Title: Bouncing of Objects

EXPERIMENT NO: 6 **Name:** SHAKYA JAY JITENDRA

Date:30-8-2025 **Reg. No:** 24BCG10123

Aim: To implement and observe a bouncing object system in Unity using mass and gravity, and the physics material of Unity.

Description/Concept:

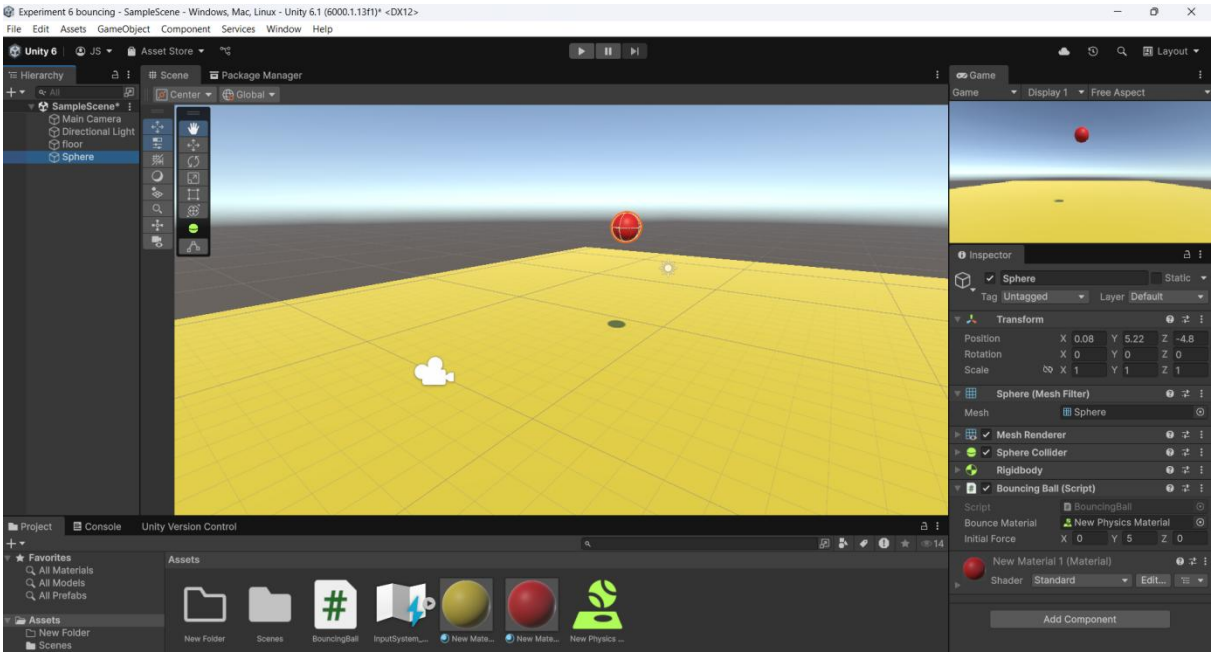
This experiment demonstrates the mechanics of gravity and collision in a 3D environment by using Unity's built-in physics system. A ball GameObject is equipped with a Rigidbody component to enable physics calculations such as gravity and momentum, and a SphereCollider to detect collisions with surfaces. A Physics Material with a high bounciness value is applied to the SphereCollider to ensure the ball bounces upon hitting the ground or other colliders. The interaction between gravity, mass, and collision response causes the ball to fall, collide, and bounce, mimicking real-world physics. Parameters like mass, drag, and bounciness can be adjusted to observe their effects on the ball's motion.

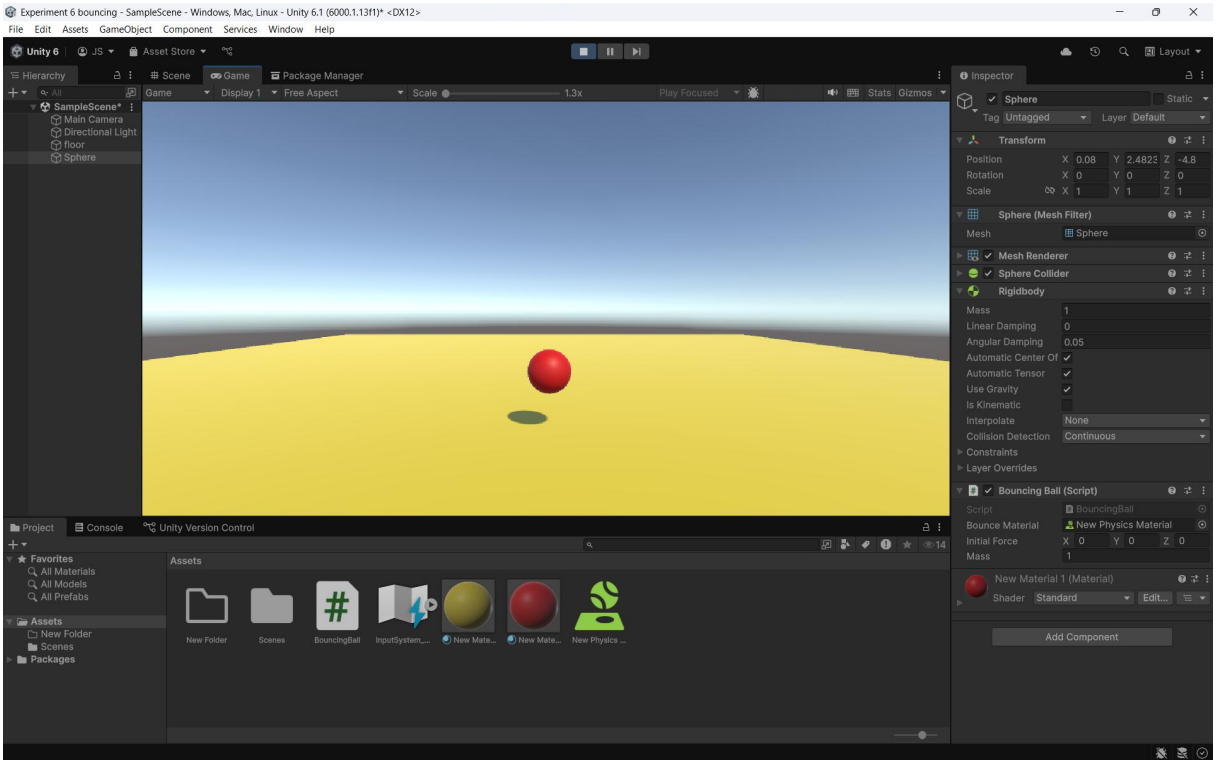
Program/Coding:

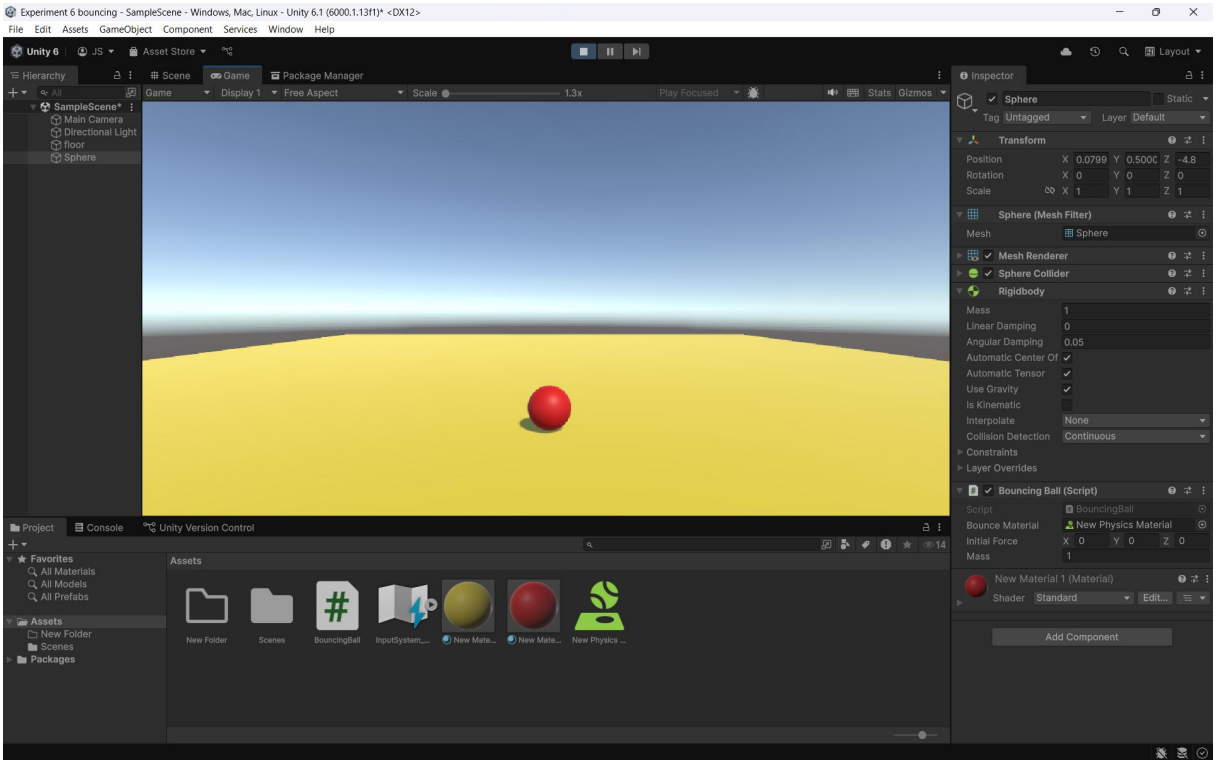
```
using UnityEngine;

[RequireComponent(typeof(Rigidbody), typeof(SphereCollider))]
public class BouncingBall : MonoBehaviour
{
    public PhysicsMaterial bounceMaterial;
    public Vector3 initialForce = new Vector3(0, 5, 0); // Initial force to start
bounce
    public float mass = 1f; // Mass of the Rigidbody, can be set in Inspector

    private Rigidbody rb;
    private SphereCollider sphereCollider;

    void Start()
    {
        rb = GetComponent<Rigidbody>();
        sphereCollider = GetComponent<SphereCollider>();

        if (bounceMaterial != null)
        {
            sphereCollider.material = bounceMaterial;
        }

        rb.mass = mass;


        rb.AddForce(initialForce, ForceMode.Impulse);

        rb.useGravity = true;
    }
}
```

Output:

When the simulation runs, the ball falls under gravity, hits the floor or collision surface, and bounces back up. The height of the bounce decreases gradually if the material absorbs energy; otherwise, it can bounce indefinitely if set appropriately. The number of bounces and bounce height visually demonstrate energy loss and conversion between potential and kinetic energy due to gravity and collisions. The ball behaves dynamically, changing position and velocity frame-by-frame based on physics simulation.







Conclusion:

Unity's physics engine effectively models real-world bouncing ball dynamics with minimal programming by leveraging Rigidbody, SphereCollider, and Physics Materials. The mass of the ball and gravity define the acceleration and momentum, while the bounciness of the physics material influences how much energy is retained after each bounce. Fine-tuning physics parameters allows for realistic or exaggerated bouncing effects, demonstrating fundamental physics concepts such as gravity, collision, impulse, and energy conservation in an interactive 3D environment. This experiment also highlights the importance of physics material settings and Rigidbody properties for achieving the desired bounce behavior

Result:

Script was created for bouncing of objects were successfully implemented and tested in Unity Game Engine 6.1 Their application in game mechanics was understood through coding and console outputs.