Gate:Crash

Giovanni E. Martinez

CST-451 Capstone Project Final Architecture & Design

Grand Canyon University

Instructor: Professor Mark Reha

Revision: 1.0.0

Date: 11/17/2022

ABSTRACT

The current state of the video game market is overly saturated with two major genres of games. Those two genres are the open world role-playing game, and the battle royale game. This does not mean that other styles of games cannot thrive in the current market, however due to the lack of differentiation in the modern game market, games that remove themselves from this model can garner more attention and possible success. That is the space that this project aims to fill.

The goal of this project is to introduce a new competitor to the current video game market, through the 2D objective based exploration game. The design philosophy for the game environment and layout will take inspiration from games in the genre such as Metroid, where the player is dropped in a large space that they will be tasked with navigating. The core gameplay loop of the game will consist of selecting a loadout and objectives from a central hub area and proceeding to the larger environment, or map, where the player will have to complete those objectives and return safely by extracting from the map to collect their rewards. The main map of the game will be random every-time that the player initiates a new instance. The game will also incorporate an escalating combat system that will force the player to take more precaution during their gameplay, to avoid being killed quickly and losing their progress.

History and Signoff Sheet

Change Record

Date	Author	Revision Notes
		Initial draft for review/discussion

Overall Instructor Feedback/Comments

Integrated Instructor Feedback into Project Documentation

☐ Yes ☐ No

TABLE OF CONTENTS

Design Introduction

    This game will make use of a 2D exploration game formula and introduce unique systems that allow it to stand as an original concept for a game within the genre of 2D style games. The game will make use of systems that are inherently unpredictable such as the randomly generated maps and objective locations, and the endlessly spawning enemies, as well as player-controlled features such as the ability for the player to create their own loadout and choose their objectives, and the ability to navigate the map in whatever fashion they choose. This combination of systems will hopefully allow for the player to remain engaged with the gameplay and story due to its replay-able format, while giving the player the sense of progression and completion as it allows the player to make use of the rewards and experience from each match to develop and customize their character.

Deliverable Acceptance Log					
ID	Deliverable Description	Comments	Evaluator (internal or external as applicable)	Status	Date of Decision
1					
2					
3					
4					
5					

Detailed High-Level Solution Design

| Proof of Concepts		
Description	**Rationale**	**Results**
1. Consruction of both a 2D and isometric environment with a player and movement.	This was to test what the most feasible approach to this project would be.	An isometric view of a 2D environment proved to provide a lot more narrative and technical freedom, and thus shall be adopted.
2. Experimentation with varying chunk sizes for the tiles that make up the map.	The purpose of this was to verify what the best size for each tile would be to best express the scale of the world relative to the player character.	The best approach would be to create 64x64 or 72x72 tile sizes, to allow for a larger since of scale to fit assets, while not overstepping design boundaries.
3. Designing enemy AI to track a player.	This was to experiment with the best approach to an enemy tracking system would be.	The game would be best to have an enemy AI that will wander around its respective chunk of the map and will seek out players within its chunk.
4. Enemy spawn logic.	The purpose of this was to experiment with the best way of introducing new enemies to the map in a reasonable way.	The game will systematically introduce more enemies into parts of the map based on where the player is.

Hardware and Software Technologies
1. Xbox controller/Gamepad, Keyboard and Mouse
2. Any device that can install Steam and Steam games. (min requirement of 8Gb of ram, Nvidia 1060 equivalent GPU, Intel i7 equivalent CPU)
3. Unity Real-Time Development Platform (version 2021.3.10f1)
4. Riptide Networking Multiplayer Framework (Out of Scope) (version 2.0.0)
5. C# (version 10.0)
6. Visual Studio 2022 (version 17.3.3)
7. Aseprite (version 1.2.31)
8. Studio One 6 or 5

Logical Solution Design:
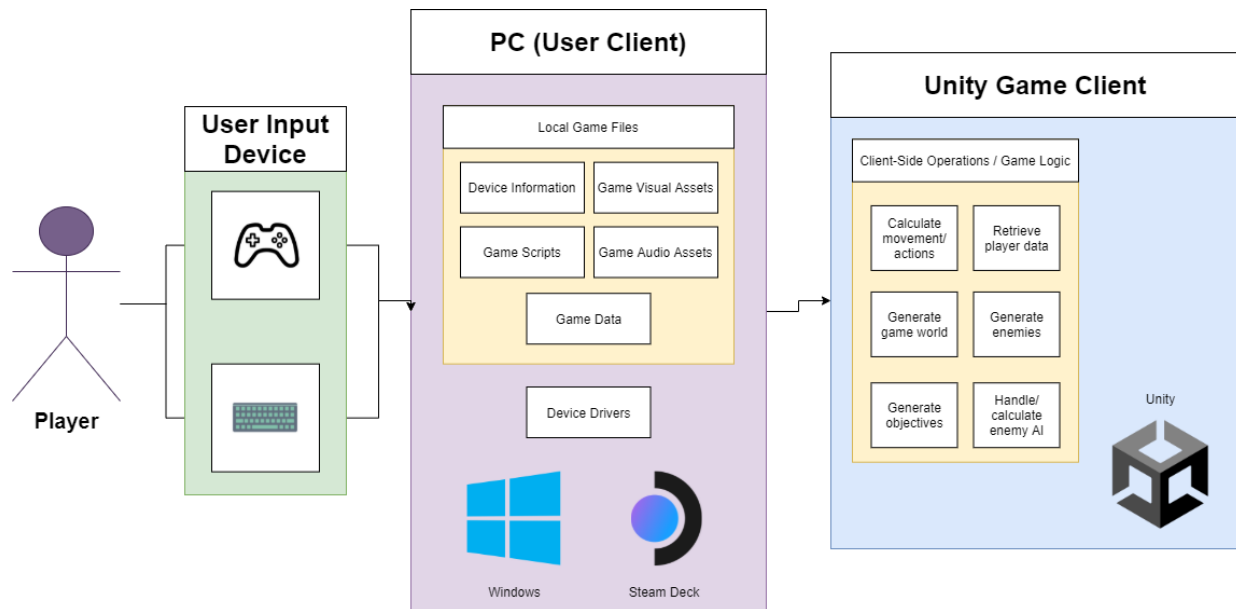
Version numbers for the different software and OS are on the physical solution design. Device input and drivers are processed through the machine OS before being processed by Unity.

Physical Solution Design:
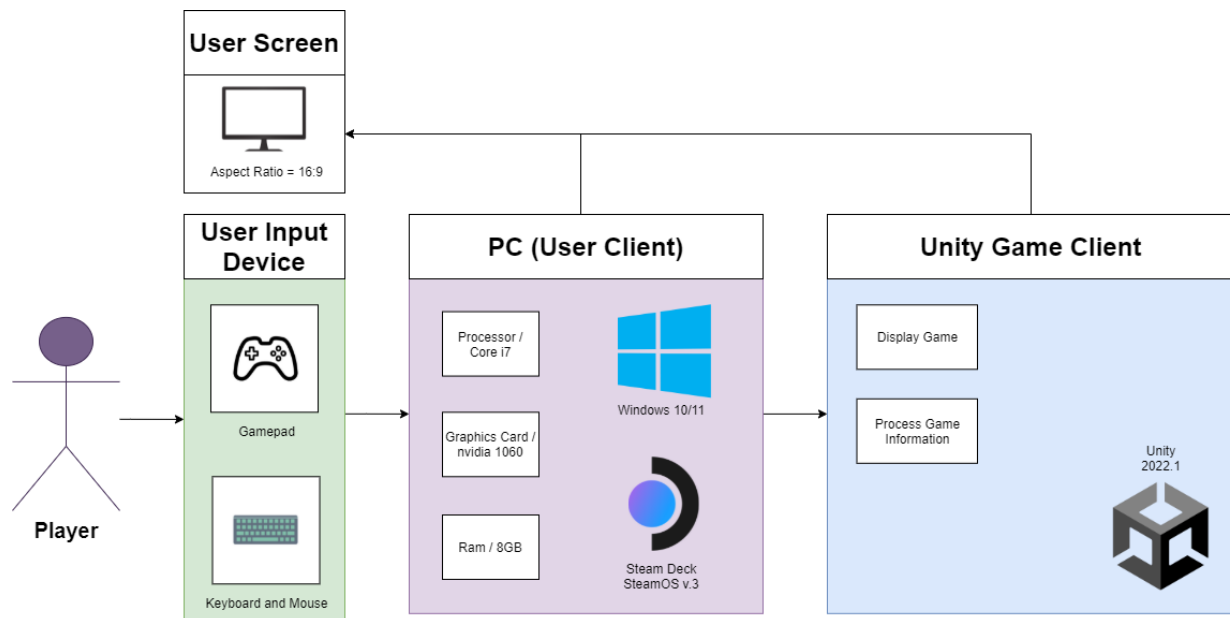
The physical solution design does not include a mention of visual aspect-ratio or resolution preferences. This is because the goal for the game is to be deployed using the custom screen resolution and resizing framework that should allow the user to resize the game window or full screen the game in any variation of a 16:9 aspect ratio.

Additionally, the gamepad to be used for the game is only limited by the different gamepads/controllers that are compatible with a standard Windows 10/11 machine.

7

Detailed Technical Design

General Technical Approach:

The approach that will be taken for the development of this project will be an iterative prototyping cycle that will allow the development team to create portions of the game in smaller development cycles. This will allow the development team to use larger portions of time on integrating each feature as it is developed, making the overall development process simpler and ideally quicker. The development of Gate:Crash will also involve small cycles where the development team will meet with volunteer testers in order to garner live feedback on features of the game, in order to know what will need to be changed or replaced to make the player experience better. The development of the game will begin through the application of what was learned through the development of the proof of concepts completed prior. The development team will also need to apply the assets of the game that are completed, to the currently developed elements and features of the game, to verify that the styles and assets fit with the game theme and narrative. Through planning sessions, a question arose as to whether it would be best to allow the player to access the entirety of the map as one big, rendered world, or if it would be best to divide the map into sections that are rendered or unrendered as the player explores each section. Development is planned to continue using the first ideology.

Key Technical Design Decisions:

- Gate:Crash will be developed in Unity, as it is the most widely supported game development engine. Its purpose is to provide the foundation and frameworks for the systems in the game.
- Visual Studio will be the IDE of choice for the development of this project, as it provides the most support for C# programming, and has easy integration to Unity.
- C# will be the language used for the development of the scripts for the game, due to the fact that Unity supports development in both C# and C++, and the development team has the most experience in C#.

- Aseprite will be the program used for the development of the visual assets of the game, as it is solely focused on the creation of 2D pixel art assets, allowing for an easy entry level design philosophy for the game.

- Studio One and Audacity will be the software used for the development of the audio assets of the game, including the music and sound effects. These were chosen, as they are the easiest to access programs for the development team for recording and sounds.
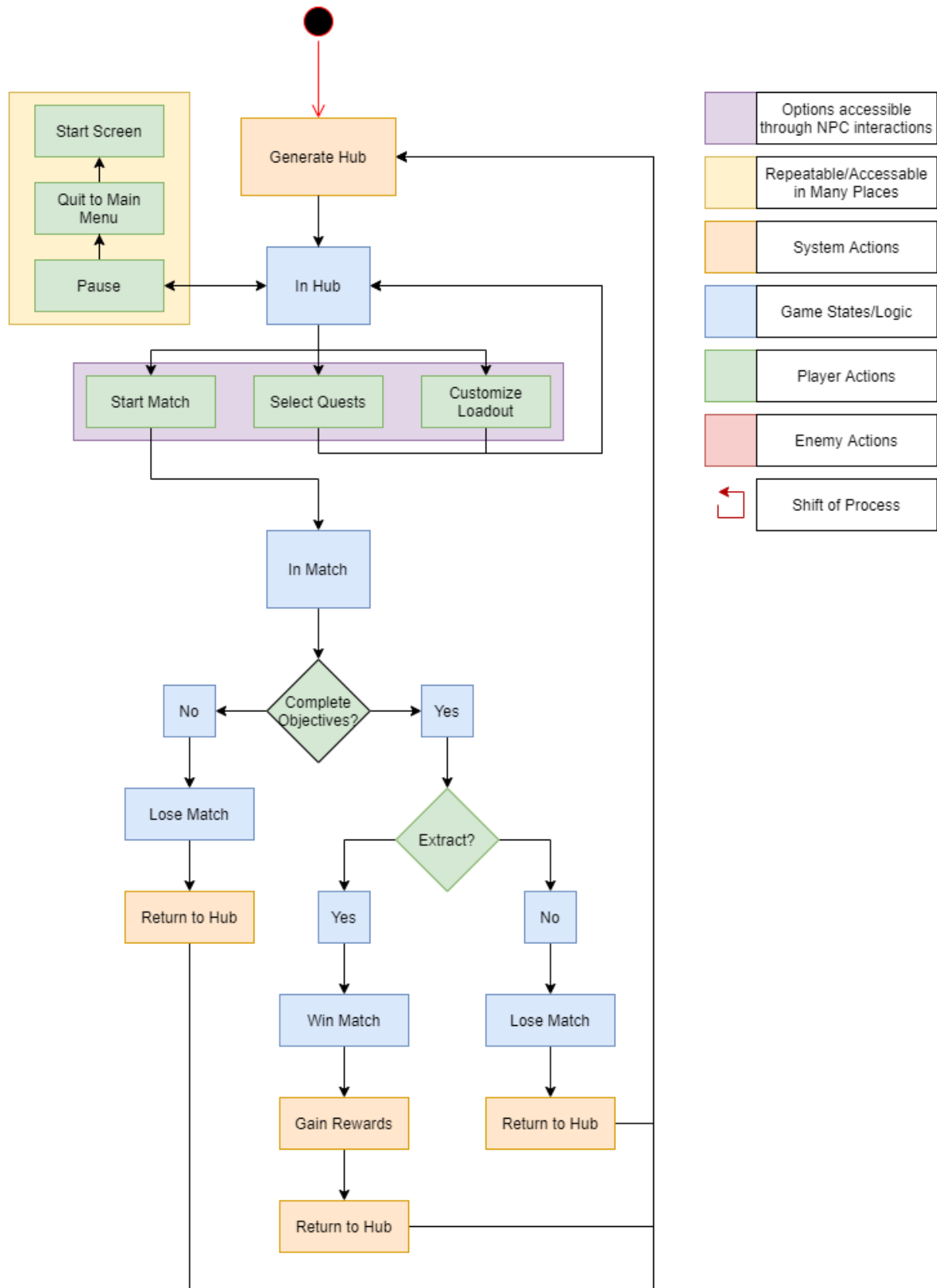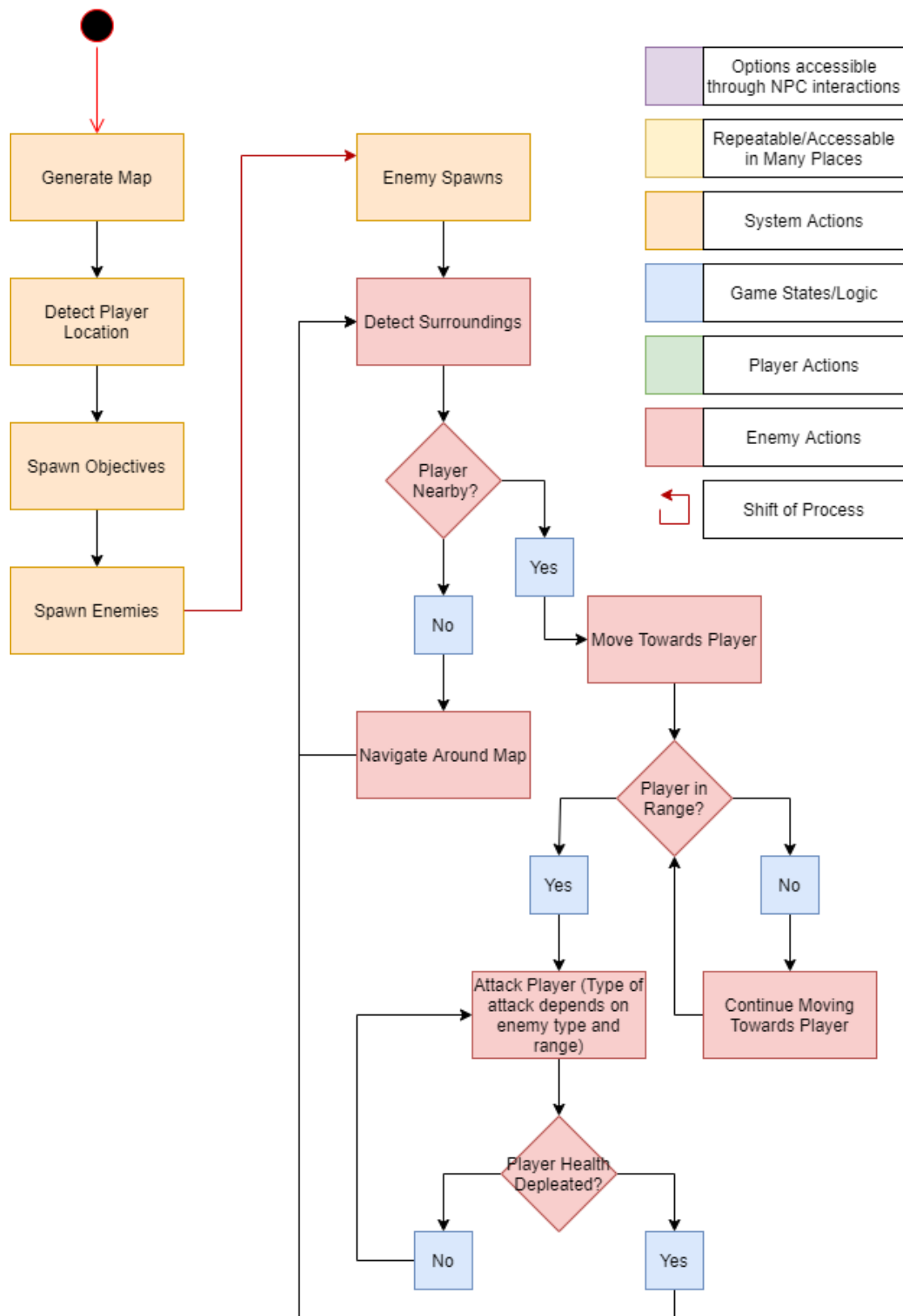
Flow Charts/Process Flows:
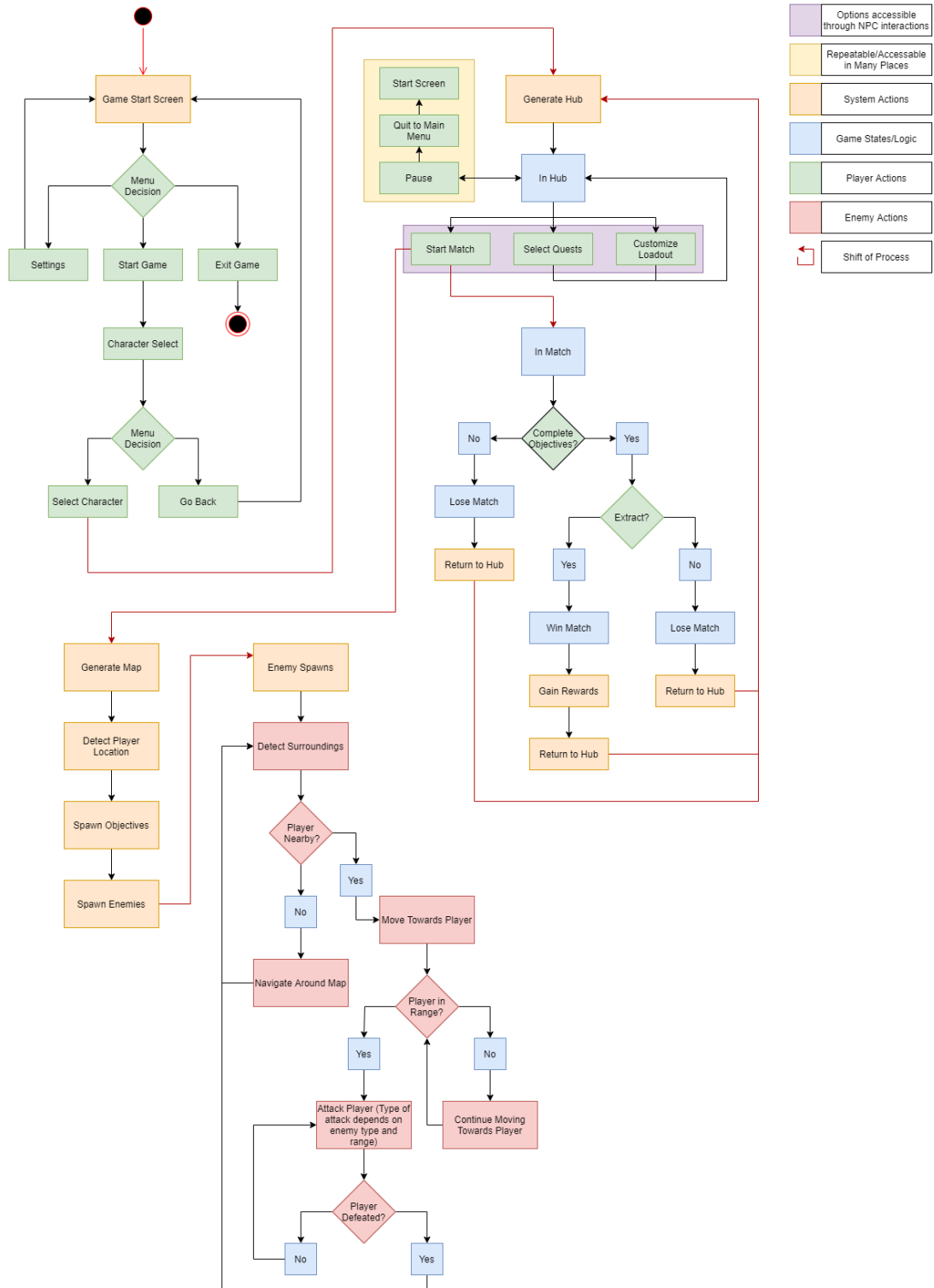
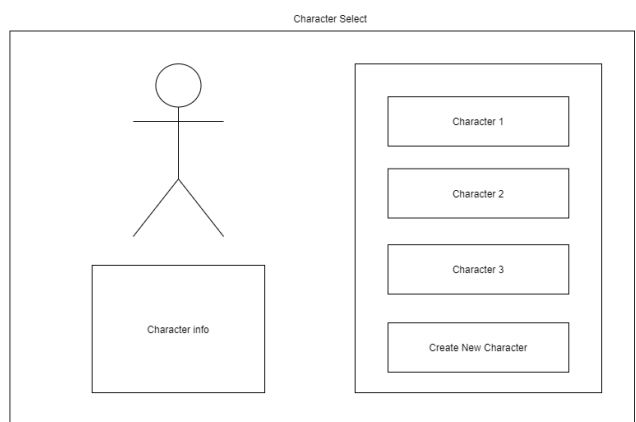Entire gameplay loop leading into each other.
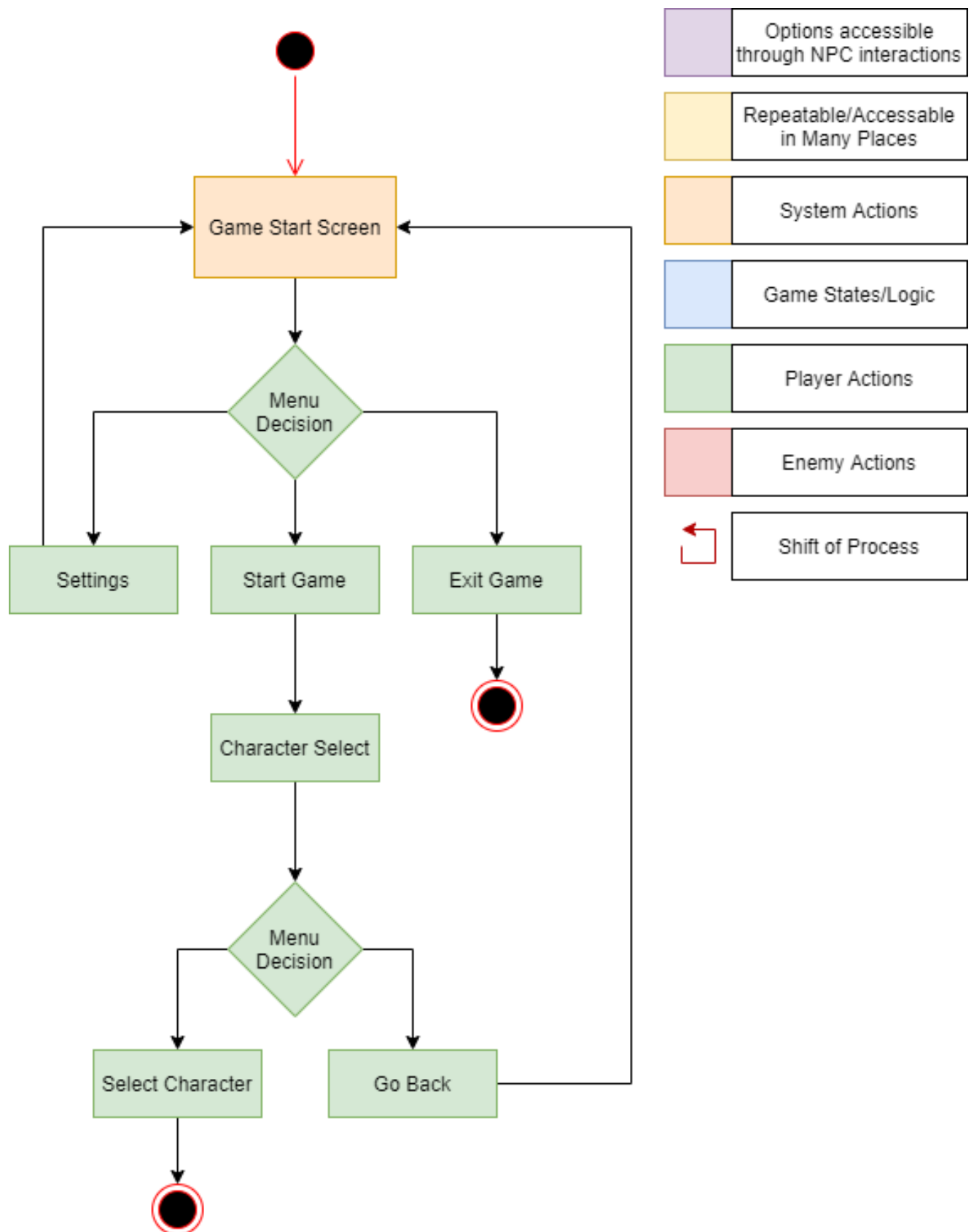
Flow for the main menu.

Player loop from hub world to main gameplay loop and back.
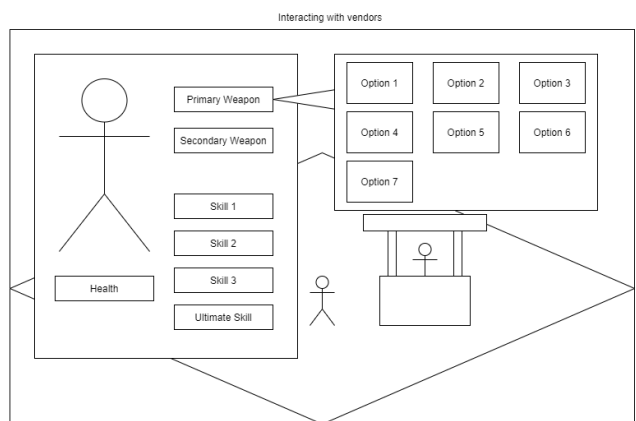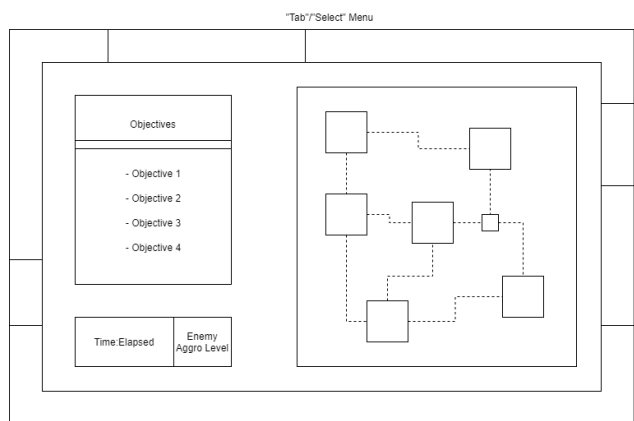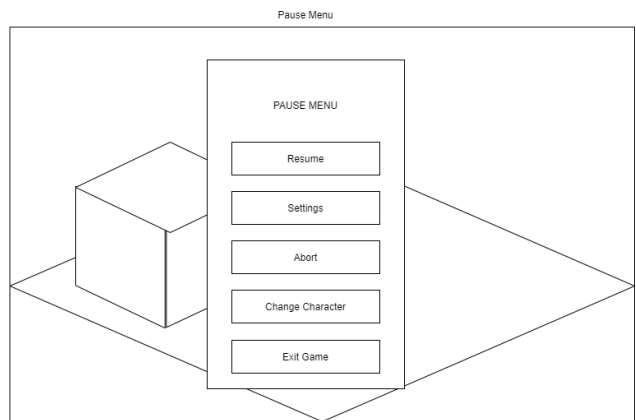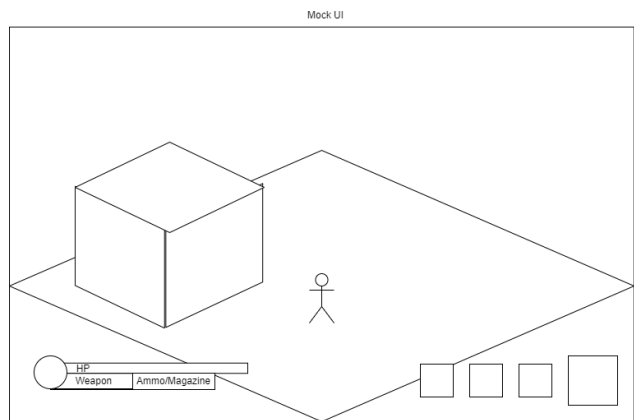
Map generation and enemy artificial intelligence loop.

User Interface Diagrams:

Mock UI

HP
Weapon | Ammo/Magazine

Pause Menu

PAUSE MENU

Resume

Settings

Abort

Change Character

Exit Game

"Tab"/"Select" Menu

Objectives

- Objective 1
- Objective 2
- Objective 3
- Objective 4

Time:Elapsed | Enemy Aggro Level

Interacting with vendors

Primary Weapon

Secondary Weapon

Skill 1

Skill 2

Skill 3

Ultimate Skill

Health

Option 1 | Option 2 | Option 3

Option 4 | Option 5 | Option 6

Option 7

Start Screen

Game Logo

Press "Space/Start" to Start

Settings

"ESC" to exit game.

Character Select

Character info

Character 1

Character 2

Character 3

Create New Character

NFR's (Security Design, etc.):

      The non-functional requirements of the game will be supported through the minimum system requirements for any machine that intends to play the game. The minimum system requirements will be subject to change once development has reached different stages and will be tested through the deployment of the game on various devices with different levels of hardware proficiency. Once the final system requirements are established, the NFR will be tested using third-party framerate and performance trackers such as the Nvidia or Windows overlays.

Other Documentation:
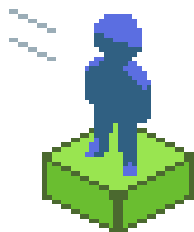
Potential silhouette for use for designing characters.
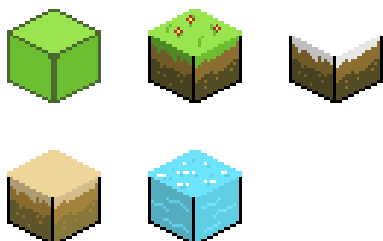
Designs for the terrain of different environments that may appear in the game.

Appendix A – Technical Issue and Risk Log

1. Use the template to identify and monitor project issues and risks.

Issues and Risk Log								
Issue or Risk	Description	Project Impact	Action Plan/Resolution	Owner	Importance	Date Entered	Date to Review	Date Resolved
R	Studio One 6 is expensive	This will increase the cost of the project by a considerable amount.	If possible, then the software will simply be purchased. However, a member of the development team has an older version of the software that offers the same level of functionality, and that will be used instead.	Giovanni Martinez	*2*	*11/15/2022*	*11/16/2022*	*11/17/2022*
R	Assets have yet to be completed for the development of the game.	Without visual assets, the game will be lifeless and thus will not attract players. This could potentially increase the cost of the project as well	Any assets that have already be created will be used throughout the development of the project. Missing assets will be purchased through the Unity Asset Store, commissioned by a third-party artist, or created throughout the development of the game.	Giovanni Martinez	1	11/17/2022	11/17/2022	Resolved throughout the development of the game.
R	(OOS) Riptide may not function as intended for the development of the multiplayer systems.	This will restrict out use of the framework to accomplish what is intended with the system.	If the framework proves to not provide the functionality that is required, the development team will shift towards using Netcode for Game Objects, the official Multiplayer API for Unity.	Giovanni Martinez	3	11/17/2022	11/17/2022	Resolved if development of multiplayer becomes feasible.