CSCI 4440 — Software Design and Documentation
Team Deliverable 4 (document version 1.0)
DUE December 1, 2020

Overview

- This fourth set of team deliverables is to be completed by your team and is due by 11:59PM EDT on the due date shown above
- One team member must define your team in Submitty in the Team Deliverable 4 gradeable, then other team members can be invited to join
- Once your team is formed in Submitty, only one of you need submit your final work; **and if submitted late, all group members must use a late day, so plan accordingly**

What to do

For this fourth set of team deliverables, your focus is on testing and implementation. As per usual, your submission must be a single well-organized PDF file.

Team logistics and evaluation

At this stage, your project team should be meeting regularly and communicating seamlessly. Team members should be filling their respective roles, and the individual success criteria identified in Team Deliverable 2 should continue to be reviewed and met.

Given the above, include the following details as the first part of your PDF:

- Your team name (e.g., "Summer Archers") and each team member's name and role
- Any changes to your team dynamics, individual roles, weekly team meetings schedule, how meetings are run, where to find meeting minutes and other artifacts, etc.

Project status reports

As with previous Team Deliverable assignments, write a project status report that summarizes the current state of your project (shortly before the deliverable due date). The project status report describes an up-to-date list of issues encountered (since Team Deliverable 3). And for each issue, include a brief summary, the person responsible for resolving the issue, and how the problem is being solved or was solved.

Remember that your project status report also must include a summary of how each team member contributed to the specific set of deliverables. This helps ensure that each team member is taking an active role in each aspect of the project.

System test plan

Document in detail at least 32 test cases by describing the information below *for each test*. Use a spreadsheet similar to what we did in our November 9 lecture.

Note that these are **not** unit tests; instead, these tests focus on functionality, behavior, performance requirements, ethical considerations, etc. Therefore, thoroughly review your requirements to find "good" test cases.

For each test case, include the following:

1. Unique test case ID (e.g., `TEST-1234`)
2. Associated requirement IDs (e.g., `REQ-23`, `REQ-25`, etc.) that this test case actually tests
3. Associated use case IDs (e.g., `UC-11`, etc.) that this test case actually tests
4. Initial setup or state of the system *before* the test is run
5. Steps of the test, i.e., how to actually run the test
6. Expected outputs
7. Expected side effects, if any

In addition to the above, please indicate any necessary testing materials, e.g., a series of documents or files that you use for your tests. For this deliverable, you can simply describe these additional testing "accessories" (if any).

Implementation plan

Write a summary of your implementation plan by providing the following details:

1. A list of programming languages you will use; while you might use one language for your core functionality, you might also make use of other languages for peripheral tasks, e.g., data import/export, log file management, etc.
2. For each programming language, a description of *what* you will use this language for, as well as *why* you think this language is your best choice
3. For each programming language, your agreed upon coding standard, which should include both language-specific guidelines and language-independent guidelines (e.g., consistent naming and spacing conventions)
4. For each programming language, specific compilation/build requirements, if any (this ensures all team members are compiling and building the software component the same way)

Note that as you write code, you should expand and/or revise items 3 and 4 above. In other words, do not consider those to be static documents.

Unit tests

Unit testing is typically considered to be part of your implementation. For at least **three** of your methods or classes, provide a comprehensive set of unit tests.

Be sure to clearly describe your unit tests such that someone unfamiliar with your project could understand what representative test cases you are testing (e.g., from our November 9 lecture, one representative test would test valid inputs for a valid scalane obtuse triangle).

Show this via code by making sure your comments are completely clear; in other words, do not just include your test code and expect us to decipher it.

What to submit

Package all of your work into a single PDF file, including your unit test code for three methods or classes. Be concise and succinct in your answers. As noted in previous assignments, we are not looking for a minimum page count or word count, so avoid "fluff" and document bloat! **Only include what is important.**

For all work in this course, make sure you use a uniform and consistent style and format.

Also, be sure to proofread your work as a team before submitting. We will assign grades (and provide feedback) based on the above details and the quality of your writing.

Though not required for this deliverable, feel free to make use of the Center for Global Communication+Design (COMM+D) at `https://info.rpi.edu/comm-d`.