INTERNATIONAL UNIVERSITY
VIETNAM NATIONAL UNIVERSITY – HO CHI MINH CITY
School of Computer Science and Engineering

PROJECT REPORT

SUSY ISLAND

OBJECT-ORIENTED PROGRAMMING (ITIT22UN21)

Semester 1 - Academic year 2023-2024
Course by Dr. Tran Thanh Tung and MSc Nguyen Quang Phu

Group members:

Nguyễn Xuân Trâm Anh (ITDSIU22177)
Đặng Phương Mai (ITDSIU22172)
Lâm Thị Bảo Ngọc (ITDSIU22169)

TABLE OF CONTENTS

CONTRIBUTION TABLE

No	Student's Name	ID	Contribution (%)
1	Nguyễn Xuân Trâm Anh	ITDSIU22177	40%
2	Đặng Phương Mai	ITDSIU22172	44%
3	Lâm Thị Bảo Ngọc	ITDSIU22169	16%

ABSTRACT

Susy Island is a 2D pixel-based platform game that has been in development for a considerable period, which is the result of an extended collaborative effort. The original inspiration for this project came from the well-known role-playing simulation game Stardew Valley, which captured our attention. Further motivation was derived from RyiSnow's educational "How to make a 2D Game in Java" YouTube video series, which offered insightful analysis and helpful technical advice throughout the game development process.

The core gameplay of Susy Island revolves around the player's exploration of a customized map, with the primary objective of navigating through the forest to discover a pathway leading to the ocean for a fishing achievement. The game ends when the player cannot catch 5 fishes before the end of the day or loses all current hearts. On the way to explore the map, the player can encounter several traps or dangerous enemies that pose threats to their in-game life or discover life-reviving items and uncover valuable treasures. In addition, the player can also interact with non-player characters (NPC) and buy, sell, or exchange items, which contributes to the overall richness of the gaming experience.

Chapter 1: INTRODUCTION

A. Objectives:

The project's goal is to create a fully playable game based on our team's discussion and research while improving our proficiency in Java coding skills and OOP techniques. The game's storyline is about the adventure of a boy discovering the world and trying to accomplish the in-game achievements within a limited time. This game can also demonstrate the four main principles of Object-Oriented Programming (OOP).

Summarily, this project aims to:

- Creating a platform game for entertaining.
- Self-learning and practicing Java programming language and OOP techniques in the Theory class.
- Going through the process of designing, developing, and optimizing code for the game.
- Evaluating the ability to build more new features on the base program.
- Gaining expertise in project management, programming, managing game mechanics, and collaborative working on a challenging project involving multiple contributors.

B. The tools used:

- IDE for programming and debugging: JetBrains IntelliJ.
- Mean of code version management: GitHub. (Figure 1.1)
- Mean of project management:
    - Manage and keep track of tasks: Google Sheets.
    - Game's rules summary: Google Docs.
    - Game's graphics/art design: Pixel Studio (Figure 1.2) and Adobe Photoshop (Figure 1.3).
- Means of Communication: Messenger. (Figure 1.4)

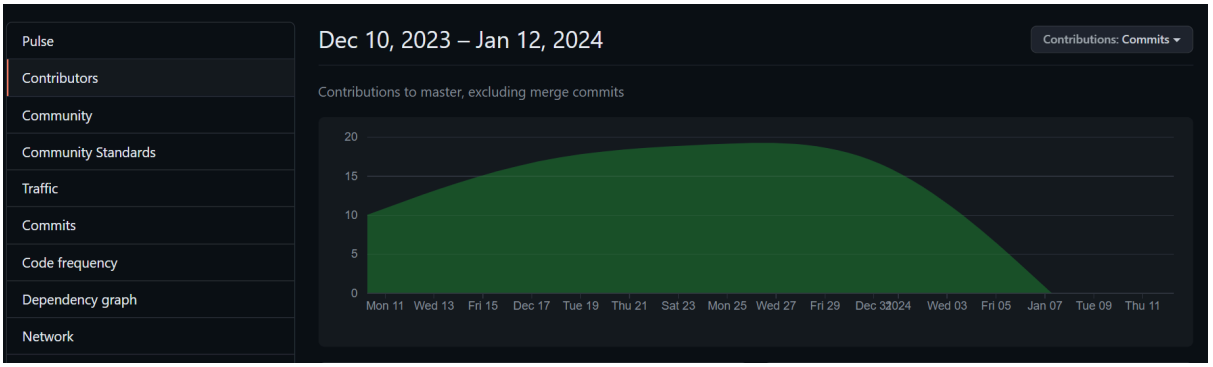


Figure 1.1. GitHub statistics

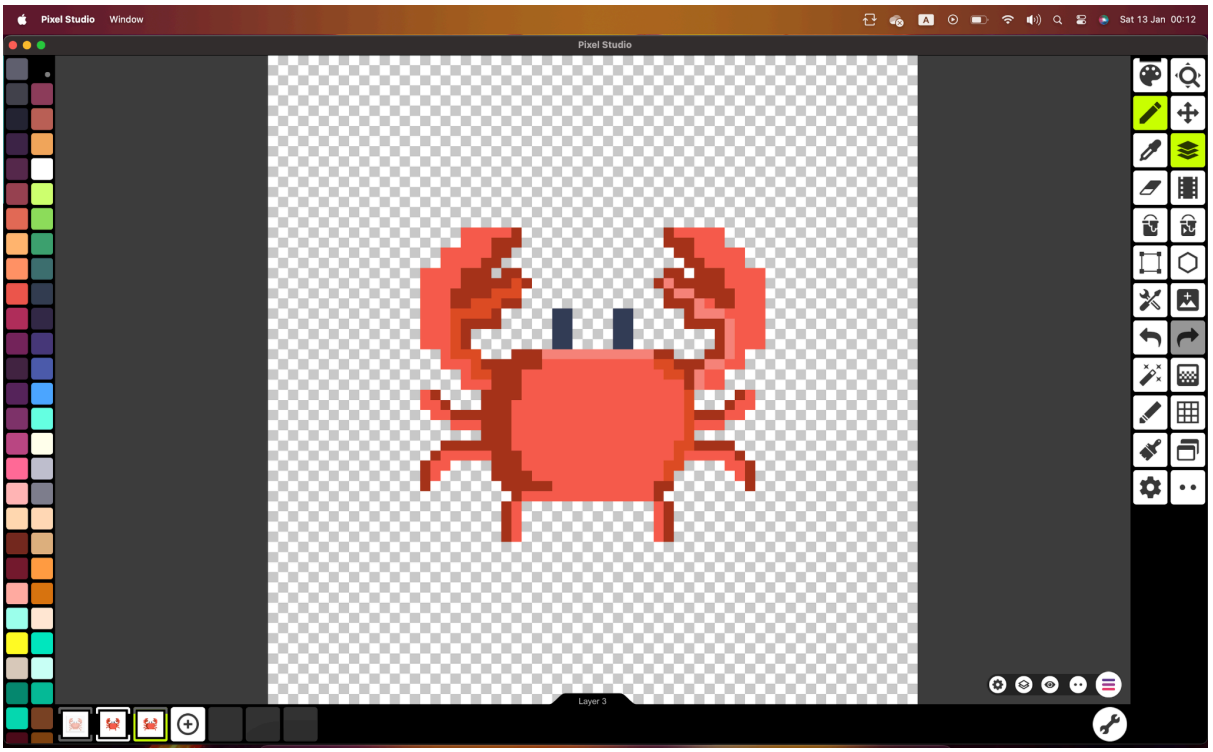


Figure 1.2. Pixel Studio

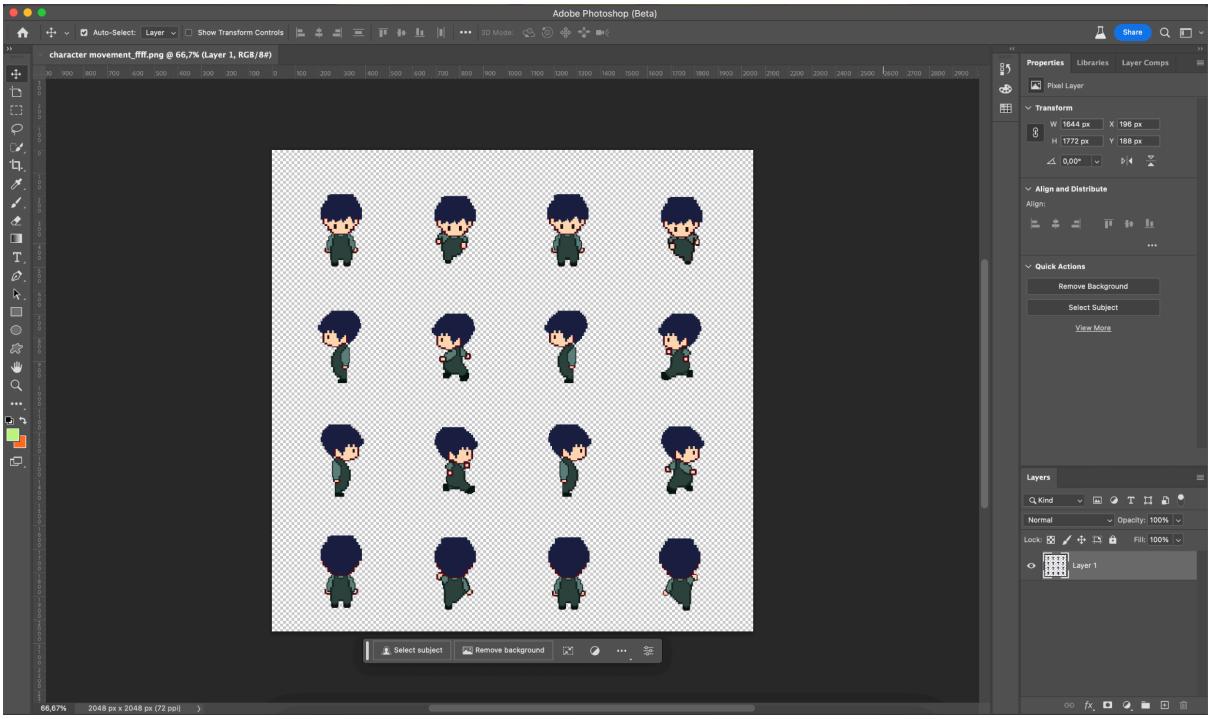


Figure 1.3. Adobe Photoshop

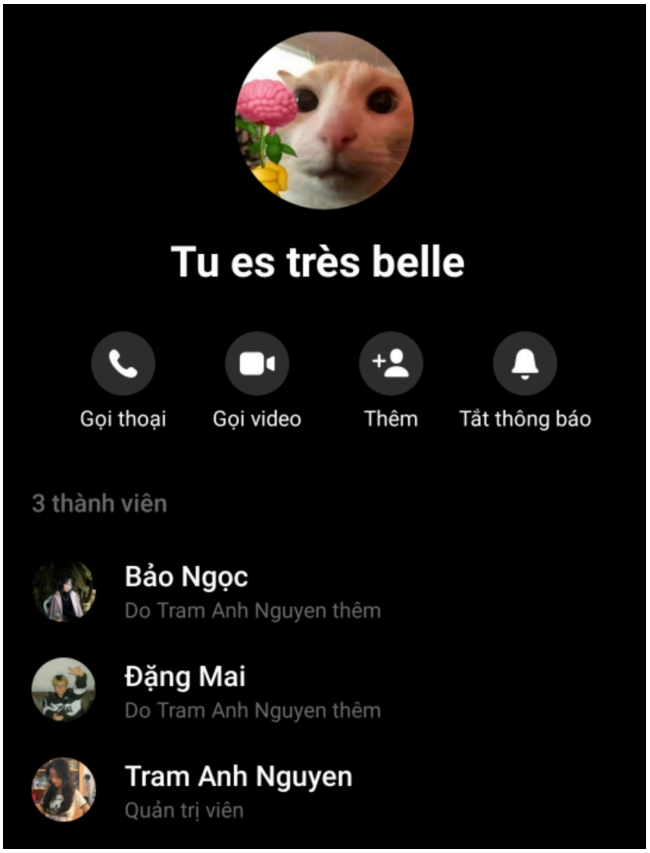


Figure 1.4. Messenger

Chapter 2: METHODOLOGY

A. Gameplay

Throughout this game, there is only one big map that has Tracking Camera Movement, “Susy Island” is divided into 2 parts: the forest and the beach.

The **gameplay** is straightforward: You (the character) are a fisher-man, you spawn at the forest (***Figure** 2.1*) on the map, you have 10 hearts (per one heart you have 2 life, the maximum life is 20). In the forest, the character faces risks that must be avoided:

- **Crab monster:** Whenever you encounter them, you will lose 1 heart. (***Figure** 2.2*)
- **Damage pits**: When moving you could fall into a damage pit, minus 1 life. (***Figure** 2.3*)
- **Poison pool:** In the game you have two pools, if you drink the poison water, you lose 2 hearts. (***Figure** 2.4*)

- Players can increase their life by drinking the healing pool water (***Figure** 2.5*). There is only 1 entry to the beach, players will receive a treasure box at the end of the road (***Figure** 2.5*). Players will achieve a rod (**Figure** 2.6), they will move to the NPC (*Figure 2.6*) to buy some fishing bait.
- At first, you (the fisher-man) will have 100 coins (default money) to buy baits. Throughout the gameplay, players can sell the fishes to make more money. If you sell your own rod, you cannot go fishing anymore.
- Make sure that you have to buy bait because if your inventory is empty you cannot go fishing (*Figure 2.7*). The game has 5 different fishes, including 4 normal fishes and 1 v.i.p fish. During the fishing state (*Figure 2.8*), we use the random function that includes all the fishes and trash so players have to face the situation that they will catch trash (*Figure 2.9*).
- In each catch, you will lose a bait that you bought, even when you catch a trash, when you run out of bait you cannot fishing (*Figure 2.7*). Per one time you catch a fish, you will lose a life, until you have one heart left, an alarm dialogue will appear to warn you(*Figure 2.10*). The playState has a timer, it's deviated into three states, day to dusk and gameover 10 seconds after the game switches to nightState. To win this game, players have to catch 5 different fishes (including the v.i.p) before the night state. You can win this game by fishing all kinds of fishes, including the v.i.p.

After that, the gameDone state will pop up (*Figure 2.11*) with the names of all members of this project.

Figure 2.1

Figure 2.2

Figure 2.3

Figure 2.4

Figure 2.5

Figure 2.6

Figure 2.6

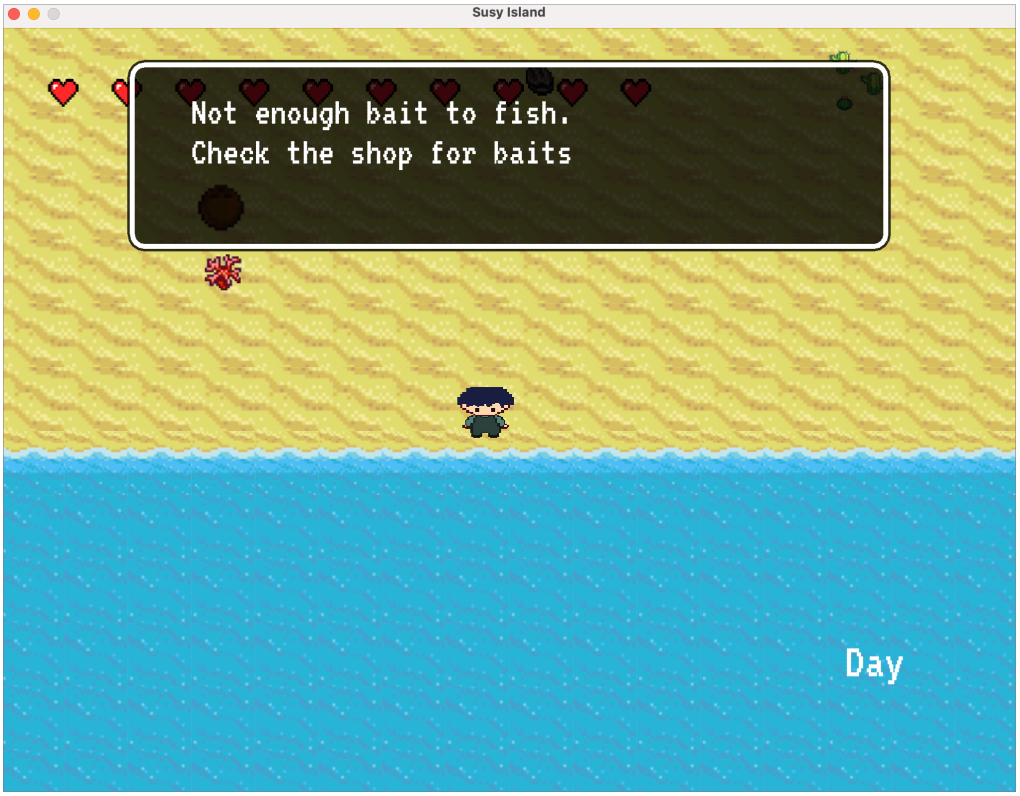


Figure 2.7

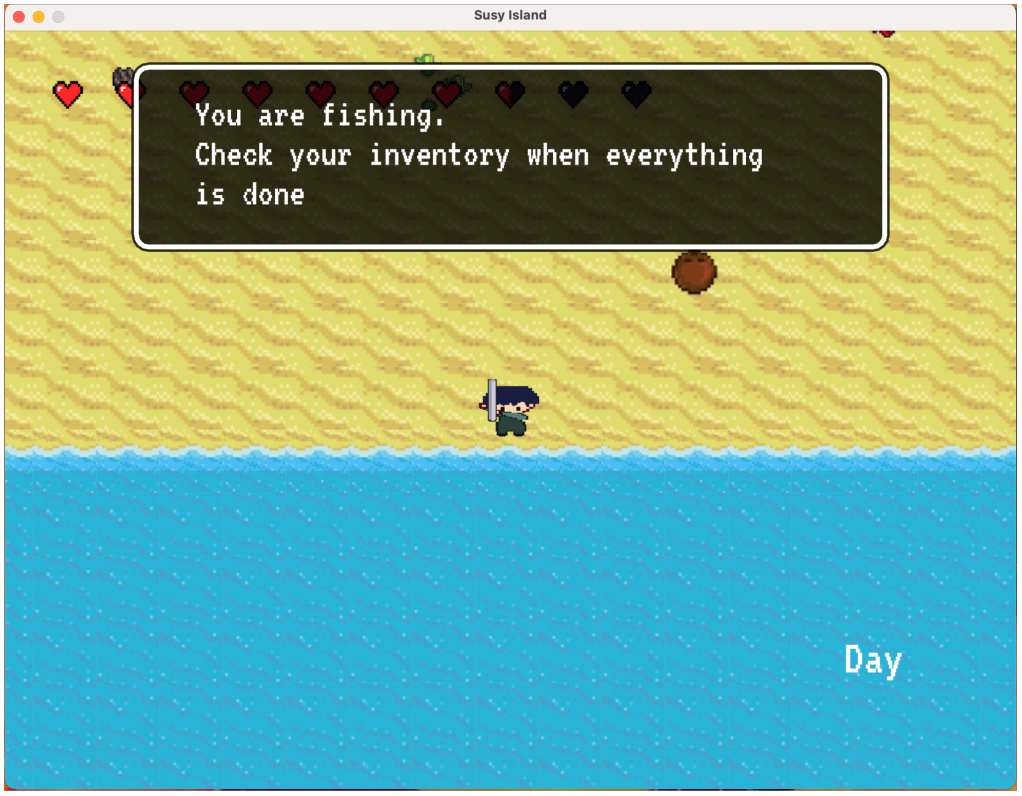


Figure 2.8

Figure 2.9

Figure 2.10

Figure 2.11

B. Design

- In this game, we decided to draw all the objects and tiles. Because while researching the resources on the internet we cannot find all of the images that match our game concept.
- At first, we wanted to have 2 different characters, one female and one male, for the player to choose.

- After that we redesign and choose only the male character to be our main character in the game. We want to make the character move smoothly so we draw frame by frame and control the movements by key inputs.

- For some objects like coin, bait,rob, fishes, coconut, trash. We scan and trace the art originally from the game Stardew valley plus from freepik. And the treasure box, we drew it.

The treasure box

4 normal fishes

the v.i.p fish

Trash

bait

coconut

fishing rod

coin

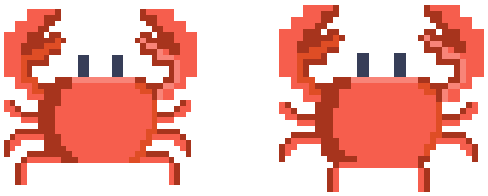
- All of the tiles that we draw, for the toxic pool and the healing pool, we draw all of the corners with rock in different ways.

- The original shop was a house but we faced problems with scaling the image into a single tile. We later designed a smaller shop to fit it into a tile size.

- For tiles we want to make the scene change smoothly so we design 6 different tiles: tree, grass, grass with sand, sand, sand with water, water. And we have

- In the sand tile, we have 4 different sand types to make the game more joyful and eye-catching.

- To make the crab monster move, we design 2 different images, and loop it constantly.



- The NPC we draw is based on a cookie run character. And because we make our NPC stay still so we just only need to design 1 image only.

C. UML Diagram

In our game, the classes in each package play quite separate roles from each other. (Figure 2.12) Package Controls plays a very important role, because it contains classes that can draw images of objects. Make the game character move, call the event to take place. The Collision is implemented and events such as touching a crab equal to losing blood, or tradingState maintain random methods in fishing events all take place in the Control package. (Figure 2.14)

Entities such as Player, NPC, Crab mostly share the same methods, so we will extend them from the Entity class (Figure 2.13). For all the objects in our game, we are using a method that we did in the Control package (class UtilityTool) to optimize the render image and in other objects class will use that method to render the image faster and we loop it in the GamePanel class.

In the Class TileManager we create a method that draws it by importing map data (in txt file). Then we use a loadMap method to scan and draw the numbers that we assigned for each tile in the map data. (Figure 2.16).

The transition from day to night as well as the control of lighting effects is handled within EnvironmentManager class and Lighting class. In order to manage the dayState and lighting effects, we create an abstract class EnvironmentSystem and class EnvironmentManager is extended from it (Figure 2.15)

Finally, this class help us to run the game (Figure 2.18)

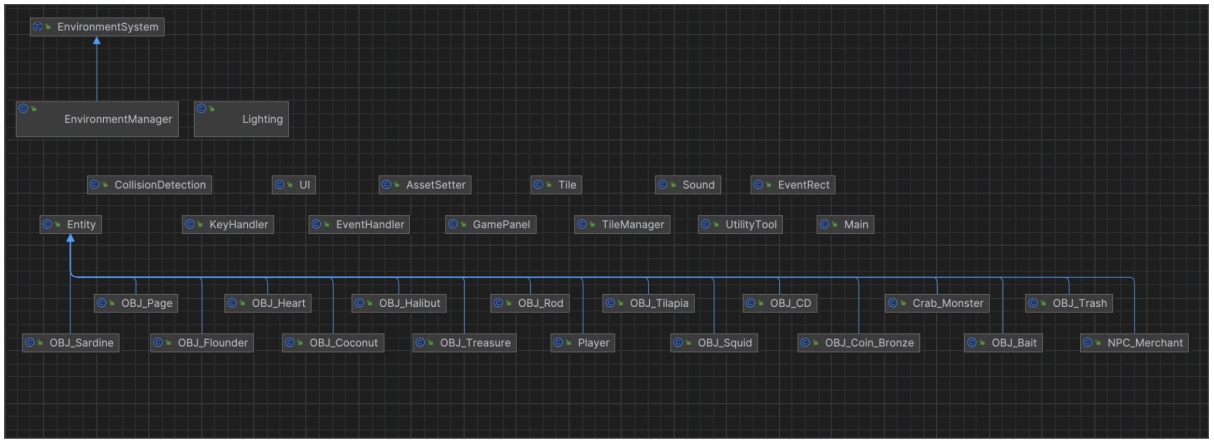


Figure 2.12 The overall

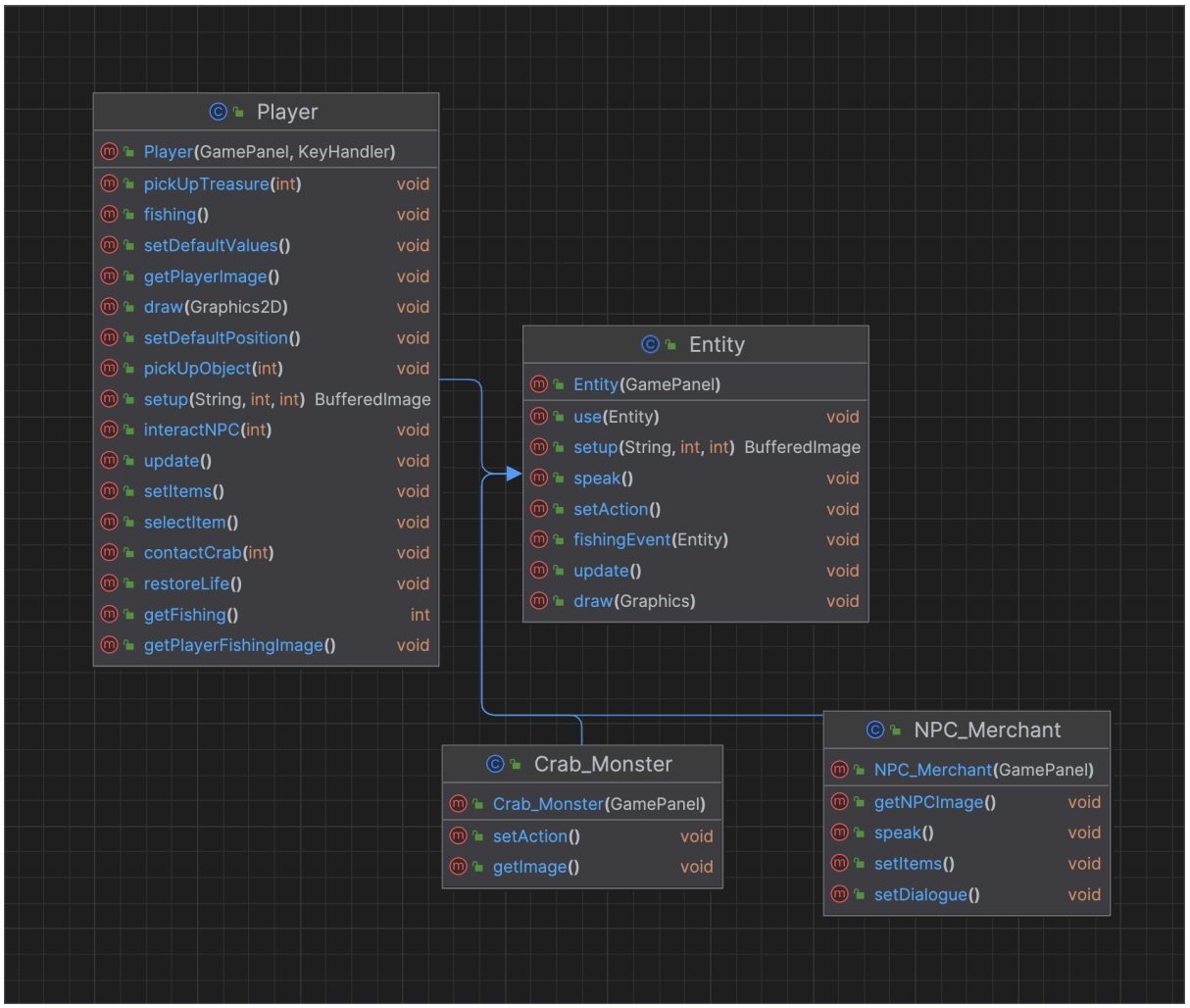


Figure 2.13 Entity package

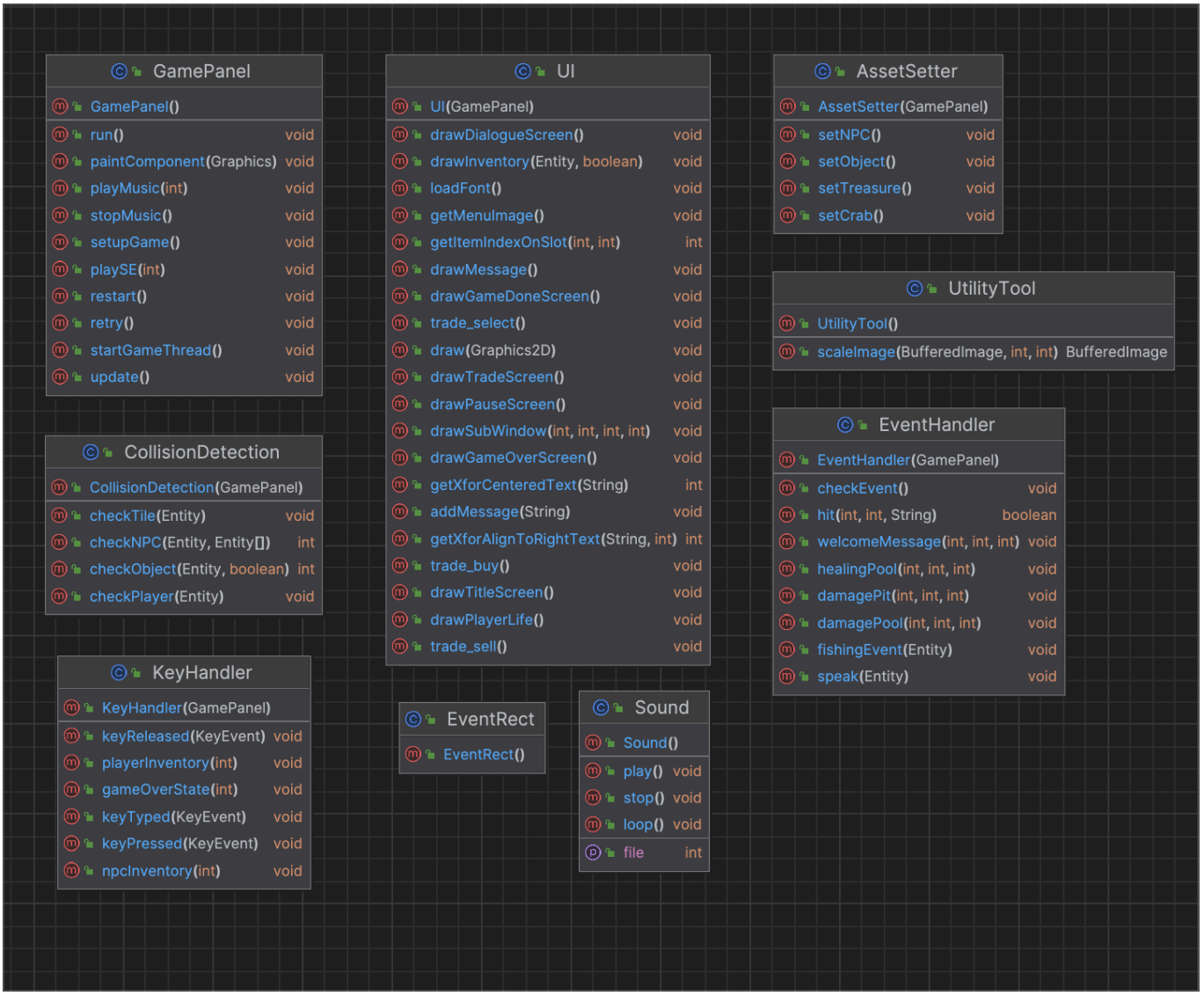


Figure 2.14: package Control

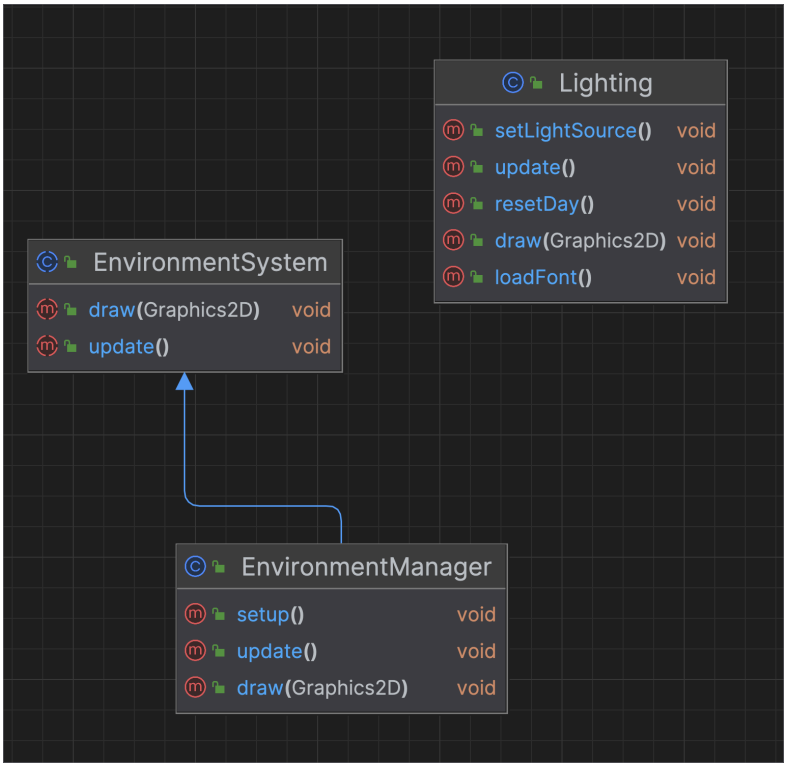


Figure 2.15 package Environment

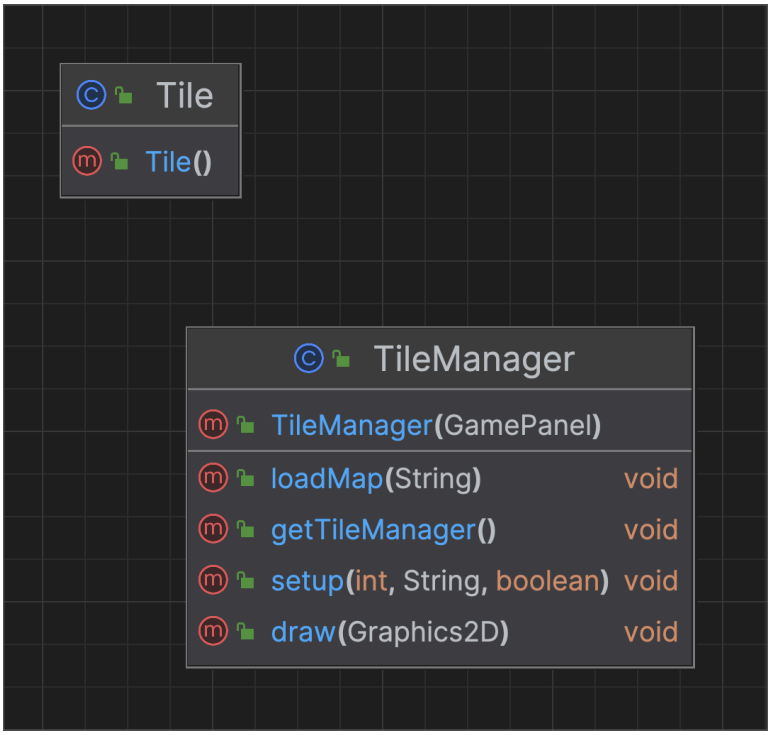


Figure 2.16 package Tile

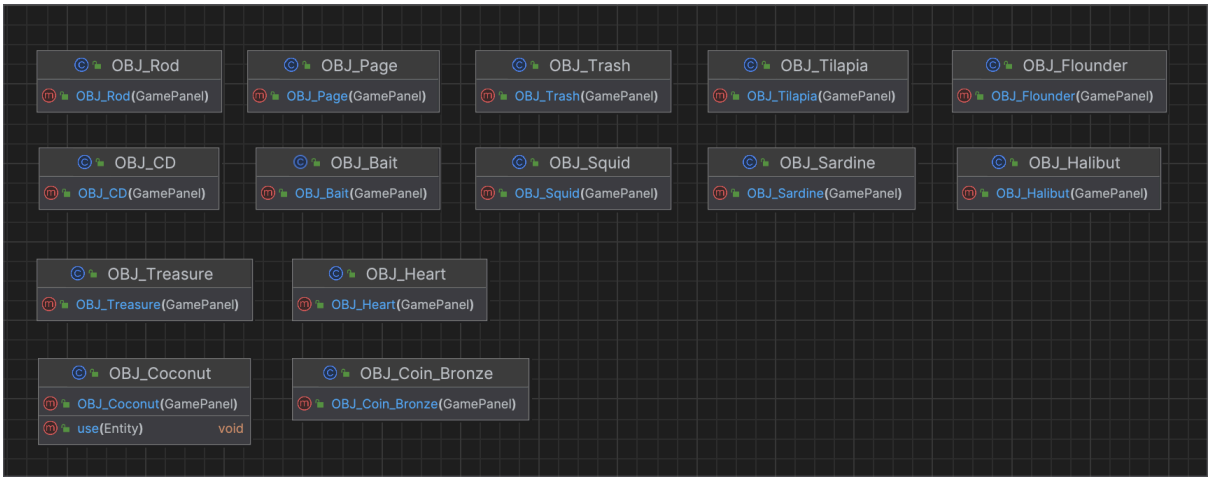


Figure 2.17 package Objects

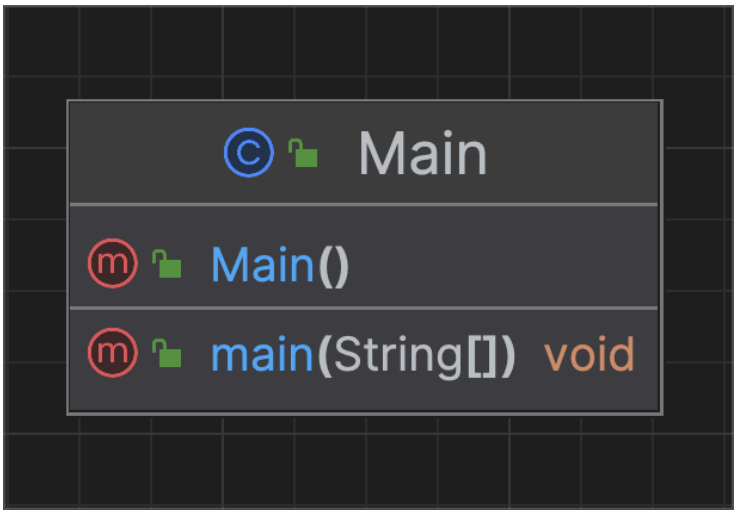


Figure 2.18 Class Main

Chapter 3: DEMO - RESULT

During developing the game, we made some adjustments where we switched from fishing only to adventure and fishing. We also added more features in order to make the game more adventurous and changed the design of the character.

Figure 3.1 : The initial design of two characters

We also extended the size of the map to 50x50, where the initial one was 20x20 and the forest was not included.

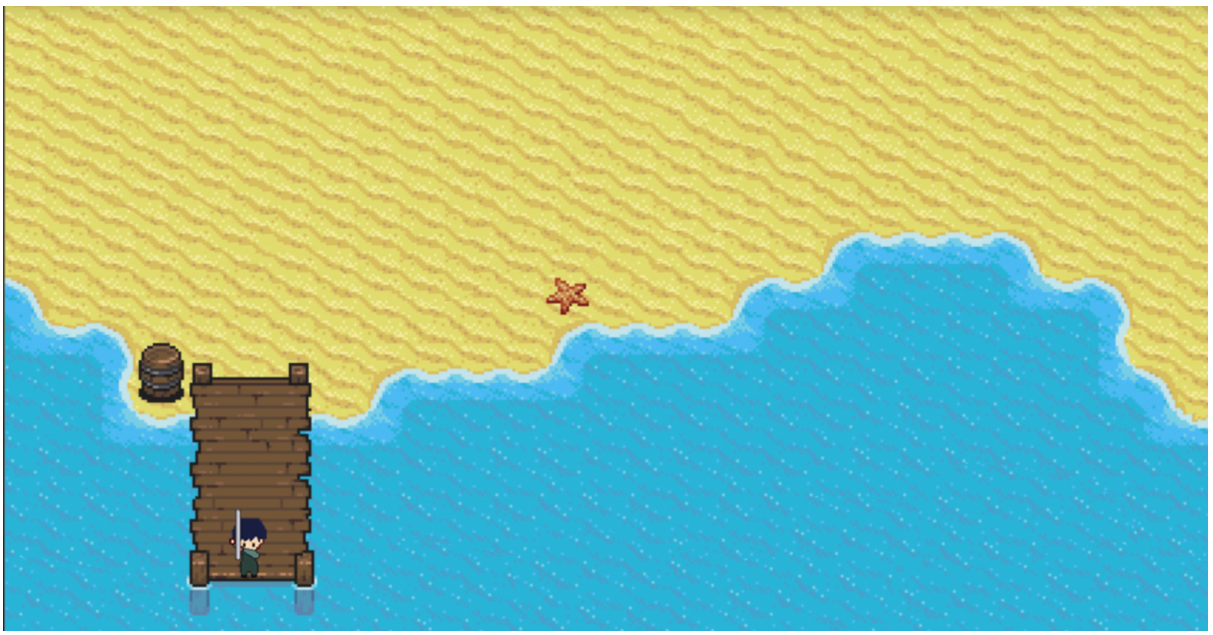


Figure 3.2: The initial map designed by photoshop

As our plan was to develop a fishing mechanism inspired by fishing minigame in famous farming game - Stardew Valley, we wanted to develop a similar fishing mechanism (**Figure 3.3**)

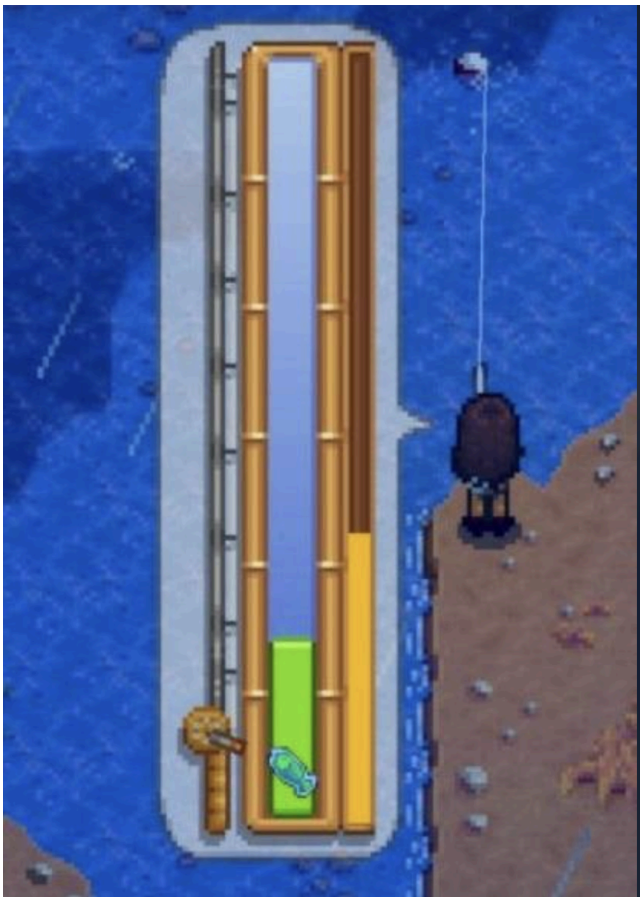


Figure 3.3: Fishing minigame in Stardew Valley

At this stage of development, we executed a 20x20 map and focused on fishing and trading only (**Figure 3.4**). We finished developing some basic features such as inventory, trading with NPC_Merchant, buying and selling system. Later on, we added the player's life, pause screen and title screen to our game. (**Figure 3.5**)

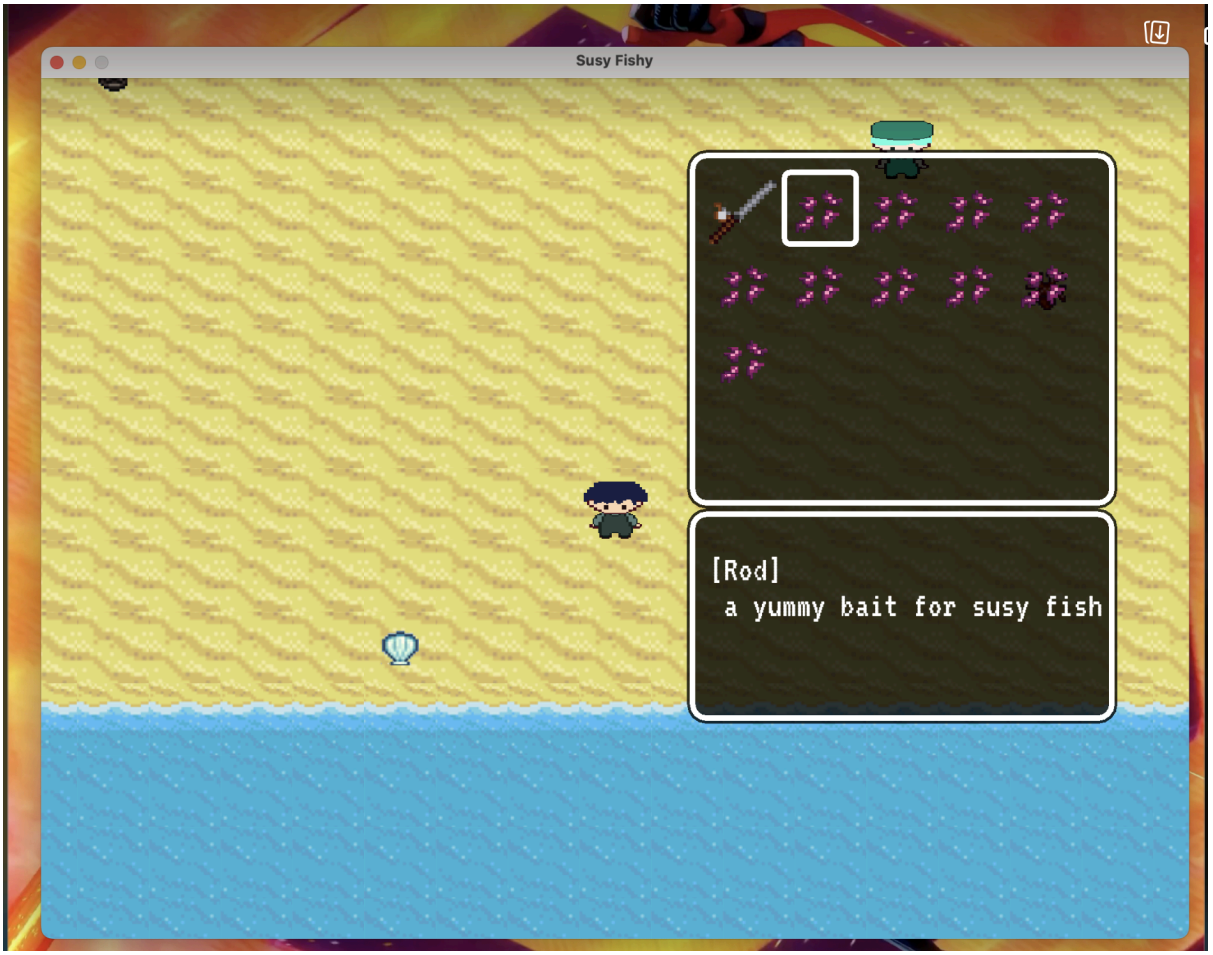


Figure 3.4: Map layout before extension

Figure 3.5.1: The main title screen

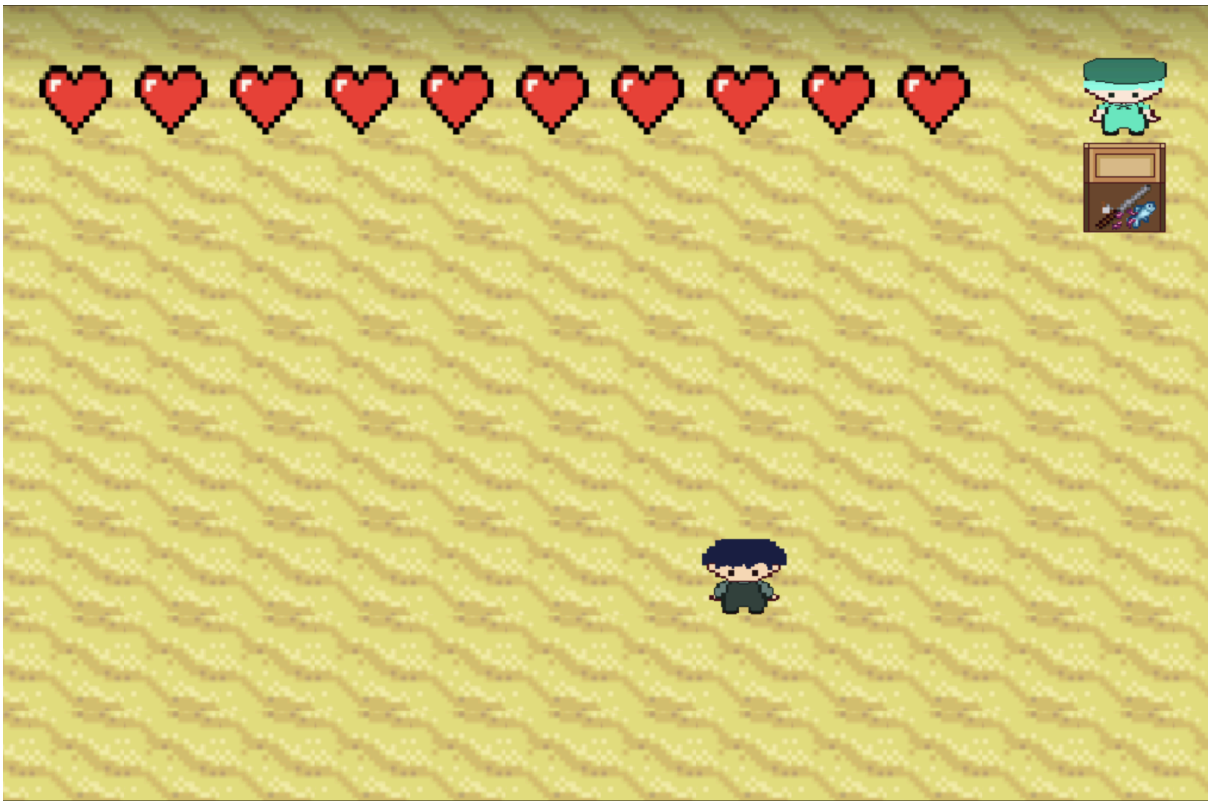


Figure 3.5.2: Player's life included

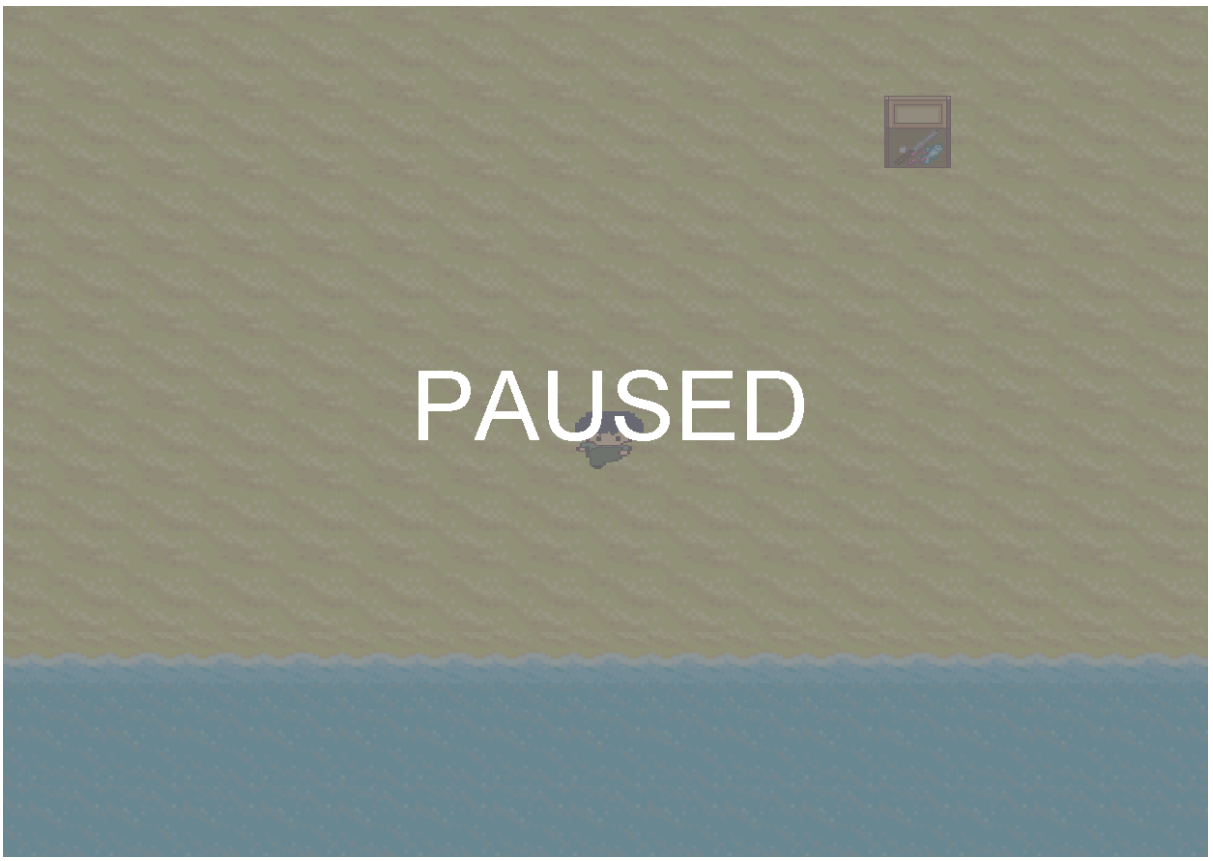


Figure 3.5.3: The first pause screen

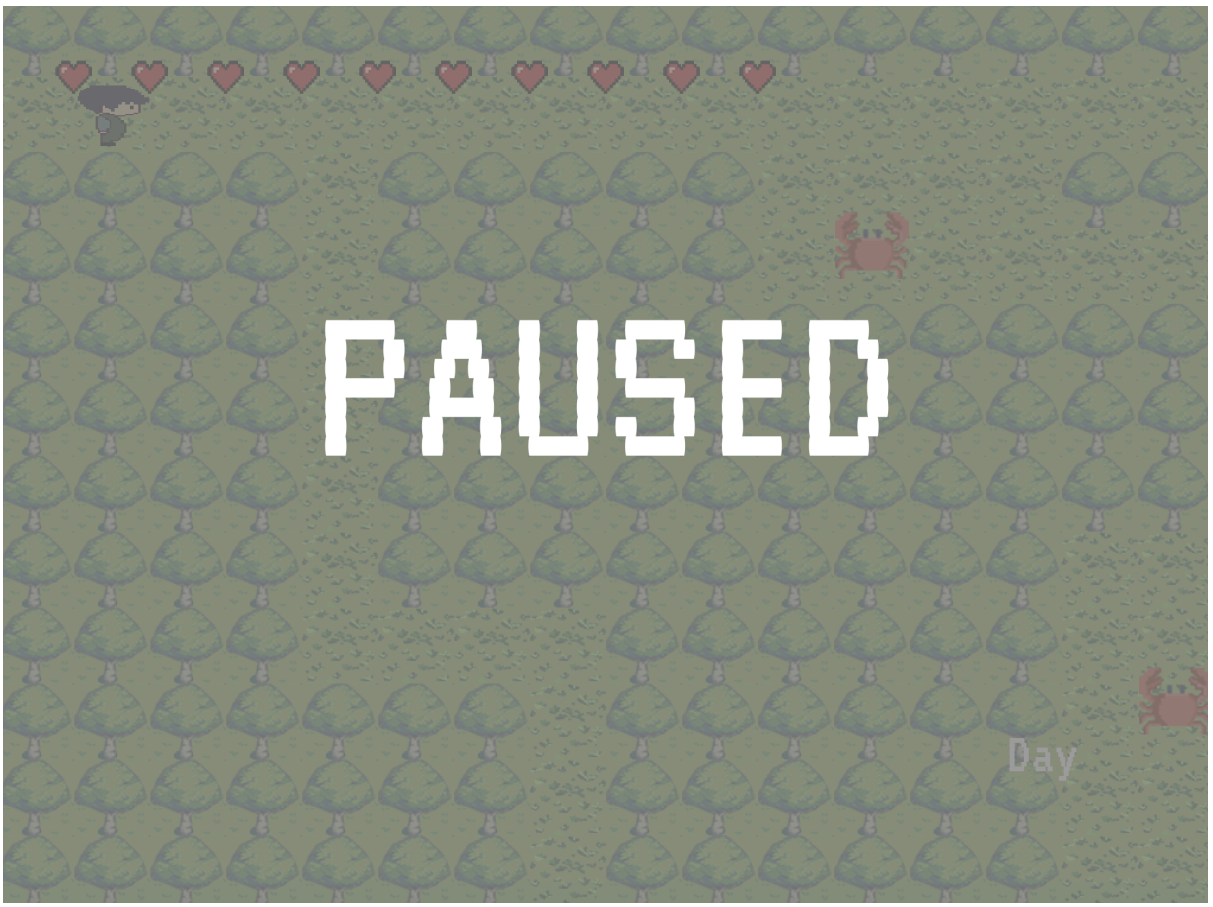


Figure 3.5.4: The current pause screen

In our latest adjustment, we expanded the size of the map to 50x50 and divided it into two parts: the forest (**Figure 3.6**) and the beach (**Figure 3.7**).

Figure 3.6: The current forest

Figure 3.7: The current beach

We developed more features for the game such as Damage pit, Healing pool, Damage pool, Crab Monster, Coconut for healing, Trash and fix bugs existed whenever we made a change in our game. Although the process was harsh initially, we were still able to pull it off and completed the game as we desired.

Chapter 4: CONCLUSION

A. Summary:

In conclusion, the game project contains all four pillars of OOP, which prove the team's effort in both game development and the overall programming process after game completion. The project's classes have covered the concept of encapsulation. Entity is the base class from which Player, NPC, and Monster are extended, which illustrates the inheritance feature. Also, the EnvironmentSystem class is where abstraction has been used.

Due to the limited acknowledgment time, team members have been challenged to learn and implement new features into the code at the same time. Although further optimization is necessary for code fluency, our team has had an incredible opportunity to absorb new information, practice with real code, and learn about the complexities of project management and collaboration. This will be the experience that team members will not get a second chance to work on.

B. Shortcoming

Although we tried our best to cover all the knowledge acquired in the OOP course, we omitted to fully use the SOLID principles and Design Patterns in our project. The lack of optimization in our code is still an issue, which causes it to be quite bulky and affects the game experience. Moreover, the fishing process gets laggy when we try to press ENTER key multiple times and the in-game sound does not disappear when the GAME OVER screen displays if the player’s life is deducted to 0.

C. Future Works

In the future our team wants to aim at optimizing the algorithms in our fishing game. Instead of using a random function, we will develop our fishing mechanism similar to the one in Stardew Valley. We also intend to add the male and female

character selection feature that we initially planned. Moreover, maybe our team will change and add a cooking selection to cook the fish caught into dishes to restore hearts. More features such as a house that the player can interact with, including another game state (the inside scenery of that house), where the player can cook and sleep to increase the number of hearts are also planned to include. Furthermore, we want to develop our game into a game that is similar to Stardew Valley.

REFERENCES

RyiSnow. (n.d.). *Home* [YouTube channel]. Retrieved Jan 12, 2023, from https://www.youtube.com/@RyiSnow

Abstract class in Java (2023) *GeeksforGeeks*. Available at: https://www.geeksforgeeks.org/abstract-classes-in-java/ (Accessed: 14 January 2024).