CS 3110 MS2 Report

Joanna Andrews
Mishita Kaja
Rishika Cherivirala
Jessica Andrews
Jeana Han

Vision

We have a clear vision of what we aim to achieve during our project. Our primary objective is to create a comprehensive garden simulator that will offer an unmatched user experience. Initially, our focus was solely on building a garden and tending to the flowers using features such as feeding, watering, and neglecting the plant, but we have now expanded our vision to include a complete ecosystem with flowers and fruits. Our new plan involves integrating a marketplace and a bakery into the simulator, which will allow users to engage in a more immersive and fulfilling experience. Through the marketplace, users will be able to sell plants and baked goods created using recipes, and in turn, gain coins that they can use to further care for their plants. This feature will not only add to the game's excitement but also enable players to hone their entrepreneurial skills. With these enhancements, our garden simulator will offer a more engaging and interactive experience that will keep users enthralled for hours.

Summary of progress

Between the milestones MS1 and MS2, we were productive in planning, beginning, and creating our implementation of our project: Garden Gameplay (GG). We began with planning which compilation files are needed for our project. As a starting point, we came up with two compilation units: garden and plant. As a result, we currently have four files in our lib directory: plant.ml, plant.mli, garden.ml, and garden.mli. In the plant.ml file, we specified a plant type with a height attribute. Along with this type, we also specified a type with four types of plants: flower, peach, strawberry, and cactus. Our plan is to allow the user to grow/feed the plants so that the heights of each plant grow. As another option, the user can choose to neglect the plants so that the heights of the plants do not grow and eventually decrease. Our goal is to let the plants die off if their heights decrease to a certain number. To code up our ideas, we implemented three functions as a starting point. The three functions we implemented include: feed, print_plant, and create_plant. To implement our idea of allowing the user to grow the plants, the feed function matches each plant type to a height x and increases the plant type's height by 1. In order to allow the user to actually visualize the plants that are in their garden, the print_plant function matches the plant type's height and determines what will be printed depending on its height. Lastly, the create_plant function matches a string representation of each plant name ("Flower", "Peach", "Strawberry", and "Cactus") and returns a plant type with its height. If a string does not match with any of the existing plants, the function will fail with "we don't have this plant yet..". The corresponding mli file, plant.mli contains the types of these three functions and type t. In our second compilation unit, garden.ml, we created a type t of Garden of Plant.t list. Along with this, five functions were implemented: create_garden, add_plant, feed_plants, printable, and print. The function create_garden creates an empty garden and add_plant allows a plant (a string) to be added to the garden. The function feed_plants maps through each plant in the garden (which is represented as a list) and feeds each one for it to grow in height. Lastly, the function printable converts the garden into a horizontal list and the print function prints this. The corresponding garden.mli file contains the type and all the described functions except printable. Lastly, in order to visualize our progress and run the code we currently have, we implemented our main.ml file. We created an empty garden using the create_garden function. We then added three plants: "Flower", "Peach", and "Peach" to this garden. Then, we created a print "menu" that allows the user to press "1" to "Feed Garden" and "2" to "Observe Garden". This will allow the user to visualize the growth of the plants or observe their garden. For now, we currently implemented it so that the user has a total of 6 times to select the two options from the menu. This is a way to simulate the days a user will have to interact with their garden. This is additional functionality we are planning to implement in the future. Completing these functions between MS1 and MS2 have allowed us to progress through our project, successfully preparing us for our first demo!

Along with the functionality of our code, we also continued exploring using git as a team of 5 members. Through trial and error, every person on the team was able to git push and git pull, allowing the team to familiarize themselves with the process of pushing and pulling with many people. We also learned, explored, and completed writing the YAML file and install file in preparation for MS2. Overall, each member of the team made successful contributions to progress the project to our milestone!

Activity Breakdown

- Jessica:
  - Task: Implement the interfaces of plant.ml and garden.ml; Brainstorm current and new features of the game
  - Delivered: Delivered the files and implemented the interfaces.
  - Hours: 2 hours
- Joanna:
  - Task: Implemented the types in plant.ml, main.ml, and garden.ml;
  - Delivered: Delivered the files and implemented types in plant.ml, main.ml, and garden.ml
  - Hours: 3 hours
- Mishita:
  - Task: Implement the feed function in plant.ml; create YAML and INSTALL files
  - Delivered: I delivered the files and implemented the feed function in plant.ml
  - Hours: 2 hours
- Rishika:

    - Task: Implement create_plant function in plant.ml; Brainstorm ideas for the game; Write parts of the report
    - Delivered: Delivered the files and implemented the create_plant function in plant.ml
    - Hours: 2 hours
- Jeana:
    - Task: Implement the feed function in plant.ml and print function in garden.ml; Schedule meetings for the demo and check-ins; Write parts of the report
    - Delivered: Delivered the files and implemented the feed function and print function
    - Hours: 2 hours

Productivity Analysis

As a team, we were productive as we were able to accomplish everything that we had planned in our sprints. For our first sprint after MS1 was due, our goals were to brainstorm ideas on how to implement the game and what features we wanted for our game. To do this, we wrote out a list of all the features that we all thought that we should implement in our project. For example, we wanted the user to have the ability to feed and water the plants in the garden, have a store where the user can buy/sell things, and have random tragedies that kill the plants in the garden. We also planned out the basic compilation units and the type of functions for the features that we wanted to code for our demo. For example, we knew that we needed a garden.ml, garden.mli, plant.ml, and plant.mli files where plant.ml and plant.mli specified the types of plants we wanted, and what attributes the plants get, like their height. We also planned that our implementation of the feed function should be in the plant.ml file. For our second sprint, we began to start the implementation of our game. For this sprint, we wanted to be able to display a small garden, feed the plants in the garden to allow them to grow/develop, and allow options for the user to either feed the plants or observe their garden. Since we were able to do this, we accomplished what we had planned in our sprints and our estimates of what we could do were accurate.