The main point of the game is to control the character and to move around the arena while collecting as many orange power ups as possible, which gives you lives. While doing this, there will be various obstacles to stop you. Running into green cactuses removes a life, getting hit by an attacker enemy removes a life, and eater enemies are constantly trying to eat the food first to make them harder to get to. The goal is to get 10 lives without going below 0 lives or falling off the map.

There are a few components to the game from the last project. This includes the player controls, camera controls, sound effects, physics, and a GUI. There are also a few components to the game from this project. This includes a built in AI system, internal navigation, and 2 enemy agent types that utilize these in order to carry out various behaviors (3 each).

From the previous project, we have the following components.

First, we have a variety of actions possible for the player. WASD control the basic player movement. In strafe mode, W moves forward, S moves backward, A strafes left, and D strafes right. In this mode, the rotation of the character is done by moving the mouse which rotates both the camera and the character. In rotate mode, W moves forward, S moves backward, A rotates the character left, and D rotates the character right. In this mode, the rotation of the character is done purely with the AD keys and the rotation of the camera is done purely with the mouse. In this mode, the camera rotation and the character rotation are completely independent of one another. The character always moves based on its own local forwards in both modes. Clicking the space bar once allows the character to jump up, and this can be done together with WASD controls. The character jumps with an initial impulse that has momentum even after the WASD keys are released. However, we have air control as well where clicking different WASD keys in midair control the direction while in midair. The character also jumps with air resistance so the movement speed is faster than the jumping speed. When the character is in the air for any reason (ex. after clicking the spacebar to jump), gravity naturally pulls the character back down. The jump impulse, the gravity, and the air resistance are all tunable parameters in the editor. The character is able to walk up certain angled platforms (possible angle amount tunable in editor) but slides down any angled surfaces it cannot walk up. The character can also interact with a ledge in a variety of ways. First, doing nothing and walking over a ledge will cause ledge fall where the character simply falls from gravity. However, when shift is clicked and held down, trying to walk over a ledge will cause ledge stop where the character is stopped before it can walk over a ledge. Lastly, when H is clicked and held down, trying to walk over a ledge will cause ledge hang where the character goes over the ledge but catches the edge of the ledge and holds on without falling. From this point, the character can jump back up onto the ledge or jump down. Clicking the spacebar with WASD controls allows the character to climb back up the ledge while clicking the spacebar with no WASD input will allow the character to jump and fall down from the ledge. The player also has the ability to glide by holding G and moving the character with normal jump and WASD controls. Holding G reduces gravity so jumping and WASD movement will be in a gliding fashion rather than a falling fashion. Releasing G turns gravity back on and stops the gliding movement. The gliding gravity is a tunable parameter in the editor. Clicking the spacebar while still in midair from a previous jump (clicking spacebar twice in quick succession) causes the player's custom movement double jump. The jump impulse, gravity, and air resistance of the single jump also applies to the double jump as tunable parameters in the editor. Lastly, the player interacts with objects in a few different ways. The character has 2 different active user interactions. First, when it tries to pick items up by clicking E. When clicking E with only cactuses or nothing nearby, nothing happens. When clicking E while near (on) an orange powerup, the powerup is collected and an extra life is gained. The character also has an interaction where it gets hurt and loses a life when it moves near and touches a cactus.

In addition to player controls, we also have camera controls. These were mentioned slightly beforehand. The camera will be positioned in a third person POV angle that follows the character as it moves around. The camera is always controlled by the mouse but has differences in the 2 modes. Moving the mouse only moves the camera in the rotate mode, but moving the mouse moves both the camera and rotates the character in the strafe mode. The camera sensitivity is a tunable parameter included in the editor. To ensure that the player is always visible, the camera checks for objects between it and the player. Any objects in the way have their visibility reduced to zero.

There are quite a few sounds implemented in the game as well. By default, there is a custom made background track that plays while the game is running. The volume can be turned up with the up arrow, down with the down arrow, and muted/unmuted with M (restarts where muted when unmuting). In addition, there are a few sound effects. First, clicking E to interact while only cactuses or nothing is around plays an empty interact sound (a clunk sound). Second, clicking E to interact while on an orange powerup plays a powerup interact sound (a squish sound). Third, touching a cactus and getting hit by it plays a pain interact sound (a hitmarker mixed with oof grunt sound). The sound effects can be toggled on and off with the comma key (,).

There are various physics forces present in our game. First, gravity (and glide gravity) acts on the player at all times where having nothing below the player will cause it to fall downwards towards the ground. Both of these parameters are tunable in the editor. We also have gravity working at angles where certain angles are too steep (the amount itself is tunable in the editor). We have basic velocity for the movement of the character itself. We also have an impulse force that allows us to both jump and double jump that acts against gravity but is still affected as it comes down from the jump. This is also tunable in the editor. There is momentum from the impulse forces where releasing a key still allows the character to finish its movement based on previous momentum. Lastly, there is air resistance where the speed of the character while jumping is slower than the speed of the character while walking. The amount of air resistance is also a tunable parameter in the editor.

Lastly, we will have an in-game GUI. This in-game GUI will display the lives that have been accumulated throughout the game, gained from collecting orange powerups and lost through colliding with green cactuses. The GUI also displays game controls and information about the game sound.

We also have an extra credit end condition. If the character collects 10 lives, they win and get a win screen. However, if the character falls off the map or ever gets a negative life value, they lose and get a lose screen based on how they lost the game.

From the current project we have the following components.

First, we have the eater enemy. This enemy chases food nodes by default. They will traverse around the map going to the food nodes and floating up in order to get to the food node. Once they are within range of the food, they eat the food and send it to another random location. After doing so, they will return back to chasing the next food node it’s close to. While chasing food, if the player is within a certain distance, the eater will retreat away from the player until it is at a safe distance. At this point, it will return back to chasing food nodes.

Next, we have the attacker enemy. This enemy chases the player node by default. They will traverse around the map trying to reach the player. Once they are within range of the player, they attack the player and remove a life from the player. After doing so, they will retreat/dodge away in order to recharge its attacking ability. Once a safe distance away and recharged, it will begin chasing the player again.

Throughout this process, the player can also attack both of the enemies by jumping on them. When the player jumps on either enemy type, the enemy is teleported away to a respawn location.