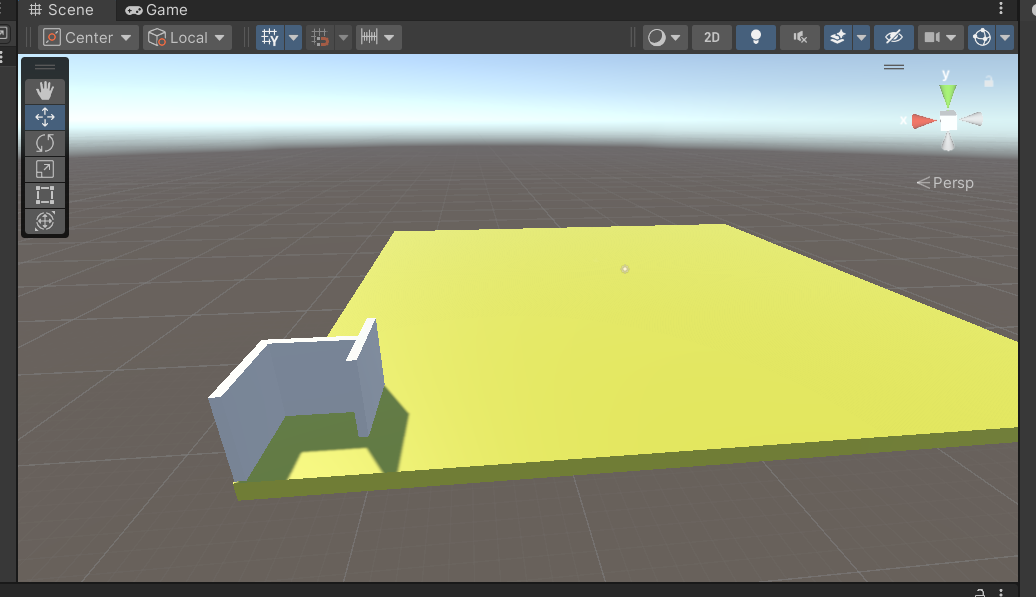
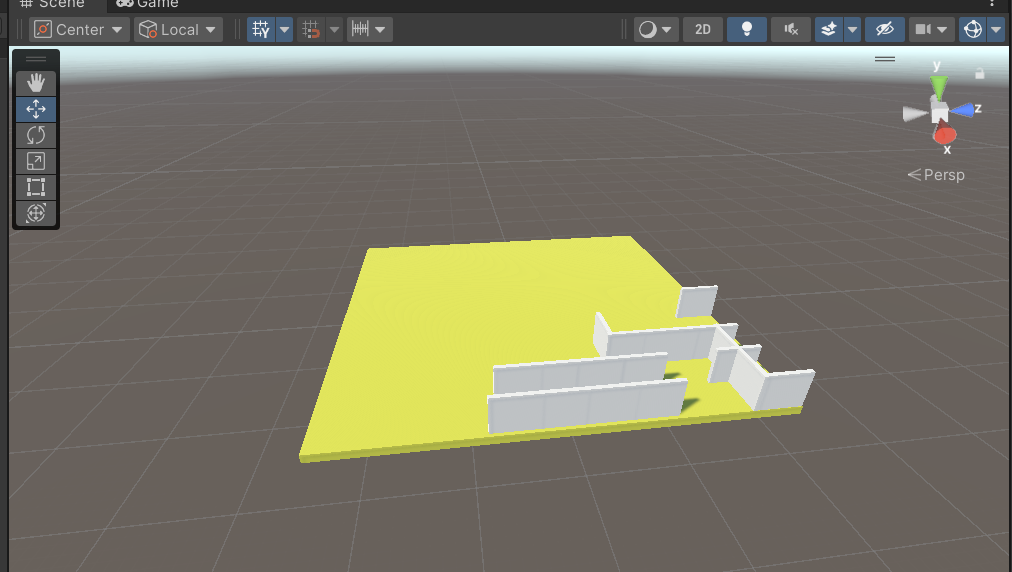
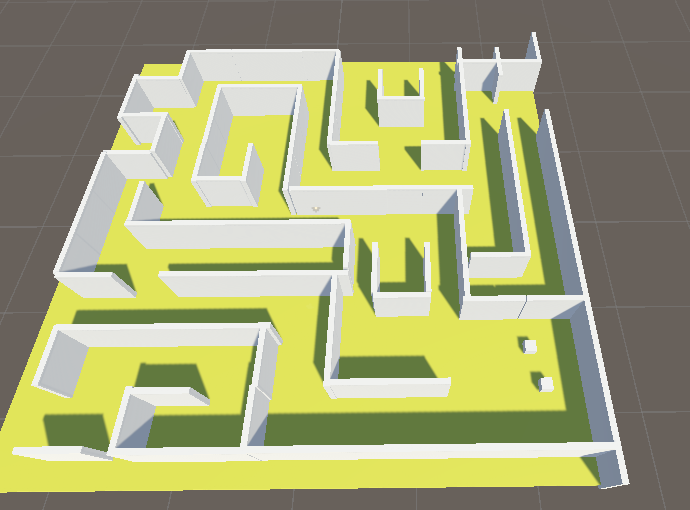
**Aim:** To create a 3D maze game

**Video**: <https://drive.google.com/file/d/14dasYG0YeugjIMswG1l2jEN74whufVhr/view?usp=sharing>

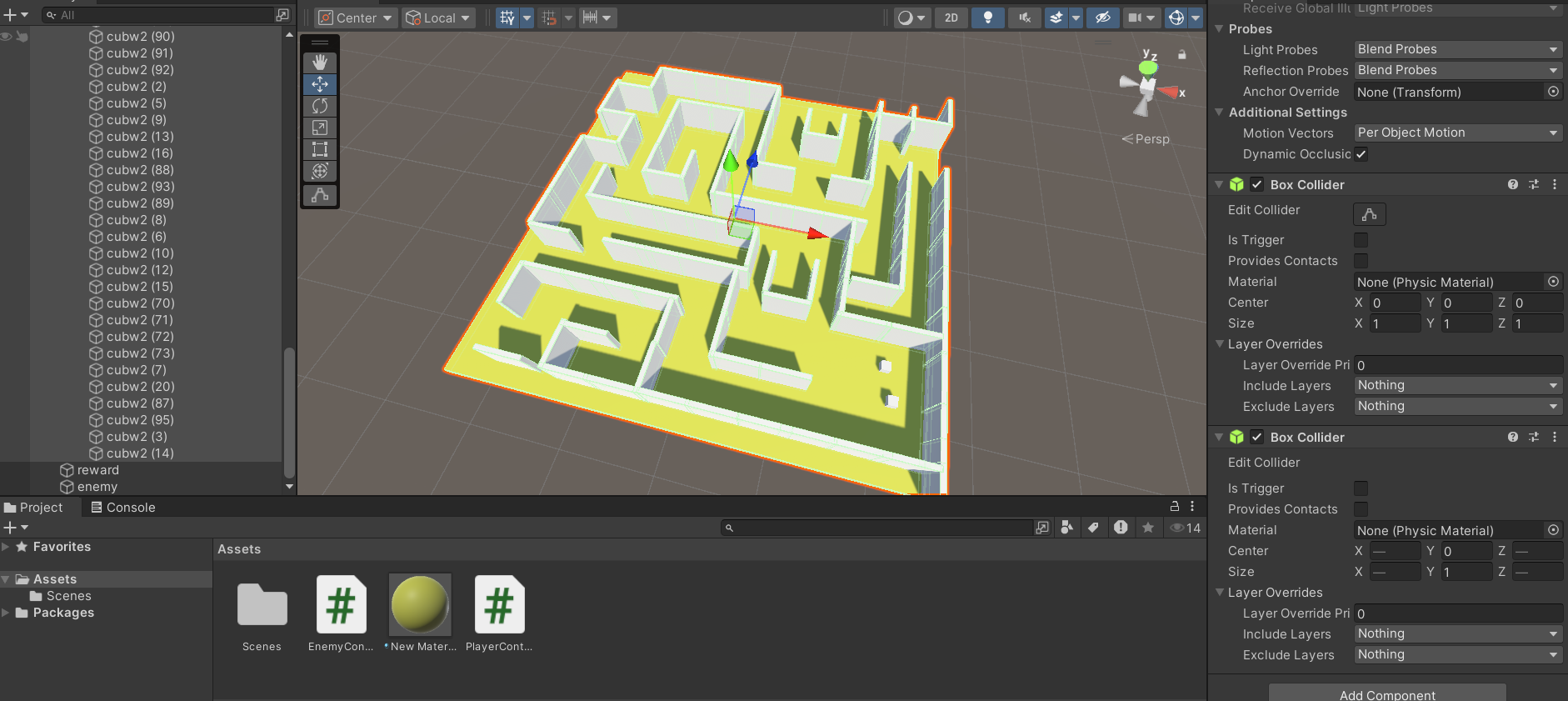
Screenshots:



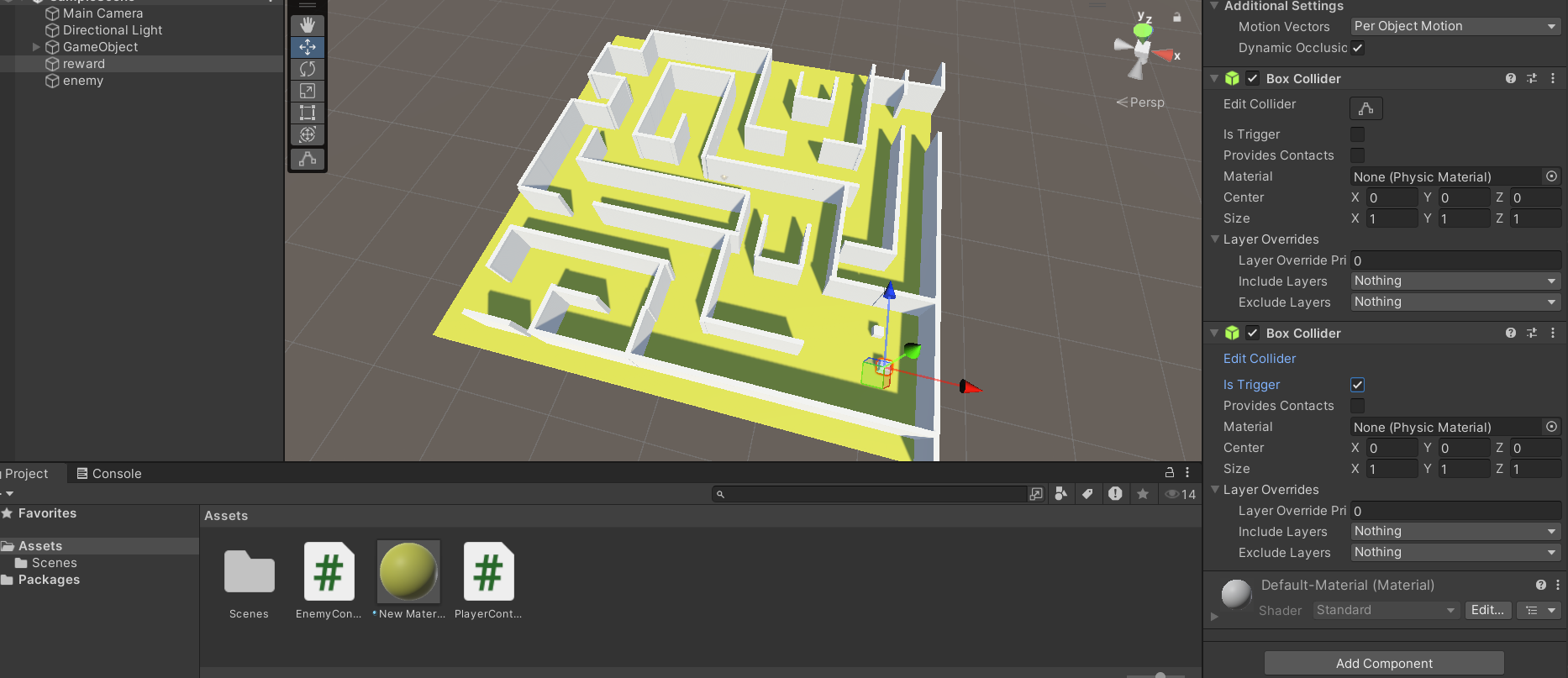




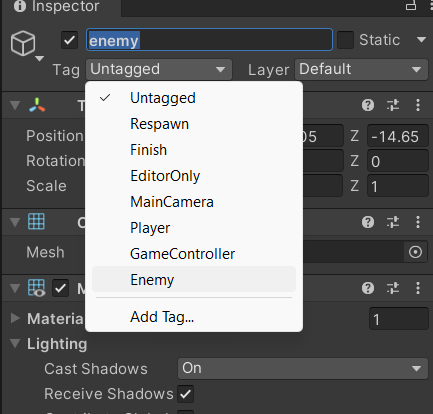
Add box collider to all the walls

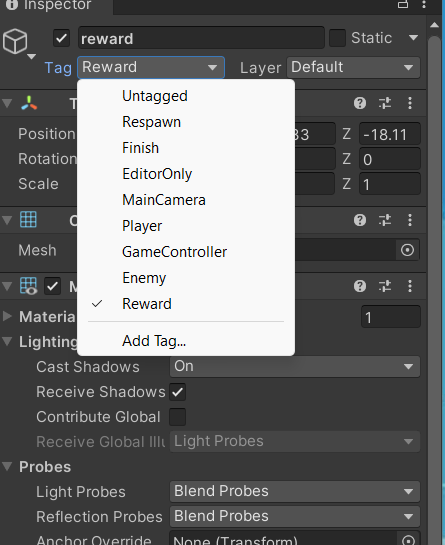


Add Box collider to enemy and reward

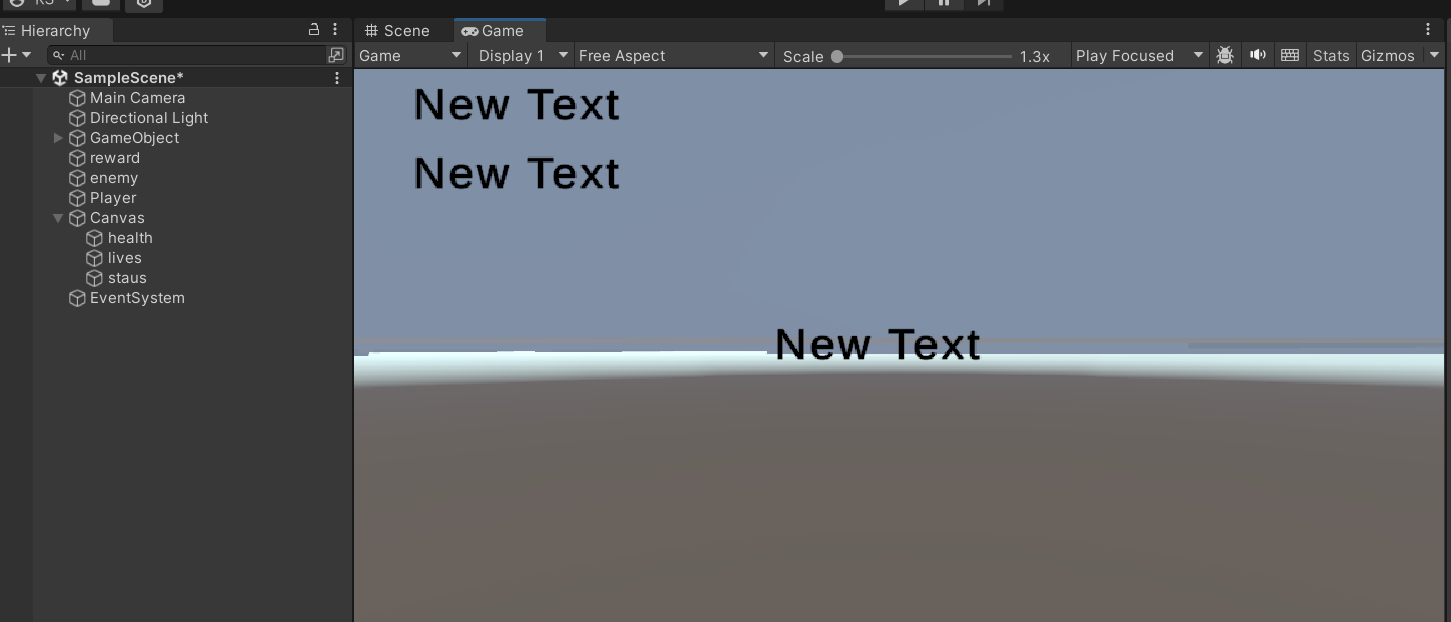


ADD tag enemy to enemy and reward to reward and player to player

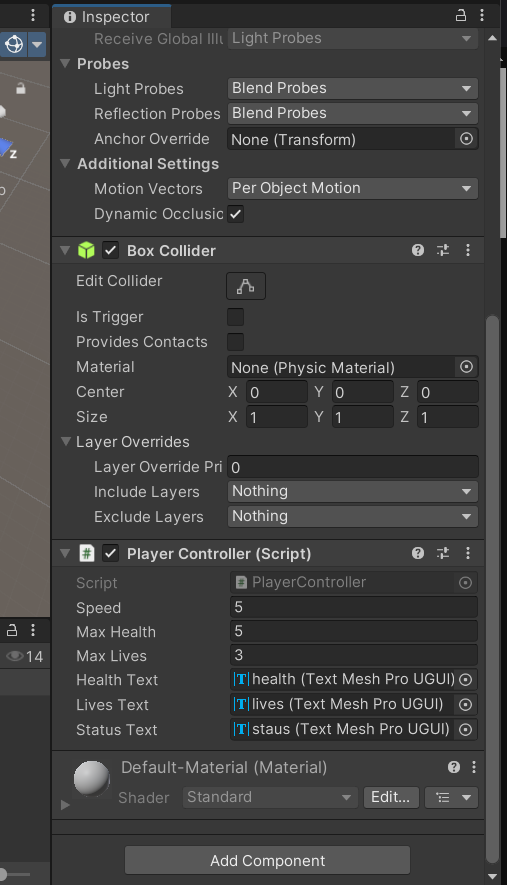




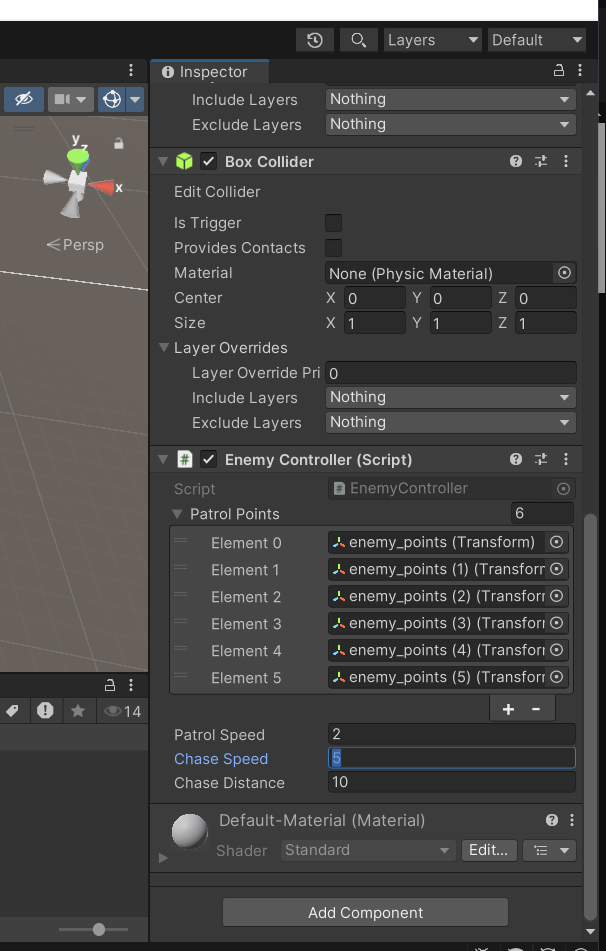
Add Text to display health life and status(game over/you won)



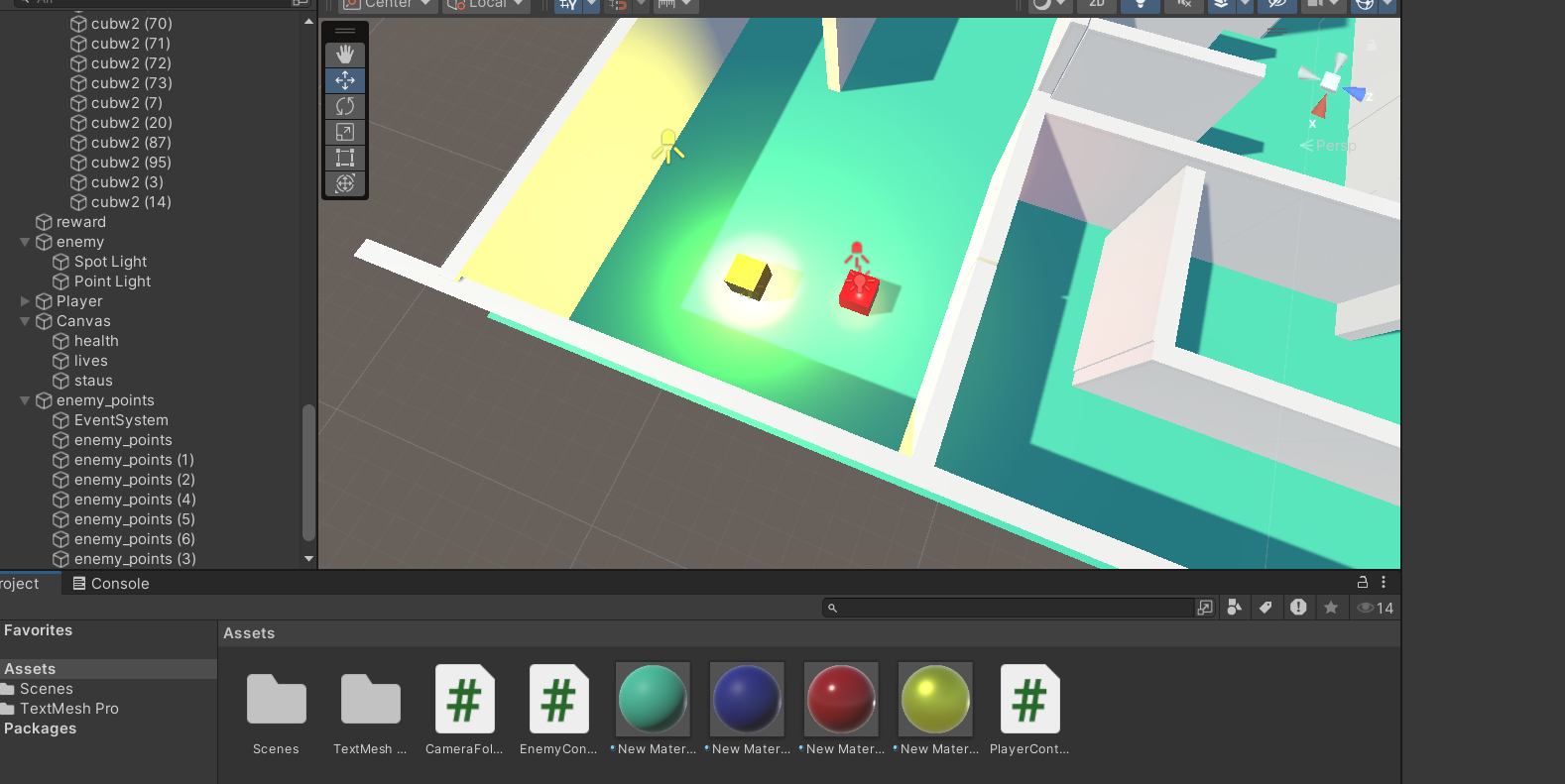
Add Script player controller to player



Add gamepoints( empty game objects markers) to locate where enemy should patrol



Add light to add visual effects



**Codes**

Player

using System.Collections;

using UnityEngine;

using TMPro;  // Import TextMeshPro namespace

public class PlayerController : MonoBehaviour

{

    public float speed = 5f;        // Player movement speed

    public float rotationSpeed = 100f;  // Speed of rotation around the Y-axis

    public int maxHealth = 5;       // Maximum health of the player

    public int maxLives = 3;        // Maximum lives player can have

    public TextMeshProUGUI healthText;  // UI TMP Text for health

    public TextMeshProUGUI livesText;   // UI TMP Text for lives

    public TextMeshProUGUI statusText;  // UI TMP Text for game status (e.g., You Won, Game Over)

    private int currentHealth;

    private int currentLives;

    private Vector3 spawnPoint;     // Player respawn point

    private void Start()

    {

        currentHealth = maxHealth;

        currentLives = maxLives;

        spawnPoint = new Vector3(22.99f, 2.08f, 22.44f); // Set initial spawn position

        UpdateUI();

        statusText.text = "";             // Clear status text at the start

    }

    private void Update()

    {

        MovePlayer();

    }

    void MovePlayer()

    {

        // Move the player forward/backward and sideways with arrow keys (or WASD keys)

        float moveHorizontal = Input.GetAxis("Horizontal");  // Left-Right movement (A/D or Left/Right Arrow)

        float moveVertical = Input.GetAxis("Vertical");      // Forward-Backward movement (W/S or Up/Down Arrow)

        Vector3 movement = new Vector3(moveHorizontal, 0.0f, moveVertical);

        transform.Translate(movement \* speed \* Time.deltaTime, Space.World);

        // Rotate player around the Y-axis when 'R' key is pressed

        if (Input.GetKey(KeyCode.R))

        {

            transform.Rotate(Vector3.up \* rotationSpeed \* Time.deltaTime);  // Rotate player around Y-axis

        }

    }

    private void OnTriggerEnter(Collider other)

    {

        if (other.gameObject.CompareTag("Enemy"))

        {

            StartCoroutine(TakeDamageOverTime());

        }

        else if (other.gameObject.CompareTag("Reward"))

        {

            statusText.text = "You Won!";

        }

    }

    IEnumerator TakeDamageOverTime()

    {

        while (true)

        {

            currentHealth -= 2;

            UpdateUI();

            if (currentHealth <= 0)

            {

                Respawn();

                break;

            }

            yield return new WaitForSeconds(1);  // Damage taken every 1 second

        }

    }

    void Respawn()

    {

        currentLives -= 1;

        if (currentLives <= 0)

        {

            GameOver();

        }

        else

        {

            currentHealth = maxHealth;

            transform.position = spawnPoint;

            UpdateUI();

        }

    }

    void GameOver()

    {

        statusText.text = "Game Over!";

        // Add logic here to stop the game

    }

    void UpdateUI()

    {

        healthText.text = "Health: " + currentHealth;

        livesText.text = "Lives: " + currentLives;

    }

}

Enemy

using System.Collections;

using UnityEngine;

public class EnemyController : MonoBehaviour

{

    public Transform[] patrolPoints;  // Waypoints for patrol

    public float patrolSpeed = 2f;    // Speed for patrol movement

    public float chaseSpeed = 4f;     // Speed when chasing player

    public float chaseDistance = 5f;  // Distance within which enemy will chase player

    private int currentPointIndex = 0;

    private Transform player;

    private bool isChasing = false;

    private void Start()

    {

        player = GameObject.FindGameObjectWithTag("Player").transform;

    }

    private void Update()

    {

        Patrol();

        float distanceToPlayer = Vector3.Distance(transform.position, player.position);

        if (distanceToPlayer <= chaseDistance)

        {

            isChasing = true;

        }

        else

        {

            isChasing = false;

        }

        if (isChasing)

        {

            ChasePlayer();

        }

    }

    void Patrol()

    {

        if (!isChasing)

        {

            Transform targetPoint = patrolPoints[currentPointIndex];

            transform.position = Vector3.MoveTowards(transform.position, targetPoint.position, patrolSpeed \* Time.deltaTime);

            if (Vector3.Distance(transform.position, targetPoint.position) < 0.2f)

            {

                currentPointIndex = (currentPointIndex + 1) % patrolPoints.Length;

            }

        }

    }

    void ChasePlayer()

    {

        transform.position = Vector3.MoveTowards(transform.position, player.position, chaseSpeed \* Time.deltaTime);

    }

}