Reflection Report on REVITALIZE

Team 13, REVITALIZE
Bill Nguyen
Syed Bokhari
Hasan Kibria
Youssef Dahab
Logan Brown
Mahmoud Anklis

[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

This section, summarizes the changes made over the course of the project in response to feedback from TAs, the instructor, teammates, other teams, and from user testers. In the final documentation changes are clearly shown by new documentation being highlighted in red (Eg. new change) and deletion of documentation is shown like this: ~~deletion~~.

1.1 SRS and Hazard Analysis

SRS Changes: Many components of the SRS were changed and some new components were added based on feedback. Minimal changes were made to the project constraints regarding the monetary cost and a new assumption was added. The work partitioning table was completely updated with more descriptive event names and more accurate inputs and outputs. The use case diagram was updated to include two missing pieces which are the update food and update exercises bubbles. The Functional Requirements were changed completely based on the feedback that the requirements should not state what the system should display but rather what the system should do. An Abbreviation and Acronyms section was added to the beginning of the document to help guide the user with specific naming conventions. There were minor updates to the fit criterion of the Non Functional Requirements to include the incorporation of user surveys. A new Non Functional Requirement relating to privacy and security was created.

The traceability matrices were updated with the new Functional Requirements and they were moved to the appendix. A risk was added relating to the POC demo. Finally, a new section for Anticipated and Unlikely Changes was added as well.

Hazard Analysis changes: A list of figures was added to the Hazard Analysis. Additionally, the recommended actions for the Login component and the Recipe component in the FMEA were updated. Also, the recommended actions for the failure mode "Login database/server down" is updated to include the frequency of the checks and how to complete the checks (automation scripts). The error messages in each recommended action section were updated and specified instead of having a general statement stating that an error message would be displayed. Formatting was also fixed in the document. The numbering of pages that display the FMEA are now placed in the correct position.

1.2 Design and Design Documentation

1.3 VnV Plan and Report

VnV Plan Changes: To start, a lot of the system tests for the functional requirements were changed to match changes from SRS. Also from the feedback we were suggested the following "Please do not include too many implementation details but focus on the functionalities (e.g., what is your input and output) of the system. Try to think in this way: what is the input for this action button and what should the system do after I click this button? The most important thing you should care about is not what components you are using but what do you want to achieve", So we completely changed the test cases for the functional requirements to focus more on functionality, rather than implementation and also focused on the inputs and outputs to be more specific with real examples. Some quality of life changes were made to the document as well, such as adding hyperlinks to the usability survey questions used to verify the nonfunctional requirements. Since the functional requirements and tests were updated, the corresponding traceability matrices were updated as well.

Also in the VnV plan it was suggested team member's roles were a bit ambiguous, so made changes in order to make it more clear what each member was testing and specifically what sections of the code / application they were testing as well. Another suggestion was to add the unit test section since MIS is now complete, in which we did with appropriate tests and traceability matrix.

2 Design Iteration (LO11)

At the start of our course, our team excelled at gathering requirements, which allowed us to lay a strong foundation for our project. By clearly defining our functional and non-functional requirements in the SRS document at the beginning of the course, we were able to minimize major changes during the design

process. However, we also recognized the importance of continuous refinement and updates to ensure that we were meeting the needs and expectations of our stakeholders. Throughout the project, our team maintained regular communication, which facilitated the identification of issues early on and enabled us to make the necessary revisions and additions as the project progressed. During the initial presentation, our project achieved the core features of our design. As we continued to refine our product, we focused on implementing the backend and database, as well as enhancing the frontend user experience. Our team was receptive to feedback and suggestions from our stakeholders, and we actively sought out ways to improve our product. By adopting an iterative approach, we were able to continually evolve and enhance our product, which ultimately led to the creation of the final version of our product, REVITALIZE. This final version fully met the needs and expectations of our stakeholders and fulfilled the vision that we set out to achieve at the beginning of the course.

3 Design Decisions (LO12)

Since this application targets users from all age groups, simplicity was a key element in our design. Because of this, our assumptions for the project were small in number. Users are only expected to: know how to use a smart phone, can read English, have a stable Internet connection, and know how to to do basic addition, subtraction and multiplication. The application is designed to meet these assumptions and its simplistic look and flow allow all users to interact with it smoothly. No instructions are needed to begin using the application.

4 Economic Considerations (LO23)

I feel as though there is a market for our product. It would be an easily-learned and friendly app for people looking to flip their lifestyle regarding health-related habits. I think marketing something like this could be done by using fitness-oriented Internet personalities (ie. on YouTube, Instagram, etc.) to promote and speak about the product. As for the cost, I think that could be estimated to at least 20 000 dollars per month (considering all of us don't get paid and do this for free). The software infrastructure that must be walled behind an app like this; the servers, the database, would probably cost something like 20 000 per month. I think other than monthly software-infrastructure subscription payments, there wouldn't be too much of an overhead cost to productionalize the app. Perhaps some licensing, trademarking, etc. but other than that it would be more of a time-sink on the engineers' (our) behalf, not a money-sink. We wouldn't charge for our product; that would completely render us as a non-option against all the other competitor apps which already exist on the app store as a free to use service. Instead, we would probably try to generate ad revenue. I think a good amount of users could be attracted, given that we use the right marketing techniques (mentioned earlier) to do so. Probably 50 000+

potential users exist as of today, and if we do a good job promoting, I think we could have a good percentage of them using our application within a year or so. That would generate a good ad revenue to keep us going.

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

For the important parts, we stuck quite closely to the development plan. We used Javascript as the main language with the React Native and Express JS frameworks for the front end and back end respectively. A technology we did not use was the performance measuring tool Sentry due to time constraints and the team deciding that other aspects of development were a higher priority. For our meetings, we did not strictly follow the exact times described in the document but in most cases were still able to meet up to twice a week. The team member roles were a lot more fluid than what was outlined in the development plan, and members picked up roles that suited their existing domain knowledge as we started the actual implementation of the application. Team communication followed the development plan quite closely, and documentation of the tasks that were assigned in call to each member were documented in our discord group. Any concerns and questions were asked in the Facebook Messenger group chat and responded in a timely fashion.

5.2 What Went Well?

Our original plan of using React Native and Express JS frameworks was quite helpful for development, as these frameworks have extensive documentation and work well with each other. Because of this, it was a relatively smooth process to integrate the frontend with the backend, which may have been more difficult due to different members of the team working on separate parts. Our choices of communication were also excellent as everyone in the team was familiar with these technologies and everything we needed to recall from the meetings were documented in these chats. We could reach each other quickly through the Facebook Messenger chat and meetings and more difficult questions were mitigated using the voice channel in the discord group. Because we documented our tasks in the discord text chat, there was no confusion on what each person needed to do. Our team communication as well as thoughtful splitting of tasks helped greatly mitigate the risks outlined in the development plan.

5.3 What Went Wrong?

To start in terms of communication plan, plan was to meet every sunday and tuesday for 2 hours, but with a lot of us having 5 other courses and personal life outside of school, there were a few occasions where these dates could not be fulfilled and when trying to find an alternate date it was sometimes difficult to set a new time. Also for a lot of the assignments/tasks we tend to set soft deadlines and strict deadlines, but what we noticed was that usually majority of the team did not meet the soft deadlines and usually followed the strict deadlines. Finally a lot of the communication/project management was done via Discord, Facebook Messenger, Git Issues and Gantt Chart, these tools were effective but sometimes the organization of tasks were not the best, so maybe using a tool such as Jira, where they show tasks via an organized table could have been more effective.

5.4 What Would you Do Differently Next Time?

To start, for communication plan every meeting was sunday and tuesday for 2 hours but should have also considered alternate dates such monday for example if team members have conflicts, in order to make meetings more consistent if a conflict arises. Also, as mentioned in what went wrong, team members tended to ignore soft deadlines and tended to only follow strict deadlines maybe should have implemented a mechanism to ensure more soft deadlines were met, such as maybe trying to divide the work into even smaller chunks and have only strict deadlines. Another thing is would have used more tools that have better visuals and improve organization for tasks such as Jira, so it is easier to keep track of work done. Also in the team meeting we did take notes of each meeting, but we could have assigned one note taker per meeting instead of all of us doing it at the same time.