Reflection Report on Farming Matters

Team #14, The Farmers
Brandon Duong
Andrew Balmakund
Mihail Serafimovski
Mohammad Harun
Namit Chopra

This document is a reflection report for Farming Matters. The document summarizes the team's thoughts and reflections on different aspects of the system that was built over the course of 8 months. It will also help demonstrate the team's learning experience throughout the project. This document will go over what documents were changed in response to the feedback received over the past 8 months. These documents include Software Requirement Specification (SRS), Hazard Analysis (HA), Design Documents, and Verification and Validation (VnV) Plan and Report. This document will also discuss the team's various design iterations, design decisions, economic considerations, and reflection on project management.

1 Changes in Response to Feedback

Throughout the project, there were meetings almost every week with the supervisors. These meetings were there to constantly verify current requirements and elicit missing requirements while also validating the product. The changes that were made in response to the TAs can both be seen in the documentation and the final product. The feedback from other teams helped to polish up the documentation and also fill in the gaps.

1.1 SRS and Hazard Analysis

There were a quite few changes made to the SRS and the Hazard Analysis to improve upon the feedback that was provided by the TAs and the issues created by teams. The first change was adding two constraints which were previously stated as requirements. These constraints were including a consent form to the players and giving them the option to delete data. A functional requirement was added to harvest fully grown crops that were missing before. There was a distinction made between seeds and crops which was not present at the time of revision one. A legal requirement was added to mention that there would be no

copyright infringement. The TA mentioned that we should be careful not to use free assets that don't have an open license. Another performance requirement was added regarding the up-time of the system. The last new requirement that was added was a security requirement which mentions the method of how sensitive data will be stored. The main change made in the hazard analysis was defining the scope more clearly as before it was not sure if there was any physical setup required. Also, failure modes were improved to better deal with the integrity requirements. Lastly, rationales for the requirements and a severity column were added to the FMEA table to emphasize the significance of the hazards.

1.2 Design and Design Documentation

Various changes were made to the Design and Design Documentation to improve upon the feedback that was provided by the TAs and the issues created by other teams. There were mostly formatting changes to improve the syntax for the MIS. Two components completely needed to be reworked based on the client's feedback. The first was the Shop which was previously mentioned as the Market. Previously, the Shop allowed users to purchase insurance separately but after discussing with the professors we felt it was best to make insurance available to purchase when buying seeds. The function which dealt with purchasing insurance was removed and it was incorporated into the buy function. For both buying and selling another parameter had to be passed in which was the price to ensure the crop at. This was due to the client's demands to be able to sell crops as individual items with different contracts as opposed to just keeping a count of each. The second component that needed to be reworked was the inventory. Since items needed to be tracked individually rather than aggregated this meant that the data structure for an item had to change as well. The functions for the inventory had to take into account this new data structure so they were changed as well. The System Design was fine for the most part and minor changes were made to address the issues created by other teams. This included adding information regarding the firmware stack and including more testing information for the timeline. For the MG and MIS, we added two more modules which included GenerateStatistics and SeasonalEvents. These two modules were included to encapsulate the random events that may occur and the seasonal changes. GenerateStatistics also meant abstracting the consultant and avatars who will provide information to the players. There were some improvements made to the GameController to handle more state variables as the complexity of the game changed over time. The GameSettings module also had to be modified to differentiate withdrawing from the experiment and deleting user data. For the CreateAccount module, there was an addition that dealt with verifying human users using Captcha. This was necessary as we had the requirement of verifying the user is human.

1.3 VnV Plan and Report

Various changes were made to the VnV Plan and Report to improve upon the feedback provided by the TAs and the issues created by teams. The first change made was removing the professor's survey results from the validation plan. This was because it was not as insightful. After all, there was not enough feedback from just one person. We also removed Prettier from the verification plan as it was not used as an automated testing tool and was only used to format code. The test for API response time was removed for redundancy purposes as this is synonymous with the database response time which was tested. There were some small formatting errors like missing symbolic parameters and proper referencing for the traceability matrix. These issues were also resolved. The goals/objectives were written with more clarity and detail. We also added more clarity in terms of flexibility of the plan to address how responsibilities would be divided if people finished tasks earlier. We addressed how the surveys were used to verify and validate different elements of the system which were mainly UI elements. Also, we added a more detailed description of how to code walkthroughs were used throughout the process. There were a few formatting errors such as tense and adding small excerpts for the subsections which were addressed. An explanation was also provided for the code coverage which came from the priorities set for certain modules. Lastly, changes were made to the system to pass failing test cases and usability testing was changed as mentioned above.

2 Design Iteration (LO11)

The first version of the design is drastically different from the end product. In the beginning, the functional and non-functional requirements were clearly defined, but not their implementation. Everyone had their idea of how the final product would look like. When we first began development, the team had implemented features with the idea of creating a minimum viable product which can be presented to the supervisor during the team's weekly meetings. During these weekly meetings, the entire team would share their opinion and feedback on the feature's implementation and what its impact would be when integrating it with the rest of the system. Many examples of this come from implemented game logic such as the insurance mechanic which was first developed in a way where players would buy insurance on a type of plant to guarantee a default floor price. If the price of the full-grown crop of that type was lower than the default floor price, the player would be able to sell the full-grown crop at the default floor price instead. Although it sounds satisfactory, its flaws became apparent when this feature was shown off in a weekly meeting. The insurance should be tied to a specific seed instead of a whole crop, and instead of a default floor price, players should be able to choose their own and pay a premium depending on their choice. As the insurance contracts were changed to be tied to specific seeds, the planting seeds popup also had to change to allow users to plant different seeds with different contracts.

Another game mechanic that changed from the first iteration was the farm grid where players grew crops and bought the land. At first, the farm grid had 10 plots of land, with each plot consisting of tiles in a 4x4 grid. After consideration in a weekly meeting, it was noted that there would be at max 160 tiles for a player to grow crops on which is impractical as planting 1 seed would take 2 clicks each. It was also found unnecessary as there would be only 3 types of seeds to plant each season. Therefore it was changed for each plot of land to be 2x2.

How the game would end was another hot topic during development where in the beginning, it was assumed that players should be allowed to play an infinite amount of turns as more data is always good. However, another requirement was that the game should be engaging and fun. Without a defined goal for the player, there is no incentive for the player to try, and so it was decided that the game would end in a set amount of time with the goal being to amass a set amount of money to win.

For saving and loading a user's game, the client requests the MySQL database and populates itself with the response. However as the game mechanics changed, the MySQL database schema for the game state also had to be changed many times to account for saving new or different data. An example of this would be when insurance was changed to be associated with specific seeds, the schema had to change to save these insurance contracts.

Finally, the team received a lot of feedback from the TA, instructor, supervisor, and testers concerning the UI/UX and how the game mechanics were unclear. The UI across different components were not consistent, so all components and their styling were standardized. The shop and inventory were also very unintuitive. For the shop, buying seeds and insurance were done separately which was a pain and so this was changed so that each specifically bought seed had their insurance tied to them. For the inventory, bought insurance was not visible and this was also solved by having the insurance contracts tied to the seeds. The game mechanics of the consultant and the insurance we found to be very confusing to understand and so tooltips were made that summarized their purpose. A tutorial popup was also added to show at the beginning of the game.

3 Design Decisions (LO12)

Many limitations, assumptions, and constraints made the system what it is today. Originally random events such as natural disasters were to have 3d models and animations run through the 3d landscape and farmland. However, we found it to cause performance issues and decided they were not worth it so we opted to use a popup with 2d animations instead. Another scrapped feature was the idea of implementing and utilizing the Black-Scholes formula for the marketplace, however, it was assumed that it would be too complicated and unnecessary for players to take advantage of. Quests from NPCs were also considered, where an NPC would request a specific crop or item in exchange for

paying above the market price. This was shut down as these could influence the player to act in such a way that their data could not be used for the study.

Other limitations came from the overall requirement that the supervisor wished to be able to deploy and build upon the final system in the future. Therefore we used MySQL for the database as the supervisor had previous experience with this. The decision to create a JavaScript-based web app was also a result of this, where the supervisor had experience developing with JavaScript in the past. The implemented code and file structure was made modular, commented thoroughly, and overall clean to further satisfy this requirement. To easily allow the supervisor to deploy the final product, we chose to use a tech stack of React, Node.js, Express.js, and MySQL as the team had experience deploying projects built upon this.

4 Economic Considerations (LO23)

From the beginning of the project, the team's objective has been consistent and unaltered: to build a farming game that will be used by the client for their research. The team nor the client will be focused on profiting from the product after its release. The product's goal is to gather information about how the game is played given the challenges and type of information (probabilistic vs deterministic).

The cost to release the product will be minimal. First, all assets used in the game have a CC0 1.0 license allowing free use for any purpose without permission from the owner. Therefore, there is no cost associated with the assets. Next, the server hosting the product is also free of cost. The client is provided with a server by McMaster University and thus, there will no additional costs once the product is loaded onto the server.

During our meetings, the client conveyed that they will take on the responsibility of finding and acquiring users. The client has created research games in the past and has experience obtaining users for their games through a variety of methods, such as asking students from their department and offering gift cards. As a result, they are no initial costs and a plan for the team to attract users for the product.

To reiterate, the product will be used for research rather than profit. The client is responsible for acquiring users and how much they choose to spend on marketing. These considerations are outside the scope of the team; however, they will provide information and assessment to the best of their ability should the client ever request it.

5 Reflection on Project Management (LO24)

5.1 How Does Your Project Management Compare to Your Development Plan

The team followed the plan throughout the development of the product outlined in an earlier document: Development Plan. The team met with the client once a week for most of the year. There was constant communication with the client, through emails, during weeks the team was busy with deliverables. The previous week's work was discussed among the team and the client. This was advantageous as the client was able to validate the product during the development process, ensuring the desired result.

The team communicated with each other using Facebook Messenger. This worked well as it's an application used daily by most group members. Other members were updated promptly during last-minute emergencies allowing for transparency and any task reassignments. Furthermore, virtual team-only calls also take place on Messenger. It accommodated differing schedules and limited efficiency was lost during work sessions. Members often needed to collaborate with select members to accomplish assigned tasks. Messenger also offers support for individual voice calls allowing for a parallel workflow.

The team member roles provided a framework for assigning responsibilities and assigning tasks. This allowed for a faster process of distributing the work and provided more time to complete the tasks.

The team used the trunk-based development as outlined in the Development Plan. Changes were approved by at least one other member before they can be merged into the main branch. All members were updated on the new changes in the main branch and ensured everyone was aware of the current state of the product. One thing that the team failed to incorporate into the workflow was running automated tests on all pull requests. This led to the introduction of bugs into core features and mechanics which should have been caught before the merge. Furthermore, the plan specifies using GitHub issues for the remaining tasks. The team made use of Messenger; however, it was not as streamlined and efficient as the listed method. Although the team was able to produce the desired product in the end, GitHub issues would have offered the team organization that was missing at times.

5.2 What Went Well?

Various things went well for the project management in terms of process and technology. The weekly meetings with the client allowed us to better understand the product. In each meeting, we were able to get clarification and solidify requirements or uncover the missing requirements. More importantly, this allowed us to gain experience with working with a client which will help us greatly when working in the industry. At first, the client wanted to use a technology stack he was familiar with. The team worked with the client to come to an agreement on the new technology stack, one both parties would be comfortable with. The

technology stack fits the project very well and is commonly used today. While most group members had prior experience with this technology stack, this was a great way to build upon that experience. In terms of the product itself, the client was pleasantly surprised and stated it exceeded his expectations. The client had previous experience with game development but did not want the risks with 3D. We were able to mitigate the risks that came with the 3D models and create an engaging game. Moreover, slightly modifying the experiment to produce more interesting results is another thing that worked well.

The client initially wanted to base the insurance on the probability that a crop will fail. By incorrectly implementing the insurance we were able to change the way insurance works by insuring items based on future prices. This will help to yield more interesting results when collecting data for the experiment.

5.3 What Went Wrong?

There were several obstacles encountered throughout the development process resulting in unexpected difficulties. The main issue was the difficulty in reaching a consensus on the final product, which was not resolved until the late stages of the project. Specifically, the game mechanics of the inventory and shop. Consequently, the insurance, inventory, and shop had to be reworked a couple of weeks before the final demonstration. The majority of the blame falls upon the team for not nailing the requirements and constraints earlier. Regardless, the team was able to rework these components in time for the demonstration; however, it was a close call and could have been avoided. Another issue was the completion of tasks too close to the deadline. With all members having a full course load, it was difficult to stay organized. As a consequence, most of the deliverables were completed on the due date. Lastly, there was no standardized procedure for project management in terms of outstanding tasks. We started to use Outlook's To-Do list but this does not translate to the industry as Asana, Jira, or Github are more commonly used. Next, there is some performance lag for older machines which is determined to be caused by utilizing 3D models. Initially, the client wanted a 2D game due to performance concerns; however, the team felt a 3D game would be more engaging and immersive, which is a major requirement. The team should have planned for an optimization stage in the process to mitigate the performance concerns of the client.

5.4 What Would you Do Differently Next Time?

If the team had to do this project differently, it would be beneficial next time to incorporate Continuous Integration and Continuous Delivery (CI/CD) for automation testing. The team has set up rules such that a pull request must be reviewed and approved by another team member before it can be merged into the main branch. Collectively as a team, we felt that unnecessary time was spent dealing with unexpected errors and outcomes as large merge requests had to be peer-reviewed. The reviewer may not have tested everything before approving the branch to be merged into the main. Thus by implementing a

CI/CD pipeline into our workflow for the project, the team can rest assured that the main branch is protected and working at all times. We could have also assigned team members specific roles in which they were to be experts in certain components of the game. Increasing the minimum amount of people needed to review a merge request and assign "experts" would ensure different subsystems of the game are still working as intended.

Another area we would possibly change for next time is the rendering library chosen. We could look for a library that is designed to handle more complex graphics to be rendered. The team experience performance issues as the complexity of the 3D rending of the environment increased. Despite some attempts to optimize key areas related to the rendering of 3D computer graphics, ideas had to be abandoned as a result. One of these ideas was rendering seasonal events to appear in the environment, such as a live animated tsunami moving along the farm and destroying the crops. Some more examples include a serve snowstorm with multiple snow particles being rendered onto the screen and a tornado gradually increasing in size as it moves across the environment. Instead, the team had to use a static demonstration (e.g. a picture of a tornado) displaying these seasonal events along with a description of them. This greatly prevented us from enriching the user's engagement and experience to a certain level related to this particular area of the game.

Another area we would want to change for next time is bug reporting, the team did not utilize the GitHub issues for bug tracking to its full potential. The team often worked on different parts, often as pairs or groups of three. As a result, all bugs known in that area were only known to the specific subteam. Others would have to recall from the previous meetings or message the group members. Instead, the team should use the bug issues to ensure all team members were aware of the current bugs. In this way, if one person had more time, they would look at these issues and see if they can make some progress on fixing the bug.

Another area we would want to change for next time is the consultant's advice. Typically this advice randomly selects advice to be statements about either a seasonal event or market event and hardly varies depending on the type of event occurring. It would be nice to add some intelligence to it and enhance the statement provided by the consultant. Some possible ideas could include observing the current game state and progress. Moreover, tracking their inventory, the different items they bought/sold, and looking at commonly bought items would inevitably help curate a more personal related response to the user and help them feel more immersed within the game.

An increased budget for this project would allow the use of different assets in the game. Although the team found adequate with the assets used for this project, assets designed by a professional could go a long way in improving the user experience and engagement. Using a third party would also allow for assets to a thematically consistent. It would also extend to multiple themes, allowing players to pick the one they prefer the most.