Reflection Report on Software Engineering

Team 14, Reach
Aamina Hussain
David Moroniti
Anika Peer
Deep Raj
Alan Scott

[Reflection is an important component of getting the full benefits from a learning experience. Besides the intrinsic benefits of reflection, this document will be used to help the TAs grade how well your team responded to feedback. In addition, several CEAB (Canadian Engineering Accreditation Board) Learning Outcomes (LOs) will be assessed based on your reflections. —TPLT]

1 Changes in Response to Feedback

[Summarize the changes made over the course of the project in response to feedback from TAs, the instructor, teammates, other teams, the project supervisor (if present), and from user testers. —TPLT]

[For those teams with an external supervisor, please highlight how the feedback from the supervisor shaped your project. In particular, you should highlight the supervisor's response to your Rev 0 demonstration to them. —TPLT]

[Version control can make the summary relatively easy, if you used issues and meaningful commits. If you feedback is in an issue, and you responded in the issue tracker, you can point to the issue as part of explaining your changes. If addressing the issue required changes to code or documentation, you can point to the specific commit that made the changes. —TPLT]

1.1 SRS and Hazard Analysis

There were several changes made to the SRS and hazard analysis as a result of feedback from Dr. Smith, our TAs, peers, and supervisors.

SRS:

- Peer feedback
  - Update the individual product use cases to be fully black-box use cases. We did this by only focusing on what the use case is, and

not describing how it could be done. This ensures the design is not constrained in the SRS (see Issue 12, this commit and this commit for more details).
  - Improve NFR-1 to make it less ambiguous (see Issue 14 and this commit for more details).
  - Improve NFR-12 to make it less ambiguous (see Issue 15 and this commit for more details).
  - Added user characteristics to make the target users more clear (see Issue 13 and this commit for more details).

- Dr. Smith and supervisors - After the revision 0 demo, both parties had very useful feedback, which we implemented in many places throughout the app and documentation.
  - Add a map to be able to see exactly where the trial/study is being held (see Issue 138 and this commit for more details). This was recommended by our supervisors.
  - Add a disclaimer/title to the profile creation page to make the purpose of the page more clear (see Issue 155 for more details). This was recommended by both Dr. Smith and our supervisors.
  - Add the ability to filter the studies/trials by distance (see Issue 178 and this commit for more details). This was recommended by both Dr. Smith and our supervisors.

- Other
  - Add FR-4, FR-5, FR-11 to the waiting room as they were not implemented.

Some other useful issues to give more insight into the changes we have made to the SRS documentation are provided below:

- Issue 251

Hazard Analysis:

- Peer feedback/TA feedback
  - Added some critical assumptions. This addresses feedback provided to us by our peers, and also feedback from the TA via the graded rubric (see Issue 35 and this commit for more details).
  - Added a definition of failure. This also addresses feedback provided to us by our peers and the TA (see Issue 36 and this commit for more details).

- Add failure mode for email generation. This also addressed feedback provided to us by our peers and the TA (see Issue 37 and this commit for more details).
- Make SR-6 less ambiguous by improving the wording of the requirement. This addresses feedback provided to use by our peers (see Issue 39 and this commit for more details).
- Removed the notification subsystem as it is currently out of scope. This addresses feedback provided to us by our peers (see Issue 38 and this commit for more details).
- Added traceability to FRs in the FMEA table (see Issue 40 and this commit for more details).

Some other useful issues to give more insight into the changes we have made to the Hazard Analysis documentation are provided below:

- Issue 250

1.2 Design and Design Documentation

There was very little feedback given for this particular section of the project, we generally performed well in this area. However, the changes made will be denoted below:

MG:

- Peer/TA Feedback.
  - Added all Functional and Non-Functional requirements to the traceability matrix in response to TA feedback.
  - Corrected some minor sentence structure issues in the document in response to TA feedback.

MIS:

- There was no feedback given for this deliverable, grammar was checked and minor flow changes were made.

System Design:

- There was no feedback given for this deliverable, grammar was checked and minor changes were made.

VnV Plan:

- Peer/TA Feedback.

- Added missing abbreviations to the symbols section in response to peer review (see Issue 45 and this commit for more details).
- Added addition characteristics for the typical user tester in response to peer review (see Issue 46 and this commit for more details).
- Added reference to the grading rubric in response to peer review (see Issue 47 and this commit for more details).
- Added clarification on what a greater than 7 score would mean in terms of testing in response to peer review (see Issue 48 and this commit for more details).
- Added an open ended additional concerns question to the usability survey in response to peer feedback (see Issue 49 and this commit for more details).
- Fixed some formatting issues including removing some whitespace from the document in response to TA feedback through the marking rubric (see Issue 255 and this commit for more details).
- Improved clarity of NFR and FR tests in response to TA feedback through the marking rubric (see Issue 256 and this commit for more details).
- Adding missing reference to hazard analysis in the relevant documents section in response to TA feedback through the marking rubric (see Issue 257 and this commit for more details).
- Added more specific plan for nondynamic testing in response to TA feedback through marking rubric (see Issue 258 and this commit for more details).

VnV Report:

- TA Feedback (No peer review was received for this deliverable).
  - Expanded on the changes made due to testing, in response to received feedback on the marking rubric (see Issue 254 and this commit for more details).3
  - Added a screenshot of the failed linting in the maintainability section to provide convincing evidence of the test failing, in response to received feedback on the marking rubric (see Issue 253 and this commit for more details).

2 Design Iteration (LO11)

[Explain how you arrived at your final design and implementation. How did the design evolve from the first version to the final version? —TPLT]

We approached the design, development, and implementation of this project using an iterative approach. We started off by coming up with the project requirements, and then splitting them into two groups: requirements that we must have and requirements that we would like to have time-permitting. Taking a look at the must-have requirements, we realized that our entire project relied on whether or not we could actually use the clinicaltrials.gov API. Since this was a huge priority, we decided for the proof of concept demo we would prove we could successfully utilize the external API.

Once we were able to verify the API worked, we officially started the development phase of the project. We worked on implementing the must-have requirements to the best of our abilities. We did this by creating tasks (using issues in GitHub) that we needed to get done in order to fully implement each requirement. For each individual task, we made the necessary code changes, did some minor testing locally, made further based on the results of the tests, merged the changes, and closed the issue for the task. In this case, we were iteratively developing on a smaller scale (on a task-by-task basis).

Once we completed implementing all these smaller tasks in order to fully implement the must-have requirements, we deployed the app and conducted some more thorough and integrated testing. This testing also included conducting a usability test with multiple users; one of the users was our supervisor. Throughout testing, we made note of things that did not work, needed improvement, or was redundant. Taking these notes, along with user feedback we received from the usability survey and our supervisors, we came up with some more tasks that we needed to complete in order to reflect our notes and feedback. This was a large-scale development iteration (of the overall integrated application).

We then repeated the process once more by making necessary changes for each task, deploying the app, and once again fully testing the overall application. Gathering our last round of feedback, we made the necessary changes and deployed the app once more before our final demo. This was where we stopped for now, but in order to keep improving our app and dealing with any bugs that we may discover or suggestions from our supervisors, we would continuously repeat this iterative process.

3 Design Decisions (LO12)

[Reflect and justify your design decisions. How did limitations, assumptions, and constraints influence your decisions? —TPLT]

4 Economic Considerations (LO23)

[Is there a market for your product? What would be involved in marketing your product? What is your estimate of the cost to produce a version that you could sell? What would you charge for your product? How many units would you have to sell to make money? If your product isn't something that would be sold, like an open source project, how would you go about attracting users? How many potential users currently exist? —TPLT]

5 Reflection on Project Management (LO24)

[This question focuses on processes and tools used for project management. —TPLT]

5.1 How Does Your Project Management Compare to Your Development Plan

[Did you follow your Development plan, with respect to the team meeting plan, team communication plan, team member roles and workflow plan. Did you use the technology you planned on using? —TPLT]

With respect to the team meeting plan, we consistently met at least once a week, like stated in our development plan, to discuss our status, any issues, and what tasks we need to complete by the next meeting. We also met regularly with our supervisors to share our progress on the project and receive their input. However, we did not meet them on a biweekly basis as stated in the development plan; instead, we met with them whenever they were available, which was about every three weeks.

The team communication plan outlined in the development was followed exactly as stated. We used Git issues to keep track of any formal tasks needed to be completed. Teams was used to host virtual meetings, as well as update the team about minor issues or concerns. We also used Google Calendar to keep track of deliverable deadlines and team/supervisor meeting times.

We did not have clearly designated member roles like we stated in the development plan. However, most of the roles were still loosely followed to some degree. Since every team member was flexible and we all had multiple skills, we weren't limited to just the roles that were designated to us in the development plan. We all took on different roles based on the current time constraint and what tasks were available for us to work on.

We followed our workflow plan exactly as it was outlined in the development plan. First, we discussed relevant tasks that needed to be completed in the following week and created issues for them on GitHub. Next, we assigned tickets to the team based on each member's skills and availability. We used the GitHub project board to keep track of the status of the issue (whether the issue is currently being worked on, in testing, completed, or not yet started). We used separate feature branches to work on changes related to each issue. After testing, we created a pull request and merged once all automated tests have passed.

We used all the technology we planned on using in the development plan. The only exception is we ended up not using Jest as a front-end unit testing framework. Due to the time constraint, as well as constant changes to the overall design of the UI and the new inputs/requests from our supervisors, we decided implementing unit tests for the front-end would hinder development with very little payoff. Instead, we tested the front-end manually.

5.2 What Went Well?

[What went well for your project management in terms of processes and technology? —TPLT]

5.3 What Went Wrong?

[What went wrong in terms of processes and technology? —TPLT]

5.4 What Would you Do Differently Next Time?

[What will you do differently for your next project? —TPLT]

6 Reflection on Capstone

[This question focuses on what you learned during the course of the capstone project. —TPLT]

6.1 Which Courses Were Relevant

The following course provided beneficial experience towards the undertaking of our capstone project:

- SFWRENG 3XA3 Software Project Management - Provided experience in coding development, but more importantly in managing the planning and documentation of a project from start to finish.
- SFWRENG 3S03 Software Testing - Provided knowledge of software testing strategies, such as static testing, unit testing, and code reviews.
- SFWRENG 3A04 Software Design III - Provided project development experience, specifically in module design and documentation.
- SFWRENG 3DB3 Databases - Provided knowledge on the operation and interaction with databases.
- SFWRENG 2XB3 Software Engineering Practice and Principles - Provided experience in software design, as well as providing knowledge in measuring system metrics.

- SFWRENG 2AA4 Intro to Software Development - Provided our first experience towards software system development, including data structures, design principles, and coding practices.

6.2 Knowledge/Skills Outside of Courses

[What skills/knowledge did you need to acquire for your capstone project that was outside of the courses you took? —TPLT]