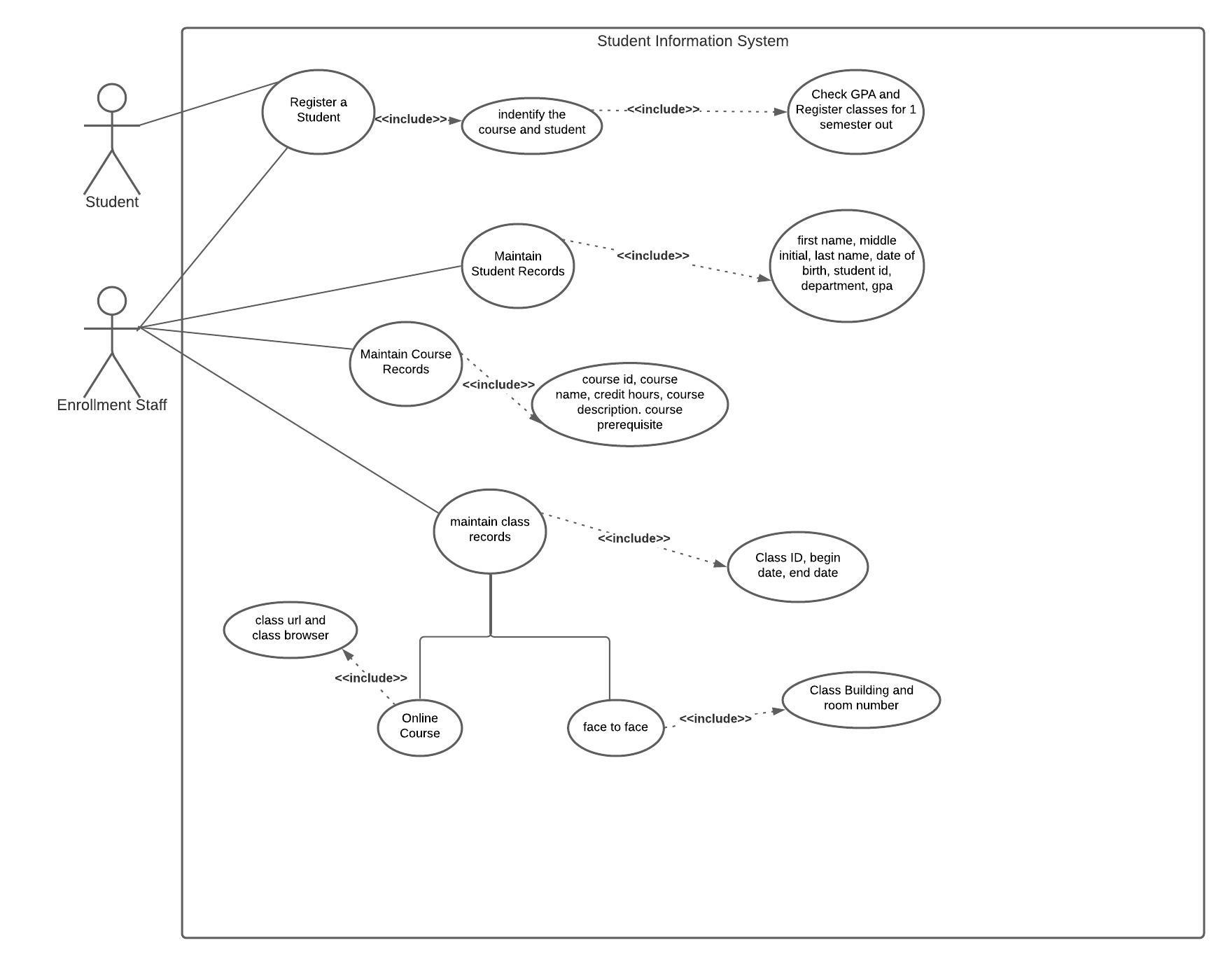
# IT 315 Final Project Part I Solution Submission Template

This template is a guide for you to organize your information. To complete it, **replace the bracketed text with the relevant information.** Some areas may be too large or too small for the information you’re inserting. Adjust the size of the areas as necessary.

**Name: Tyler Pickard**

**Date:** 09/21/2020

1. **Creation:** Generate your student information system (SIS) use case diagram. Refer to textbook pages 121–129.



|  |  |  |
| --- | --- | --- |
| Use Case Name:  Register a student | ID:  1 | Importance Level:  High |
| Primary Actor:  Enrollment staff and students | Use Case Type:  System | |
| Stakeholders and Interests:  [Insert information.] | | |
| Brief Description:  Students and enrollment staff should have the ability to register for classes. | | |
| Trigger:  [Insert information.]  Type:  [Insert information.] | | |
| Relationships   * Association:   Student and enrollment staff   * Include:   Identify the student, identity the course and show all classes for that course | | |
| Normal Flow of Events:  First the system will identify the student selected is correct. Next the course will be identified and display all classes after that start date. The system will have the ability to register for 1 semester out. Next the GPA will be checked because this determines how many courses a student can register for. Duplicate classes will be checked for. Online courses require an acknowledgement that you understand the software and hardware required. If any registration rules aren’t followed then an error message will be displayed. Finally, the system will display a confirmation of the classes selected. | | |
| SubFlows:  [Insert information.] | | |
| Alternate/Exceptional Flows:  [Insert information.] | | |

|  |  |  |
| --- | --- | --- |
| Use Case Name:  Maintain Student Records | ID:  2 | Importance Level:  High |
| Primary Actor:  Enrollment staff | Use Case Type:  Business | |
| Stakeholders and Interests:  [Insert information.] | | |
| Brief Description:  Keep records of student’s: first name, middle initial, last name, date of birth, student id, department, and gpa. | | |
| Trigger:  [Insert information.]  Type:  [Insert information.] | | |
| Relationships   * Association:   Enrollment staff   * Include:   Stored records   * Extend:   Ability to add, modify, or delete records | | |
| Normal Flow of Events:  Enrollment staff can access student records. They can also update, add, or delete any records. | | |
| SubFlows:  [Insert information.] | | |
| Alternate/Exceptional Flows:  [Insert information.] | | |

|  |  |  |
| --- | --- | --- |
| Use Case Name:  Maintain course records | ID:  3 | Importance Level:  High |
| Primary Actor:  Enrollment staff | Use Case Type:  Business | |
| Stakeholders and Interests:  [Insert information.] | | |
| Brief Description:  Maintain the course information: course id, course name, credit hours, description, and prerequisite classes. | | |
| Trigger:  [Insert information.]  Type:  [Insert information.] | | |
| Relationships   * Association:   Enrollment staff   * Include:   Maintain the course information: course id, course name, credit hours, description, and prerequisite classes.   * Extend:   Ability to add, modify, or delete course information. | | |
| Normal Flow of Events:  Enrollment staff can access course information. They can also update, add, or delete any information. | | |
| SubFlows:  [Insert information.] | | |
| Alternate/Exceptional Flows:  [Insert information.] | | |

|  |  |  |
| --- | --- | --- |
| Use Case Name:  Maintain class records | ID:  4 | Importance Level:  High |
| Primary Actor:  Enrollment staff | Use Case Type:  Business | |
| Stakeholders and Interests:  [Insert information.] | | |
| Brief Description:  Ability to maintain class records and information. There are also two inherited or child cases for online classes and face to face classes. | | |
| Trigger:  [Insert information.]  Type:  [Insert information.] | | |
| Relationships   * Association:   Enrollment staff   * Include:   Class id, begin date, and end date   * Generalization:   Online courses, face to face courses | | |
| Normal Flow of Events:  Class id, begin date, and end date will apply to both online and face to face classes. Then you will move down to the second level and chose the type of class. Online courses will include the class url and browser. Face to face will include building name and room number. | | |
| SubFlows:  [Insert information.] | | |
| Alternate/Exceptional Flows:  [Insert information.] | | |

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

Check your diagram against the SIS requirements and write this review. In doing verification, the objective is to make sure that you are building software according to user specifications. Ask questions like these: Does each use case have the required functionality? Do all the use cases combined perform as a complete coherent system?

After reviewing the use-case diagram, I have determined that all the functions that were requested are displayed and upon completion of the project, the system should function as requested.

I have simplified my use-case diagram from the original for a cleaner and more readable diagram. The basis of the main use-cases are covered without overloading the diagram with details of the individual use-cases.

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

Explain why you chose these particular use cases and why you chose the relationships between them. Explain your approach to creating your functional model and the design decisions you made to create it.

I wanted to break the overall problem into a few main categories in the form of use-cases. After reviewing the document, I came up with four use-cases to break down: register a student, maintaining student records, maintaining course records, and maintaining class records. The program can be built for these four use-cases.

First, we have the use-case of register a student. This use-case will need to be accessed by both the student and enrollment staff since both have the ability to register classes. This is the most complex of the use-cases with many additional steps than the others. First the student must be identified and then the course can be identified for that student. Once a course is chosen, all the classes will be displayed for all dates after the current one. The system has the ability to register courses for 1 semester out. The GPA must be checked next because the student’s GPA will determine how many course the system allows them to register for. After the classes are selected, the system will check for duplicates. If any errors in the registration process occur then an error message will be displayed to the user. Online courses have an extra step that will require the user to acknowledge the software and hardware requirements of the course. Finally, the system will display a confirmation of the classes selected. This use-case will be using the previous use cases to actually enrollment students in the courses. I see this as the most important and complex of the use-cases. For a clean use-case diagram, not all of these details are covered in this particular diagram. A separate use-case diagram could be made for registering students that would involve all the additional details of the class.

Next is the maintain student records. The enrollment staff will be using this use-case to update, add, or delete the student records. Students will not access this use-case.

Maintaining courses is very similar to the setup of maintaining student records. Enrollment staff will have the ability to modify, add, or delete course records.

Maintaining class records use-case is set up in a generalization relationship meaning that the main use-case will be the parent and then there are two child cases that inherit the information of the parent but also add some unique qualities of their own. The two children cases are the types of classes: online and face to face. Online classes will have the unique feature of a class url and browser. Face to face will need to show the building name and room number.

When all the use-cases are put together the system will operate as requested. This breakdown will allow for smaller problems to be worked on and then put together to create the software. As previously mentioned, the use-case diagram does not show all the details of the main use-cases because the diagram would become difficult to read. For a workable use-case I have simplified the details for each use-case in this particular diagram. Separate use-case diagrams could be made for each additional use-case to break them down further.

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

These are your reflections on what you learned. Address what you found challenging and what you found easy. Discuss your experience creating your functional model and the lessons you learned from it. Specifically, draw connections between your experience and the object-oriented techniques and methods discussed in this course.

The most challenging part of this assignment for me was seeing how complicated the use-case diagram can get as it comes together. Past programming courses have taught me how to breakdown large projects into smaller problems and then placing them all together to create the final product. With this past experience, I did not find forming the use-cases to be as challenging as I would have if not for the experience. This is a great way to approach large problems to keep them from becoming too overwhelming and can give some direction on where to start. This assignment also showed me how a large project can be worked on by several team members at once. If there were five programmers on a team, each could program one use-case and then put the classes together at the end. I also see how creating this diagram can act as a roadmap to visually show team members what needs to be completed for a use-case.

Something I tried to focus on was keeping my use-case diagram simple. My original diagram had close to 20 ovals describing different aspects of each use-case. I believe the original diagram covered everything that was required but was ultimately difficult to read and not visually appealing. Here I have made a much cleaner, less detailed diagram based on the feedback I received. At time I find myself adding more detail or description than a project requires, so this has been a challenge. However, I am happy with the final product I am submitting.