CENTENNIAL
COLLEGE

**College: Centennial College**

**Course Title: Software Requirements Engineering**

**Course Code: COMP 225 SEC.001**

**Professor: Dr. Mina Labib**

# Team Project – Part B

**Student Name:**

**Karimi, Arash (301303157)**

**Korra, Priya Shruthi (301223736)**

**Lee, Chui (301260201)**

**Leung, Ka Ying (301245290)**

**Lin, Ling (301292283)**

**Date: 17th March 2023**

## Table of Content

### Deliverable #1

### Deliverable #2

### Deliverable #3

### Deliverable #4

### Deliverable #5

### Deliverable #6

### Appendix E

**Deliverable #1 (Ka Ying)**

| Use Case Name | List of Related Requirements ID | Actor(s) | Brief Description |
|---|---|---|---|
| View user profile | FR01, FR02 | Student, Professor, School Staff | The actors will log in by using their credentials. They will click on their profile picture, and in response the application will present their profile. |
| Edit user profile | FR02 | Student, Professor, School Staff | The actors will log in by using their credentials. They will click on their profile picture, in response the application will present their profile. The actors will click on the 'Edit' button and edit their personal information, including password, e-mail address, and phone number. They will then click on the 'Save' button to save the edits. |
| View calendar | FR03 | Student, Professor, School Staff | After logging into the application, the actors click on the 'Calendar' icon, in response the application will show the actors their calendars in month view. By clicking the 'Week' button, the calendar will be shown in week view. By clicking the 'Day' button, the calendar will be shown in day view. Tasks will be displayed each day. |
| Send school announcement | FR04 | School Staff (primary) | After logging into the application, the actors click on the 'Admin' button and select 'Announcement' and then 'Send Announcement', in response the application will present the actor with a form. In the form, the actor will select the recipient groups, type of announcement and enter the details. The actors will click |

| | | | |
|---|---|---|---|
| | | | ‘Schedule Message’ and select the time and date of the delivery. After clicking ‘Preview’, the application will automatically generate a preview of the announcement page. The actor will then click ‘Submit’ to send notification to the corresponding user accounts. |
| | | Professor and Student (secondary) | After receiving mobile notification, the actors will click into the notification and read the announcement. |
| Search for answers relating to a particular issue | FR05, FR07 | Student, Professor, School Staff (primary) | After logging into the application, the actors search for any information by clicking the ‘Search’ icon, in response the application will present the actors with the results in the FAQ page. |
| | | School Admin Staff, IT Administrator (secondary) | After logging into the application, the actors click on the ‘Admin’ button and select ‘Site Management’, in response the application will present the actors a list of all pages. The actors then select ‘FAQ’ and click ‘Edit’. The actors can either click ‘Save’ or ‘Submit’ after editing. |
| Access virtual ID card | FR02, FR06 | Student | After logging into the application, the actors will click on their profile picture, in response the application will present the actors their profile. By clicking the ‘myCard’ button, the virtual card with barcode will be shown. |
| | | Professor, School Staff | After logging into the application, the actors will click on their profile picture, in response the application will present the actors their profile. |

| | | | |
|---|---|---|---|
| | | | By clicking the “staffCard’ button, the virtual card with barcode will be shown. |
| Upload course materials | FR08 | Professors, School Staff | After logging into the application, the actors will click on the particular course, in response the application will prompt the actors to select the section number from a list. After choosing the section number, the actors will click on ‘Content’ and select the place they want to upload the materials. The actors will then click ‘Add Content’. The application will show a form in which the actors will select the recipient groups, enter the details and upload materials. After clicking ‘Preview’, the application will automatically generate a preview of the page. The actor will then click ‘Submit’ to upload. |
| View and update attendance records | FR09 | Student | After logging into the application, the actors will click on the particular course. From the drop-down menu in the course page, the actors will click on ‘Course Tools’ and then ‘Attendance’, in response the application will show the attendance records. |
| | | Professor | After logging into the application, the actors will click on the particular course. From the drop-down menu in the course page, the actors will click on ‘Course Tools’ and then ‘Attendance’ in response the application will show the attendance records. By clicking ‘Update’, the actors can update the |

| | | | |
|---|---|---|---|
| | | | attendance records by checking the boxes next to the students’ names. |
| Seek help with assignments | FR04, FR10 | Student, Professor | After logging into the application, the actors will click on the course. From the drop-down menu in the course page, the actors will click on ‘Discussion’, and then click ‘Create Topic’, in response the application will show a form in which the actors will enter the subject and details. After clicking ‘Preview’, the application will automatically generate a preview of the topic. The actor will then click ‘Submit’ to submit. Students and professors who are in this course will then receive a notification and are able to answer the question by commenting in the discussion forum. |
| Grade assignments | FR11 | Professor (primary) | After logging into the application, the actors will click on the course, in response the application will prompt the actors to select the section number from a list. After choosing the section number, the actors will click on ‘Grades’ and select the assignment to be graded. A table with student names, student numbers and their submitted assignments will then be shown. The actors will open the students’ assignments one by one and then input their grades in the column ‘Grades’. The actors can either click ‘Save’ or ‘Submit’ after inputting the grades. |
| | | Student (secondary) | The actors will receive a mobile notification after assignments/tests are graded. An alternative way to |

|  |  |  |  |
|---|---|---|---|
|  |  |  | view their grades is to log into the application and click on a particular course. From the drop-down menu in the course page, the actors will click on ‘Grade’, in response the application will show a list of grade items and the grades. |
| Submit assignments | FR11 | Student | The actors log into the application and click on the course. From the drop-down menu in the course page, the actors will click on the "Assignments", then select a specific assignment. The app allows the actors to upload their completed assignment and submit it to the professors. |
| View campus events | FR13 (amendment in Part A) | Student, Professor (primary) | After logging into the application, the actors click on the "Events" tab, then selects a specific date. The app displays a list of campus events happening on that day, including lectures, workshops, and social activities. |
|  |  | School Staff (secondary) | After logging into the application, the actors click on the ‘Admin’ button and select ‘Event Management’, then selects a specific event. The app will show real-time updates on the event, such as the number of participants and any updates. The actors can also click on ‘Event Report’ to generate reports on the event's success based on attendance and other metrics. |
| Register for campus events | FR14 (amendment in Part A) | Student, Professor (primary) | After logging into the application, the actors click on the "Events" tab, then selects a specific event. The app allows the user to RSVP for the event and provides any necessary details, such as time, location, and dress code. |

|  |  |  |  |
|---|---|---|---|
| View campus news | FR04 | Student, Professor (primary) | After logging into the application, the actors click on the "News" tab, then selects a specific article |
|  |  | School Staff (secondary) | After logging into the application, the actors click on the 'Admin' button and select 'Announcement'. The actors will then select a specific news from a list to view it in detail. The app provides analytics to help the actors measure the impact of their announcements, including the number of views and shares. |
| View campus map | FR15 (amendment in Part A) | Student, Professor (primary) | After logging into the application, the actors click on the "Map" icon, then select the campus they want to view. The app displays a map of the campus, including buildings, parking, and other points of interest. |
|  |  | School Staff (secondary) | After logging into the application, the actors click on the "Map" icon, then select the campus they want to view. The actors can edit any existing location on the map by navigating to the "Edit Location" button. If a location is no longer relevant or has been removed from the campus, the actors can delete it by clicking the 'Delete Location" button. |
| Access library resources | FR16 (amendment in Part A) | Student, Professor, School Staff | On the app dashboard, the actors click on the "Library" tab, in response the application will prompt the actors to the existing school library page. |
| View course outline | FR07, FR17 (amendment in Part A) | Student (primary) | After logging into the application, the actors will click on the course. From the drop-down menu in the course page, the actors will click on 'Content', in response the application will show the course goals, learning outcomes, |

| | | | |
|---|---|---|---|
| | | | assignments, grading criteria, and instructor information. |
| | | Professor (secondary) | After logging into the application, the actors will click on the particular course, in response the application will prompt the actors to select the section number from a list. After choosing the section number, the actors will click on 'Content' and select 'Course Outline'. |
| Register for course | FR18 (amendment in Part A) | Student | The actors will log in by using their credentials. They will click on their profile picture, in response the application will present their profile. From the drop-down menu in the profile page, the actors will click on 'Registration' and view the list of available courses. The actors select the courses they want to register for by clicking on the course name or adding them to their cart. The app shows the course details, including the course description, prerequisites, and availability. The app displays the actors' schedule and any conflicts with the courses they have selected. If there are conflicts, the app recommends alternative courses or sections. Once the actors have selected their courses and verified their schedule, they click the "Register" button to confirm their registration. |
| View student records | FR19 (amendment in Part A) | School Admin Staff | After logging into the application, the actors click on the 'Admin' button and select 'Records Management'. The actors select the student whose records they want to view or update. The app will show the student's personal information, academic history, and other relevant data. The actors can click on 'Edit' to update the record. The records are stored in the database and can be |

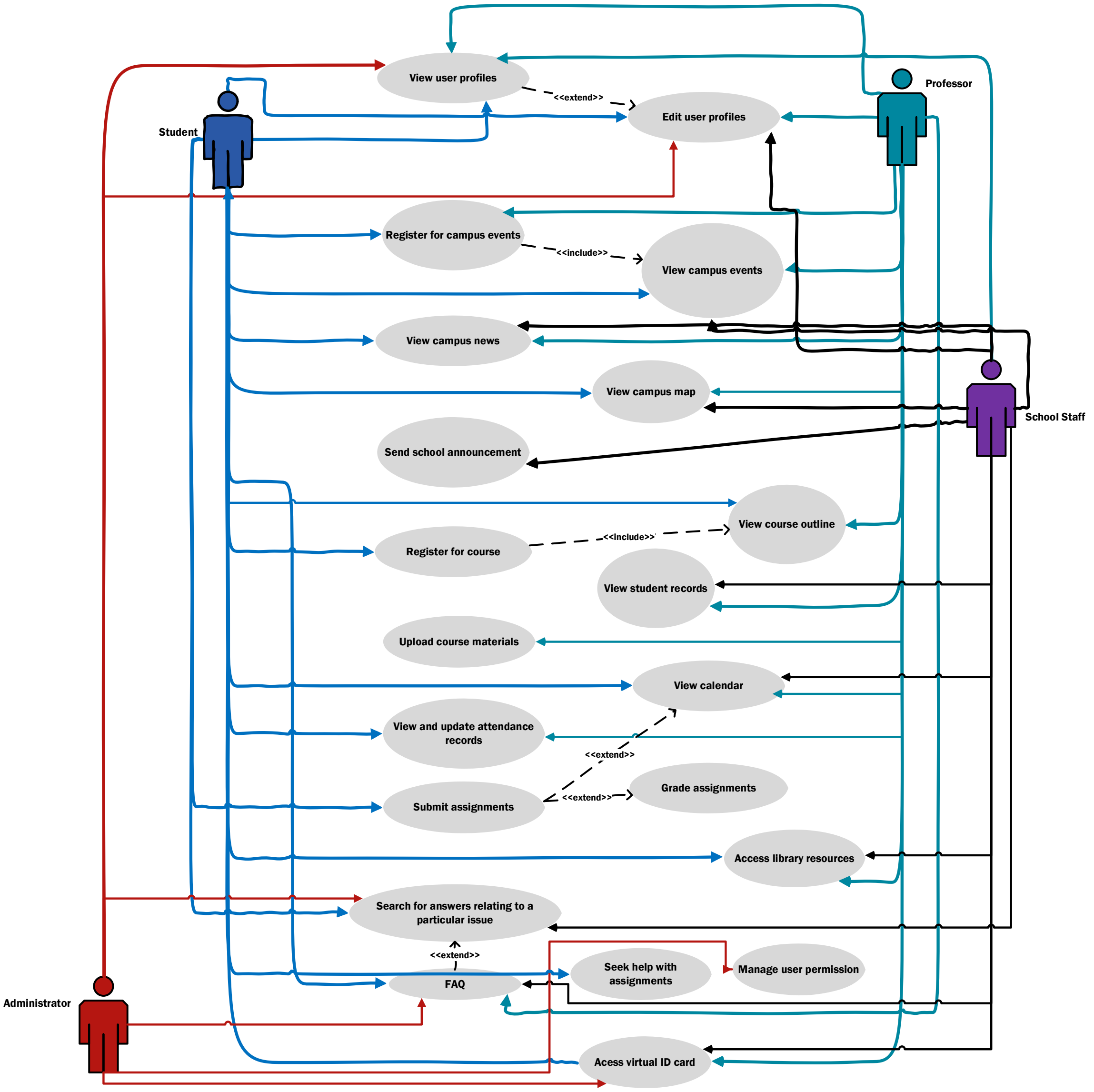
| | | | |
|---|---|---|---|
| | | | used for accreditation and compliance purposes. |
| Manage user permission | FR20 (amendment in Part A) | IT administrator | After logging into the application, the actors click on the 'Admin' button and select 'User Management'. The app will display a list of all user accounts and their roles (student, teacher, staff). The actors can select a user to view their details and edit their account information. They can also set up access controls and permissions by selecting the "Permissions" tab, which allows the actors to specify which resources (such as files, applications, or websites) each user can access. In case of any security breaches or concerns, the actors can use the "Security" tab to view user activity logs and investigate any suspicious behavior. |

**Deliverable #2 (Priya)**

| | |
|---|---|
| Use Case Name | View Calendar |
| Priority | Must |
| Actors | Student, Professor, Student admin |
| Pre-conditions | • The user is logged into the application.<br>• The student is enrolled in at least one course.<br>• The user has created a calendar or has at least one calendar associated with their account. |
| Post-conditions | • The user can view and edit their calendar. |
| Main Scenario | 1. The user opens the application.<br>2. The user navigates to the "Calendar" section.<br>3. The system displays a list of calendars associated with the user's account.<br>4. The user selects the calendar they want to view or edit.<br>5. The system displays the selected calendar.<br>6. The user can view all events in the calendar, including event name, date, time, location, and any additional notes.<br>7. The user can edit or delete existing events.<br>8. The user can add new events to the calendar.<br>9. The system prompts the user to provide information for the new event, including event name, date, time, location, and any additional notes.<br>10. The user provides the necessary information and saves the new event.<br>11. The system adds the new event to the calendar.<br>12. The user can view the updated calendar with the new event. |

## Deliverable #3 (Ling)

### Use Case Diagram



**Deliverable #4 (Ling)**

Based on the use case diagram, here is a list of analysis classes identified using the Noun technique:

1. User (generalization of Student, Professor, system administrator and School Staff)
    a. UserID
    b. UserName
    c. UserRole
    d. UserRight

2. Calendar (course, lesson, semester)
    a. CalendarType (Quiz, Assignment, Test)
    b. CalendarDescription (brief description of the task)

3. Announcement (course, lesson, semester)
    a. AnnouncementType (course-based, school-based)
    b. EmergencyLevel (High, Medium, Low)
    c. AnnouncementDescription (brief description of the task)
    d. Recipient(Individual, Class, Group)

4. Course
    a. CourseTitle (The name or title of the course.)
    b. CourseCode
    c. CourseSection
    d. CourseDuration (The length of time the course)
    e. CourseSemester (Fall2022, Win2023, Sum2023, Fall2023)
    f. CourseInstructor
    g. CourseFormat (in-person, online or a hybrid format)
    h. CourseCredit

User:

The User class is a generalization of Student, Professor, system administrator, and School Staff. It has the following attributes:

UserID: a unique identifier for each user in the system

UserName: the name of the user

UserRole: the role of the user in the system (e.g., student, professor, staff, administrator)

UserRight: the level of access rights for the user in the system (e.g., read-only, read-write)

Calendar:

The Calendar class represents the schedule of courses, lessons, and semesters. It has the following attributes:

CalendarType: the type of task scheduled in the calendar (e.g., quiz, assignment, test)

CalendarDescription: a brief description of the task scheduled in the calendar

The Calendar class is related to the Course class as each calendar event is associated with a course. It is also related to the Announcement class as both classes can be associated with a lesson or semester.

Announcement:

The Announcement class represents announcements related to a course, lesson, or semester. It has the following attributes:

AnnouncementType: the type of announcement (e.g., course-based, school-based)

EmergencyLevel: the level of emergency of the announcement (e.g., high, medium, low)

AnnouncementDescription: a brief description of the announcement

Recipient: the recipient(s) of the announcement (e.g., individual, class, group)

The Announcement class is related to the Course class as each announcement is associated with a course.

Course:

The Course class represents a course offered in the school. It has the following attributes:

CourseTitle: the name or title of the course

CourseCode: a unique identifier for the course

CourseSection: the section of the course

CourseDuration: the length of time the course takes

CourseSemester: the semester during which the course is offered

CourseInstructor: the instructor(s) teaching the course

CourseFormat: the format of the course (e.g., in-person, online, hybrid)

CourseCredit: the number of credits the course is worth

The Course class is related to the Calendar class as each course has a schedule of tasks and events associated with it. It is also related to the Announcement class as announcements related to a course are associated with it. The Course class has a composition relationship with the CourseInstructor class, as each course has one or more instructors teaching it.

**Deliverable #5 (Arash)**

Before the brainstorming session, we needed to have a clear understanding of what the app is meant to accomplish and who the target users are. This involved defining the scope of the app, identifying the key features and functions that it should have, and establishing the goals we hope to achieve with it.

Once we had a clear understanding of the app's scope and goals, we started brainstorming possible features and classes that could be included. This step involved using techniques such as mind mapping or discussions on scrum meetings with team.

After generating a list of potential features and classes, I started to group them together based on their similarities and dependencies. For example, group features related to course management such as getting enrolled, view and manage course information and content into a "Course" class, while features related to student identity and access to virtual ID card included into an "ID Card" class.

At last, when we had a rough grouping of features into classes, we started to define the responsibilities and interactions between those classes. This step involved creating CRC cards for each class, which outlined the class's responsibilities, its collaborators, and its dependencies. These CRC cards are shown in appendix E.

Finally, I created the Visio file with all CRC cards included in them. Each card has a Responsibility and Collaboration section which announces their roles and rules. Each class is based on the notes that I took from scrum meeting that team had.

**Deliverable #6 (Lee)**

Here is the updated domain class diagram based on the responsibilities and collaborations identified in the CRC index cards:

| |
|---|
| Class: User |
| Attributes:<br>UserID (private): a unique identifier for each user<br>UserName (private): the username of the user<br>UserRole (private): the role of the user, such as student, professor, school administrator, or school staff<br>UserRight (private): the set of permissions granted to the user |
| Methods:<br>Login(username: string, password: string): Allows the user to log in to the application using their credentials<br>EditProfile(): Allows the user to edit their personal information, including their password, email address, and phone number<br>ViewProfile(): Allows the user to view their profile information<br>receiveAnnouncement(): Allows the user to receive announcements related to their activities in the system<br>EnrollCourse(): Allows the user to enroll in a course<br>SeekHelp(): Allows the user to seek help with a particular activity in the system |

| |
|---|
| Subclass: Student |
| Attributes:<br>Major (private): the student's major<br>Year (private): the year of study for the student |
| Methods:<br>Enroll(): allows the student to enroll in courses<br>ViewTranscript(): allows the student to view their academic transcript |

| |
|---|
| Subclass: Professor |
| Attributes:<br>Department (private): the department the professor belongs to<br>Office (private): the office location for the professor |
| Methods: |

| |
|---|
| CreateCourse(): allows the professor to create a course<br>GradeAssignments(): allows the professor to grade student assignments |

| Subclass: SchoolAdministrator |
|---|
| Attributes:<br><br>AdminRole (private):  the specific role of the administrator, such as registrar or dean<br>AccessLevel (private):  the level of access granted to the administrator |
| Methods:<br><br>manageUsers(): allows the administrator to manage users in the system<br>manageCourses(): allows the administrator to manage courses |

| Subclass: SchoolStaff |
|---|
| Attributes:<br><br>StaffRole (private):  the specific role of the staff, such as IT support or facilities management<br>Department (private):  the department the staff member belongs to |
| Methods:<br><br>assist(): allows the staff to provide assistance to users<br>manageResources(): allows the staff to manage school resources, such as equipment or facilities |

| Class: Calendar |
|---|
| Attributes:<br><br>CalendarType (private): the type of event on the calendar, such as a quiz, assignment, or test<br>CalendarDescription (private): a brief description of the task on the calendar |
| Methods:<br><br>ViewCalendar(viewType: string): Allows the user to view the calendar in month, week, or day view |

| Class: Announcement |
|---|
| Attributes:<br>AnnouncementType (private): the type of announcement, such as course-based or school-based<br>EmergencyLevel (private): the level of urgency for the announcement, such as high, medium, or low<br>AnnouncementDescription (private): a brief description of the announcement<br>Recipient (private): the intended recipient of the announcement, such as an individual, class, or group |
| Methods:<br>SendAnnouncement(recipient: Recipient, type: AnnouncementType, description: string, schedule: Date): Allows the school staff to send an announcement to the recipient<br>ViewAnnouncement(): Allows the user to view the announcement they have received |

| Class: Course |
|---|
| Attributes:<br>CourseTitle (private): the name or title of the course<br>CourseCode (private): a unique identifier for the course<br>CourseSection (private): the section number for the course<br>CourseDuration (private): the length of time for the course<br>CourseSemester (private): the semester during which the course is offered<br>CourseInstructor (private): the instructor for the course<br>CourseFormat (private): the format of the course, such as in-person, online, or hybrid<br>CourseCredit (private): the number of credits for the course |
| Methods:<br>RegisterForCourse(): Allows the student to register for the course<br>ViewCourseOutline(): Allows the student to view the course outline<br>UploadMaterials(section: int, recipient: Recipient, description: string, materials: Material[]): Allows the professor or school staff to upload course materials<br>ViewAttendance(course: Course): Allows the student or professor to view the attendance records for a course<br>SeekHelpWithAssignments(topic: string, description: string): Allows the student or professor to seek help with assignments through the discussion forum<br>GradeAssignments(assignment: Assignment, grades: int[]): Allows the professor to grade assignments |

SubmitAssignments(assignment: Assignment, submission: Submission): Allows the student to submit an assignment

Class: Grades

Attributes:

GradeID (private): a unique identifier for the grade
StudentID (private): the ID of the student who received the grade
AssignmentID (private): the ID of the assignment for which the grade was received
GradeValue (private): the value of the grade received

Methods:

calculateGrade(): calculates the grade based on the value of the grade received

Class: Events

Attributes:

EventID (private): a unique identifier for the event
EventTitle (private): the title of the event
EventDate (private): the date of the event
EventLocation (private): the location of the event
EventType (private): the type of event, such as lecture, exam, or discussion

Methods:

registerEvent(): allows the student to register for the event
presentEvent(): displays the details of the event

Class: Assignment

Attributes:

AssignmentID (private): a unique identifier for the assignment
AssignmentTitle (private): the title of the assignment
AssignmentDesc (private): the description of the assignment
DueDate (private): the due date of the assignment

Methods:

submitAssignment(): allows the student to submit the assignment
presentAssignment(): displays the details of the assignment

| Class: News |
|---|
| Attributes:<br><br>NewsID (private): a unique identifier for the news article<br>NewsTitle (private): the title of the news article<br>NewsDesc (private): the description of the news article<br>NewsDate (private): the date of the news article |
| Methods:<br><br>getNewsInformation(): retrieves the information about the news article<br>presentNewsInformation(): displays the details of the news article |

| Class: Maps |
|---|
| Attributes:<br><br>MapID (private): a unique identifier for the map<br>MapTitle (private): the title of the map<br>MapDesc (private): the description of the map<br>MapType (private): the type of the map, such as campus map or venue map |
| Methods:<br><br>getMapInformation(): retrieves the information about the map |

| Class: Library |
|---|
| Attributes:<br><br>LibraryID (private): a unique identifier for the library resource<br>LibraryTitle (private): the title of the library resource<br>LibraryDesc (private): the description of the library resource |

| Class: Records |
|---|
| Attributes:<br><br>RecordID (private): a unique identifier for the record<br>StudentID (private): the ID of the student to whom the record belongs<br>RecordDesc (private): the description of the record<br>RecordDate (private): the date on which the record was created |

| Class: Help |
|---|
| Attributes:<br><br>HelpID (private): a unique identifier for the help resource |

HelpTitle (private): the title of the help resource
HelpDesc (private): the description of the help resource
HelpDate (private): the date on which the help resource was created

Use Case Scenario Narrative:

**1. Logging in to the application:**

The user enters their username and password to log in to the application. If the credentials are correct, the user is granted access to the system. If not, the user is prompted to re-enter their credentials.

**2. Viewing and Editing User Profile:**

The user can view their profile information by clicking on the "View Profile" button. This allows them to see their personal information, including their name, email address, phone number, and role in the system. If the user wants to edit their profile, they can click on the "Edit Profile" button, which allows them to change their personal information, including their password, email address, and phone number.

**3. Receiving Announcements:**

The user can receive announcements related to their activities in the system. When an announcement is sent to the user, it appears in their notification center. The user can then click on the announcement to read its description and view any attached files.

**4. Enrolling in a Course:**

The user can enroll in a course by selecting the course from a list of available courses and clicking on the "Enroll" button. If the course has a limited number of seats, the user may need to wait until a seat becomes available.

**5. Viewing Course Outline:**

The user can view the course outline by clicking on the "View Course Outline" button. This allows them to see the topics covered in the course and the schedule of assignments and tests.

**6. Uploading Course Materials:**

The instructor or school staff can upload course materials by selecting the course section, recipient, description, and materials to upload and clicking on the "Upload Materials" button.

**7. Viewing Attendance Records:**

The user can view the attendance records for a course by selecting the course from a list of courses they are enrolled in and clicking on the "View Attendance" button.

**8. Seeking Help with Assignments:**

The user can seek help with assignments by selecting the assignment they need help with, describing the topic they need help with, and clicking on the "Seek Help" button. This allows them to access the discussion forum for the assignment and ask for help from other users.

**9. Grading Assignments:**

The instructor can grade assignments by selecting the assignment from a list of assignments and entering the grades for each student.

**10. Submitting Assignments:**

The user can submit an assignment by selecting the assignment from a list of assignments and clicking on the "Submit Assignment" button. This allows them to upload the file containing their completed assignment.

**11. Viewing Calendar:**

The user can view the calendar by selecting the type of view (month, week, or day) and clicking on the "View Calendar" button. This allows them to see the events and deadlines on the calendar.

**12. Sending Announcements:**

The school staff can send announcements to the recipient by selecting the recipient, type, description, and schedule for the announcement and clicking on the "Send Announcement" button.

**13. Viewing Announcements:**

The user can view the announcements they have received by clicking on the "View Announcement" button. This allows them to see the description of each announcement and any attached files.

**14. Registering for Events:**

The user can register for events by selecting the event from a list of available events and clicking on the "Register Event" button.

**15. Presenting Events:**

The user can present the details of an event by clicking on the "Present Event" button. This allows them to see the date, location, and description of the event.

### 16. Viewing News:

The user can view the news articles by clicking on the "Get News Information" button. This allows them to see the title and description of each news article.

### 17. Viewing Maps:

The user can view the maps by clicking on the "Maps" icon on the home screen. This will display a list of available maps, including campus maps and venue maps. The user can then select a map to view its details, such as the location and layout of buildings or the seating arrangement of a venue. The user can also use search filters to find specific locations on the map, such as a particular building or room. Additionally, the user can zoom in or out and pan the map to view different areas.

### 18. User accesses library resources:

The user can access library resources by searching for them through the system's library feature. They can enter keywords, titles, authors, or other relevant information to search for the desired resources. Once they find the resource they need, they can access it by clicking on the resource's title or description. Depending on the resource, they may be able to view it online or download it to their device.

### 19. User views records:

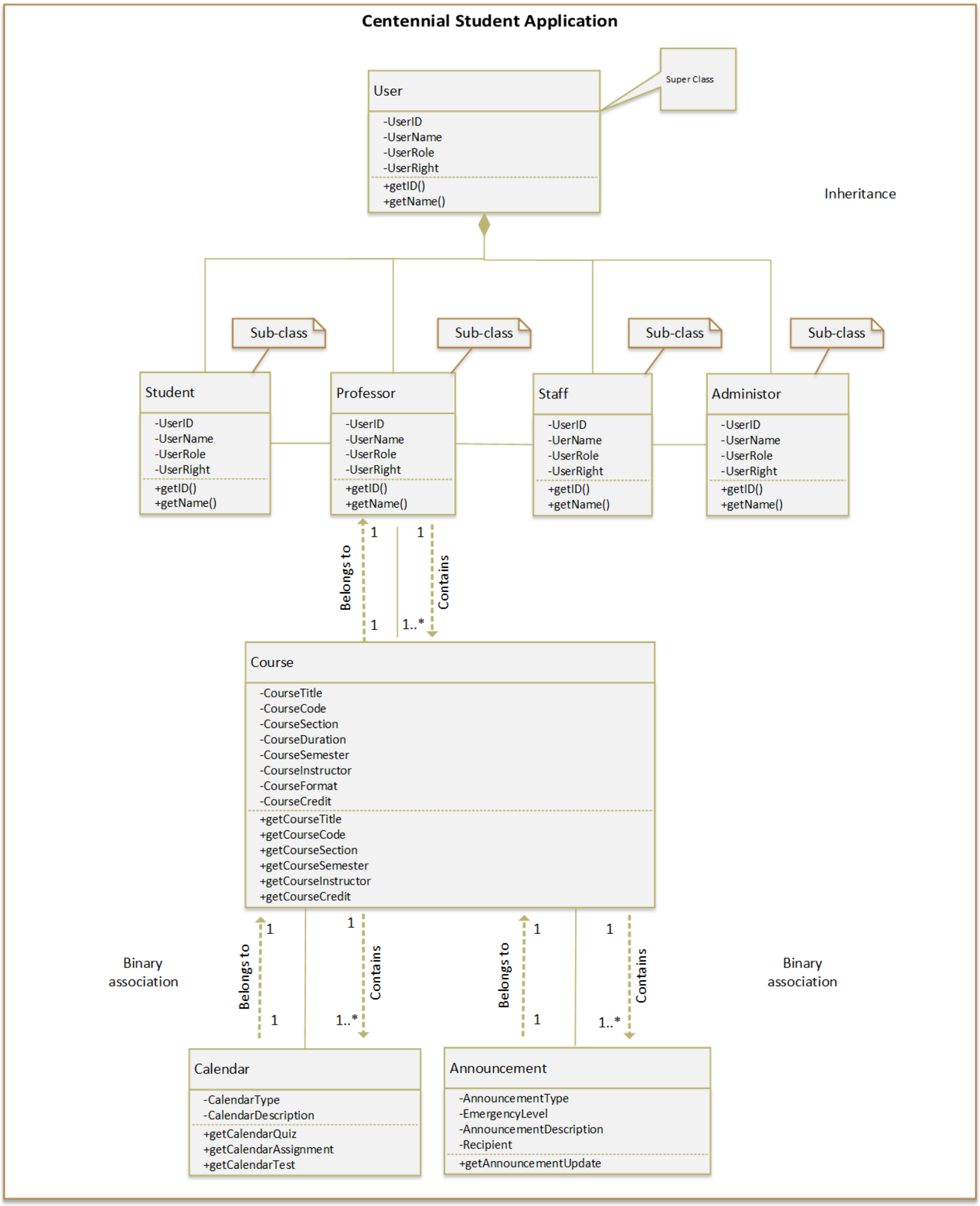
User views records by navigating to the "Records" section of the application. From there, the user can select the specific record they want to view by either searching for it or selecting it from a list. Once the record is selected, the user can view its details, including the description and date it was created.

### 20. User seeks help:

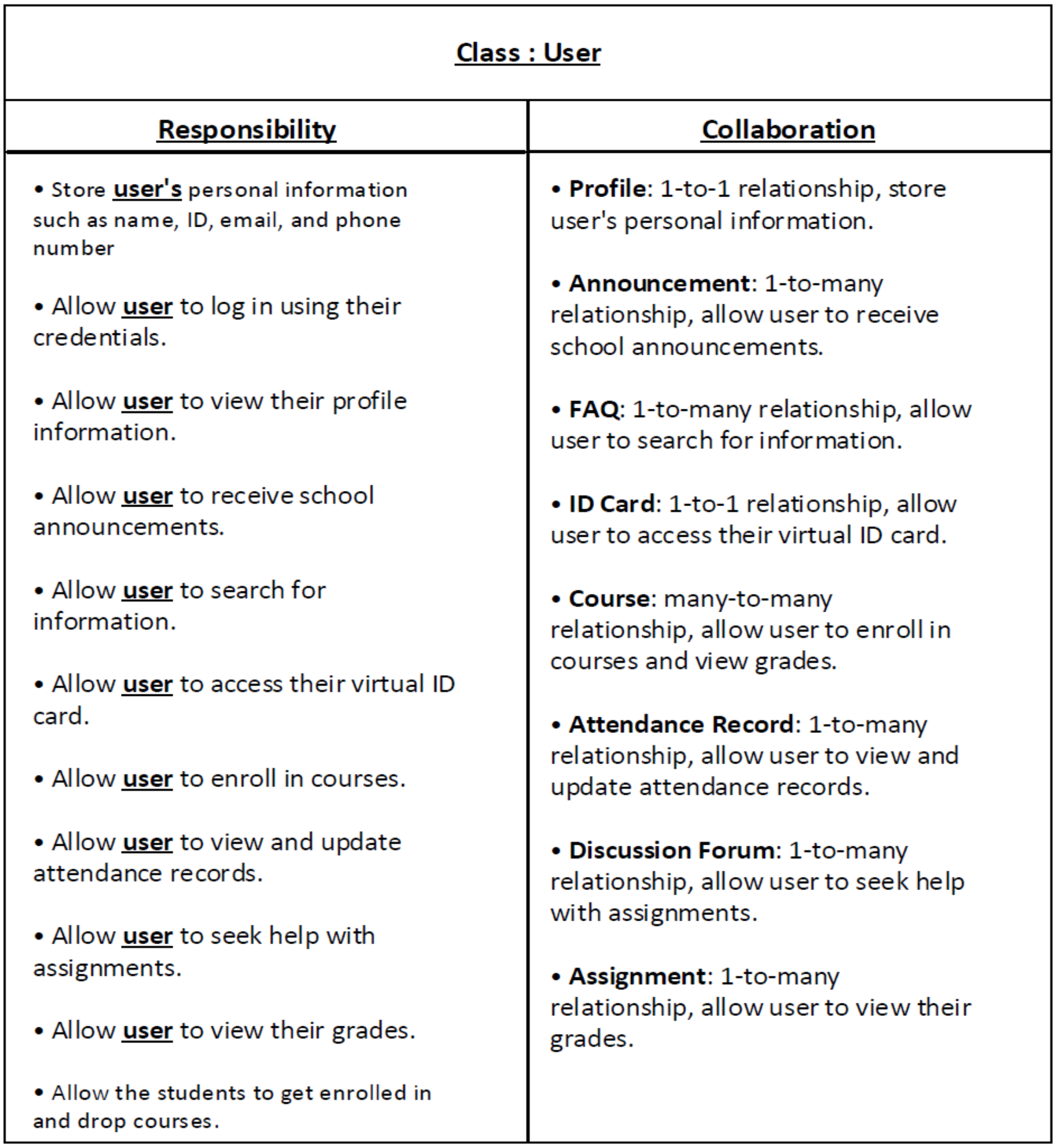
User seeks help by accessing the "seekHelp" method in the User class. The method allows the user to request assistance with a particular activity in the system. The user may enter a topic and description of their issue, which will be directed to the appropriate support staff or discussion forum. They can then receive assistance from the support staff or other users in the system who have expertise in the particular topic. The user can also access the discussion forum to browse for similar issues or seek solutions to their problem.

## Appendix E

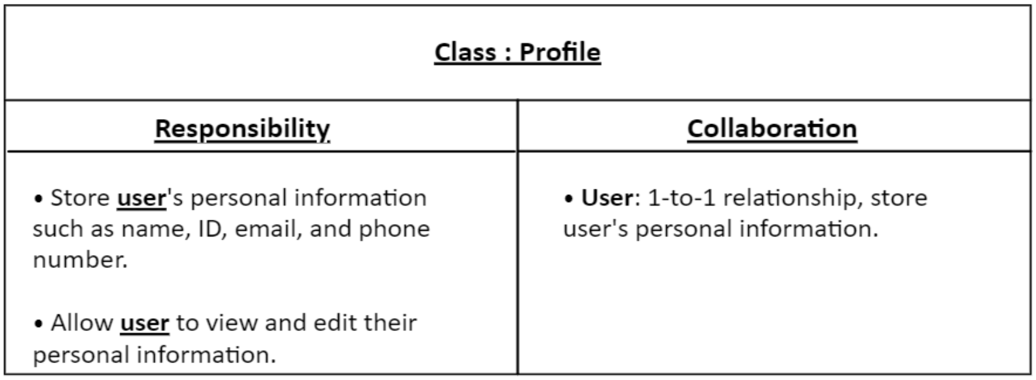
## Domain class diagram (deliverable #4)



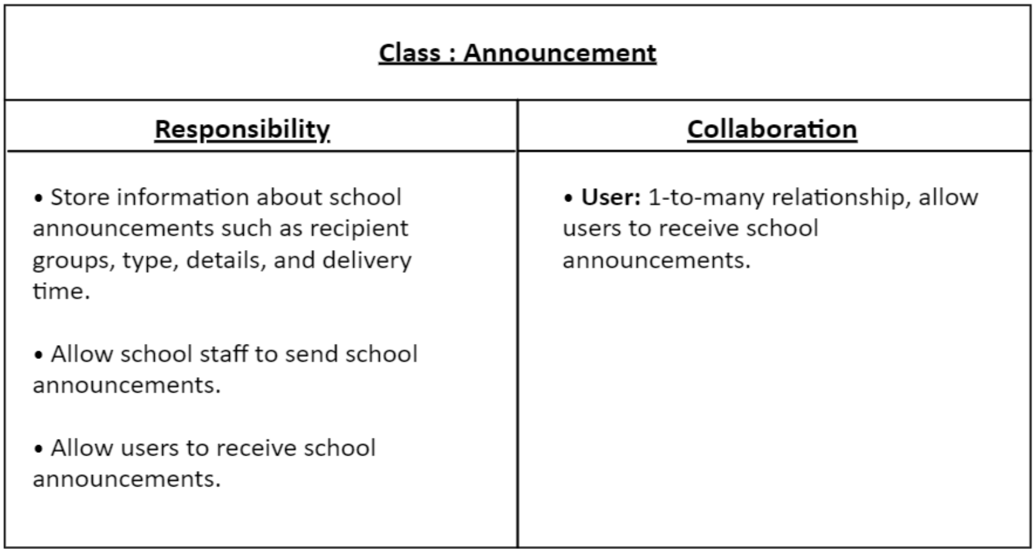
## CRC index cards (deliverable #5)



| Class : User | |
|---|---|
| **Responsibility** | **Collaboration** |
| • Store **user's** personal information such as name, ID, email, and phone number<br>• Allow **user** to log in using their credentials.<br>• Allow **user** to view their profile information.<br>• Allow **user** to receive school announcements.<br>• Allow **user** to search for information.<br>• Allow **user** to access their virtual ID card.<br>• Allow **user** to enroll in courses.<br>• Allow **user** to view and update attendance records.<br>• Allow **user** to seek help with assignments.<br>• Allow **user** to view their grades.<br>• Allow the students to get enrolled in and drop courses. | • **Profile**: 1-to-1 relationship, store user's personal information.<br>• **Announcement**: 1-to-many relationship, allow user to receive school announcements.<br>• **FAQ**: 1-to-many relationship, allow user to search for information.<br>• **ID Card**: 1-to-1 relationship, allow user to access their virtual ID card.<br>• **Course**: many-to-many relationship, allow user to enroll in courses and view grades.<br>• **Attendance Record**: 1-to-many relationship, allow user to view and update attendance records.<br>• **Discussion Forum**: 1-to-many relationship, allow user to seek help with assignments.<br>• **Assignment**: 1-to-many relationship, allow user to view their grades. |



| **Class : Profile** | |
|---|---|
| **Responsibility** | **Collaboration** |
| • Store **user**'s personal information such as name, ID, email, and phone number.<br>• Allow **user** to view and edit their personal information. | • **User**: 1-to-1 relationship, store user's personal information. |



| **Class : Announcement** | |
|---|---|
| **Responsibility** | **Collaboration** |
| • Store information about school announcements such as recipient groups, type, details, and delivery time.<br>• Allow school staff to send school announcements.<br>• Allow users to receive school announcements. | • **User:** 1-to-many relationship, allow users to receive school announcements. |

| **Class : FAQ** | |
|---|---|
| **Responsibility** | **Collaboration** |
| • Store frequently asked questions and their answers.<br>• Allow administrators to manage the FAQ.<br>• Allow users to search for information. | • **Administrator**: 1-to-many relationship, allow administrators to manage the FAQ.<br>• **User**: 1-to-many relationship, allow users to search for information. |

| **Class : ID Card** | |
|---|---|
| **Responsibility** | **Collaboration** |
| • Store the user's virtual ID card information such as barcode.<br>• Allow users to access their virtual ID card. | • **User:** 1-to-1 relationship, store the user's virtual ID card information. |

**Class : Course**

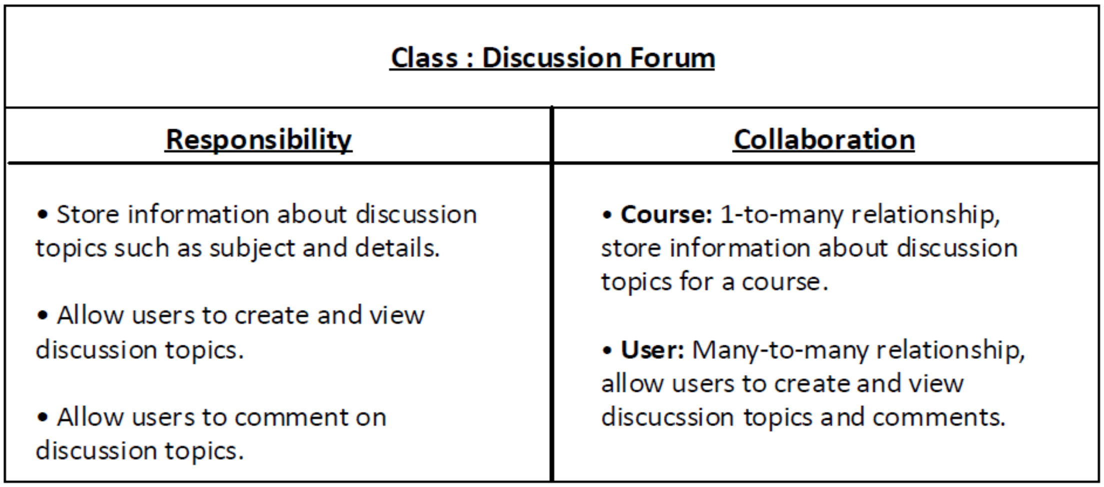
| **Responsibility** | **Collaboration** |
|---|---|
| • Store information about a course such as name, code, description, instructor, sections, content, and assignments.<br>• Allow users to enroll in a course.<br>• Allow users to view course information.<br>• Allow instructors to manage course information such as sections, content, and assignments. | • **User**: many-to-many relationship, allow users to enroll in a course and view course information.<br>• **Section**: 1-to-many relationship, store information about sections for a course.<br>• **Content**: 1-to-many relationship, store information about content for a course.<br>• **Assignment**: 1-to-many relationship, store information about assignments for a course. |

**Class : Section**

| **Responsibility** | **Collaboration** |
|---|---|
| • Store information about a section such as section number.<br>• Allow Instructors to manage section information for a course. | • **Course:** 1-to-many relationship, store information about sections for a course. |

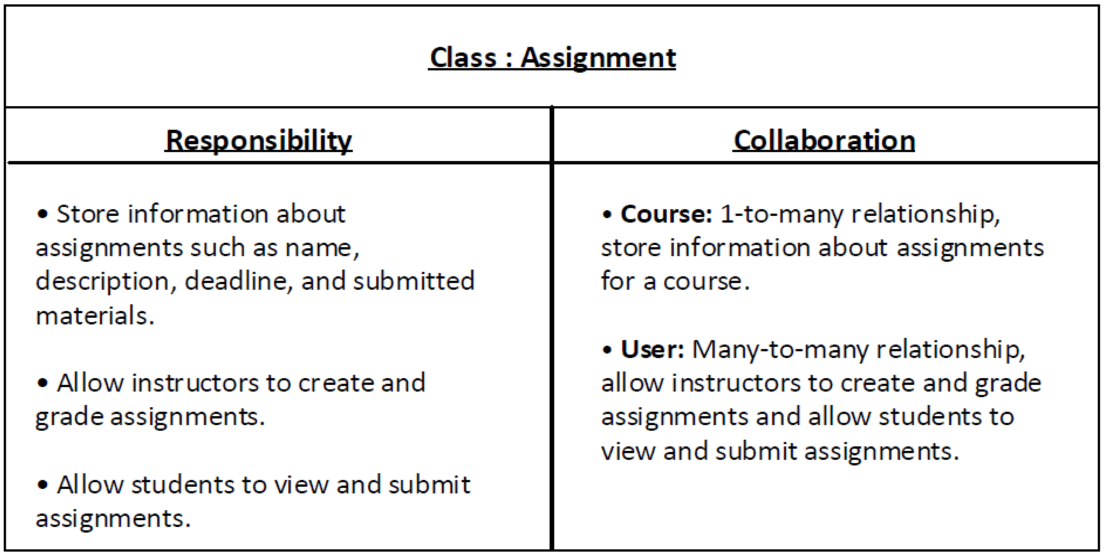
| **Class : Content** | |
|---|---|
| **Responsibility** | **Collaboration** |
| • Store information about a section such as recipent groups, details, and materials.<br>• Allow Instructors to manage course content information for a course. | • **Course:** 1-to-many relationship, store information about content for a course. |

| **Class : Attendance Record** | |
|---|---|
| **Responsibility** | **Collaboration** |
| • Store information about attendance records for a course such as date, status, and student ID.<br>• Allow Instructors and students to view and update attendance records. | • **Course:** 1-to-many relationship, store information about attendance records for a course.<br>• **User:** Many-to-many relationship, allow instructiors and stuents to view and update attendance records. |



**Class : Discussion Forum**

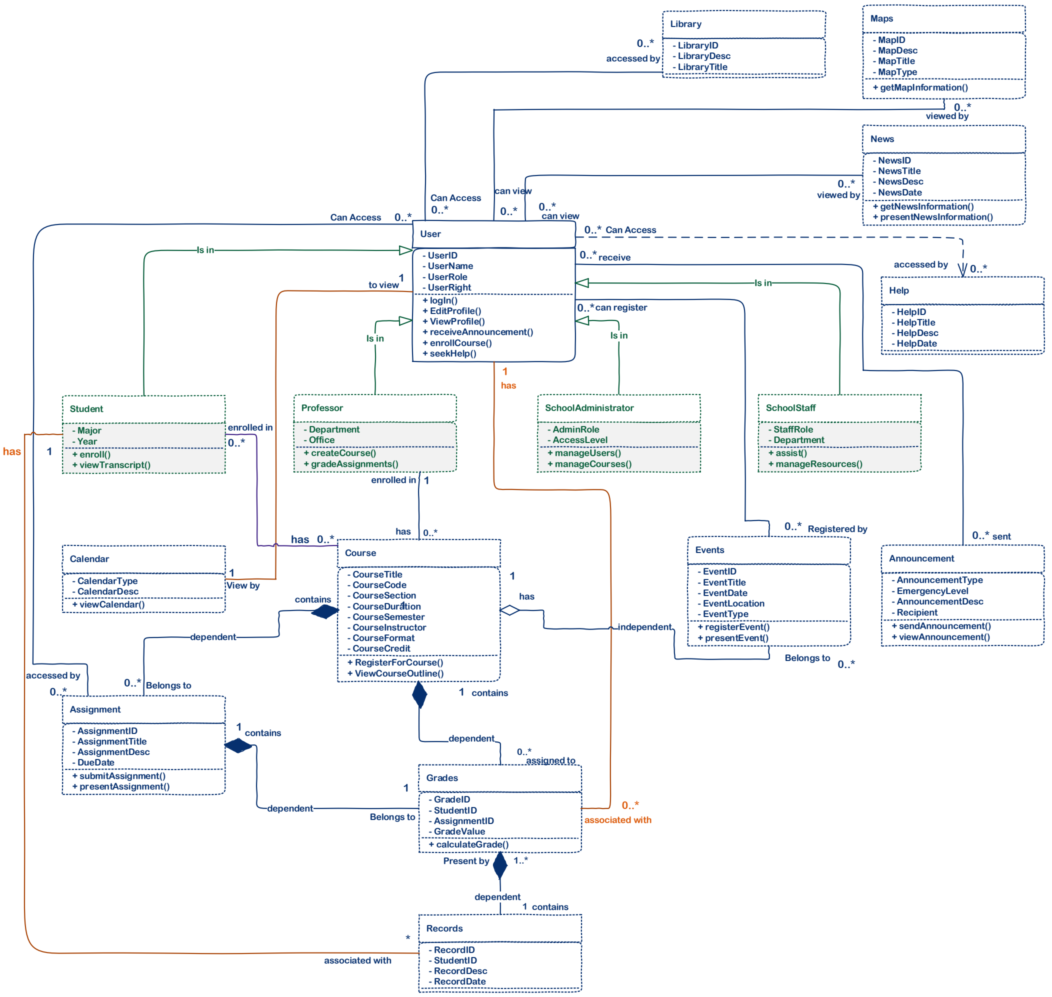
| Responsibility | Collaboration |
|---|---|
| • Store information about discussion topics such as subject and details.<br>• Allow users to create and view discussion topics.<br>• Allow users to comment on discussion topics. | • **Course:** 1-to-many relationship, store information about discussion topics for a course.<br>• **User:** Many-to-many relationship, allow users to create and view discucssion topics and comments. |



**Class : Assignment**

| Responsibility | Collaboration |
|---|---|
| • Store information about assignments such as name, description, deadline, and submitted materials.<br>• Allow instructors to create and grade assignments.<br>• Allow students to view and submit assignments. | • **Course:** 1-to-many relationship, store information about assignments for a course.<br>• **User:** Many-to-many relationship, allow instructors to create and grade assignments and allow students to view and submit assignments. |

## Class Diagram (deliverable #6)


Library
- LibraryID
- LibraryDesc
- LibraryTitle
0..*
accessed by
Maps
- MapID
- MapDesc
- MapTitle
- MapType
+ getMapInformation()
0..*
viewed by
News
- NewsID
- NewsTitle
- NewsDesc
- NewsDate
+ getNewsInformation()
+ presentNewsInformation()
0..*
viewed by
can view
0..*
can view
Can Access
0..*
0..*
Can Access
0..* Can Access
User
- UserID
- UserName
- UserRole
- UserRight
+ logIn()
+ EditProfile()
+ ViewProfile()
+ receiveAnnouncement()
+ enrollCourse()
+ seekHelp()
0..* receive
accessed by
0..*
Help
- HelpID
- HelpTitle
- HelpDesc
- HelpDate
Is in
1
to view
Is in
Is in
0..* can register
Is in
1
has
Student
- Major
- Year
+ enroll()
+ viewTranscript()
enrolled in
0..*
has
1
Professor
- Department
- Office
+ createCourse()
+ gradeAssignments()
enrolled in
1
SchoolAdministrator
- AdminRole
- AccessLevel
+ manageUsers()
+ manageCourses()
SchoolStaff
- StaffRole
- Department
+ assist()
+ manageResources()
Calendar
- CalendarType
- CalendarDesc
+ viewCalendar()
1
View by
has 0..*
has
0..*
Course
- CourseTitle
- CourseCode
- CourseSection
- CourseDuration
- CourseSemester
- CourseInstructor
- CourseFormat
- CourseCredit
+ RegisterForCourse()
+ ViewCourseOutline()
contains
1
has
independent
0..* Registered by
Events
- EventID
- EventTitle
- EventDate
- EventLocation
- EventType
+ registerEvent()
+ presentEvent()
Belongs to
0..*
0..* sent
Announcement
- AnnouncementType
- EmergencyLevel
- AnnouncementDesc
- Recipient
+ sendAnnouncement()
+ viewAnnouncement()
dependent
accessed by
0..*
0..*
Belongs to
Assignment
- AssignmentID
- AssignmentTitle
- AssignmentDesc
- DueDate
+ submitAssignment()
+ presentAssignment()
1 contains
1 contains
dependent
0..*
assigned to
Grades
- GradeID
- StudentID
- AssignmentID
- GradeValue
+ calculateGrade()
1
dependent
Belongs to
0..*
associated with
Present by
1..*
dependent
1 contains
Records
- RecordID
- StudentID
- RecordDesc
- RecordDate
*
associated with