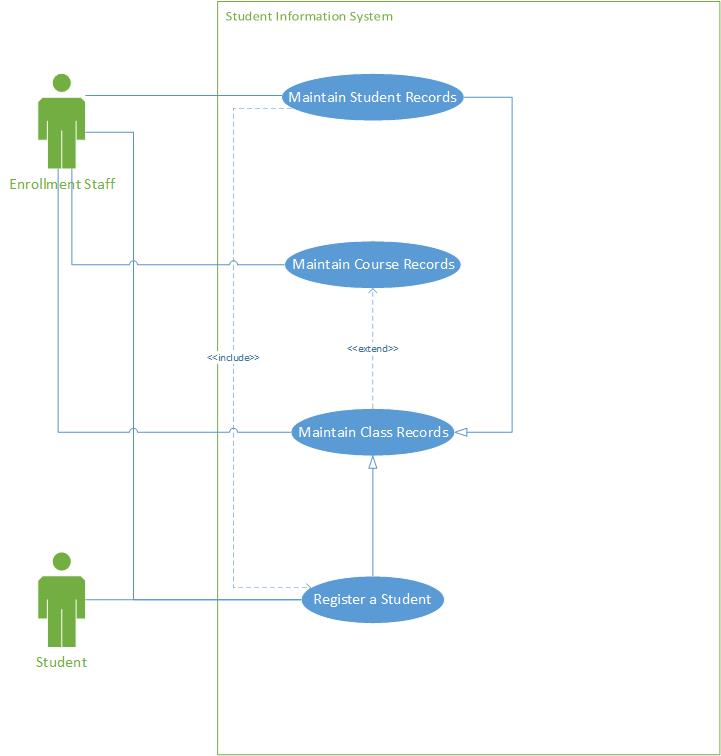
# IT 315 Final Project Part I Solution Submission Template

**Name: Dylan W. Sanderson**

**Date:** 1/17/2020

1. **Creation:**



|  |  |  |
| --- | --- | --- |
| Use Case Name:  Maintain Student Records | ID:  1 | Importance Level:  High |
| Primary Actor:  Enrollment Staff | Use Case Type:  Detail, Essential | |
| Stakeholders and Interests:  Enrollment Staff - Maintain full student records.  Students - Communicate and verify content of records with Enrollment Staff. | | |
| Brief Description:  This use case will describe how the SIS system will maintain and alter student records. | | |
| Trigger:  Enrollment staff member accesses the system to make an alteration to a new or existing student record.  Type:  External | | |
| Relationships   * Association:   Enrollment Staff   * Include:   Register a Student   * Extend: * Generalization:   Maintain Class Records | | |
| Normal Flow of Events:  1. The Enrollment Staff contacts the Student, via phone, email, or pre-filled form, to obtain information.  2. The Enrollment Staff chooses a subflow based on what alteration is necessary.  If the student is new, S-1 Create New Student subflow is performed.  If the student has had any information change, S-2 Update Existing Record subflow is performed.  If the student’s records need to be removed, S-3 Delete Record subflow is performed.  3. If necessary, the Enrollment Staff confirms the subflow is complete, and terminates the communication. | | |
| SubFlows:  S-1: Create New Student  1. The Enrollment Staff verifies that the student is not already the SIS.  2. The Enrollment Staff obtains the Student’s full name and date of birth, and inputs them.  3. The Enrollment Staff informs the Student of their system-created Student ID number.  4. The Enrollment Staff sets the Department as desired, or leaves it as undeclared if unknown.  5. The GPA data field remains null until courses are completed.  S-2: Update Existing Record  1. The Enrollment Staff obtains what information should be changed.  2. The Enrollment Staff updates only the necessary fields, changing the information to the new values.  S-3: Delete Record  1. The Enrollment Staff verifies the student record is slated for deletion.  2. The Enrollment Staff deletes the record. | | |
| Alternate/Exceptional Flows:  1a1: Record Already Exists  1. The Enrollment Staff informs the student their record already exists.  2. The Enrollment Staff verifies the current information. If any information has changed, execute S-2. | | |

|  |  |  |
| --- | --- | --- |
| Use Case Name:  Maintain Course Records | ID:  2 | Importance Level:  High |
| Primary Actor:  Enrollment Staff | Use Case Type:  Detail, Essential | |
| Stakeholders and Interests:  Enrollment Staff - Maintain Records of Courses.  Student - Be informed of what courses are available. | | |
| Brief Description:  This use case will describe how an enrollment staff member may alter the course records in the SIS. | | |
| Trigger:  Enrollment staff member accesses the system to make an alteration to a new or existing course record.  Type:  External | | |
| Relationships   * Association:   Enrollment Staff   * Include: * Extend: * Generalization: | | |
| Normal Flow of Events:  1. The Enrollment Staff receives information regarding a change in information for a course.  2. The Enrollment Staff chooses a subflow based on what alteration is necessary.  If a course is being added, S-1 Create Course subflow is performed.  If course information is being changed, S-2 Edit Course subflow is performed.  If a course is being deleted, S-3 Remove Course subflow is performed. | | |
| SubFlows:  S-1: Create Course  1. The Enrollment Staff verifies that the course is not already the SIS.  2. The Enrollment Staff obtains the course name, credit hours, description, and prerequisites, and inputs them.  3. The Enrollment Staff verifies the system creates a course ID number.  S-2: Edit Course  1. The Enrollment Staff obtains what information needs to be altered.  2. The Enrollment Staff updates only the necessary fields, changing the information to the new values.  S-3: Remove Course  1. The Enrollment Staff verifies the course is slated for deletion.  2. The Enrollment Staff deletes the course. | | |
| Alternate/Exceptional Flows:  1a1: Course Already Exists  1. If the course already exists in the SIS, S-2 is executed.  2. S-2 is used exclusively to verify the stored information is current, and correct that which is not. | | |

|  |  |  |
| --- | --- | --- |
| Use Case Name:  Maintain Class Records | ID:  3 | Importance Level:  High |
| Primary Actor:  Enrollment Staff | Use Case Type:  Detail, Essential | |
| Stakeholders and Interests:  Enrollment Staff - Maintain records of classes.  Students - Be informed of classes and their availability. | | |
| Brief Description:  This use case will describe how an enrollment staff member may alter the class records in the SIS. | | |
| Trigger:  Enrollment staff member accesses the system to make an alteration to a new or existing class record.  Type:  External | | |
| Relationships   * Association:   Enrollment Staff   * Include: * Extend:   Maintain Course Records   * Generalization: | | |
| Normal Flow of Events:  1. The Enrollment Staff receives information regarding a change in information for a class.  2. The Enrollment Staff chooses a subflow based on what alteration is necessary.  If a course is being added, S-1 Create Class subflow is performed.  If course information is being changed, S-2 Edit Class subflow is performed.  If a course is being deleted, S-3 Remove Class subflow is performed. | | |
| SubFlows:  S-1: Create Class  1. The Enrollment Staff verifies that the class is not already the SIS.  2. The Enrollment Staff obtains the course ID the class is tied to, as well as the start and end dates, and inputs them.  3. The Enrollment Staff inputs URL and browser, if the course is online.  4. The Enrollment Staff inputs class building and room, if the course is in-person.  S-2: Edit Class  1. The Enrollment Staff obtains what information needs to be altered.  2. The Enrollment Staff updates only the necessary fields, changing the information to the new values.  S-3: Remove Class  1. The Enrollment Staff verifies the class is slated for deletion.  2. The Enrollment Staff deletes the class. | | |
| Alternate/Exceptional Flows:  1a1: Class Already Exists  1. If the class already exists in the SIS, S-2 is executed.  2. S-2 is used exclusively to verify the stored information is current, and correct that which is not. | | |

|  |  |  |
| --- | --- | --- |
| Use Case Name:  Register a Student | ID:  4 | Importance Level:  High |
| Primary Actor:  Enrollment Staff, Student | Use Case Type:  Detail, Essential | |
| Stakeholders and Interests:  Enrollment Staff - Register a student to a specific class.  Student - Register themselves in specific classes. | | |
| Brief Description:  This use case will describe the interactions with the SIS that allow a student to become registered for one or more classes. | | |
| Trigger:  A Student needs to be registered for a class.  Type:  External | | |
| Relationships   * Association:   Enrollment Staff  Student   * Include: * Extend: * Generalization:   Maintain Class Records | | |
| Normal Flow of Events:  1. Either themselves or with the Enrollment Staff as a proxy, the Student’s information is entered into the SIS.  2. The desired course information is entered, and the system displays all future classes.  3. A class for a maximum of one semester forwards is selected.  4. The subflow S-1 Class Validation is performed.  5. The process returns to step 2 and repeats cyclically until it is voluntarily terminated. | | |
| SubFlows:  S-1: Class Verification:  1. It is verified that the desired registration does not already exist.  2. If the class is online, the student acknowledges that they have the hardware requirements to access it.  3. The student’s GPA is checked. If it is below 3.5, they are not allowed to register for more than three classes. The number of classes they are registered for is checked. | | |
| Alternate/Exceptional Flows:  1a1: Class Verification Failure  1. If any of these conditions are not met, the normal flow is returned to step 2. | | |

1. **Testing:** Verify and validate your use case diagram and use case descriptions against the SIS requirements definition.

In comparing to the requirements of the documents, the use case diagram paired with the use case description to effectively cover the needs of the system. Each use case covers all required functions, with subflows for specific actions within a particular portion of the system, and alternate/exceptional flows that catch the uncommon or rare occurences where certain criteria within the flows are not met. All together, the use cases cover the four main areas that are needed to have the system fulfill it’s intended purpose.

There are only two requirements that are not directly commented on within the use cases, as they are less direct interaction descriptions and more software engineering side. Those two requirements are that the system is secure and the system is both mobile and pc friendly. The cross-platform compatibility is something that would be handled on the software engineering side, and should not need a direct description within the use cases. The security (I.e. login credentials etc.) are implied by the only two actors being Student and Enrollment Staff and that what is accessible to at least one of the actors is limited.

1. **Approach Explanation:** Explain your approach to the problem, the decisions you made to arrive at your solution, and how you completed it.

I chose these four use cases as they are the four main areas of functionality. Any specific requirement within that area could be handled within subflows to keep the diagram simple and not overpopulated. The diagram itself benefits from concealing information and focusing on main points and their interrelationships. The use case descriptions themselves fill in the information gaps.

The associations are specific and important to take note of. First, it is important to note that all of the use cases are accessible by the Enrollment Staff, but only one is accessible by the Student. This fits the description provided, as students should not be altering courses, classes, or their own profile without validation. Most universities, however, allow students after their first semester to enroll in classes themselves, usually with a verification code from an enrollment staff member, which would coincide with the implied security features.

It is also important to note that their are no direct relationships between the use cases. Each process is entirely self-contained, with the only exception being that a class can reference a course ID to show what it is a part of. From a programming standpoint, for exception handling, it is likely the software engineer would allow that data point to default to null, and prevent database crashes from a lack of information in the category. Therefore, while one object may reference another, the objects themselves do not depend on each other to exist, making it not necessary for such a relationship within the use case diagram to exist.

Addendum: We have reflexively added some relationships to the Use Case Cards. We added a generalization relationship between Register a Student and Maintain Class Records, as the student will have to be registered for classes. This relationship was also added under Maintain Student Records. We added an extends relationship on Maintain Class Records referencing Maintain Course Records, as a class cannot exist without referencing a course. Lastly, we added to Maintain Student Records an includes relationship for Register a Student, as registering a student is part of maintaining the records of the student, and any functionalities for maintaining the student’s records would be accessible when first registering them.

1. **Self-Reflection:** Reflect on this experience and the lessons you learned from it.

Previously, I had not simplified a use-case diagram setup enough. This was partially due to my lack of understanding as to the specifics of use-case descriptions. I actually did my use-case descriptions first, to determine exactly how to lay out my use-case diagram. As such, I was able to maximize on the concept of concealing information, and fold multiple other items I previously would have placed as use-cases with an include relationship into subflows within the actual use case.

There were a few areas I struggled with in this particular assignment. One of these was the alternate/exceptional flows. Reflexively, it makes sense to include these in the use-case descriptions. However, I had not considered both conceptually and in descriptive implementation how to discuss what should happen if either a user or the product itself misbehaves. If I had left that idea to the wind, were this an actual project for a company, it would have increased production time, as the programmers would not necessarily have known to layer in protection for these cases. Best case, in an agile system, the testers would have caught it. Worst case, in a waterfall system, it wouldn’t have been caught until post-release, and would have required time, money, and patching to resolve.

The trigger descriptions also proved to be challenging, as previously I had not though of why a use case need exist, simply that it did. I also had not considered the source of the trigger being internal vs. external, which is of vital importance. Lastly, the importance level are always a difficult decision. In this case, since all four use cases are the four core functions, I decided to label all four as high, as all four would need to be implemented in the software before it could be considered alpha phase.