The main point of the game is to control your character and to move around the arena while collecting as many orange power ups as possible, which gives you lives. There are 2 game modes: both singleplayer and multiplayer. In singleplayer, the main goal is to get 10 lives by collecting power ups without going below 0 lives or falling off the map. In multiplayer, there are 2 players both trying to eat these power ups at the same time and jumping on each other takes lives away. The goal is to get 10 lives first (before the other player) without going below 0 lives or falling off the map.

From the previous project, we have the following components (player controls and physics).

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.

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.

For the current project, we have added the following components (networking).

First, we have implemented multiple game states. When the game is first started, the user is given a selection between single-player mode and multiplayer mode. In single-player mode, the player is loaded into the game and tries to get 10 power ups without falling off the map. If they lose, they get a lose screen with the reason they lost. If they win, they are congratulated with a win screen that says why they won. Both scenes will prompt the player to restart the game again or to quit to the main screen. In multiplayer mode, the player is loaded into a second GUI menu where they are given the selection to host or join a game.

This second GUI menu in multiplayer mode is our lobby system. One player will add their username and host the game to start themself as the server. The second player will add their username and then join the game to add themself as a client to the server. Once both players have joined, the players can start the game. Either player can then click the start button to do this. Both players are then transferred into the game scene when the players can then race to get the 10 power ups first. Their usernames are displayed above them.

We have set up smooth handling of both connections and disconnections. If a player joins midway through a game (started game), the player is added as a client to the existing server and both players are updated with each other. Additionally, if either player disconnects (server or client) then they are removed from the game and the other player wins the game by default (will be given a winning screen).
We have set up game end conditions for both modes as well. When the single player ends, they have the choice to restart the game or to return to the main screen. When the multiplayer ends, they have the same options as they do in singleplayer (restart or exit), but here the restart takes you back to the lobby instead of restarting the game automatically. In the lobby, players can host, connect to, and start a new game.

For our extension, we have set up dead reckoning. This allows the game system to predict the player's position based on current movement so that the game doesn't have to send as many movement update packets to the other player. This allows for smoother gameplay, as it reduces latency by not sending and processing as much information.