# Bullet Deflector

## Objective:

The objective of the Player in Bullet Deflector will be to deflect bullets/lasers that are shot at them by a set of turrets surrounding them. The Player will earn points for each deflection and Turret destruction or lose health if they miss. Some enemies will not be completely stationary and will move towards the Player as time passes and explode if they are within range, causing a game over.

*Game could be endless if Player has good timing and skill*

Deflections will occur if the Player hits the input with the correct timing. When a deflection occurs, a set of code will determine the point of impact between the Sword and the Bullet and using that information, calculate a reaction force for the bullet.

The Player will use a Sword to perform the deflections. The Sword will be animated through the use of coded Physics. The Player will only control the Input, the Sword will rotate on the y-axis depending on the direction of the incoming bullet. Will use an overlapping sphere to determine the direction.

Bullets will also use coded Physics and will be destroyed if they collide with anything other than the Sword. An explosion force will also be emanated when they collide with anything other than the Sword giving the Player an opportunity to destroy the Turrets even if the deflected bullet does not hit the Turret directly.

Players will need to time their input according to the speed of the bullet and distance from the Player.

Players will also have access to a number of guns that will allow them to inflict ranged damage as well. Player guns will make use of different bullet types to create different physics effects, for example, a regular bullet that is given an initial velocity but is then affected by gravity and air resistance as it travels through the air once shot, and/or another bullet that will add acceleration in the direction of travel after it is shot (similar to a rocket). These bullets will also have varying levels of explosion damage and force.

The bullet types discussed previously will also be usable by the enemy turrets.

## *Gameplay Mechanics:*

Physics Features:

1. Physics-based Animations and Movement
2. Impact Collisions and Reaction Forces
3. Explosive Force for Projectiles
4. (Wishlist) Platforming to add verticality to game

AI Features:

1. States to differentiate between having a target, looking for a target, and waiting for a target
2. Subject and Observer pattern to notify neighbouring AI enemies of the Player
3. Seek and Wander pattern for patrolling enemies
4. AI entities will make use of a component system with each handling a different aspect of the AI

## *Production Timeline:*

Week 1:

**Task 1: Create Basic AI turret template**

Description: Setup the first enemy in the game, a turret that looks and aims at the player within a certain range and returns to its original rotation once the Player is out of range.

Task Estimate: 1 - 2 days depending on the complexity that is implemented right away, ie. using a state machine or simply setting up the components to do the target finding and aiming

**Task 2: Setup a Shooting Object for the Player**

Description: Setup a base gun component that lets the Player shoot projectiles, pools projectiles, and provide projectiles with an initial velocity so we can see some shooting mechanics working

Task Estimate: 1 - 2 days, also depending on amount of complexity intended this early on in development. Simple implementation would be ideal

**Task 3: Setup Player Sword**

Description: Setup a base sword component that will let the Player initiate an animation of the sword swinging in one direction and to make it extensible enough so that more animations can be added in at a later date.

Task Estimate: 1 - 2 days, creating the animation may be the only time consuming part since it will be made through Physics manipulation

**Task 4: Setup Incoming Projectile Notification system**

Description: Implement an UI or another visual to let the Player know that there is a projectile headed towards them. Ie, setting up multiple levels of Trigger Colliders on the playerObj that change the nature of the notification the deeper the into the levels the projectile gets until it hits the player

Task Estimate: 1 - 2 days

**Task 5: Setup Projectile Deflection**

Description: Add implementation of projectile by simply giving the shot projectile a random direction back towards where it came from. Ie. if it was travelling in a forward direction, make the bullet travel in the backwards direction with additional values added to the other two directions

Task Estimate: 1 - 2 days as this may require Physics code and a little bit of testing

<u>Week 2:</u>

**Task 1: Update enemy AI to use State Machines**

Description: Add state machine behaviour to turrets so they are able to differentiate between Aim state, Searching state, and Death state

Task Estimate: 1 day, since the implementation is still basic and relies on if the turret has a target or not

**Task 2: Implement a modular Health system**

Description: Create a health system that all game entities can use to make damage dealing easier to handle in the future

Task Estimate: 1 day

**Task 3: Test Player and Enemy deaths**

Description: Add damage logic to the projectiles that are setup and ensure that both the player and enemy can go into their death states. Also add a reset option so that the game starts over after player death

Task Estimate: 1 day

**Task 4: Add a scoring system and UI**

Description: Implement a score tracker so that Player can have a sense of progression and display it on screen

Task Estimate: 1 day

**Task 5: Add a spawning and wave system**

Description: As enemies are killed, they will need to be respawned to continue gameplay. Using a wave system will also give the player an opportunity to end the game between waves.

Task Estimate: 1 - 2 days, depending on complexity of wave system

**Task 6: Implement projectile effects**

Description: Add explosive force with varying radii and damage based on distance from impact point. Add rocket type projectile for additional acceleration during flight time. If time permits, add a homing projectile functionality

Task Estimate: 1 - 2 days

**Task 7: Add perfect deflection**

Description: When Player hits the input to deflect a projectile and the projectile is within a "sweet zone", the projectile will travel back towards the entity that shot it. Projectiles will have a "BelongsTo" component attached to them that will be used to determine target and travel path after deflection

Task Estimate: 1 - 2 days

Week 3:

**Task 1: Add Subject Observer pattern to AI**

Description: Implement subject and observers to AI, even though Player is still stationary, preparing for allowing player movement will be efficient

Task Estimate: 1 day

**Task 2: Implement Player Movement**

Description: Give player control of moving through 4 zones to fight groups of turrets and perhaps find better positioning for deflections. Ensure all physics code and implementation still works after adding Player movement

Task Estimate: 1 day

**Task 3: Add a moving enemy**

Description: Implement a second AI that is also able to move and follow the Player and makes use of the Seek and Wander pattern, as well as the Subject and Observer pattern

Task Estimate: 1 - 2 days

**Task 4: Add quality features**

Description: Add visuals and sound effects to the game, clean up UI, ensure wave progression and score tracking is working correctly, play game and look for bugs to fix

Task Estimate: 1 day

**Task 5: Fix Bugs**

Description: After playing the game and getting others to play it, go through all the identified bugs, or as many as possible, and solve to ensure a smooth gameplay loop.

Task Estimate: Whatever amount of time is remaining