Table 1: Revision History

Date	**Developer(s)**	**Change**
09/25/2022	Namit Chopra, Brandon Duong Andrew Balmakund, Mohammad Harun Mihail Serafimovski	Finished First Version

Development Plan
Farming Matters

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

This document discusses the development of the project. It outlines the team meeting plan, communication plan, member roles, workflow plan, and proof of concept demonstration plan. Furthermore, it also details the technology being used, coding standing as well as project scheduling.

1 Team Meeting Plan

The team members will meet at least twice a week at flexible times, either in person or online. Formal meetings with the supervisor and members will occur every Wednesday at 10:45 am. The meeting scribe will be responsible for recording the contents and topics discussed at each meeting. Other team members will create a list of activities and topics to discuss as well as time estimates for each item. All attendees will have a chance to contribute to the discussion and have their input taken into account. Group decisions and work will be part of a team-only discussion. Individual tasks will be assigned during these meetings to be completed before the next meeting. The meeting scribe will post the contents summary of the meeting in the Facebook chat which is available to all team members.

All members must be present for all meetings. If a member is absent, they must notify the rest of the group one hour before the meeting with a reason. If the meeting cannot be postponed, the absent member must reach out to inquire about the content discussed in the meeting.

2 Team Communication Plan

The team will have a Facebook Messenger group to communicate any administrative and miscellaneous issues with each other. Facebook Messenger will be the main form of communication. It will be used to set up meeting times, and

discuss small ideas about different aspects of the project, and documentation issues. For code-related issues, the team will track and communicate these tasks using GitHub issues. The team will be using E-mail to communicate with the professor and supervisor.

3 Team Member Roles

Building a large project in a team of 5 requires each member must occupy multiple roles. There is not a team leader but rather a liaison. Namit will be responsible for communicating with the professor and supervisor. Namit will also be responsible for being the lead assets designer. Brandon is responsible for being the lead tester and the git expert. Andrew will be appointed as the latex expert. Mohammad will be appointed as a scribe for each meeting, recording information and decisions discussed. Mihail is the moderator for the team and supervisor meetings. He is also the react expert. Furthermore, all members will be developers, involved with documentation, and testers. Although the roles have been assigned to each team member, new roles may arise further into the project. Member roles may change depending on the needs of the projects at different points in time. Any role change and addition must be discussed among the group members first.

Member Names	Roles/Expert
Namit Chopra	Liaison, Developer, Documentation, Tester, Lead Assets Designer
Brandon Duong	Developer, Documentation, Lead Tester, Assets Designer, Git Expert
Andrew Balmakund	Developer, Documentation, Tester, Assets Designer, LaTex Expert
Mohammad Harun	Meeting Scribe, Developer, Documentation, Tester, Assets Designer
Mihail Serafimovski	Moderator, Developer, Documentation, Tester, React Expert

4 Workflow Plan

For the Git workflow, we will use trunk-based development. The main branch will be protected such that no update can be pushed to it directly. This will ensure that the main branch will always be stable. A new and separate branch will be created for every change that is based on the main branch. A pull request must be created and approved before it can be merged into the main branch. All tests should pass and at least one developer, other than the author or the pull request collaborators, must review the code before approving the pull request. The feature branch will be deleted once the feature branch is merged into the main branch.

Each task will be added as a GitHub issue and classified using the default labels including bug, documentation, and feature. Developers will assign themselves issues based on priority and familiarity. Once a developer's task is finished and merged into the main branch, the corresponding issue will be closed. A template will be set up on GitHub issues for a standard bug report and feature request.

5 Proof of Concept Demonstration Plan

5.1 Risks

- The data collected for research may be biased or inaccurate. Will the data collected be useful for the research?
- The game may fail to immerse the user causing them to quit before sufficient data is collected.
- Making the game accessible and playable across multiple platforms like smart phones, laptops, tablets, and desktops might require too much development effort.

5.2 Overcoming Risks

To overcome these risks, the V model will help ensure we are following an iterative process.

- The team will incorporate several UI design principles to improve the users experience with the game.
- The team will demonstrate that the decisions critical for research are indistinguishable from unimportant decisions

5.3 Proof of Concept Demonstration

For the proof concept, the following will be demonstrated:

- Successful logging of users decision
- Basic interaction for user decisions
- Successful interpretation of data collected
- Basic interface completed using good design principles
- Decisions for the first turn is completed and simulate a singular event
- The game can be played on different screen resolutions

6 Technology

- HTML, CSS, JavaScript
  - Since we need the client-side code to run on browsers (as specified by the project supervisor), HTML/CSS/Javascript are necessary.
- For front-end, we will use the React framework for JavaScript.
  - Although the React framework adds more learning on top of the bare-bones HTML/CSS/JS stack, the team felt that because we all have existing experience with React, it would streamline development.
  - React is very popular, there is a lot of documentation, examples, and extra packages available that make development easier. We can easily extend the functionality of React by using packages for whatever we need.
  - One possible restriction is that React has more of an impact on system performance and requires a stronger network connection. Seeing how many websites use React today, and considering that the app's demographic is not specifically people who have slow internet or old computers, this restriction is very unlikely to have an impact.
- For back-end, we will use the Node.js framework.
  - Node.js is a back-end runtime for Javascript that executes Javascript code outside of a web browser. Node.js is very commonly paired with React as they both use Javascript, allowing for easy communication between front-end and back-end components. It also will increase the speed of development as the team doesn't have to learn a new language that some of us may be unfamiliar with.
  - To easily create a REST API, we can extend the functionality of Node with the Express package.
- VSCode will be used as a coding editor by all developers. It will provide extensions and features that will help keep good programming practices and is helpful for debugging code.
- For front-end testing, we will use Jest. For back-end, we will use Mocha.
  - Mocha is the most popular Javascript unit testing framework. It is designed to be simple, extensible, and fast. Mocha provides a lot of useful functionality out of the box such as fixtures, mocking, etc. It also provides a lot of features for async testing, which will be useful for testing back-end components that need to do async operations such as writing to a database (for example).
  - Jest is also very popular, however we will specifically use it for front-end testing as it provides a suite of useful features such as snapshot testing that can be used to ensure the layout of a page doesn't change.

- For code coverage, we will use istanbul.js.
  - Istanbul.js is a very simple, lightweight code coverage tool that bundles and runs all unit tests with a single command. It generates a very readable and understandable output and prints it to the terminal. An example can be seen in the linked website.
- Latex will be used for documentation
- CI will be set up using GitHub actions. The CI system will run automated tests triggered by a change to the main branch to ensure the main branch is always stable.
  - Mocha and Jest both have the necessary features needed to automate testing with GitHub actions.
- Performance measuring tools are not needed for this project as performance is not a requirement or limiting factor.
- Linter: prettier

7 Coding Standard

The team will follow a React and JSX Style Guide developed by Airbnb. The main purpose of the coding standard is for consistency throughout all files. The coding standard will also used for following the best guidelines and practices for writing code.

8 Project Scheduling

- Meet with supervisor every week to explain which deliverable is currently worked on, and to ask questions related to said deliverable
- Complete the first draft of the document four days before the deliverable due date
- All project deliverables, supervisor meetings, and team meetings will all be monitored on Google Calendar that will be shared amongst the development team and supervisor