CSCE 247 - Project 1 - Draft Requirements

Due Date: Sunday, February 10th, 11:59 PM (As PDF documents, via the course webpage)

Problem Statement

As unimaginable as it may be to some, most students eventually graduate. However, before a student can graduate, they must fulfill a set of rules. Currently, checking these rules is a laborious manual task where a Graduate Program Coordinator (GPC) must take a student's transcript and a paper-based degree program form and verify that all requirements have been met. The University of South Carolina Computer Science & Engineering Department has commissioned a software-based solution where, using data populated from student records, both individual students and Graduate Program Coordinators can check progress, amend data, and perform other necessary functions.

Overall Project Description

Your mission, should you choose not to fail the assignment(s), is to create the **processing back-end** for the graduation rule auditing project. This backend will receive as input various user requests from a variety of other systems, including front-end user interfaces (you are **not** implementing a user interface), can load student records (a set of key-value pairs) from a student database, and will perform three primary functions (as well as other functions that you will elicit from the customer):

1. Check the student records against the set of graduation requirements for their degree (Computer Science, Computer Engineering, and Computer Information Systems), and determine which requirements have been fulfilled and which remain.
2. Present a summary view to the GPC or student viewing the records indicating degree progress, completed milestones, and other relevant information.
3. Allow students to submit planned courses, after which, the system will inform the students of the impact of those courses on graduation requirements.

Later in the semester, you will be given a description of the interface between external systems (that you will **not** build) and the back end processing system (what you **will** build), the database abstractions (data that can be loaded into your system), and a description of the inputs to the system. We may make use of these interfaces when we test and grade the system, so they cannot be changed.

Graduation Rule Assessment Data System (GRADS)

Here you will get a set of informal “user requests” of what this system shall do. Note that this is purposely incomplete and vague and that it is your job to sort it all out (i.e., by eliciting requirements). Other features may emerge as part of the online elicitation session.

GRADS shall aid in the auditing of degree requirements for graduate students.

The system should support multiple user account types – including a student and a Graduate Program Coordinator. GRADS will verify the account type and provide functionality for each:

1. **For a student:**
    a. Present a summary report that describes their progress toward their degree (Bachelors of Science in Computer Science, Computer Engineering, or Computer Information Systems) as well as the courses that will be used to satisfy the degree requirements. This includes general requirements, like the CarolinaCore, and degree-specific requirements (for example, CIS majors must complete a Business Information Management minor).
    b. Display a student's current GPA.
    c. Display other relevant information (including their advisor, semester that the student began their program, etc.).
    d. Allow a student to enter the courses they are thinking about registering for and see the impact of their selections on degree progress.
2. **For a Graduate Program Coordinator:**
    a. Display a list of all students in the program and their student ID numbers.
    b. Allow the GPC to view the summary for any student in the program (see the Student list above)
    c. Display the raw transcript for any student (including for each course, the course name, semester taken, and final grade).
    d. Allow the GPC to amend the records for any student (for example, to update grades or list new transfer courses).
    e. Allow the GPC to leave notes on a student’s transcript.

Your Task

Your task in this assignment, as outlined above is to develop use-cases to help you elicit the requirements, and then to develop a requirements document for GRADS. You may choose any organization of your document, but always keep in mind that is must be readable, changeable, and capture all the essential information we have discussed in class. The same applies to your requirements - you may choose a template, but your requirements must be detailed, consistent, complete, and testable. Remember, you will build this system later, and if you have a poor requirements document, you will run into problems. You will be graded on what the system ought to do (as opposed to what you think it should do).

A requirements document template will be made available on the web page for the class. In addition, a template for individual requirements will also be made available on this page. You may choose to use these templates as-is, adjust them to fit your needs, or create your own. There are also some examples of use-case diagrams and scenarios in the slides and a use case template will be available on the same page. Finally, several checklists will be made available that you can use to determine if your requirements are up to professional standards.

The Requirements Elicitation

On Monday, January 28, a digital elicitation session will be opened on the course website on Dropbox. This will be your opportunity to ask questions directly to the “customers” and formulate your requirements. Before this session, be sure to come up with questions to ask (and start coming up with requirements).

Program requirements are linked from here: https://cse.sc.edu/undergraduate

We may make certain abstractions and changes to reduce complexity, but you may base your initial requirements questions on these documents.

Helpful Hints

Do not invent many unneeded requirements. Focus on the core functionality of GRADS and do not add “things that would be nice to have”. “Gold plating” the requirements by adding all kinds of nice, but unneeded, functionality and checks will lead to an excessively large and complex document.

Focus on the features asked for in this document and in the elicitation session. You must write detailed requirement specifications that capture what the behavior of the final system should be, including defining functionality, specifying properties that the output of the system must follow to be considered correct, and posing non-functional requirements that govern how the system operates with regard to performance, security, etc.

Remember that your system may need to check the results of the graduation rules for each program, but the document should not contain requirements such as “Must pass CSCE 240”. That is something a student must do to complete a degree, but it is not something the software does. Requirements should encompass how - in general - the software checks for milestone completion (and what it does with the passed and remaining milestones).

Deliverables

You are required to turn in the requirements document, your use-cases, and a use-case diagram. You can put your use-cases and diagram as a chapter in your requirements document, or you can submit them as a separate document.

Note that the first deliverable will be graded with a light hand and is intended for you to get feedback so you can prepare suitable document for the second deliverable. That said, take this assignment seriously, ensure you have time to complete the assignment, and focus on demonstrating that you can write good requirements.