System Design for Farming Matters

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

April 5, 2023

1 Revision History

Date	Version	Notes
18/01/2023	1.0	Finished First Version
05/04/2023	1.0	Finished Second Version

2 Reference Material

This section records information for easy reference.

2.1 Abbreviations and Acronyms

symbol	**description**
Farming Matters	Final Year Software Engineering Capstone Project name
HTTPS	Hypertext Transfer Protocol Secure
React	Frontend framework
Node.js	Backend framework
MySQL	Database management system
Redis	Database management system
Firebase	Cloud computing services provided by Google
API	Application Programming Interface
Auth Token	Used for users to verify their identity
Game State	Current state of the game (i.e money, inventory, seeds planted)
UI/UX	User interface/User experience

Contents

List of Tables

List of Figures

3 Introduction

The following document outlines the system design of Farming Matters. The project aims to conduct survey research through an interactive and engaging activity. This will further help understand genuine decisions from the users to help with the research of understanding risk-making decisions.

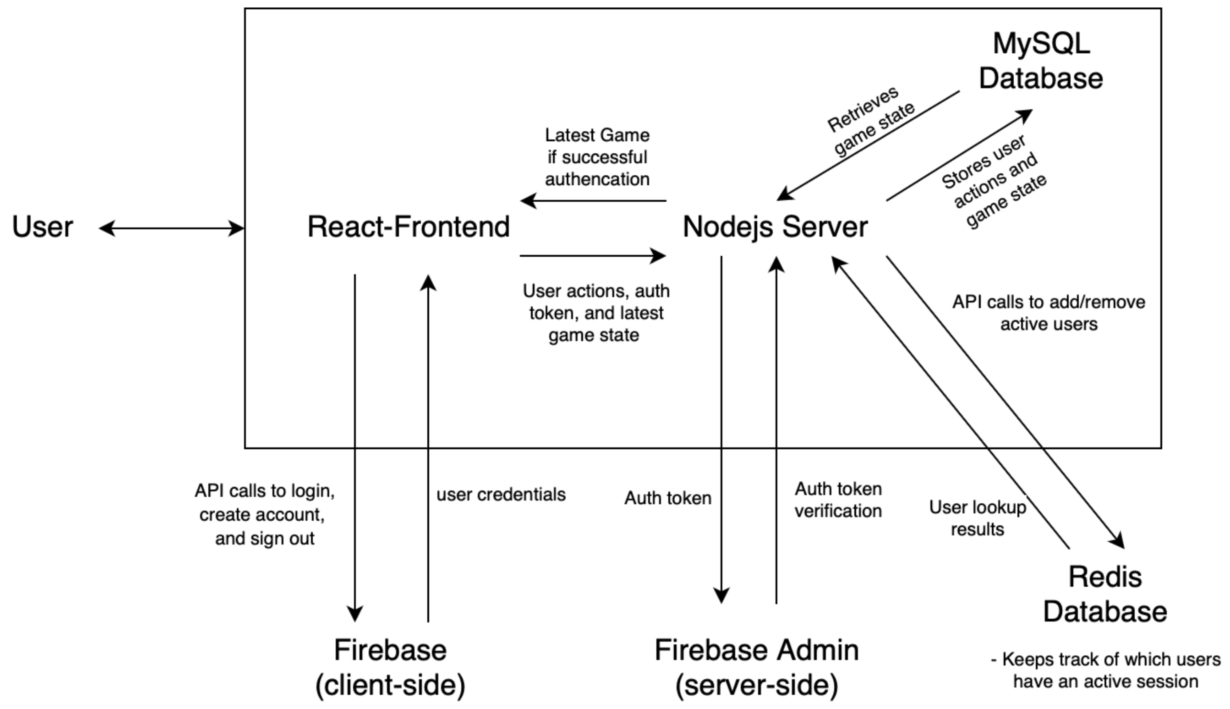
Farming Matters has been developed incrementally in order to better understand different non-technical aspects before starting the technical side. The Problem Statement and Development Plan were the first documents to be completed for the project. It helped the team better understand the problem, what was expected in the solution, and how the team will go about the development. One of the most important documents, Software Requirements Specification, was created in conjunction with Dr.Yiannakoulias who is the supervisor of this project. The Hazard Analysis provides an analysis regarding hazards, how to mitigate each hazard and provide safety and security requirements. The Verification and Validation Plan outlines the testing the document once the implementation is completed. All the documents leading up to this one had to be taken into account when developing the system design document for Farming Matters.

4 Purpose

The system design document outlines the best possible design for Farming Matters given the constraints and requirements outlined in the previous documents. The design of the system is composed of two other documents, Module Guide and Module Interface Specification. These documents detail different design aspects and work together to provide the design of the overall system. The scope of the project is provided to illustrate the system boundaries. Furthermore, the document outlines the normal behaviour, undesired event handling, component diagram, and the design decisions made to incorporate requirements provided in previous documents. The interface designs, communication protocols, and the timeline for completing the project are also included.

5 Scope

The developed system, which the user interacts with, consists of the React frontend, Node.js server, and the MySQL database. The system leverages external services provided by Google's Firebase for authentication, including signing up, logging in and out, and verifying users as authorized. The system also leverages a Redis server to ensure an account only has one logged-in session at a time. The React frontend sends user actions, auth token, and current game state to the Node.js server. The Node.js server responds with a loaded game state if auth token is successfully verified. The Node.js server stores user actions and the game state within the MySQL database, and the MySQL database responds with the stored game state. The final developed and deployed system will be available to users through the web browser.



MySQL Database
Retrieves game state
Stores user actions and game state
Latest Game if successful authencation
User
React-Frontend
Nodejs Server
User actions, auth token, and latest game state
API calls to add/remove active users
API calls to login, create account, and sign out
user credentials
Auth token
Auth token verification
User lookup results
Redis Database
- Keeps track of which users have an active session
Firebase (client-side)
Firebase Admin (server-side)

Figure 1: System Boundary Diagram

6 Project Overview

6.1 Normal Behaviour

This web game is intended to be played by the participants of the research study. The participants will play on either a desktop computer or laptop with a monitor resolution of at least 1280 by 720, and a stable connection to the internet. This consistency ensures that all research-related decisions and relevant game actions are recorded without failure, to not bias the study.

The normal behaviour of the system includes logging in and out of accounts, buying and selling items, planting and harvesting a crop, making research-related decisions (i.e whether or not to pay for consulting information or insurance), and ending a turn. Upon ending a turn, all relevant actions are recorded and saved to the database which is used for loading a player's game and also analyzed as a part of the research.

6.2 Undesired Event Handling

Undesired events include users trying to play without any account, users trying to share accounts, users creating bots to play the game, users having an unstable connection to the internet, and users having multiple sessions open.

To tackle these problems, only authorized users will be able to save their games and have their data tracked. In the case of account sharing, participants in the study will have to understand and accept the guidelines that rule against this. If there is suspicion of account sharing or botting, the account's data will also not be included in the study. If the player loses internet connectivity in between turns, their data would not be able to be uploaded to the database and so to deal with this, the game will resume at the most recently saved state. Lastly, in the undesired case where a user is trying to log into their account in multiple tabs, the system will automatically log them out of the older session to avoid any race conditions.

6.3 Component Diagram

Similar to the system boundary diagram, the system consists of the Frontend, Consultant, Insurance, Plant Harvest, Shop, Inventory, Server, Authentication, and the Database module. The Consultant module deals with the game mechanic of buying consultant advice, and whether it returns deterministic or probabilistic data. The Insurance module deals with the game mechanic of tracking insurance for planted crops, and at what floor price they were insured. The Plant Harvest module deals with the game mechanic of planting, growing, and harvesting crops, buying land, and applying fertilizer. The Shop module deals with the game mechanic of buying seeds and fertilizer. The Inventory Module deals with the game mechanic of tracking what items the user currently has, including seeds, harvested crops, and fertilizer. The Frontend module uses the Consultant, Insurance, Plant Harvest, Shop, and Inventory module for dealing with any game logic needed. The Frontend module also handles UI/UX, and how the user interacts with the system. The Frontend module also uses the Authentication module, which consists of handling user sign-up, login, and auth token verification. The Server module deals with logging user actions, saving the game state, and loading the game state, all of which use the Database module. The Database module consists of the MySQL database and stores user actions and game states received from the Server module. The Frontend module uses the Server module to request the user's stored game state, save the user's current game state, as well as to store user action data.

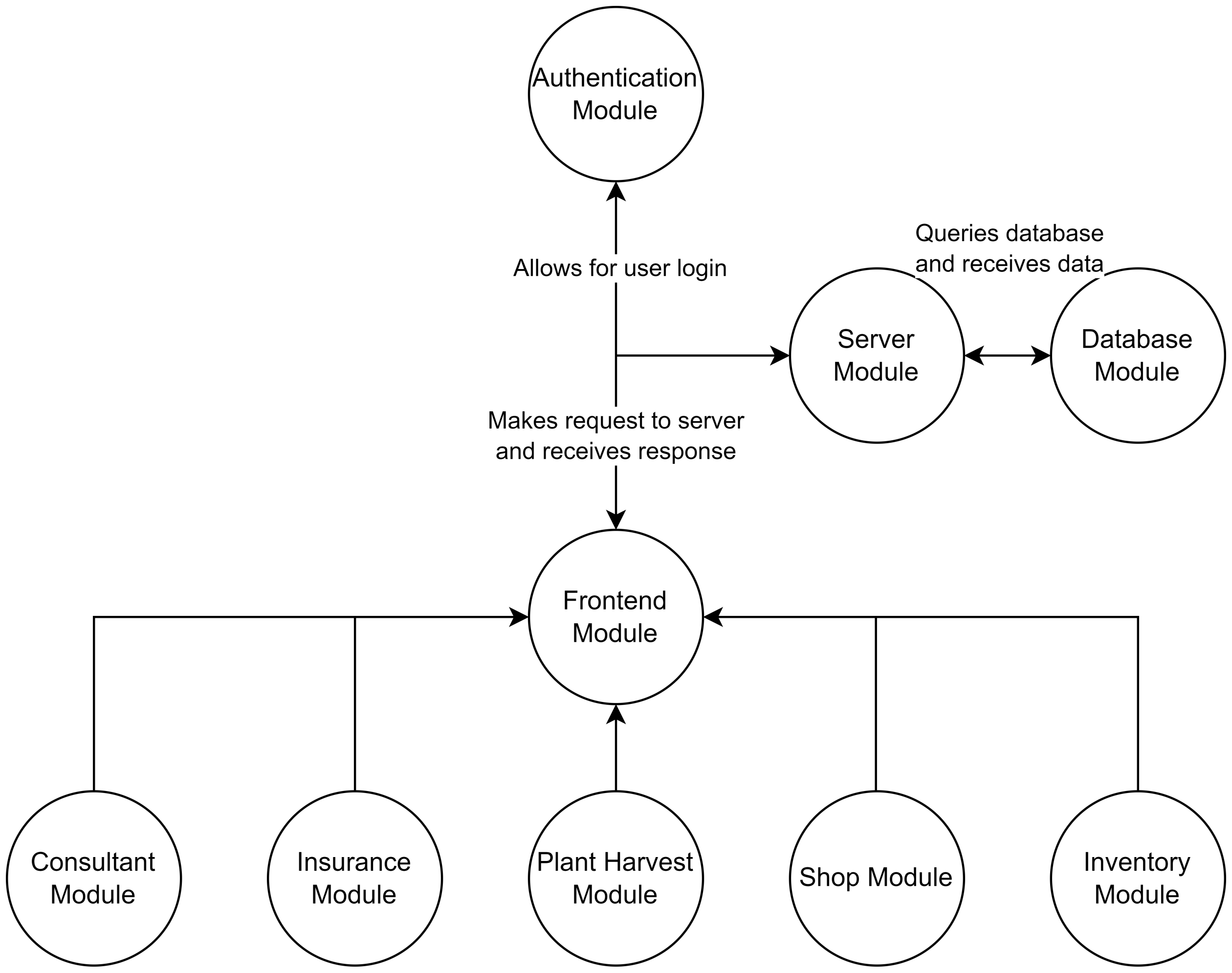


Figure 2: Component Diagram for the whole system

6.4 Connection Between Requirements and Design

1. **FR3** To satisfy the requirement of accumulating in-game currency, players will be able to sell seeds, items, and harvested crops for money. Players may also gain money from random events.

2. **FR5, SR1, PR1** To satisfy the requirement of verifying new users as humans, users will have to complete a captcha before creating an account.

3. **FR6** To satisfy the requirement of a shop to purchase items, users will be able to buy seeds, fertilizer, farm buildings, and land.

4. **FR7** To satisfy the requirement of growing crops, users will be able to plant seeds and fertilizer. Different seeds can be planted in different specific seasons, and they all have their growth length. Different fertilizers either allow for the harvested crop to be sold for more money than normal, or for the seed to grow faster.

5. **FR11** To satisfy the requirement of prompting for consulting advice, players will not be able to end their turn unless they respond to the consulting advice prompt. Other NPCs and random events will help disguise this research's relevant decisions.

6. **FR12** To satisfy the requirement of prompting for insurance, players will not be able to end their turn unless they respond to the insurance prompt. Other NPCs and random events will help disguise this research's relevant decisions.

7. **FR13** To satisfy the requirement of logging user decisions, the system will log all game actions that are relevant to the study. These include buying and selling items, and whether or not the player buys consulting advice or insurance.

8. **FR14** To satisfy the requirement of saving the user's game state, the system will store current money, turn, farm grid, and inventory.

9. **FR15, LR1** To satisfy the requirement of deleting a user's data, they will have to alert the research conductors and they will remove the user's data from the database.

10. **FR20** To satisfy the requirement of assigning users to focus groups, participants will be randomly assigned to either always receive deterministic or probabilistic information from the consultant.

11. **FR21** To satisfy the requirement of random events, users will have a chance to be prompted by one after ending their turn. These random events include natural disasters, lucky harvest conditions, and gifts.

12. **UH1** To satisfy the requirement of being easy to learn, there will be a heavy emphasis on using both icons and text for any button.

7 System Variables

N/A

8 User Interfaces

The interface design shown below is the layout of Farming Matters. After the initial phase of development and use, these designs are the most intuitive for navigation and presentation. These designs are open to change during the development and completion of the system. Different colour schemes will also be tested before deciding on a final one. The final user interface will be decided based on the feedback from the developers and initial users. In the designs, all circles will be populated with an appropriate icon.

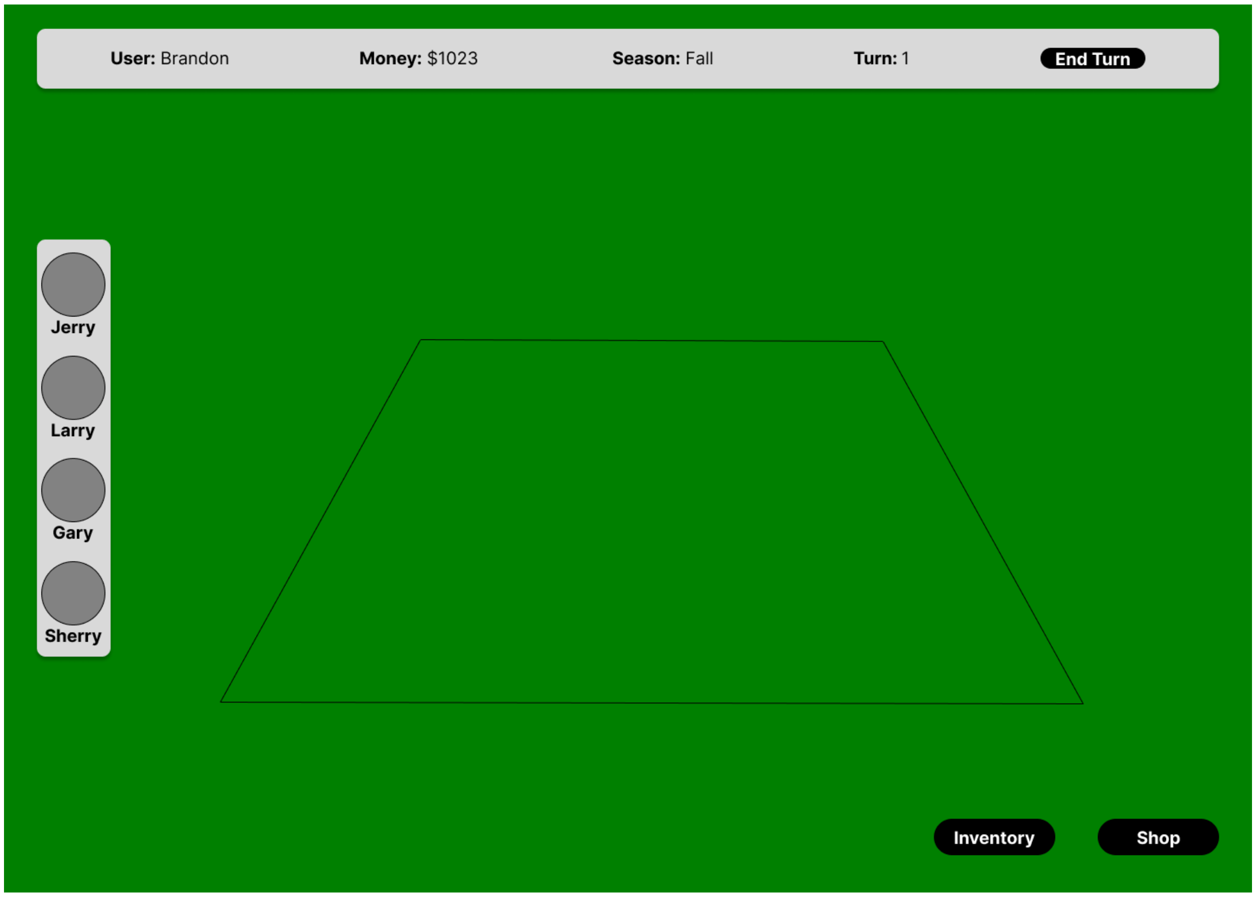


Figure 3: Base UI with nothing selected

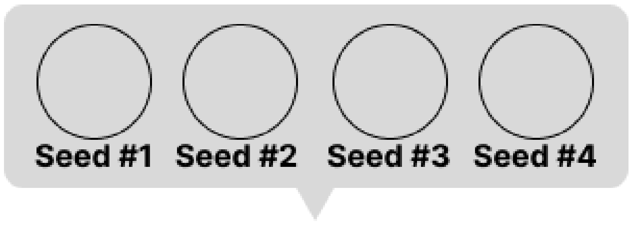


Figure 4: Popover for planting a seed

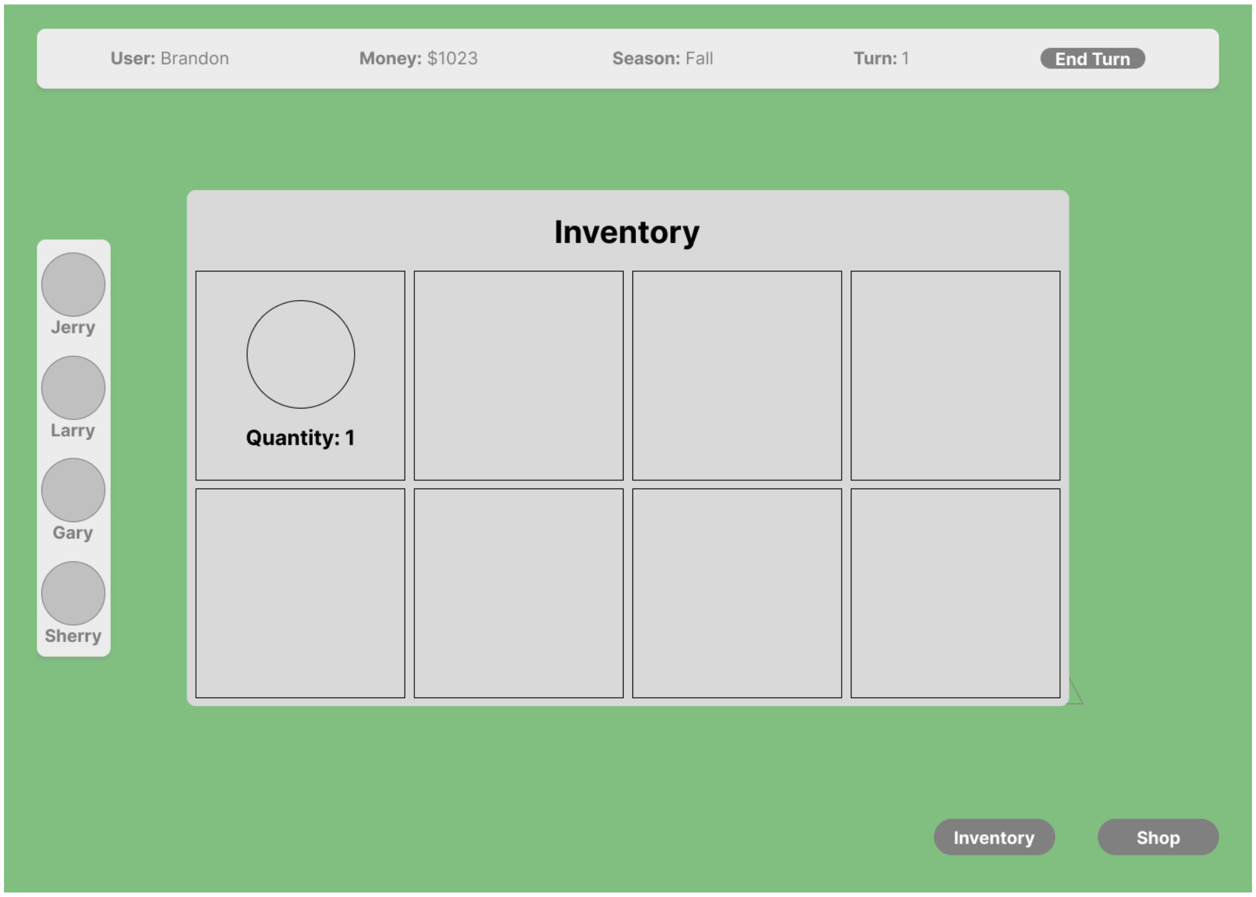


Figure 5: UI for Inventory Screen

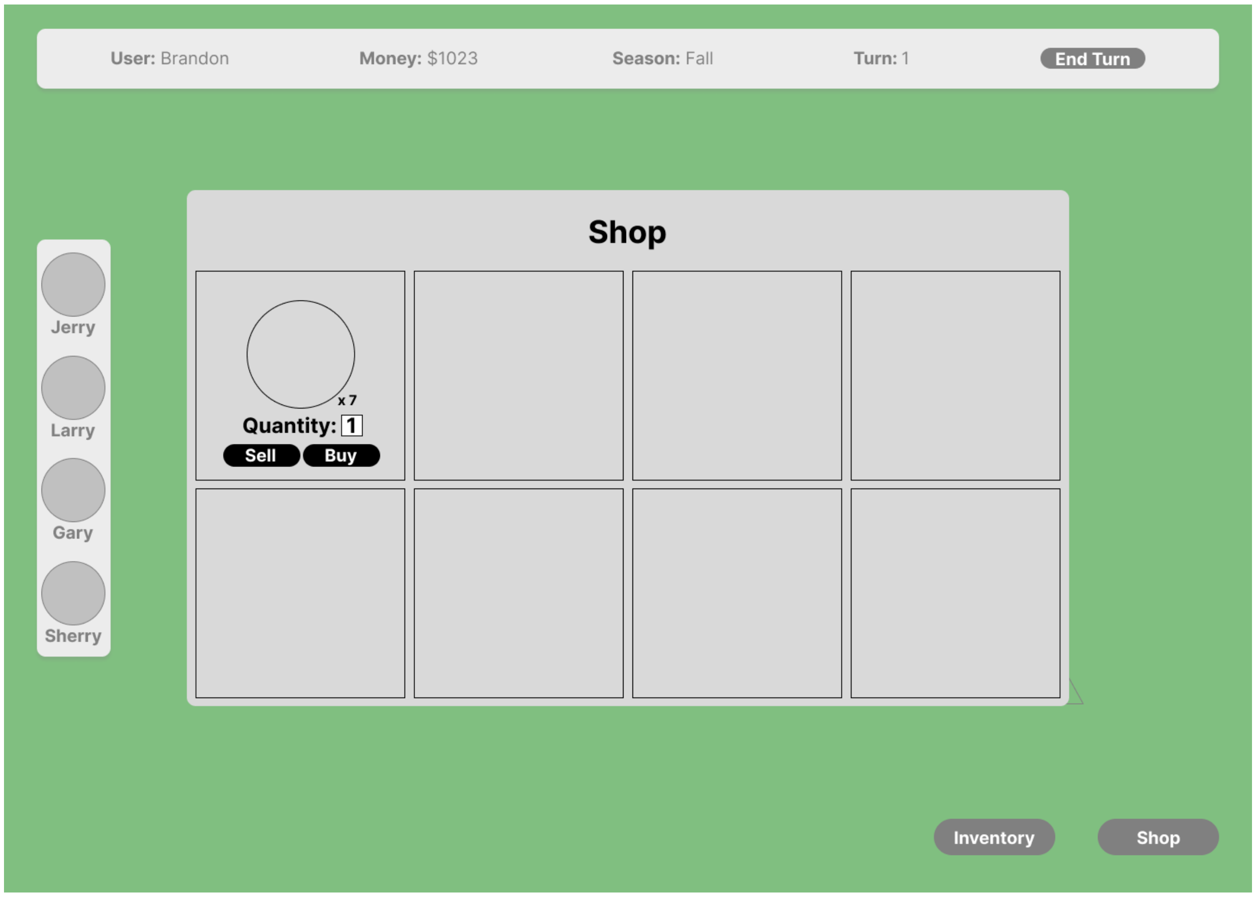


Figure 6: UI for Shop Screen

9 Design of Hardware

N/A

10 Design of Electrical Components

N/A

11 Design of Communication Protocols

The system relies on both traditional HTTPS requests and a Websocket connection which persists as long as the user has the application open. HTTPS requests are used to send user actions and game states from client to server, and the server can send the client a saved game state via an HTTPS response. Headers are used in the requests for authorization, so an auth token must be included in any request header. HTTPS is preferred over plain HTTP because the system has login functionality which deals with sensitive data (email/passwords).

Websockets are used to track a persistent connection between the client and server. Specifically, WebSockets are leveraged to keep track of each unique user to ensure that a user may not have multiple active sessions. Event handlers on the WebSocket are used to track the state of the user (logged in or not) and set the 'active' state of the user respectively. When a client signs out or disconnects, the 'active' state is removed.

12 Timeline

Module Name	Team Member	Due Date
Avatar	Andrew	Jan. 1, 2023
AvatarMenu	Andrew	Jan. 1, 2023
Consultant	Andrew	Jan. 27, 2023
OtherAvatar	Andrew	Feb. 3, 2023
Manual Testing of Consultant Functionality	Andrew	Feb. 10, 2023
Item	Mohammad	Jan. 15, 2023
Inventory	Mohammad	Jan. 13, 2023
Manual Testing of Inventory Functionality	Mohammad	Jan. 20, 2023
Market	Mohammad	Jan. 27, 2023
Manual Testing of Shop Functionality	Mohammad	Feb. 3, 2023
DatabaseOperations	Namit	Jan. 29, 2023
Manual Testing of Logging, Saving, and Loading	Namit	Feb. 4, 2023
MusicPlayer	Namit	Feb. 5, 2023
GameSettings	Namit	Feb. 10, 2023
Manual Testing of Music and Settings Functionality	Namit	Feb. 17, 2023

Table 1: Farming Matters Module Completion Timeline Part 1

Module Name	Team Member	Due Date
AuthState	Mihail	Jan. 15, 2023
Socket	Mihail	Jan. 20, 2023
CilentFirebase	Mihail	Jan. 8, 2023
User	Mihail	Jan. 1, 2023
AuthError	Mihail	Jan. 1, 2023
CreateAccount	Mihail	Jan. 13, 2023
Login	Mihail	Jan. 13, 2023
ServerFirebase	Mihail, Namit	Jan. 29, 2023
RedisClient	Mihail	Jan. 1, 2023
Server	Mihail, Namit, Andrew	Feb. 3, 2023
Manual Testing of Authentication Functionality	Mihail	Feb. 10, 2023
Seed	Brandon	Jan. 20, 2023
FarmTile	Brandon	Jan. 27, 2023
FarmGrid	Brandon	Jan. 31, 2023
Manual Testing of Planting, Growing, and Harvesting Functionality	Brandon	Feb. 7, 2023
GameController	Andrew, Namit, Brandon, Mihail, Mohammad	Feb. 12, 2023
Manual Testing of Game Controller	Andrew, Namit, Brandon, Mihail, Mohammad	Feb. 19, 2023

Table 2: Farming Matters Module Completion Timeline Part 2

A Interface

N/A

B Mechanical Hardware

N/A

C Electrical Components

N/A

D Communication Protocols

- HTTPS
- Websocket

E Reflection

The information in this section will be used to evaluate the team members on the graduate attribute of Problem Analysis and Design. Please answer the following questions:

1. What are the limitations of your solution? Put another way, given unlimited resources, what could you do to make the project better?

   The limitations of the solutions include the assets and sound used in the game. The assets were mostly free and given unlimited resources perhaps the look and feel of the project could be improved. Another limitation would be the modularization of the solution. This could be perhaps because of the use of React which supports functional programming better. Another limitation of the solution is the data that can be collected. The ethics board limits us from gathering all data that can be considered unethical and there may be cases where we require certain data but it is not approved. Lastly, the solution has not been thoroughly tested and multiple users need to provide input regarding the solution. With unlimited resources, we could gather more data from a significant number of users.

2. Give a brief overview of other design solutions you considered. What are the benefits and tradeoffs of those other designs compared with the chosen design? From all the potential options, why did you select documented design?

We explored trying to create our authentication implementation but due to time constraints and complexity, we decided it was better to use Firebase for help with authentication. Another design that was considered was hosting the MySQL database on the cloud. The main stakeholder rents a server and preferred that the logs were sent to his server and that the server would support an SQL database. Some software design considerations the team focused on was deciding where to store the game implementation-specific details such that this game logic is kept a secret from being exploited by users. Intuitively, we considered putting game-specific logic in the backend since the user will only have access to the client side of the application. However, this will increase the number of requests from the server side to the client side, thus putting more stress on the bandwidth and throughput of what the server can handle, causing further delay gameplay-wise and hindering user enjoyment. In the end, we decided to move the game logic to the client side. We have researched and found a way to obfuscate the client-side code to prevent others from exploiting the game.