Unity-Based Fighter

CS 462: Senior Software Engineering Project
Team 27

Contributors
Brayden Tremper
Cameron Markwell
Gauge Hartwell
Aaron Koffel

Abstract: This paper aims to provide an overview of the alpha functionality of the Unity Based Fighter game created by the Senior Project Group 27. A lot of progress has been made toward the project goals of including game mechanics such as limb damage and characters who react to the game's music.

Table of Contents

1 Introduction

This report provides information regarding the overall plan for our project. Our project consists of developing a full scale video game using Unity3D in an attempt to explore the relationship between an enemy AI and a human player. With this, the report outlines the overall vision and requirements for this project alongside our measures for success as well as constraints and risks which may affect the success of the project. Branching off this, the overall plan for the project is displayed based on the desires of the stakeholders while considering the overall constraints and taking them into account while developing an iteration plan for the project.

2 Background

When considering the idea of a video game, they are often made as an escape from reality. The player often takes the form of the protagonist who must face a number of obstacles and challenges. This causes the player to often recognize this protagonist as themselves, bringing forth the need for immersion and allowing this experience to be supported. In games, this immersion is supported by a believable interaction between the protagonist and their enemies. Boss fights explore this idea heavily. The player must wait until the boss is done attacking before attempting to attack themselves. With this, the player is able to get into the boss's rhythm, creating the opportunity for a "dance" between the player and boss to be implemented. This game seeks to explore this relationship between the virtual boss and the player.

Many games have boss fights; They have been a staple of video games since arcade games. Boss fights are generally fights at the end of a level in which the enemy being fought is stronger, larger, has more health, faster, or any varying combinations of abilities that make the enemy a powerful foe. Some bosses attack in predictable patterns that are telegraphed at the beginning of their attack, and some of these bosses attack with a particular rhythm. The player, usually too weak to simply take the full force of the attack, then has to move, block, deflect, or perform some other moves in time with the bosses' attacks in order to minimize the damage taken. Bosses of this type will often then hesitate before attacking again in order to allow the player to attack them and deal some damage. This relationship, communication, and mutual respect between the player and the enemy is reminiscent of the relationship of two dancers. However, there are not many games that emphasize this relationship, or explore it in detail. We aim to change this by building a combat system to produce a fight that evokes feelings similar to those evoked while dancing.

3 Vision Statement

Our goal is not one of financial interest, or one that improves humanity in a direct way as a whole. Rather, our goal is more similar to that of a piece of art in that we want to evoke a particular feeling within the user. We want to develop a game that gives the feeling of dancing to the users and explores that feeling. This will be accomplished by the use of rhythms, telegraphs, predictability in attack sequences, shifting who can be on the offense, and synchronizing actions with music.

In addition to this, we want to make a game that feels good to play and is fun. This will be accomplished by interesting gameplay choices, unique bosses, allowing for different play-styles, and impactful gameplay mechanics.

In order to limit ourselves and ensure that we have the time to develop a viable product, we aim to only develop a proof of concept to start with. This would consist of only a single boss-fight. This is all that is needed to show that such a game could actually evoke the feelings we aim to evoke with gameplay that is fun and interesting.

3.1 Hypothesis

3.1.1 Growth Hypothesis

Our project will offer a unique take on third-person fighting games. By taking an in-depth dive into the similarities between the combat and dancing, our game will differentiate itself from other, similar games.

3.1.2 Value Hypothesis

When viewing the value of this product, it seeks to provide an engaging, immersive, and unique experience to the end user. The immersion, applied through dance by the use of telegraphed fights, seeks to simulate a real-world dance. If this simulation is accurately tracked according to the game's rhythm with the player and boss “dancing” to the rhythm, emulating a real world dance, then this section can be considered to provide a valuable and immersive experience to the end-users. Following this, the product seeks to be unique through the emulated dance, but also other elements such as limb damage (if this damage can be tracked accurately, with damage being applied to that specific limb, its value can be recorded). Branching off of this, the end user ultimately seeks to have an overall enjoyable experience such as having the previous elements above but also ideas such as appealing graphics, smooth movement, a variety of moves, and well tracked attacks. In order to test the value of this, every element needs to be accounted for, allowing the user to have an overall beneficial experience.

3.2 Requirements

3.2.1 Functional Requirements

- The project will be implemented using 3D graphics.
- The player character will respond to inputs given via either a keyboard and mouse or a controller with the appropriate action.
- Opponents in the game will utilize a simple Artificial Intelligence to choose which action to take.
- Opponents in the game will follow predictable attack sequences, give the player opportunities to attack back, and sync all of this with the music.

- The player will have a variety of movement options such as walking, running, jumping, dashing, and attacking.

3.2.2 Non-Functional Requirements

- The game controls and functions will be intuitive to the players and beta-testers.
- The game will run smoothly given computers capable of smoothly running games of similar complexity.
- The scripts compiled will run in a fast manner with low run-time complexity.
- Enemy AIs will respond in a timely manner when applicable according to the telegraph of attacks.
- The final game file will be a size less than 10GB.
- The loading time between different scenes in the game will be less than 1 minute.

4 Success Measures

Because our goal is to evoke a particular feeling, our success measures are subjective. The best way to measure our success, then, is to use beta-testers and listen to their reviews. Throughout the development of the game, we will reach out to friends and those interested and search for volunteer beta-testers. These beta-testers can then give short answers to a few survey questions to determine if they feel like the game is fun, if it feels nice to play, and if it evokes a feeling of dancing.

At the time of project completion, success will be measured as a minimally “complete” or playable game - at least one “boss” is completed as a fully working entity within the game, a limb damage mechanic will be fully implemented within the game for all characters, and all player-enemy interactions will be a good basis for building a “dance” out of. A “boss” within the game will be considered complete when the character is operating as intended (correctly interacts with player - fights player, limbs react to taking damage, character is synchronized with the background music). The “limb damage” mechanic will be considered complete when each character within the game successfully reacts to having one of their limbs damaged. As far as the feeling of “dancing” where the enemy is concerned, completion will be measured with the operation of the fight itself, as in the interaction between the player and boss will be a working fight synced to the music, which can be tuned later to better fit the subjective feeling of a “dance”. Project completion will be measured based on whether the enemy responds to actions in correspondence with the music. In addition, our stretch goal of making the game feel like a dance will be measured subjectively whether the game feels like a “dance” (where the player must time their attacks correctly to get a shot in) or just another non-player character (NPC) fight.

5 Prioritized Project Constraints

Time

This project is constrained by a 9-month (minus spring and winter break) project time. It is intended to be completed and ready for showcasing at the end of the 9-month period given for its completion. If it does not appear that the project will be completed before the time-limit given, more man-hours will be brought to the project in order to assist with completion of the project in the time given. This will mostly consist of everyone in the group devoting a larger portion of their time to the project. The scope of this project is intentionally minimal so that the project team has a clear and concise target to achieve in order to call the project “done” at the end of the 9-month period.

Resources

The resources for this project can be changed as needed to meet deadline requirements and project scope. The main resource needed for this project will be the time investment of the project developers. Both during the 3 terms, and during breaks. There is no project budget, so monetary constraints are not considered here. Other resources that will be needed for this project include outside expertise (provided partially by the project partner), documentation on the tools needed for the project development (such as Unity 3D documentation, Github documentation) and the development software/tools themselves (Unity 3D, Github).

Scope

The scope of this project is not as flexible as the resources available for the project, but more flexible than the time allocation for the game. The scope currently includes the things outlined in the “success measures” section of this document. If the project can not be completed (e.g. the scope is too large for the project), then some features will need to be omitted.

6 Stakeholders

- **Outside experts**
  - Helps development team with assistance based on their expertise on a given subject, such as music, dancing, graphic/character/environment design.
- **The Players**
  - Plays the game and enjoys their time with it. Enjoys the feeling of “dancing” with their opponent within the game.
- **The development team**
  - Creates the game to the specifications outlined in this document.
- **The project partner**
  - Benjamin Brewster, a professor at Oregon State University, outlines the vision for the game and how it should look/feel. Wants the project with the requested features completed at the end of the 9-month project development time.

7 Risk

Table 1: Project Risk and Mitigation strategy

Risk	Likelihood	Impact	Mitigation Strategy	Early Detection	Consequence
The deadline may not be possible with the size of the minimal required scope of the project.	Unlikely	High	To mitigate this, a review of current progress and an update to the plan will be done on a bi-weekly basis so more resources can be brought in, if required, to accomplish the minimal required scope on time. Each update will provide an opportunity to assess current progress and decide if the project should move forward or if it should be scrapped or postponed	Bi-weekly plan updates result in an iteration plan that goes beyond the deadline.	Should the mitigation strategy fail to prevent/avoid the risk, the project scope may be reduced in order to allow for minimum project completion.
The project is not satisfactory to the stakeholders after completion.	Unlikely	Medium	To mitigate this, an awareness of the vision of the project partner (through meetings/regular communication) will be maintained.	Bi-monthly updates/meetings with the project partner to help maintain focus and direction of the project.	Should the finished project not conform to the vision of the project partner or the players, the project will be completed anyway to create a deliverable/base for the project team to possibly improve.
The project developers do not have the same vision of what the project will be upon completion.	Unlikely	Low	To mitigate this, regular meetings among the project development team will be held in order to keep each team member on the same page with the rest of the team.	Weekly meetings will be held among the project team so that the current goal and project vision can be understood among the team.	Should the project team fail to come to an understanding on the project vision or goal, the team (including project partner) will come together to discuss the

					goal/vision of the project to re-align themselves with the project goal.

8 Scope

By the end of the capstone sequence, we hope to have a fully functional game with one boss-fight. This game should demonstrate each part of the full game. The player should possess a varied moveset which can be built upon to create an extremely varied moveset. There should be at least one melee weapon and one ranged weapon so that there is a working basis for the weapon mechanics and all that is needed is to add more weapons. There should be at least one boss that demonstrates what a boss-fight should be, and all that needs to be done is to design more bosses.

8.1 Process Flows

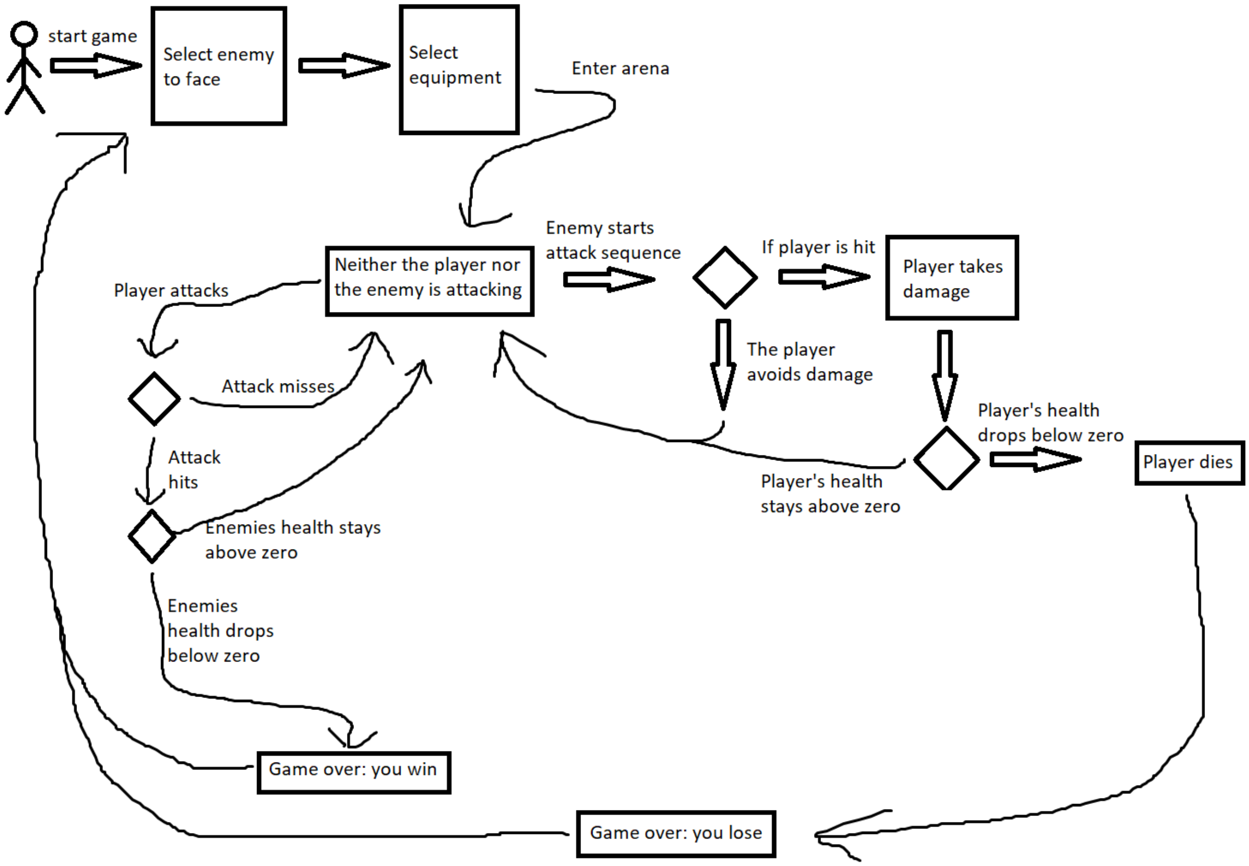

Figure 1: Flow of Gameplay

The essential components of the game are the player and the enemy. These two characters interact and fight each other throughout the bulk of the game (the rest of the game is choosing which enemy to face, and what equipment to use). Throughout a fight, either character may attack the other. Should one of the characters be hit by the others attack, that character will receive damage and, if they've taken enough damage, die. Alternatively, seeing an attack coming, a character may dodge and avoid taking damage. When one character dies, the match ends and a new match may begin.

8.2 User Stories

8.2.1 Player

1. As a player I want to be able to move around in the world so that I properly engage in the battle.
2. As a player, I want to be able to make my character jump so that there is another dimension available to me during a fight.
3. As a player, I want the camera to smoothly follow my character and always face the correct direction so that I can quickly and easily get a bearing of my surroundings.
4. As a player, I want enemies to fight so that I have a clear goal and a way of beating the game.
5. As a player I want every weapon to have a simple attack that's relatively easy to execute so that I can always have an acceptable default for an attack.
6. As a user, I want to switch weapons so that I can have a wider arsenal available at any time.
7. As the player, I want to have a movement speed faster than the standard movement so that I can quickly maneuver long distances around an enemy.
8. As a player I want to quickly move a short distance so that I can dodge oncoming attacks.
9. As a player I want to quickly move a short distance and be invulnerable for a moment so that I can dodge oncoming attacks.
10. As a player, I want to be able to jump in mid-air so that I gain extra height.
11. As a player, when hitting some things with a melee attack in mid-air, I want to be pushed back in the opposite direction they were looking so that I can bounce off of enemies and their attacks.
12. As a player, I want to be able to sync my actions with the universal rhythm tracker so that I can sync my actions with the enemies.
13. As a player, I want each weapon to have a special attack so that I have an interesting extra option with each weapon.
14. As a player, I want to have the option to quickly jab at the enemy so that I can damage him in cases where I might not be able to otherwise.
15. As a player, I want to be able to deflect an oncoming attack without moving so that I have an interesting option when being attacked.
16. As a player, after taking damage to a particular limb, I want penalties applied to things the limb is used for for a short time.
17. As a player, I want to be able to easily see how much health I have left so that I can gauge how riskily I am willing to play and see when I need to heal.
18. As a player, before going into a fight, I want to be able to choose which weapons, items, and abilities I have active so that I can equip myself in a variety of ways for different bosses and different playstyles.

19. As a player, I need to have a visual representation of the player, opponent, and environment so that while I play I know what I am controlling, attacking, and how to interact with the environment.

20. As a player, I want to be able to move my character in different directions so that I can interact with the game world.

21. As a player, I want to be able to defend myself against my enemies so that I can prevent them from damaging me.

22. As a player, I want to be able to use items from my inventory to heal myself so that I can continue to interact with my enemies.

23. As a player, I want to see my character's health so that I can decide whether I should change how I play.

24. As a player, I want to see my character's currently equipped weapons and items so that I can decide what to do with them.

25. As a player, I want to see the current status of my character's weapons so that I can make informed decisions on what my next action should be.

26. As a player, I want to see my enemies' health so that I can make an informed decision on whether to attack them.

27. As a player, I want to see the player character as a person and enemies as identifiable things (humans, robots, animals, whatever) so that I can empathize with them and understand what's happening.

28. As a player, I want enemies to exist within the game environment so that I can enjoy fighting them.

29. As a player, I want my enemies to die when they have taken enough damage to reduce their health to 0 so that the game is not impossible to beat.

30. As a player, I want my enemies to be able to focus on me so that I can have a meaningful fight/interaction with them.

31. As a player, I want my enemies to react to my actions within the game world and attack me based on when I am vulnerable.

32. As a player, I want the other characters within the game to move based on the game soundtrack so that I can feel like I am dancing with the enemies while fighting them.

33. As a player, I want my enemies to take damage specific to their various limbs and receive debuffs to provide a unique experience for the player.

34. As a player, I want my enemies to have health bars that are visible to me so that I can properly estimate their current health.

35. As a player, I want my character's and my enemies' limbs to be susceptible to damage so that the game has a unique experience.

8.2.2 Developer

1. As a developer. I need to have a visual representation of the player, opponent, and environment so that I can create and debug both the movement and animation system.

2. As a developer, I want the enemy boss to take offensive/defensive actions based on the current and history of the players past actions so that the AI can have accurate data to train on.

3. As a developer, I want to be aware of which direction the camera is facing so that I can make the player move effectively and play correct animations.

4. As a developer, I want the player and enemies to have smooth animations so that I can debug collisions and be aware of the current state of the characters collider.

5. As a developer, I want the environment in the game to contain boundaries so that the player cannot leave the map in an unexpected manner potentially breaking the game.

8.2.3 Project Partner

1. As a project partner, I want the game to have appealing graphics, smooth animations, and responsive colliders so that the end user can enjoy their experience.

2. As a project partner, I want the game to follow a rhythm so that the combat can feel like a dance for the final users.

3. As a project partner, I want all the user stories to be implemented so that the final results are as expected.

4. As a project partner, I want the player and boss to be susceptible to limb damage so that an interesting and difficult game mechanic can be implemented.

8.3 Iteration Plan and Estimate

Table 2: Sprint Plan

Term	Sprint #	Category	User story / Addition
Before the start of Fall	N/A	Movement	As a player I want to be able to move around in the world so that I properly engage in the battle.
			As a player, I want to be able to make my character jump so that there is another dimension available to me during a fight.
			As a player I want to quickly move a short distance so that I can dodge oncoming attacks.
		Graphics	As a player, I need to have a visual representation of the player, opponent, and environment so that while I play I know what I am controlling, attacking, and how to interact with the environment. (partial)
		Setup	Initialize Unity project and distribute with other members through GitHub. Also, begin watching Unity tutorials to better understand commands and workflow.

Fall term	Sprint 1	Movement	As a player, I want the camera to smoothly follow my character and always face the correct direction so that I can quickly and easily get a bearing of my surroundings.
		Graphics	As a player, I want to see the player character as a person and enemies as identifiable things (humans, robots, animals, whatever) so that I can empathize with them and understand what's happening. (partial: character model)
		Enemies	As a player, I want enemies to fight so that I have a clear goal and a way of beating the game.
		Weapons	Worked on weapons. Made progress on the gun
		Testing	Test that the camera follows the player correctly from various input directions and rotations.
	Sprint 2	Movement	Improve and expand upon player movement
		Enemies	Set up deterministic movement for enemy AI
		Graphics	Improve player character and enemy graphics and animations.
		Weapons	Add in melee weapons and perhaps a new type of gun
		Testing	Test that the updated movement system doesn't interfere with previous, graphics don't interfere with game functionality, and animations correspond with character movement.
Winter term	Sprint 3	Movement	Improve and expand upon player movement
		Enemies	Update enemies to behaver non-deterministically so that they can respond to the player's actions
			Expanding on enemies abilities
		General	Add more weapons and items
		Testing	Take time to more thoroughly test what has been done thus far and its interactions with everything else.
	Sprint 4	Movement	Finish character moveset
		Enemies	Expanding on enemies abilities
		Graphics	Add in any missing animations and smooth out existing ones
		Environment	Add in arena to fight in
		Music	Add in soundtrack
			Start work on synchronizing the enemy, the player, and the

			music.
	Sprint 5	Environment	Upgrade map and environment to be visually appealing and to provide interesting gameplay
		Sound effects	Add in player and enemy sound effects
		Choreography	Begin work on fight choreography
			Adjust movement, attack, and defense actions of the player and of enemies to fit choreography
		Testing	Take time to more thoroughly test what has been done thus far and its interactions with everything else.
			Start beta-testing
Spring term	Sprint 6	Choreography	Ensure that everything is synchronized
			Continue work on fight choreography
		Screens and menus	Start work on the title screen menu, the boss select menu, the item and weapon equip menu, and the victory/death screens
		Testing	Continue beta-testing
	Sprint 7	Choreography	Finish work on fight choreography
			Adjust animations to match choreography
		Screens and menus	Finish work on menus
		Testing	Continue beta-testing
			Take time to more thoroughly test what has been done thus far and its interactions with everything else.

9 Report of Alpha Functionality

9.1 Brayden Tremper

9.1.1 Bio

I am in my final year at Oregon State University for a degree in Computer Science with an applied option in Game and Simulation programming. I have had previous experience in Unity3D and C# through the Video Game Development Club, personal projects, and an internship. With this, I have completed game development projects for 2D, 3D, and VR environments.

9.1.2 Description of work contribution

The main contributions I have made for the project involve the music analysis, input tracking, third-person camera, a dynamic reticle, character ragdolls, health system, general movement animations, and graphics within the game. The music analysis created allows for a selected song, given its beats per minute, to be broken down further into its respective keynotes, 1 through 4, which can be tracked allowing for an action to take place at the desired keynote. This has been applied to the game environment allowing for lights to be intensified at given keynotes in sync with the song's beat as well as enemy actions where the enemy has the ability to select an action at a given keynote based on their location in respect to the player. Following this, I have set up the input tracking for both mouse and keyboard as well as gamepad controller support where each input is binded to a selected button, mouse, or thumbstick allowing each to return a value or function, depending on the case, to then be used to control the player's actions. In addition to this, I set up the third-person camera, using Unity's cinemachine package, allowing it to follow the player where it can be rotated when given a user input which in turn rotates the player so the back is always directed towards the camera. Building on this camera system I implemented an aim functionality that adjusts the field of view for the camera as well as a dynamic reticle which changes size based on this zoom as well as player movement.

I also created the character health system which can be applied to both the player and enemies which tracks the character's current health, allows them to take damage, heal when desired, and implements a death method that turns the character into a lifeless ragdoll. Building on the ragdoll, this is set up allowing the character model to function as a humanoid while also applying colliders on each of the character's limbs where limb damage is implemented allowing damaged limbs to exhibit a negative effect which will be implemented soon. Finally, I set up the graphics within the game for the character's models along with the map environment. These highlight the main contributions I have made for the project thus far but I have also spent time assisting with bugs that other teammates have encountered and aiding in management for the project. In the future development of the project I am seeking to work on making the enemy move and transition between states more smoothly and expanding on the enemies available actions which can be triggered on beat with the music for situations when the enemy is patrolling around the map, following the player, and in the attack range of the player.

9.2 Cameron Markwell

9.2.1 Bio

I am a senior in Applied Computer Science (Game and Simulation Programming specifically) at Oregon State University. I have limited experience with Unity and game development in general and this will easily be the largest, most complex, and most complete game that I'll have ever worked on. While I've started games in the past (that is how I learned and became familiar with Unity) I've never been able to get very far.

9.2.2 Description of work contribution

I was the one to start and initiate this project. Before Capstone started, I wanted to develop a game (since this was my best opportunity to create a well-made and complex game) but did not see any good projects for that. So, I found out how to develop my own project. I found a group, we collaborated on what kind of game we wanted to build, we found a sponsor, and got assigned to our project.

As a first step, I laid the groundwork. I developed the scene in Unity, made a player character (a simple sphere), applied controls to him, and ensured that basic physics worked. After that, I developed the framework for how the code should work for the weapons and for the characters. I set up a method of inheritance for both the characters in the game and for the weapons and I defined how mechanisms such as using a weapon or moving would use this framework. I added what I call ‘action states’ to determine the length of time a move's telegraph, action, recovery, and cooldown should last. I also implemented basic movement options such as dashing and jumping (this was much simpler then than it was to improve and change them because there were no animations yet). Much of this has since been adjusted, built upon, or even removed.

After that I made a number of small contributions. This includes but is not limited to adjusting the camera, adjusting how the dash works with the rest of movement, and adding bullet drop.

My next major addition was adding a melee attack. This took a while since animations existed at this point.

Since then (in addition to some minor adjustments) I’ve been working on weapon systems. I got both swords and bullets to do damage to either the player or to the enemy, I added primary and secondary attacks for weapons, I added weapon cycling so the player could cycle through all their weapons, and I got the character’s hand to always follow the gun (if the gun is equipped) regardless of what animations are playing. These steps took extra long due to animation and merging difficulties.

9.3 Gauge Hartwell

9.3.1 Bio

I am in my final year of college for a degree in cybersecurity. I have little experience with game development, and this project has pushed me out of my comfort zone, allowing me to gain more experience with fields outside of my own field of study. My experience prior to this project involved subjects within cybersecurity, such as social engineering, port scanning, malware, and cryptography.

9.3.2 Description of work contribution

I have added the inventory system to the game, as well as the pause menu and the player death menu. In addition to this, I implemented healing items and an interface for new inventory items to be implemented within the game. An inventory item is defined as an item that can be picked up and placed in the player's inventory and then later selected from the inventory and used in the game. The pause menu required implementation of several methods that interact with the game engine itself, such as resetting the game back to the beginning state and resuming the operation of the game scene. The player death menu is similar to the pause menu, in that it allows the player to reset or quit the game. The inventory itself required a detailed understanding of how objects interact with each other in a game environment. It involves a class for the inventory itself, which keeps track of the objects within it (Inventory Items) and handles adding and removing those objects. There is a class for the inventory interface, which handles displaying items that are in the player's inventory, as well as using those items when the player selects them from the inventory. Healing items required an understanding of work that Brayden had done with the character health system beforehand. I extended his work to allow for Inventory Items that could heal the player character and their limbs. Earlier in the project, I added in the enemy character that is seen wandering around the sample scene (the one used for testing purposes during game development) and implemented its basic behavior (patrolling, following, and attacking states). The patrolling state of the enemy involves the character finding a random spot in the game environment and going to it, then finding another spot to move to upon reaching the last spot. The following state occurs when the player gets close to the enemy character and involves the enemy character facing the player and moving toward them. The attacking state occurs when the player is within attacking range of the enemy character. In the first sprint of our project, I handled the "project owner" role and made sure that the team's sprint meetings were on time and documented.

9.4 Aaron Koffel

9.4.1 *Bio*

I am a senior at Oregon State University pursuing a degree in Computer Science with a focus in Simulation and Game Programming. Prior to this project I had started a Unity project or two, but only got far enough to become familiar with the basics of the engine, and how to move objects around. In completing this project, I have accomplished my personal goal of getting my feet wet in the game development industry.

9.4.2 Description of work contribution

Over the last few months I completed a wide variety of tasks ranging from adding objects and physics to adding animations and programming them. First, I added a reticle object to the heads up display so that the player could see what they were aiming at. Next, I followed Cameron's "action states" example for dashing movement and created similar action states for any tool (typically a weapon) the player is using. The action states of the tool define how long the character will telegraph their move, how long the action will take, the recovery time, and the cooldown until the tool can be used again. Later, I tackled several parts of the handgun. Cameron had already set up the gun so that it could fire projectiles, but several mechanics needed to be added in order to get the intended gun behavior. I added an aim assist that works by shooting a raycast from the camera through the aiming reticle and it checks to see if an enemy is in range. If an enemy is in range and the player places their reticle over them, the mouse/controller sensitivity decreases to make it easier to stay on target. The next thing I tackled was recoil. In order to make the gun feel more realistic on screen, I added an impulse each time the mouse button was clicked, which results in the camera and reticle being "bumped" up slightly. As my final contributions to the gun so far, I added a fire rate limit to the gun, and a reload function that keeps track of the amount of ammunition in the player's magazine. If the player shoots every last round in their magazine, they will be unable to shoot while the gun reloads for several seconds.

As I had just come out of the "Intro to Computer Graphics" class last term, I was anxious to try my hand at some of the animations in our project. Previously, Cameron had implemented a simple dash, where the player would lean forward and move quickly for a second or two. I made some changes to the dash movement system so that the player can now dash/dodge in all directions: forward, backward, left, right, diagonals. I added several new animations (forward, backward, left, right) and made use of the forward and backward animations in the diagonal directions by rotating the player when appropriate. I also added new animations for walking backwards and diagonally backwards. Finally, I worked on smoothing and debugging the jump actions and animations that Cameron made. So far, I was able to smooth out both the jump launch and midair animations. The final animation of the jump, the landing, has proven to be more difficult than I had anticipated.