|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Report Cheating | |
| **Scenario** | Proctor wants to report cheating case | |
| **Triggering Event** | The exam proctor wants to report a cheating case on the exam he/she is assigned to | |
| **Brief Description** | The Proctor see all ongoing exam, proctor can report cheating case for ongoing exam | |
| **Actors** | Proctor | |
| **Related Use Cases** |  | |
| **Stakeholders** | Student, Lecturer, Scoring Department | |
| **Precondition** | 1. There should be an ongoing exam schedule 2. The proctor was assigned for that particular exam | |
| **Postcondition** | 1. At the end of the use case, the database will be inserted with the new data | |
| **Flow of Events** | **Actor** | **System** |
| 1. Click exam page | 1.1 Fetch exam page |
|  | 1. Click Report Cheating Button on exam page | * 1. Fetch report cheating page   2. Enable report button for students that have not get any cheating reports |
|  | 1. Click Report Cheating for a student | * 1. Pop up confirmation dialog |
|  | 1. Click [‘Yes’/’No’] | * 1. Insert to database if user click yes   2. Show success message |
| **Exception Condition** | If the whole student in the class has been reported cheating | |

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Update Forum | |
| **Scenario** | Update an existing forum | |
| **Triggering Event** | Lecturer or student wants to update the forum they made | |
| **Brief Description** | Lecturer or student made the forum before and willing to update the forum, where’s the content and title can be modified. | |
| **Actors** | 1. Lecturer 2. Student | |
| **Related Use Cases** | It can be triggered via View Forum use case | |
| **Stakeholders** | 1. Lecturer 2. Student | |
| **Precondition** | 1. There should be a class first 2. There forum should be created first | |
| **Postcondition** | 1. The updated forum values should be a valid value 2. At the end of the use case, the forum inside the database will be updated with the newest updated values | |
| **Flow of Events** | **Actor** | **System** |
| 1. The lecturer or student clicks one of forum | 1.1 System fetch the forum thread data |
| 1. Lecturer/Student click update button on forum thread | * 1. System fills in current data to forum   2. Fetch the new form fields |
| 1. Input new forum content |  |
| 1. Click Update Button | * 1. Validate content values   2. Show confirmation dialog |
| 1. Click [‘Yes’/’No’] | * 1. Insert to database if user click yes   2. Show success message |
| **Exception Condition** | Empty form values | |

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Create Class | |
| **Scenario** | Creating a new class | |
| **Triggering Event** | Administrative Department wants to create a new class | |
| **Brief Description** | Administrative Department wants to create a new class to allocate new students and lecturer at a certain course | |
| **Actors** | 1. Administrative Department | |
| **Related Use Cases** | View All Class | |
| **Stakeholders** | 1. Student 2. Lecturer 3. Administrative Department | |
| **Precondition** | 1. There should have any available schedule list 2. There should have a course first | |
| **Postcondition** | 1. Form values should not be empty 2. Form values should be only filled with a valid value 3. Assign student and lecturer for the newly created class | |
| **Flow of Events** | **Actor** | **System** |
| 1. Administrative Department click create class button | * + 1. Fetches create class form to page     2. Fetch All lecturer and student list     3. Fetch all available courses list |
|  | 1. Input forum fields |  |
|  | 1. Click Create Button | * + 1. Validate content values     2. Show confirmation dialog |
|  | 1. Click [‘Yes’/’No’] | * + 1. Insert to database if user click yes     2. Show success message |
| **Exception Condition** |  | |

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Input Score | |
| **Scenario** | Input score for given task (assignment / exam) | |
| **Triggering Event** | Lecturer wants to input score for his/her student score | |
| **Brief Description** | Lecturer wants to give the students score for what they have worked for | |
| **Actors** | 1. Lecturer | |
| **Related Use Cases** | It can be trigger via View Class Detail use case | |
| **Stakeholders** | 1. Student 2. Lecturer 3. Scoring Department | |
| **Precondition** | 1. Assignment or exam should be created beforehand | |
| **Postcondition** | 1. If student have cheat record, his/her score will be 0 automatically 2. If the student has no answer, then his/her score will also be 0 by default | |
| **Flow of Events** | **Actor** | **System** |
| 1. The lecturer click input score button for certain task | * 1. System will fetch input score page   2. System will fetch all score fields      1. If task is assignment, system will fetch student answers and automatically fill 0 for score fields if exam is empty         1. If task is exam, system will fill 0 for students with cheat records |
|  | 1. Lecturer input students score |  |
|  | 1. Lecturer click save button | * 1. Validate score values   2. Insert the score data to database   3. Show success message |
| **Exception Condition** | Lecturer input invalid values to the score fields | |

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Create Assignment | |
| **Scenario** | Create a new assignment | |
| **Triggering Event** | Lecturer wants to create a new assignment | |
| **Brief Description** | The lecturer chooses a course, and then pick any available class/classes in that certain course and creates the new assignment by entering the content and the submission date. | |
| **Actors** | Lecturer | |
| **Related Use Cases** | Can be access through View Assignment use case | |
| **Stakeholders** | Lecturer, Student | |
| **Precondition** | 1. The lecturer prepared the assignment content | |
| **Postcondition** | 1. The assignment should be created with valid values 2. The lecturer should only assign to classes that he is associated with 3. At the end of the use case, the database will be inserted with new assignment data | |
| **Flow of Events** | **Actor** | **System** |
| 1. The lecturer click create assignment button | * 1. System fetches all available courses associated with the lecturer |
| 1. The lecturer chooses a course | * 1. System fetches all classes associated with the course and the lecturer into the form |
| 1. The lecturer input the form values | * 1. System validates input   2. System inserts the form assignment data to database |
| **Exception Condition** | * 1. The content is empty   2. The submission date and time is not bigger than current date and time   3. The lecturer didn’t pick any class | |