**Overview of the game design:**

You are a box that can be controlled using WASD or arrow keys. And you press Space to shoot. You can also emote by pressing V, B, N or M

The objective is to earn 50 points before your opponents. You earn points for shooting your opponents.

The game supports up to 6 players, but requires at least 2 players.

You can play with more than 6 players, but player 6 and onwards will spawn in the centre of the map.

**Implementation details of the network features:**

Before a match the Host has the ability to start a new match, but not the other clients.

Starting a match updates the currentGameState NetworkVariable to Playing, allowing all players to move and shoot. The currentGameState NetworkVariable can only be written to by the server.

The starting positions of the players are determined by their OwnerClientId. The Player script gets the starting positions from the GameManager, which has a list of transforms referencing the positions.

When a player shoots a Bullet, that bullet spawns with the Ownership from the player who shot it. When the Bullet detects that it has hit a player that isn’t its owner, it makes its owning player run the ScoreUpRPC method, which is sent to the server, and then despawns.

The ScoreUpRPC increases the player’s score NetworkVariable by one. The score NetworkVariable can only be written to by the server.

Emotes are sent to everyone, but are instantiated locally sense it’s not necessary to sync it’s movement or scale, due to the fact that it always moves and shrinks the same way every time and cannot be influenced during the time it’s instantiated.

**Challenges faced and solutions implemented:**

I sort of missed until just a few days ago that you had to document your development process in the assignment. So there might not be as much to write about.

In the GameManager script, I wanted to give WinCheckRPC a Player parameter. But I got an error that said that it doesn’t know how to deserialize it. So I tried to make the parameter a networkVariable outside of the method.

But then I got an error telling me that I must implement IEquatable to my variable. I couldn’t figure out how to do that.

My solution was to rename WinCheckRPC to PlayerWin and have a separate method named PlayerWinCheckRPC that does the checking.

I then made the Player scripts have the GameManager subscribe PlayerWinCheck to their score.OnValueChanged.

Then I change PlayerWinCheckRPC to a none Rpc method (now PlayerWinCheck) so that the win message gets displayed for the clients as well.

After that I implemented emoting without any major problems. The only problem was that I accidentally wrote the ids from 1 to 4 instead of 0 to 3 in Update at first.

When the match is over, all players go back to their starting positions. I would run a foreach loop that find all of the players in the scene, and finds which one is owned by the device. It is in a method that is sent to everyone.

But the problem was that only the host’s position was reset instead of everyone’s. I tried removing the if check in the foreach loop that checks for if the OwnerClientId is matching the currently iterated Player’s OwnerClientId. And it worked.

And finally, I turned off Interpolation for the Bullet prefab. The interpolation sort of made the bullet’s position update slower on everyone else’s screens, and disappear without it looking like it hit anything. Turning it off fixed that.

**A reflection on the learning experience:**

I have been taught NetcodeForGameObjects before this, but I haven’t used it much at all.

This exercise made me remember how it works and how to implement it.

I learned more about the Rpc decorator (SendTo.), learned about ownership, and I learned how to use GUILayout buttons.

I feel like if I read the assignment more carefully and tried a little harder at the start of the project, then the end result would have turned out better. But it turned out much better than I thought it would at least.