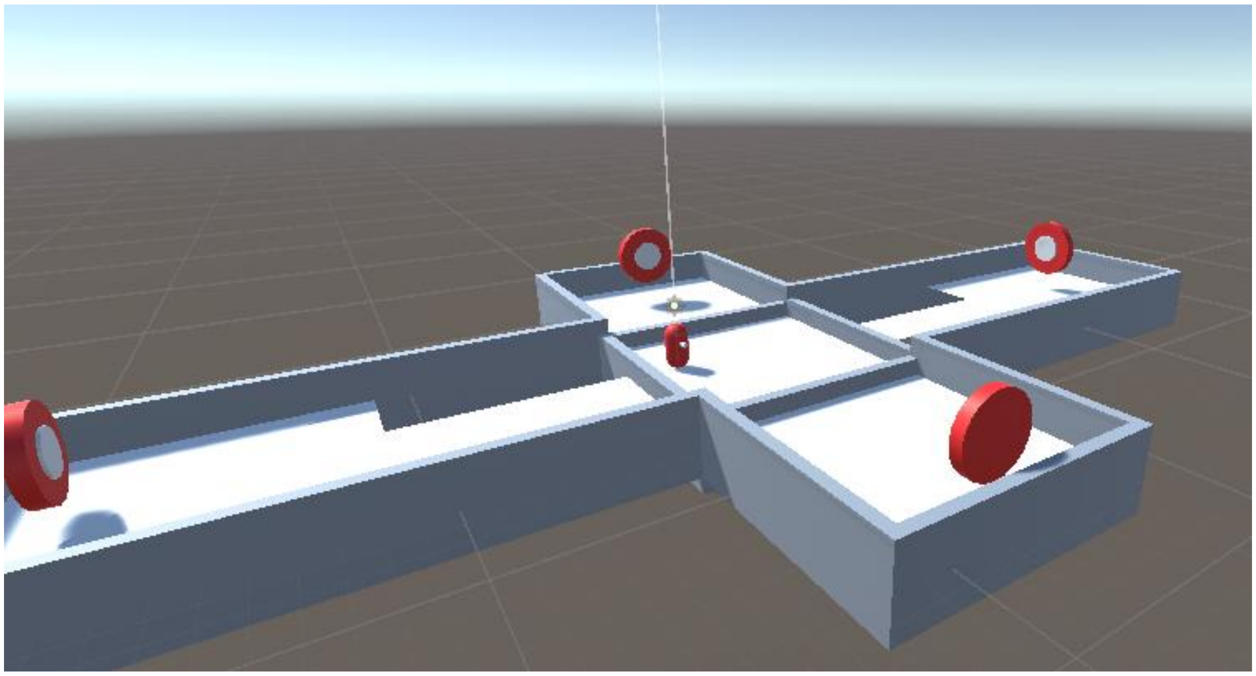
Lab 3: Instantiate, Invoke, and Object Lifetime

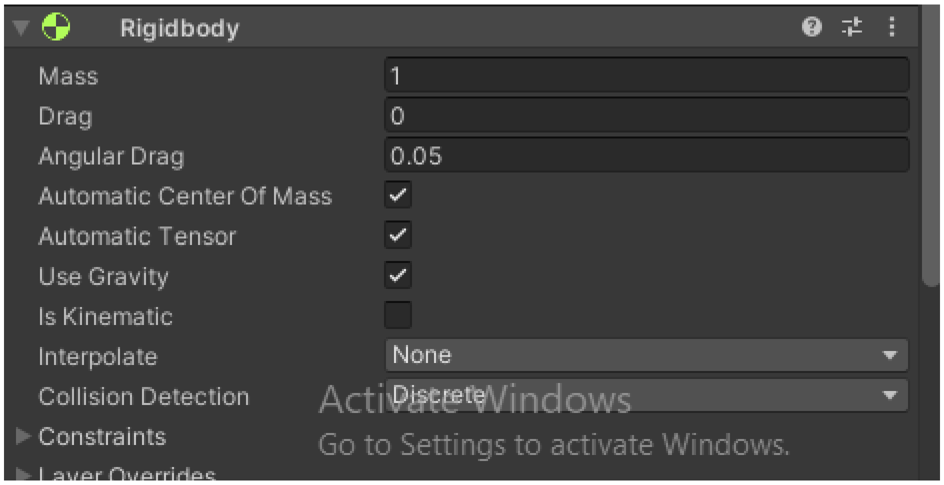
Setup:

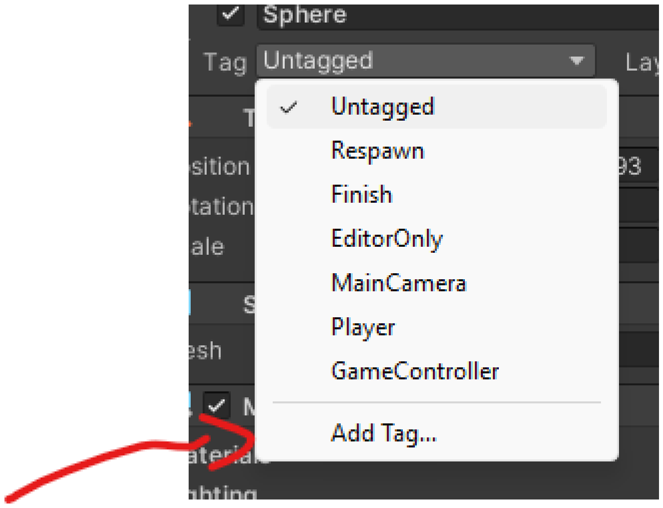
1. Clone the repository:
   - Open this link : https://classroom.github.com/a/UOppABh8
   - Follow the instruction to clone the repo with Github Desktop
2. Open the sample scene in scene folder

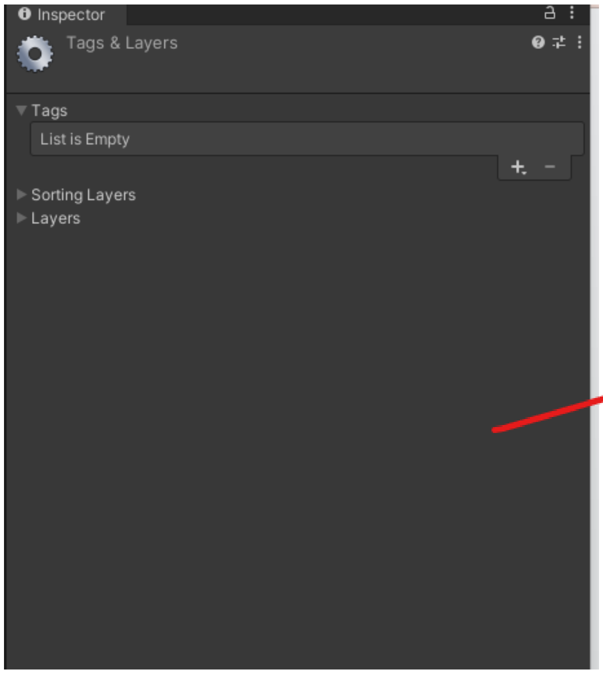


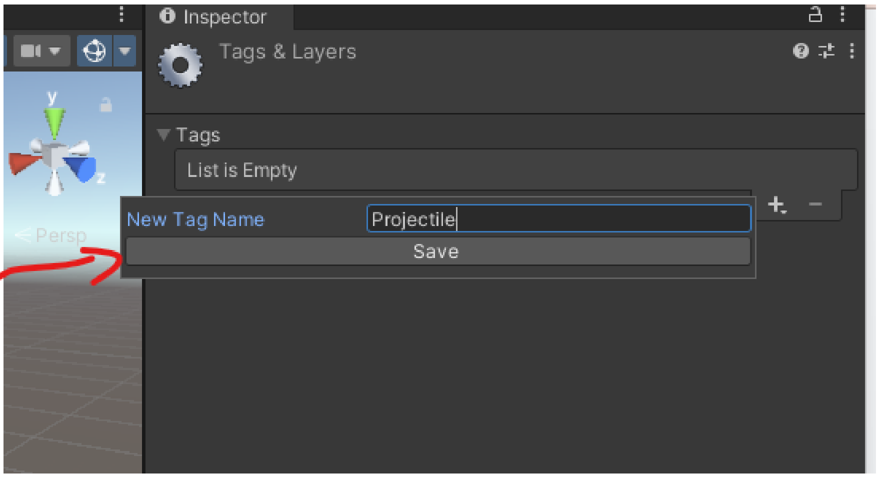
Task 1: Create a Bullet/Canonball Prefab

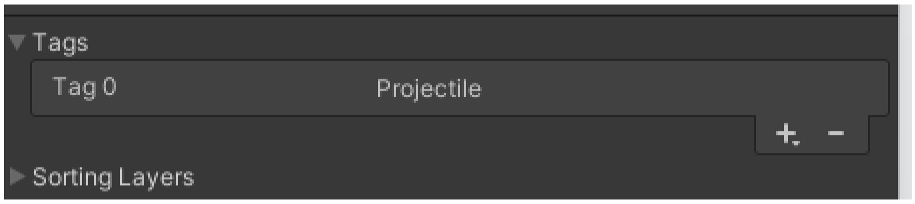
1. **Create a new GameObject**:
   - Right-click in the *Hierarchy* and select Create > 3D Object > Sphere.
2. **Add Rigidbody Component**:
   - Select the sphere in the *Hierarchy*.
   - In the *Inspector* panel, click Add Component and search for Rigidbody.
   - Make sure the Rigidbody's Use Gravity checkbox is ticked to allow gravity to affect the bullets and make sure it is not a Kinematic object by making sure that the IsKinematic Checkbox is NOT ticked.

- 
Rigidbody
Mass 1
Drag 0
Angular Drag 0.05
Automatic Center Of Mass ✓
Automatic Tensor ✓
Use Gravity ✓
Is Kinematic
Interpolate None
Collision Detection Discrete
Constraints
Layer Overrides
Activate Windows
Go to Settings to activate Windows.

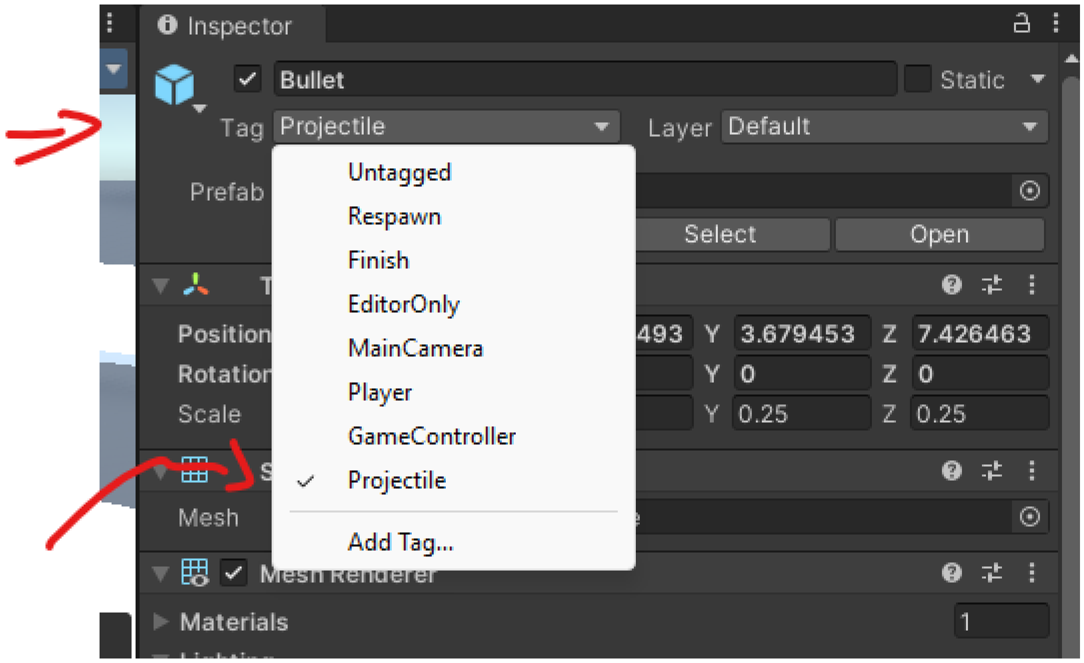
3. **Add Tag**:
   - Click the sphere again, go to the *Inspector*, and click on the Tag dropdown.
   - Select or add a new tag called "Projectile" by clicking Add Tag, then choose Projectile from the list.
   - 



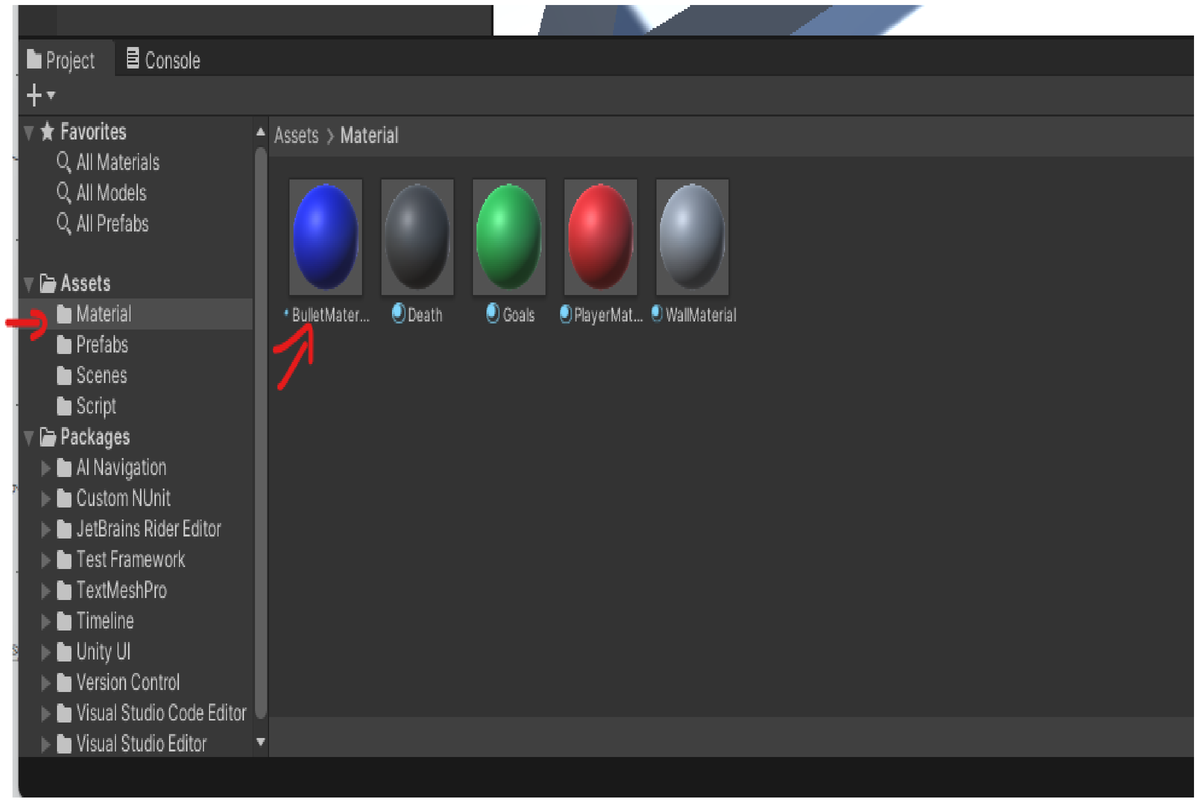


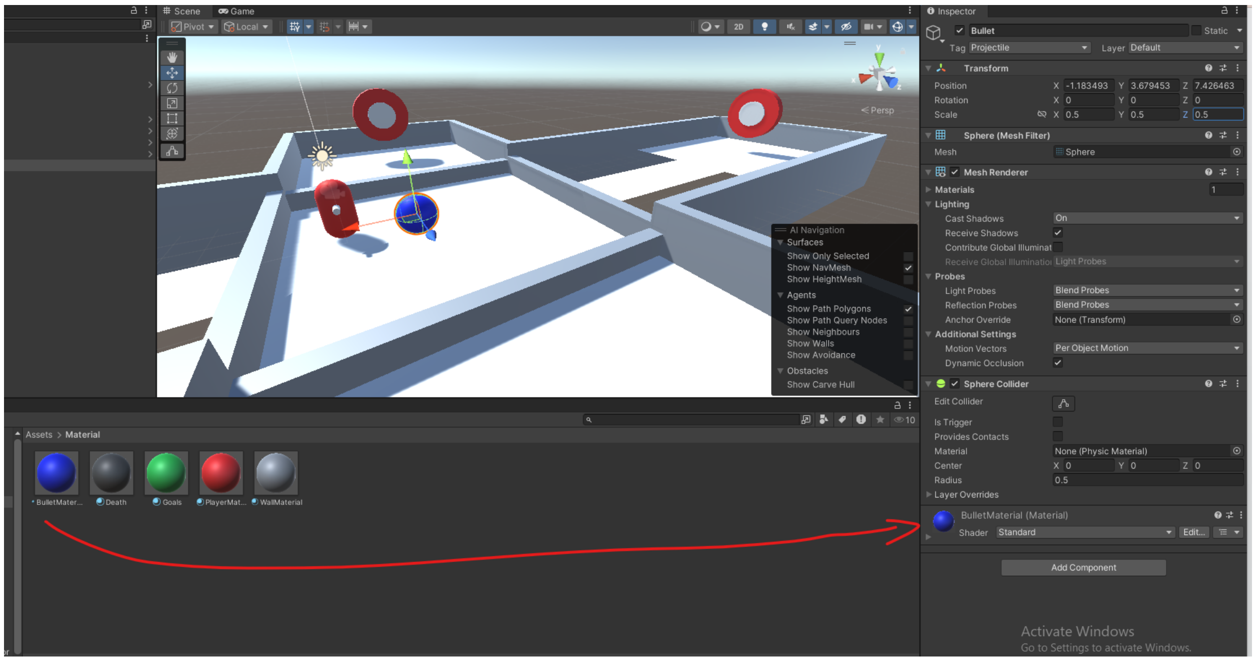
- 

- Make sure you choose projectile for you bullet after creating the tag.



4.
5. Apply the bullet material





Bullet ✓ Static
Tag Projectile Layer Default

Transform
Position X -1.183493 Y 3.679453 Z 7.426463
Rotation X 0 Y 0 Z 0
Scale X 0.5 Y 0.5 Z 0.5

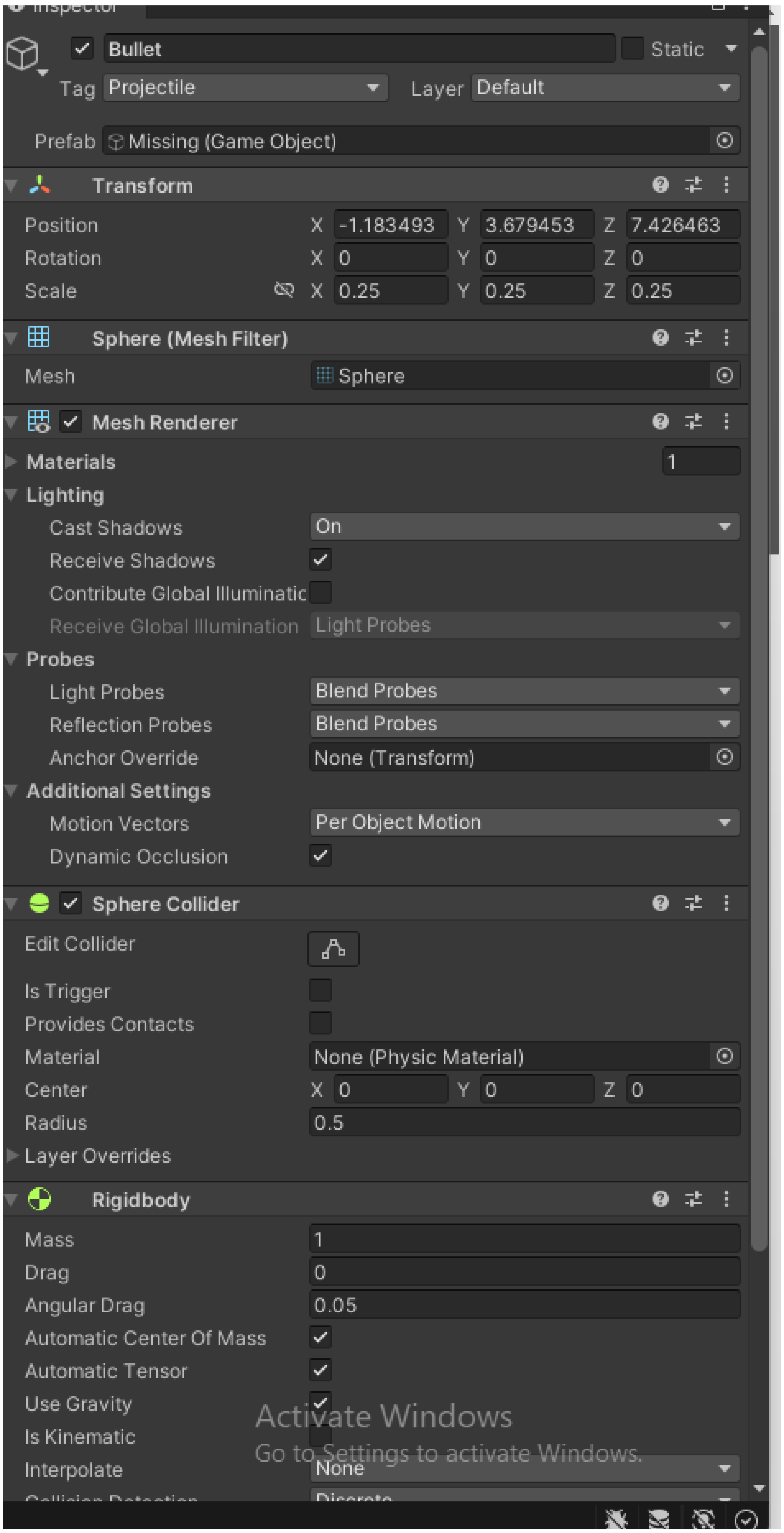
Sphere (Mesh Filter)
Mesh Sphere

✓ Mesh Renderer
Materials 1
Lighting
Cast Shadows On
Receive Shadows ✓
Contribute Global Illuminat
Receive Global Illuminatio Light Probes
Probes
Light Probes Blend Probes
Reflection Probes Blend Probes
Anchor Override None (Transform)
Additional Settings
Motion Vectors Per Object Motion
Dynamic Occlusion ✓

✓ Sphere Collider
Edit Collider
Is Trigger
Provides Contacts
Material None (Physic Material)
Center X 0 Y 0 Z 0
Radius 0.5
Layer Overrides

BulletMaterial (Material)
Shader Standard Edit...

- Your bullet should look something like this :


Inspector

✓ Bullet | Static

Tag: Projectile | Layer: Default

Prefab: Missing (Game Object)

Transform

	X	Y	Z
Position	-1.183493	3.679453	7.426463
Rotation	0	0	0
Scale	0.25	0.25	0.25

Sphere (Mesh Filter)

Mesh: Sphere

✓ **Mesh Renderer**

Materials: 1

Lighting
- Cast Shadows: On
- Receive Shadows: ✓
- Contribute Global Illuminatic: ☐
- Receive Global Illumination: Light Probes

Probes
- Light Probes: Blend Probes
- Reflection Probes: Blend Probes
- Anchor Override: None (Transform)

Additional Settings
- Motion Vectors: Per Object Motion
- Dynamic Occlusion: ✓

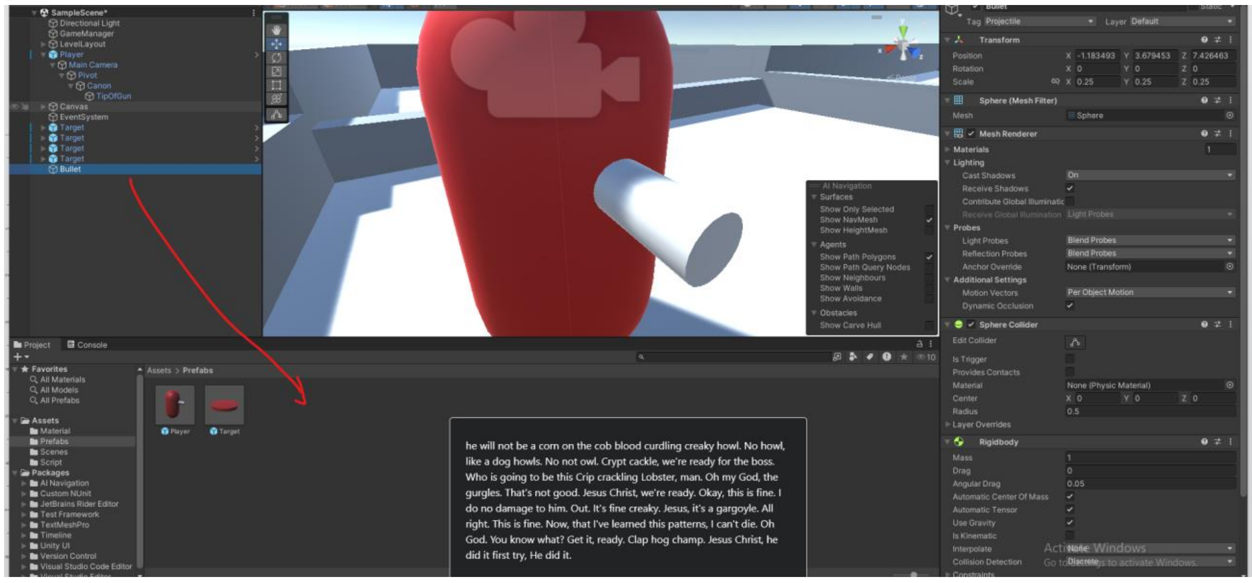
✓ **Sphere Collider**

- Edit Collider
- Is Trigger: ☐
- Provides Contacts: ☐
- Material: None (Physic Material)
- Center: X 0 Y 0 Z 0
- Radius: 0.5
- Layer Overrides

Rigidbody

- Mass: 1
- Drag: 0
- Angular Drag: 0.05
- Automatic Center Of Mass: ✓
- Automatic Tensor: ✓
- Use Gravity: ✓
- Is Kinematic: ☐
- Interpolate: None
- Collision Detection: Discrete

○

6. **Save it as a Prefab**:
   - Drag the sphere from the *Hierarchy* into the *Project* panel to save it as a prefab.
   - 

Bullet
Tag Projectile Layer Default

Transform
Position X -1.183493 Y 3.679453 Z 7.426463
Rotation X 0 Y 0 Z 0
Scale X 0.25 Y 0.25 Z 0.25

Sphere (Mesh Filter)
Mesh Sphere

Mesh Renderer
Materials 1
Lighting
Cast Shadows On
Receive Shadows ✓
Contribute Global Illuminatic
Receive Global Illumination Light Probes
Probes
Light Probes Blend Probes
Reflection Probes Blend Probes
Anchor Override None (Transform)
Additional Settings
Motion Vectors Per Object Motion
Dynamic Occlusion ✓

Sphere Collider
Edit Collider
Is Trigger
Provides Contacts
Material None (Physic Material)
Center X 0 Y 0 Z 0
Radius 0.5
Layer Overrides

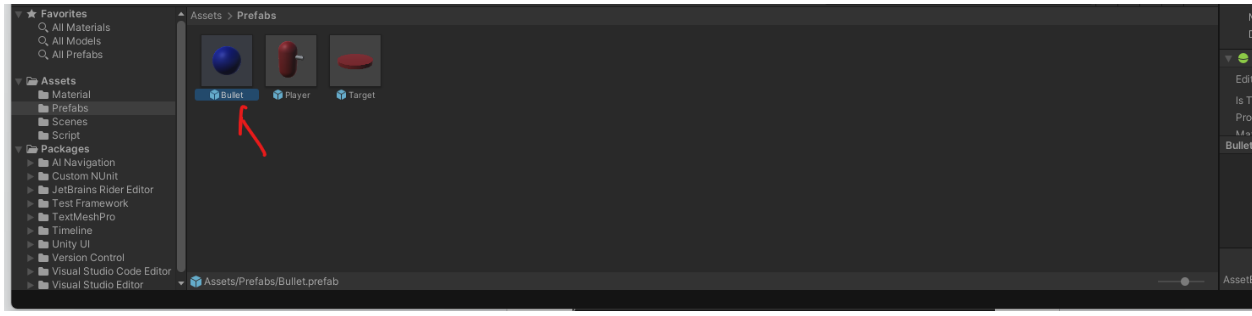
Rigidbody
Mass 1
Drag 0
Angular Drag 0.05
Automatic Center Of Mass ✓
Automatic Tensor ✓
Use Gravity ✓
Is Kinematic
Interpolate None
Collision Detection Discrete
Constraints

he will not be a corn on the cob blood curdling creaky howl. No howl, like a dog howls. No not owl. Crypt cackle, we're ready for the boss. Who is going to be this Crip crackling Lobster, man. Oh my God, the gurgles. That's not good. Jesus Christ, we're ready. Okay, this is fine. I do no damage to him. Out. It's fine creaky. Jesus, it's a gargoyle. All right. This is fine. Now, that I've learned this patterns, I can't die. Oh God. You know what? Get it, ready. Clap hog champ. Jesus Christ, he did it first try, He did it.

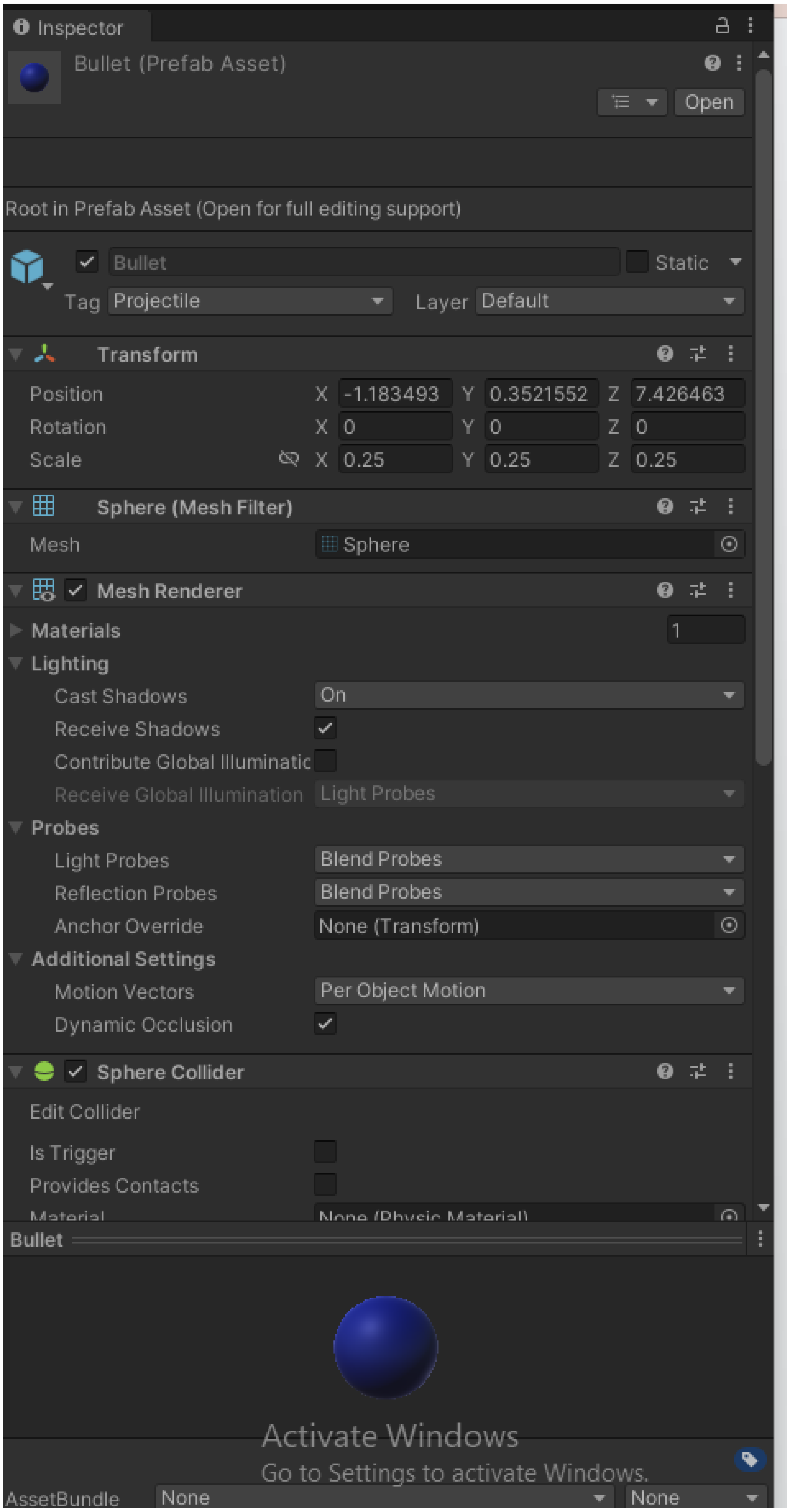
Task 2: Create a BulletComponent Script

1. **Create a Script**:
   - In the *Project* panel, right-click and select Create > C# Script.
   - Name it BulletComponent.
2. **Edit the Script**:
   - Open BulletComponent.cs in the code editor.
   - The bullet component will contain the code for destroying the ball

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;



public class BulletComponent : MonoBehaviour
{

    void Start()
    {
        // Destroy object after a few seconds
        Destroy(gameObject, 5f);
    }
}
```

3. **Attach the Script**:
    - Attach the BulletComponent script to your bullet prefab by dragging it from the *Project* panel to the prefab's *Inspector*.
      

    -



Inspector

Bullet (Prefab Asset)

Open

Root in Prefab Asset (Open for full editing support)

✓ Bullet ☐ Static

Tag Projectile Layer Default

Transform

	X	Y	Z
Position	-1.183493	0.3521552	7.426463
Rotation	0	0	0
Scale	0.25	0.25	0.25

Sphere (Mesh Filter)

Mesh: Sphere

✓ **Mesh Renderer**

Materials: 1

Lighting
- Cast Shadows: On
- Receive Shadows: ✓
- Contribute Global Illuminatic: ☐
- Receive Global Illumination: Light Probes

Probes
- Light Probes: Blend Probes
- Reflection Probes: Blend Probes
- Anchor Override: None (Transform)

Additional Settings
- Motion Vectors: Per Object Motion
- Dynamic Occlusion: ✓

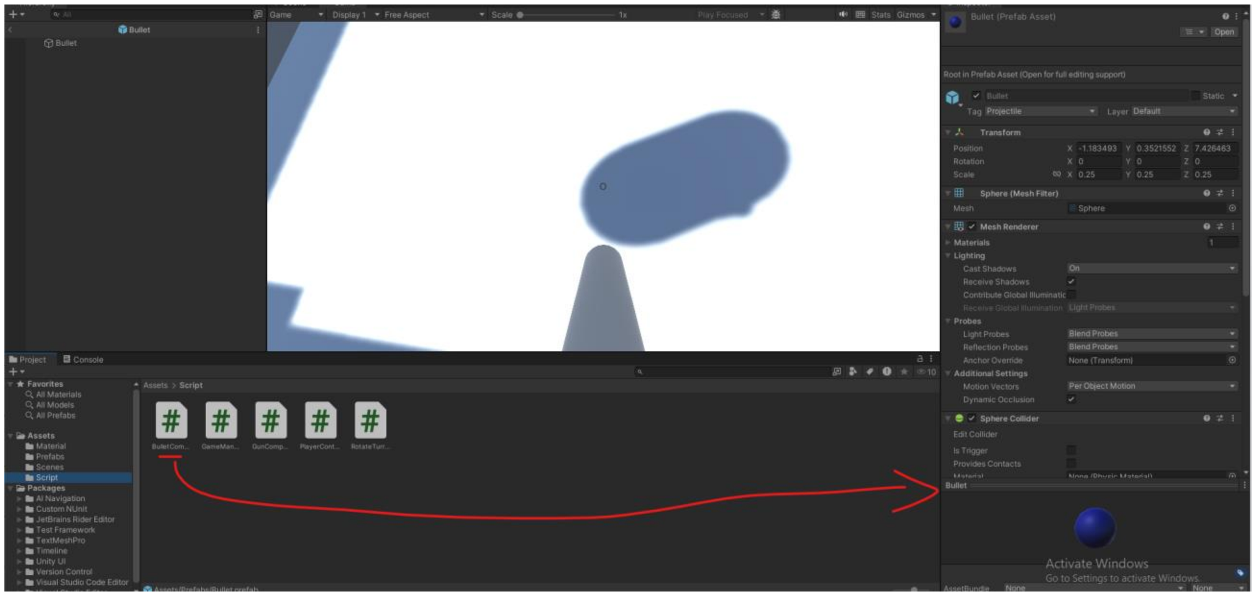
✓ **Sphere Collider**

Edit Collider
- Is Trigger: ☐
- Provides Contacts: ☐
- Material: None (Physic Material)

Bullet

Activate Windows
Go to Settings to activate Windows.

AssetBundle None None



Bullet (Prefab Asset)
Root in Prefab Asset (Open for full editing support)
Bullet — Static
Tag: Projectile — Layer: Default

Transform
Position: X -1.183493, Y 0.3521552, Z 7.426463
Rotation: X 0, Y 0, Z 0
Scale: X 0.25, Y 0.25, Z 0.25

Sphere (Mesh Filter)
Mesh: Sphere

Mesh Renderer
Materials: 1
Lighting
Cast Shadows: On
Receive Shadows: ✓
Contribute Global Illumination
Receive Global Illumination: Light Probes
Probes
Light Probes: Blend Probes
Reflection Probes: Blend Probes
Anchor Override: None (Transform)
Additional Settings
Motion Vectors: Per Object Motion
Dynamic Occlusion: ✓

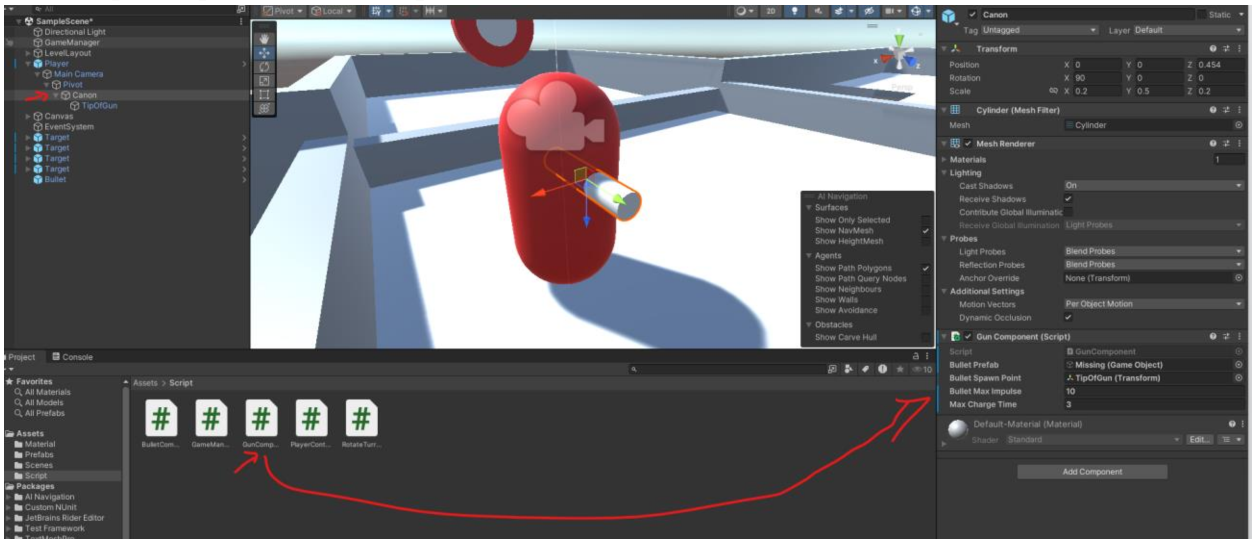
Sphere Collider
Edit Collider
Is Trigger
Provides Contacts
Material: None (Physic Material)

AssetBundle: None — None

Task 3: Modify the Gun component to Spawn the bullets

1. **Edit the Script**:
    - Open GunComponent.cs and implement the following:

```
using UnityEngine;

public class GunComponent : MonoBehaviour
{
    public GameObject bulletPrefab;
    public Transform bulletSpawnPoint;
    public float bulletMaxImpulse = 100.0f;
    public float maxChargeTime = 3.0f;
    private float chargeTime = 0.0f;
    private bool isCharging = false;

    void Update()
    {
        // TODO add the logic to track player keeping the input down.
        if (Input.GetButtonUp("Fire1"))
        {
            ShootBullet();
        }
    }


    void ShootBullet()
    {
        GameObject bullet = Instantiate(bulletPrefab, bulletSpawnPoint.position,
bulletSpawnPoint.rotation);
        Rigidbody rb = bullet.GetComponent<Rigidbody>();

        // TODO change that equation so that it adds an impulse that follows
charge time
        float bulletImpulse = bulletMaxImpulse;

        // An impulse is a force you apply on a object in a single instant.
        rb.AddForce(bulletSpawnPoint.forward * bulletImpulse, ForceMode.Impulse);
    }
}
```

2. **Configure the Gun**:
   - The gun component should already be assigned to the Gun gameObject (A child of the player) You will need to assign the bullet prefab and the bullet spawn point in the *Inspector*.
   - 

Canon — Tag: Untagged — Layer: Default

Transform: Position X 0, Y 0, Z 0.454; Rotation X 90, Y 0, Z 0; Scale X 0.2, Y 0.5, Z 0.2

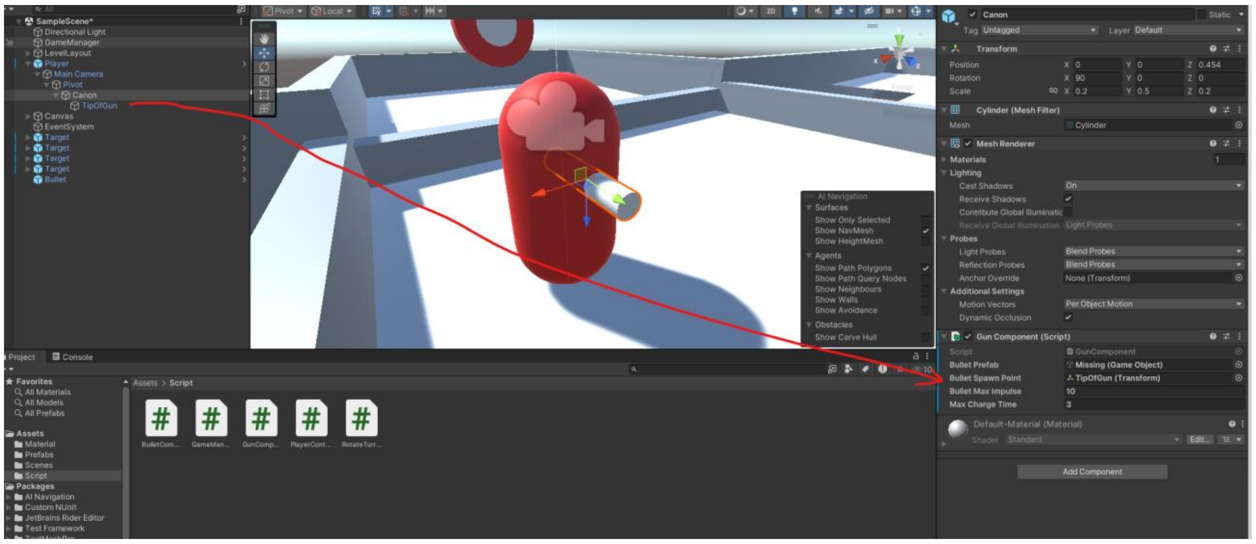
Cylinder (Mesh Filter): Mesh Cylinder

Mesh Renderer: Cast Shadows On; Receive Shadows ✓; Light Probes Blend Probes; Reflection Probes Blend Probes; Anchor Override None (Transform); Motion Vectors Per Object Motion; Dynamic Occlusion ✓

Gun Component (Script): Script GunComponent; Bullet Prefab Missing (Game Object); Bullet Spawn Point TipOfGun (Transform); Bullet Max Impulse 10; Max Charge Time 3

Default-Material (Material): Shader Standard

Assign **the tip of gun transform** as the bullet spawn point.

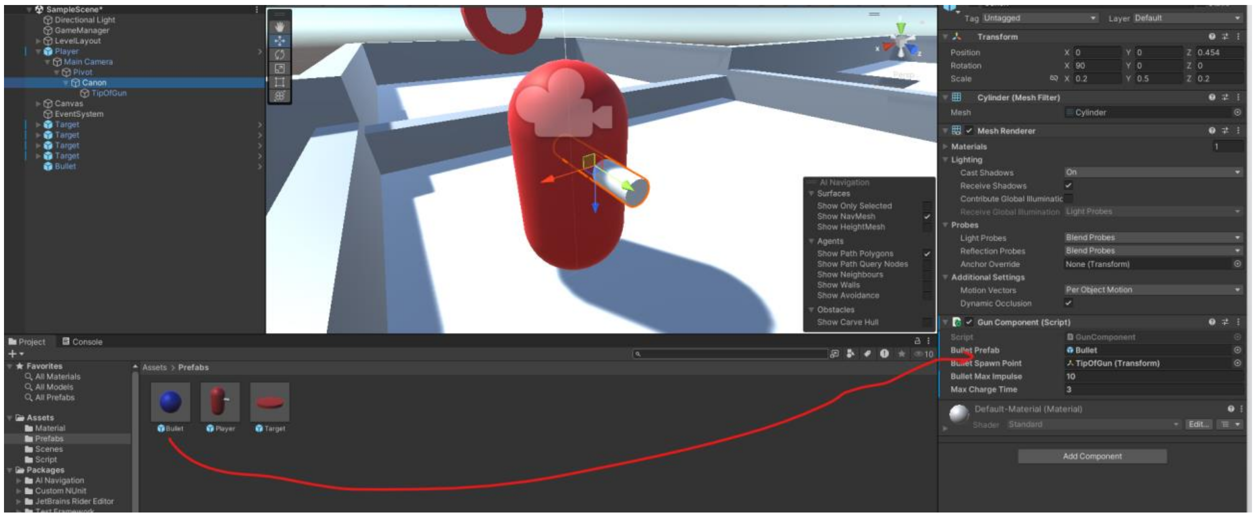


Canon — Tag: Untagged — Layer: Default

Transform: Position X 0, Y 0, Z 0.454; Rotation X 90, Y 0, Z 0; Scale X 0.2, Y 0.5, Z 0.2

Gun Component (Script): Script GunComponent; Bullet Prefab Missing (Game Object); Bullet Spawn Point TipOfGun (Transform); Bullet Max Impulse 10; Max Charge Time 3

And the bullet prefab as the Bullet Prefab Proprety



Tag Untagged Layer Default

Transform

Position	X 0	Y 0	Z 0.454
Rotation	X 90	Y 0	Z 0
Scale	X 0.2	Y 0.5	Z 0.2

Cylinder (Mesh Filter)

Mesh	Cylinder

Mesh Renderer

Materials	1
Cast Shadows	On
Receive Shadows	✓
Contribute Global Illumination	
Receive Global Illumination	Light Probes
Light Probes	Blend Probes
Reflection Probes	Blend Probes
Anchor Override	None (Transform)
Motion Vectors	Per Object Motion
Dynamic Occlusion	✓

Gun Component (Script)

Script	GunComponent
Bullet Prefab	Bullet
Bullet Spawn Point	TipOfGun (Transform)
Bullet Max Impulse	10
Max Charge Time	3

Default-Material (Material)
Shader Standard

Add Component

Task 5: Modify the Gun Component

You will need now to modify the gun component Update function to add a charging mechanic for bullet speed. (IE Charge longer, Bullet goes further)

Here are few hints :

- This will be done within the Update() function.
- You need to detect when the player starts holding the fire button. Use an if condition with the boolean Input.GetButtonDown("Fire1") to detect when the button is initially pressed.
  - When a player first press down the button the chargetime should reset to 0.

- Using the boolean Input.GetButton("Fire1") increments the charge time every frame the button is held down. By adding the Time.deltaTime to the charge time.
  - You should increase the chargeTime value over time using Time.deltaTime, which gives you the time passed between each frame.

- To avoid excessively fast bullets, you can apply a maximum limit to the chargeTime.
  - Use Mathf.Clamp to keep the chargeTime between 0 and a certain max value (like 3 seconds).

    -
      ```
      chargeTime = Mathf.Clamp(chargeTime, 0, maxChargeTime);
      ```

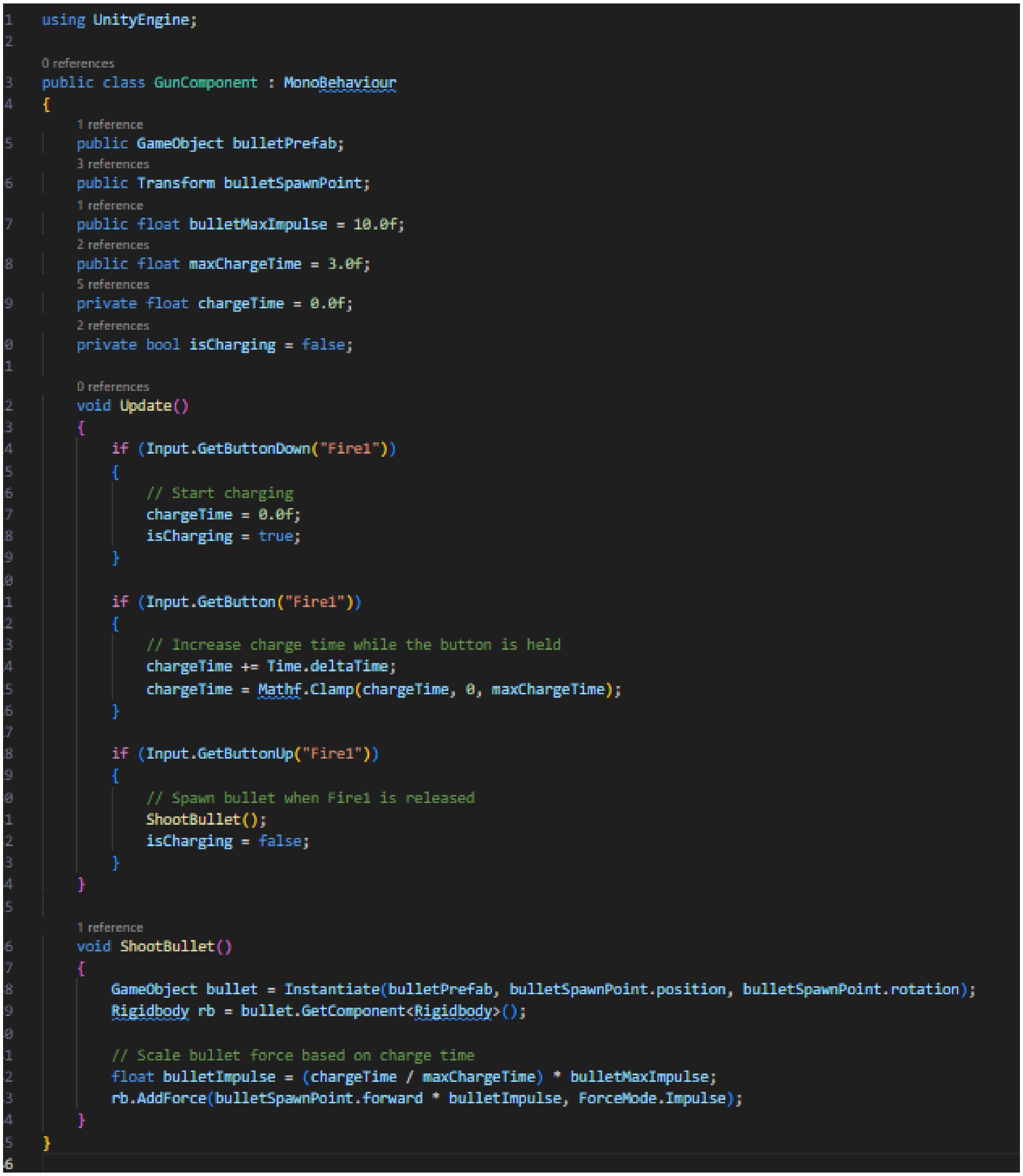
Copy the code you came up with here :

```
void Update()
  {


   if (Input.GetButtonDown("Fire1")) {
       isCharging = true;
       chargeTime = 0f;
   }
   if (Input.GetButton("Fire1") && isCharging) {
       chargeTime += Time.deltaTime;
       chargeTime = Mathf.Clamp(chargeTime, 0f, maxChargeTime);
   }
   if (Input.GetButtonUp("Fire1") && isCharging) {
       ShootBullet();
       isCharging = false;
       chargeTime = 0f;
   }


  }
```

- Now change the ShootBullet function to

```
void ShootBullet()

{

        GameObject bullet = Instantiate(bulletPrefab, bulletSpawnPoint.position, 
bulletSpawnPoint.rotation);

        Rigidbody rb = bullet.GetComponent<Rigidbody>();
        // Scale bullet force based on charge time

        float bulletImpulse = (chargeTime / maxChargeTime) * bulletMaxImpulse;

        rb.AddForce(bulletSpawnPoint.forward * bulletImpulse, ForceMode.Impulse);

    }
```

In the end your GunComponent should look something like this **(NOTE : the next code snippet is a picture)**



```
using UnityEngine;

0 references
public class GunComponent : MonoBehaviour
{
    1 reference
    public GameObject bulletPrefab;
    3 references
    public Transform bulletSpawnPoint;
    1 reference
    public float bulletMaxImpulse = 10.0f;
    2 references
    public float maxChargeTime = 3.0f;
    5 references
    private float chargeTime = 0.0f;
    2 references
    private bool isCharging = false;

    0 references
    void Update()
    {
        if (Input.GetButtonDown("Fire1"))
        {
            // Start charging
            chargeTime = 0.0f;
            isCharging = true;
        }

        if (Input.GetButton("Fire1"))
        {
            // Increase charge time while the button is held
            chargeTime += Time.deltaTime;
            chargeTime = Mathf.Clamp(chargeTime, 0, maxChargeTime);
        }

        if (Input.GetButtonUp("Fire1"))
        {
            // Spawn bullet when Fire1 is released
            ShootBullet();
            isCharging = false;
        }
    }

    1 reference
    void ShootBullet()
    {
        GameObject bullet = Instantiate(bulletPrefab, bulletSpawnPoint.position, bulletSpawnPoint.rotation);
        Rigidbody rb = bullet.GetComponent<Rigidbody>();

        // Scale bullet force based on charge time
        float bulletImpulse = (chargeTime / maxChargeTime) * bulletMaxImpulse;
        rb.AddForce(bulletSpawnPoint.forward * bulletImpulse, ForceMode.Impulse);
    }
}
```

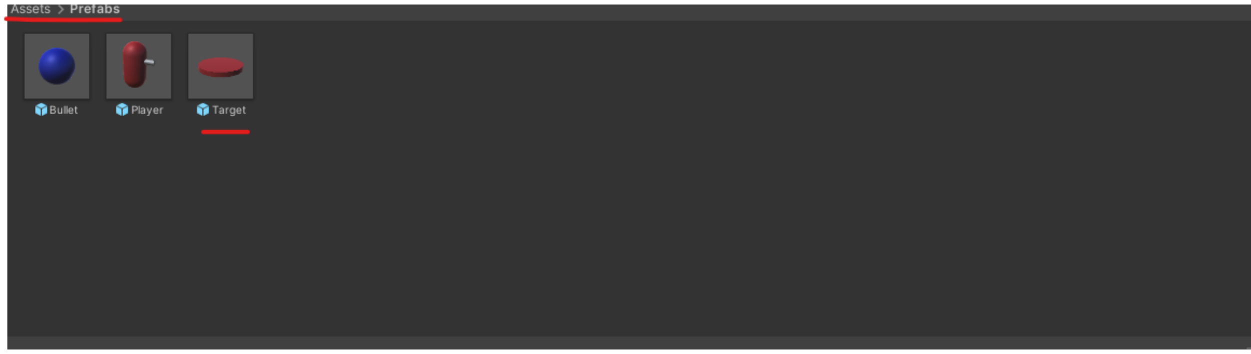
Task 4: Target Interaction and Scoring

1. **Create TargetComponent Script**:

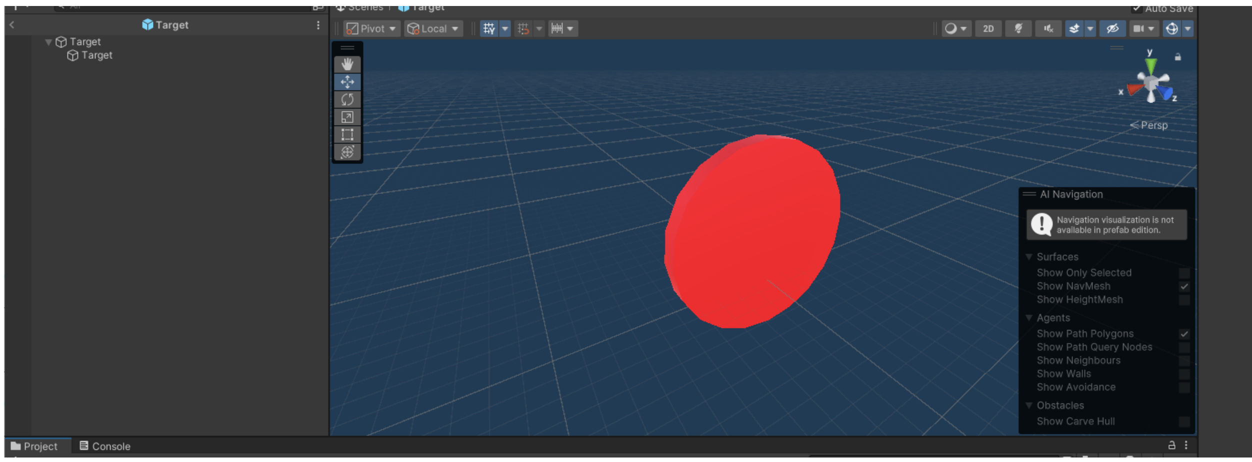
  - In the *Project* panel, create a new C# script called TargetComponent.
2. **Edit the Script**:
  - Open TargetComponent.cs and implement collision handling:

```
using UnityEngine;
public class TargetComponent : MonoBehaviour
{
    private Renderer targetRenderer;
    private Color originalColor;
    public Color hitColor = Color.green; // Change to any color you want

    private void Start()
    {
        targetRenderer = GetComponent<Renderer>();
        if (targetRenderer != null)
        {
            originalColor = targetRenderer.material.color;
        }
    }

    private void OnCollisionEnter(Collision collision)
    {
        if (collision.gameObject.CompareTag("Projectile"))
        {
            GameManager.Instance.IncrementScore();

            // Change color
            if (targetRenderer != null)
            {
                targetRenderer.material.color = hitColor;
            }

            // Restore color and hide target after 5 seconds
            Invoke("ResetColor", 5f);
        }
    }

    private void ResetColor()
    {
        if (targetRenderer != null)
        {
            targetRenderer.material.color = originalColor;
        }
    }
}
```

3. **Attach the Script**:
   - Select the target prefab in your Prefab folder and double click it.

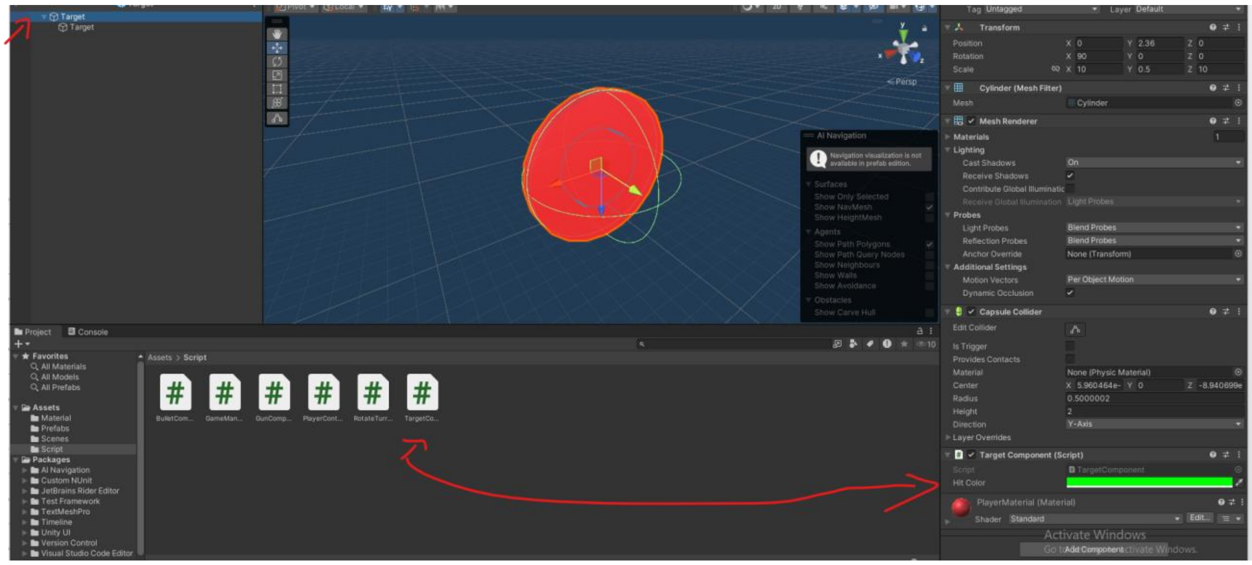


   - It will change the viewport to a blue background with only your prefab where the scene would be



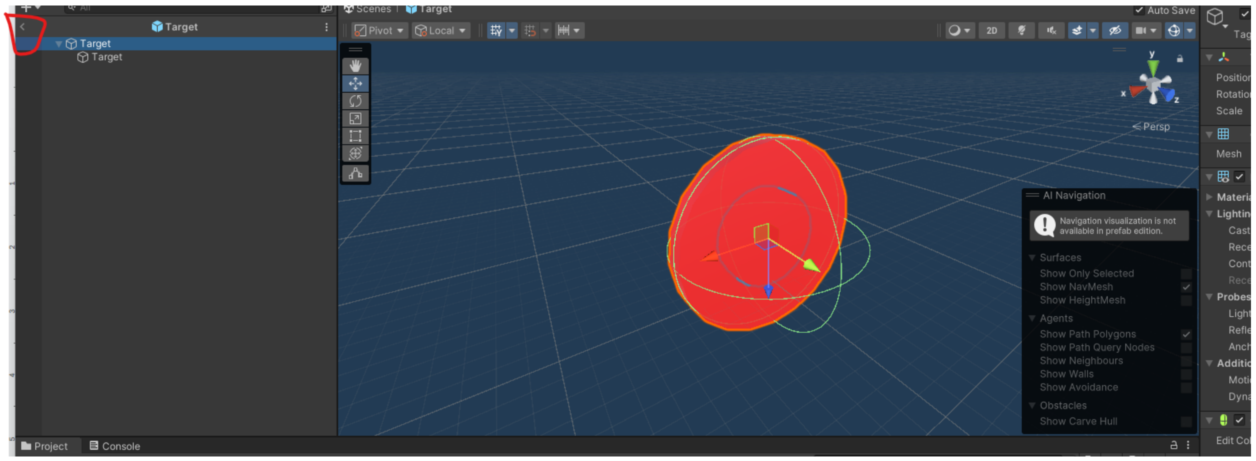
Navigation visualization is not available in prefab edition.

   -
   - Now click on the parent object and add the TargetComponent to it



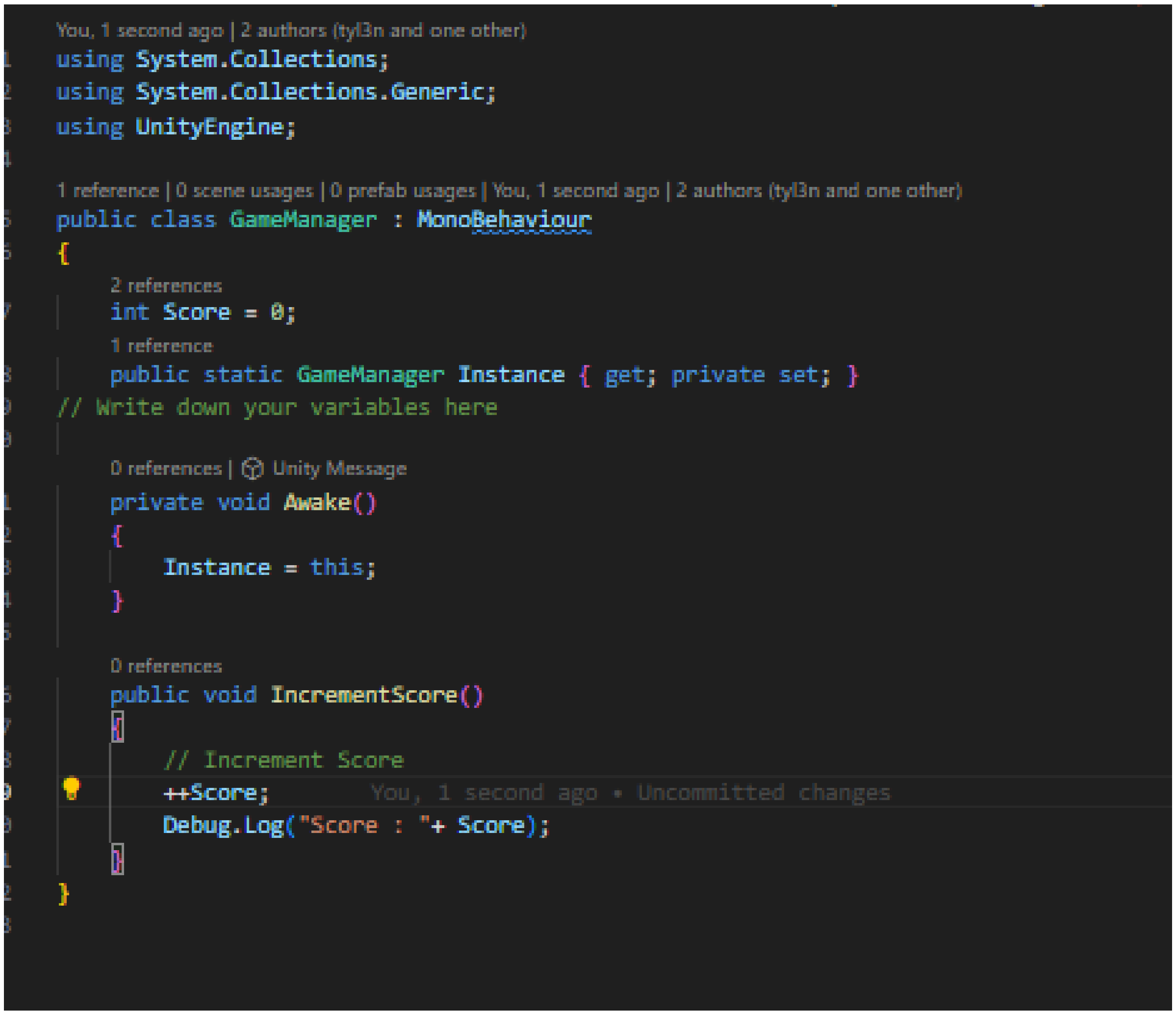
-
- This will add the target component on all the Target object in your scene.

You can go back to your scene by pressing the arrow



4. **Connect to GameManager**:
   - Ensure your GameManager script has an IncrementScore() method. This will update the score when a bullet hits the target.

     Make sure the GameManager IncrementScore Is properly implemented



```
You, 1 second ago | 2 authors (tyl3n and one other)
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

1 reference | 0 scene usages | 0 prefab usages | You, 1 second ago | 2 authors (tyl3n and one other)
public class GameManager : MonoBehaviour
{
    2 references
    int Score = 0;
    1 reference
    public static GameManager Instance { get; private set; }
// Write down your variables here

    0 references | Unity Message
    private void Awake()
    {
        Instance = this;
    }

    0 references
    public void IncrementScore()
    {
        // Increment Score
        ++Score;        You, 1 second ago • Uncommitted changes
        Debug.Log("Score : "+ Score);
    }
}
```

Submission

1. **Add this file to your Asset folder and commit to your repo.**