drop Request table:* Contains student id, instructor id, request status, and time stamp information.

*Withdraw Request table:* Contains student id, instructor id, course id, request status, time stamp information.

*Login Form:* Contains two text fields, one for username and one for password. The form also contains a login button to provide, the functionality of submitting the form, to the user.

*Personal Information Form:* Contains the user’s (the Student or the Instructor who is logged in to Campus Online++) personal information and text fields which are used to display the personal information of the user and are also used to edit/update some personal information. The form also contains a button named “update” so that the user can submit the form.

*Acknowledgement Notice:* Contains a message which indicates the process has been completed.

*Curriculum Notice:* Contains semester, course code, course full name and credit of the courses information.

*CCR:* Contains courses’ code, courses’ full name, courses’ credit and letter grade of the course taken by the Student.

*Schedule Notice:* Contains rows for weekdays and hours columns of each day row, and fills columns of the rows with the course name and course classroom that the user has that hour on that day.

*Student Schedule Form:* Contains a text field for the purpose of entering the student id and a “Search” button to submit the Student Schedule Form.

*Evaluation Form:* Contains two different parts namely Course Evaluation and Instructor Evaluation. In the Course Evaluation part, there are some questions for the course which was selected by the Student. Likewise, in the Instructor Evaluation part, there are some questions for the instructor who offers the selected course.

*Transcript:* Contains student name, student id, course names which the student has taken so far and grades of the courses.

*Password Change Form:* Contains three text fields namely “password”, “new password”, “confirm new password” and a button named “send” so that the user can submit the form.

*Advisor Functions Form:* Contains “Waiting For Approval” option which the Advisor uses to see the students who are waiting for approval, the form also contains “Approved” option that the Advisor uses to see the students who were approved by the Advisor.

*Course Quotas Notice:* Contains names of all courses and their quotas.

*Reservation Room Form:* Contains the list of available rooms and a button named “reserve” so that the Instructor can submit and reserve the room.

*Offered Courses Form:* Contains the offered courses and their sections list. The form also contains some options namely “Class List”, “Submit Grades” and “Attendance List”.

*Course Form:* Contains the list of all courses’ names.

*Prerequisites Notice:* Contains prerequisite courses’ name for selected course.

*Class List Form:* Contains names, surnames, and student id information of students who enrolled at the course.

*Attendance Form:* Contains students' name, photo, attendance information in terms of percentage, and select option to mark the student as present.

*Final Grades Form:* Contains the id of students who enrolled at the selected course, and a choice option which contains possible letter grades (“AA”, “BA”, “BB”, “CB”, “CC”, “DC”, “DD”, “F”) for each student on the screen.

*Consent Requests Form:* Contains the list of consent requests that waiting for the Instructor to answer. The form also contains “Accept” and “Decline” options.

*Type Error Form:* Contains some information about the possible error types and their explanations which may happen during registration, and their explanation.

*Register Form:* Contains selected courses by the Student and course List. Course List part contains the courses that filtered according to the Student’s information and quota.

*Course List Form:* Contains the courses that filtered according to the Student’s information and quota.

*Selected Course Forms:* Contains the courses which the student selected during the registration process.

*Waiting Students Form:* Contains schedule approval request which sent to the Advisor by students.

*Waiting Schedules Form:* Contains the schedule of the selected student’s which is waiting for approval. Also contains “Accept Schedule” and “Decline Schedule” options.

*Approved Students Form:* Contains the list of student ids (students whose schedule approved by the Advisor) and 'View Schedule' option.

*Approved Schedules Form:* Contains the schedule of the selected student’s which is approved by the Advisor.

*Course Quota Form:* Contains the courses and their remaining quotas.

*Set Quota Form:* Contains the course’s name and the course’s quota. Also the form contains two different options named Special Quota for an Individual Student

*Withdraw Form:* Contains current courses’ names of the student.

*Withdraw Request Form:* Contains withdraw request sent to the Advisor by Students.

*User Enrolment Form:*Contains department select option which is composed of all departments of the school and contains user type select option (“Student” or “Instructor”). The form also contains personal information parts such as Identification, Address, Education and Job, Health and Medical or Previous University Information.

*Add Course Form:*Contains text fields to enter course id, section, title, maximum quota number and select option to choose department information.

*Add Schedule Form:*Contains select option to choose course id hours and select options to choose lecture hours, classrooms for the selected course.

# References

1. Bruegge B. & Dutoit A.H.. (2010). *Object-Oriented Software Engineering Using UML, Patterns, and Java*, Prentice Hall, 3rd ed.
2. Current Campus Online system of Işık University
3. Lecture presentations of the course (the presentations were provided by the Instructor who offered the course).