PROJECT PLANNING DOCUMENT

1. PROJECT TEAM

Team Member	Role	Responsibilities
Monica Torres	Project Developer	Builds the product itself with the mindset of completing the biweekly goals set during each Sprint.
Chairo Pela	Project Developer	Builds the product itself with the mindset of completing the biweekly goals set during each Sprint.
Jennifer Menjivar	Project Developer	Builds the product itself with the mindset of completing the biweekly goals set during each Sprint.
Aaron Youch	Scrum Master	Guides the Scrum Team and helps when issues arise. Manages the Scrum Team in order to keep the workflow organized and productive.
Joseph Jose	Project Developer	Builds the product itself with the mindset of completing the biweekly goals set during each Sprint.
Gregory Zacharko	Product Owner	Mainly responsible for the organization and releases of Product Backlog Items via Trello. Represents stakeholders and ensures that requirements of the product are met each sprint.

2. PROJECT INFORMATION AND SCOPE

1. Project/Product Name: Robo Rescue
2. Project Description and Scope
    - ***a. What is it?***

      The game is a 2D Platformer game where the user will be tasked with retrieving various pieces of missing computer parts through 4 different levels, where each level will reward them with one of the missing pieces upon completion. Then, in the fifth and final level, the user will need to put all of the computer parts into their proper places to fix the broken computer, like a puzzle. The game allows the user to have fun traversing and jumping through this world while also learning the main parts of a computer.

b. What does it do?

The game exposes children to modern technology in a friendly, educational environment without them realizing that they are learning. Having work disguised as a game captures children's attention more and they tend to have more fun and accomplish more work. The audience is children who are at the elementary school level, namely children through the ages of 5 and 8. This age range was chosen due to the fact that it is around these ages that a child is able to grasp elementary concepts in computing. Thus, this is the age range to start getting kids interested in STEM subjects.

c. Why is this project noteworthy?

This project will assist teachers, parents, and caretakers wanting to introduce their children to the world of computers and technology. It can serve as a fun medium while also serving as a learning tool for STEM related courses, and/or activities for kids. The fact that the product will be free also makes it accessible and convenient for wherever/whoever chooses to use it.

3. PROJECT RISKS, CONSTRAINTS, AND FEASIBILITY

1. Can the project be completed on time?

With approximately fifteen weeks in the course session, we plan to complete this project by the time we conduct our final presentation during finals week. We will work in biweekly Sprints, with Sprint reviews at the end of every Sprint. We will also keep updated our backlogs and Trello board to assist in our plans of completing the project by the end of the semester.

2. Does the project integrate with other existing systems? If so, explain.

The project readily integrates with the innumerable off-the-shelf game engines and readily available technologies used to construct interesting games. There are also widely available tutorials dedicated to the production of games and their inner systems, as well as ensuring that they are engaging to play.

3. Are there any limitations to the success of the project?

It should be noted that most schools and younger children are often restricted to low-end hardware or mobile devices. Ideally, the project should be able to run on minimal hardware so it is accessible to wider audiences.

4. How would you rank risks to the project?

The largest risks would be the less related fields used in game making such as graphic design and sound production. While many of the related skills overlap with CS, they also have many unrelated and more artistic parts. Another large risk would be the psychological aspect of producing a game targeted at children and the related difficulties. Making a game is “easy”, making a “fun” game involves much more psychology and is significantly more difficult, and attempting to make an educational game, especially one intended for children, that has to include more than just “fun” is even harder.

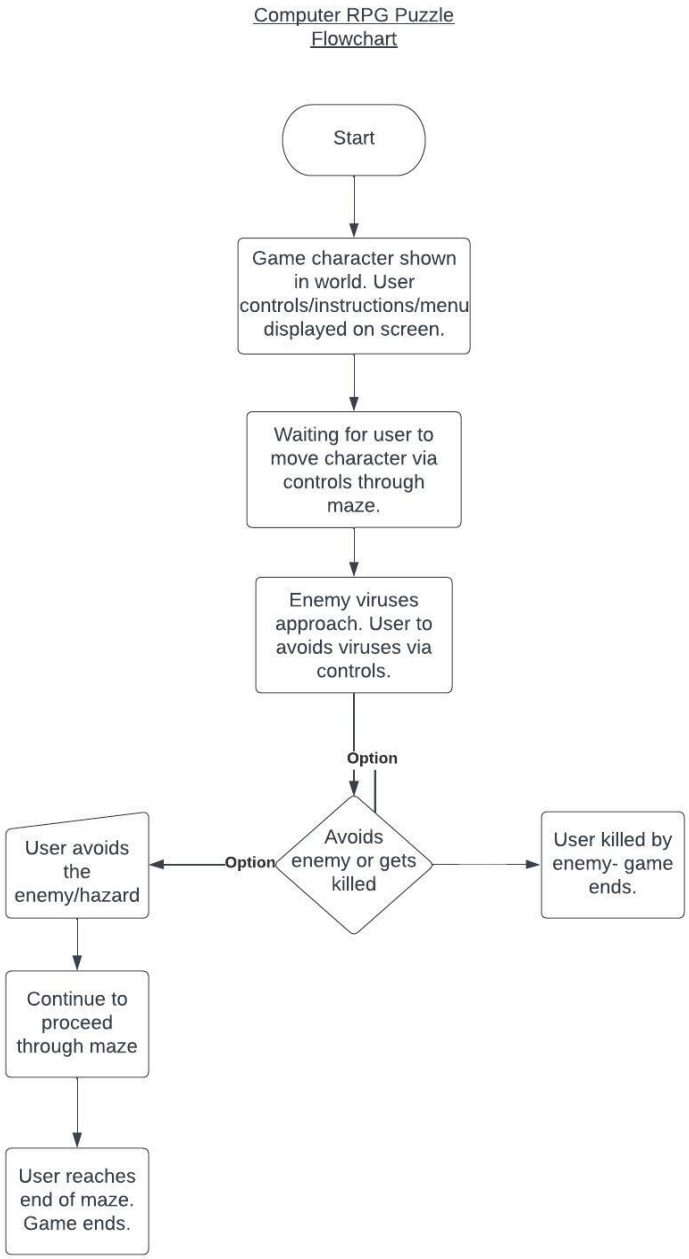
5. ***Are there any technical unknowns?***

As of now, there are no technical unknowns discovered. The product will be free and we are using a free game engine to design our product. If we encounter any problems while designing we will make note of it, but currently, we are foreseeing a generally smooth process.

6. ***Do you foresee any single point of failures?***

Currently no, we do not anticipate any failures in the making or designing of this project.

4. VISUALIZE THE PRODUCT: MAP THE PROJECT ARCHITECTURE



Project/Product Name: Robo Rescue

5. PROJECT METHODOLOGY

How will the team track and document progress, changes, and setbacks?

Throughout this semester, our group will be following scrum methodology in our project planning. Because of this, our team consists of a scrum master, a product owner, and developers. Scrum events that we have integrated into our production include 2-week-long sprints where we work towards completing the product backlog items specified for that sprint, sprint reviews at the end of the sprint where we present our progress over the last sprint, and weekly scrum meetings to address our progress. Our process includes the use of scrum artifacts like the product backlog item list to address the items that we need to accomplish overall and the sprint backlog that addresses what is slated to be done during a particular sprint.

6. PROJECT LANGUAGE AND TOOLS: ARCHITECTURE REQUIREMENTS

1. ***What are the development and production environments?***

   Visual Studio Code (VS Code) and Unity will serve as our development environment, and a web-host platform such as itch.io will be used as our production environment to deliver our final product.

2. ***List product requirements***

   a. **Hardware, specifications, and locations**

      Hardware minimum requirements:

      - Windows: Windows 7
      - Mac OS: Sierra 10.12.6+

      CPU minimum requirements: Intel Core i5

   b. **Inputs and outputs**

      Input Devices:

      - Keyboard

      Output Devices:

      - Monitor

   c. **Input and output devices**

      Input Devices:

      - Keyboard

      Output Devices:

      - Monitor

   d. **Servers with descriptions**

      Not applicable

   e. **Database requirements**

      Not applicable

   f. **Hosting**

      Web hosting on the itch.io platform

g. Development and production environments

Game Design:

- Unity
- VS Code for coding game

h. Interfaces or integrations with other hardware

Not applicable

i. Software requirements

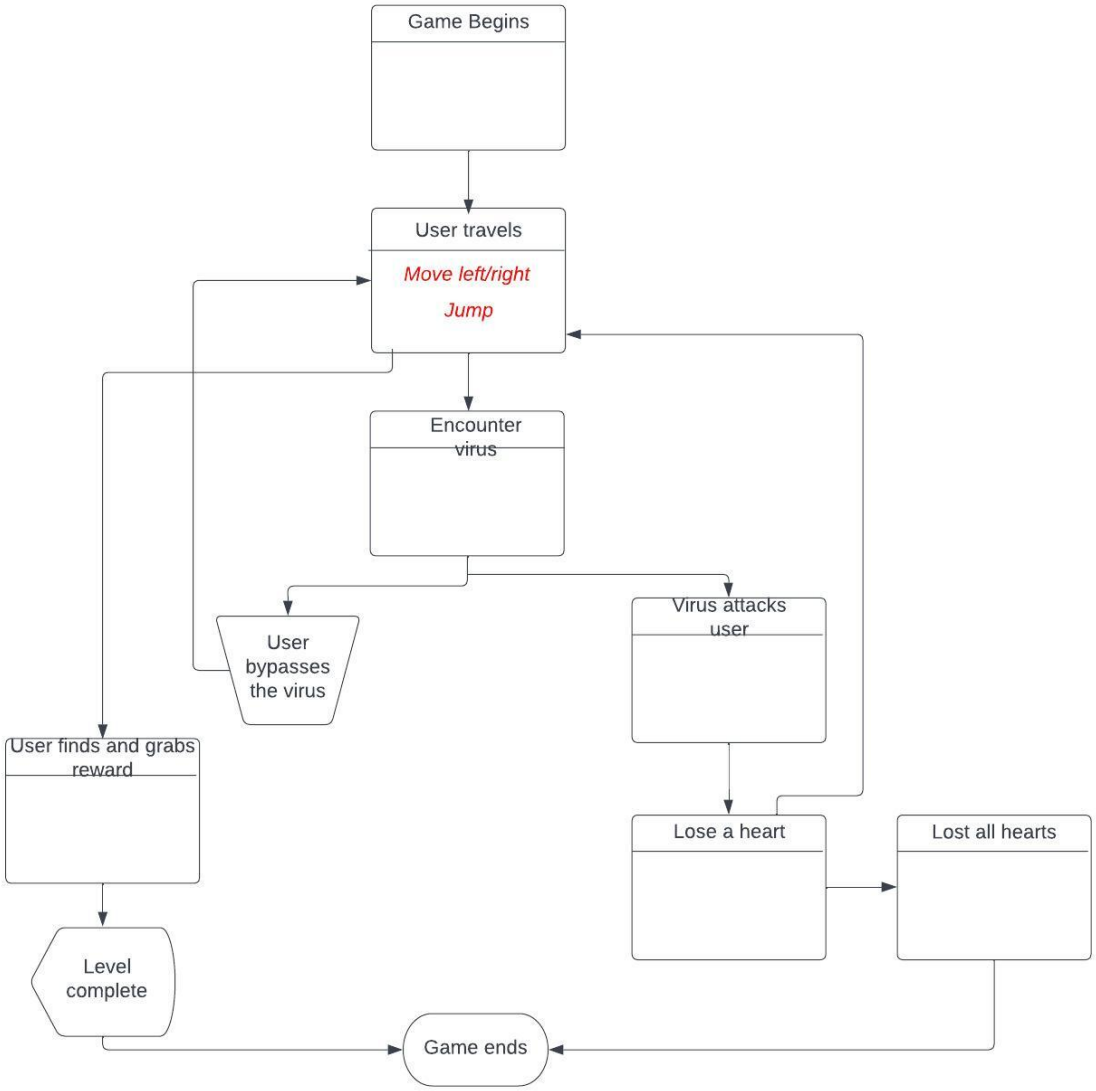
- Unity Hub
- VS Code

j. Capacity, space requirements

Disk space minimum:

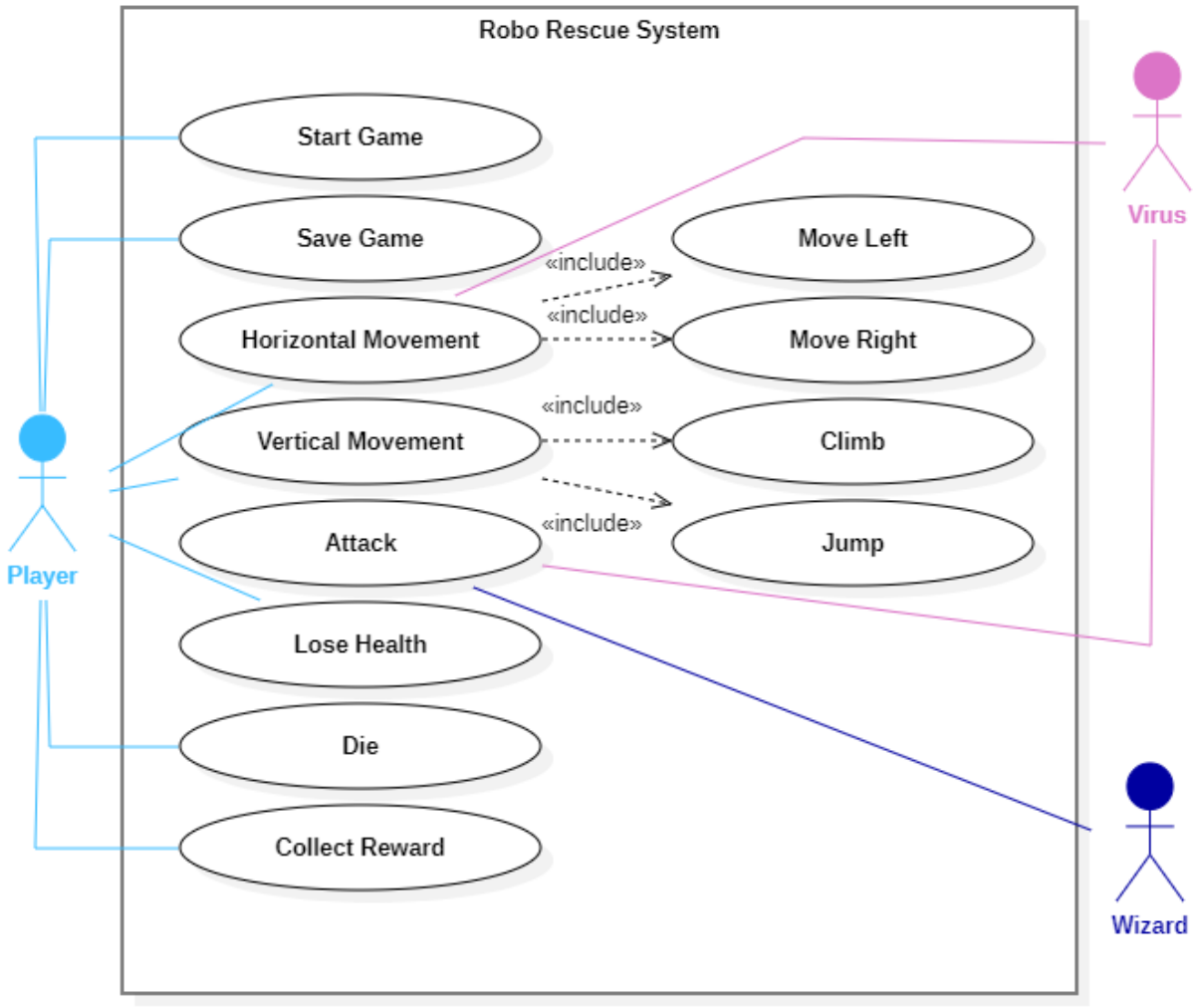
- 80GB

7. PRODUCT FEATURES AND ENTITIES



Project/Product Name: Robo Rescue

8. MAP THE PROJECT ARCHITECTURE: USE CASE DIAGRAM



9. LIST ALL THE HIGH-LEVEL FUNCTIONS

1. ***List all High-Level Functions Based on the Use Case Diagram***

Start (*Player*):

- The player is capable of starting the game and initiating gameplay

Save Game (*Player*):

- The player is able to save the game to their last completed level

Horizontal Movement (*Player and Virus*):

- Both the player and viruses are able to move forward and backwards

Vertical Movement (*Player*):

- The player is able to jump around the level, onto platforms, and over viruses and other enemies and hazards to avoid them

Attack (*Virus and Wizard*):

- Viruses and Wizards are able to attack the player and causes the player to lose health

Lose Health (*Player*):

- The player is capable of losing health if they were successfully attacked by a virus or wizard

Die (*Player*):

- The player is capable of dying once their health has depleted fully

Collect Reward (*Player*):

- The player can collect a reward (i.e. a piece of the missing computer) after finding it within each level

10. LIST ALL USER INTERACTIONS

1. ***List User Inputs (text fields, lists, etc.)***

User Inputs:

- Not applicable

2. ***List User Interactions (buttons, mouse/touch interactions, drop-down selections, etc.)***

User Interactions:

Buttons:

- **A Key:** Player is able to travel to the left
- **D Key:** Player is able to travel to the right
- **W Key:** Player is able to jump

11. DATA MANAGEMENT

Unity, the game engine we adopted to create our game, utilizes a JsonUtility class for serializing and deserializing JSON files. So we will be working with the JsonUtility class, which manages JSON files. Unity utilizes JSON so that it can keep track of levels and collected items of the game.

12. PROJECT DEVELOPMENT

Weekly updates on project status.

13. HOW IS SUCCESS DEFINED?

To us, success is achieved when the game has been created with all the desired levels and gameplay mechanics in it, has been tested by children who are within our target audience and age range for the game, and when the game has passed all its tests and trials.

The metrics that measure success are:

- The number of levels in the game
- The smoothness of the gameplay
- Did children enjoy playing the game (simple yes/no question)
- Did children learn something brand new that they didn't know before playing the game (yes/no)

The process/plan that ensures product quality is unit testing and confirming that the code runs smoothly without any errors.

14. CODE AND PRODUCT TESTING

1. Unity's compiler builds without warnings or errors being observed including in runtime.
2. Code is peer reviewed.
3. Code is documented; classes and functions have comments detailing their use.
4. Functionality testing was conducted in the form of continuous play testing.
5. Unit testing is accomplished using Unity Testing Framework.
6. Tested on a range of machines with various performance capabilities and operating systems, including Windows 10 and Mac OS.

15. FINAL PRODUCT SPRINT REVIEW