Danielle Anderson

A. Create an Avatar with a Character Controller

1. You can make a new character controller, or use one from a previous project.
2. You must know how to manipulate the character controller script to adjust the various values (e.g., speed, acceleration, jumping, etc.). In other words, if you import a character controller that isn't of your own creation, you must make changes to it - link me to the original character controller and tell me how you modified it (this will show me that you are able to modify it, should you need to).

This has been met. There is an avatar with a character controller that allows it to move, dash, jump and shoot. The movement is animated.

B. Create a \*Useful\* Camera

1. First person or third person are acceptable. The camera must follow the player; no static cameras are allowed.
2. In later projects, you may choose to convert back to a static camera from the dynamic camera required for this project. For now, I need to assess your ability.

This has been met. The game uses a first person camera that is in front of the avatar and follows their movement as well as the mouse.

C. Incorporate Collision/Trigger(s)

1. At least one type of collision and/or trigger must have a scripted behavior. The game must respond \*in an obvious manner\* to this collision/trigger.

This has been met. There is a collision called when the bullets hit the target. There is also a collision check when spawning targets using Physics.OverlapSphere.

D. Incorporate Procedural/Random Generation

1. Use scripting to introduce controlled randomness into your game.
2. You may do anything you like with this portion of the project, \*but\* it must be meaningful/impactful to the game, demonstrate substantial work and thought, and enhance the game in some way. It should make sense with the rest of the game (e.g., don't introduce randomness just to do it - make sure it has a useful purpose, and clearly makes the game better),
3. The randomly generated assets/data should look as though they belong (i.e., make sure they're properly placed, rotated, styled, etc.)

This has been met. The targets are randomly selected from an array and a random position is chosen. If there is no collision based on Physics.OverlapSphere then it will set its position there and set as active. If there is a collision it will appear at a default position. This way the game does not spend too much time looking for a possible position but still keeps the random element. Having a set array for each target type also manages how many can appear at once and makes giving them a default location easier.

Since the player is in different spots facing different directions throughout the game, the targets rotate automatically when spawned. This makes them easy to notice as well as avoids the possibility of them looking out of place by being the wrong way.

E. Incorporate data persistence (saving/loading) in a substantial and useful manner

1. You must have data that saves / loads between playthroughs. You must be able to do something in the game - save - and have your actions / stats / something persist in the next playthrough.
2. APIs are not required. Using PlayerPrefs is not sufficient for meeting this requirement.
3. This data persistence must be substantial and important to the game's functionality. Saving / loading something just because you need to meet a requirement will not suffice for this assignment. It should improve the game.

This has been met. There is a highscore leaderboard feature that uses json saving and loading. At the start of the game in the menu it will check if there is any saved data and if so load the top 3 highscores (if there is none it will show a default blank entry). When starting the game the player will be asked to input their initials. After the game is done, these initials along with the player’s time, points, and hits per second will be saved as an highscore entry if the hits per second is greater than the other entries on the leaderboard. If the player got a highscore, then they will see their entry on the leaderboard.

F. Incorporate a **substantial** unique element, that you have researched and learned on your own:

1. This can be anything you like, as long as it is substantial and impressive. Look at the point values in the rubric, relative to the point values of other aspects of this project. Use this relationship to identify how much work you should put into this unique element. If you are not sure if yours is substantial enough, ask me and I will be happy to let you know!
2. I recommend thinking about something you're personally interested in and use this as an opportunity to practice it. Again, I'm happy to answer any questions you have about this.

This has been met. I chose to incorporate a NavMesh for the first time as it is something I will likely need in the future. I used this for a moving target that occasionally spawns throughout the game. This target has 3 different spots it will go to when spawned, but most avoid some obstacles to do so. I made this target a NavMesh agent that will adjust its destination when it reaches on the goal points I set. I then selected some obstacles that would be in its way and marked them as such in the NavMesh window.