Source Code

ClickToMove

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

namespace CompleteProject
{
    public class ClickToMove : MonoBehaviour
    {
        public float shootDistance = 10f;
        public float shootRate = .5f;
        public PlayerShooting shootingScript;

        private Animator anim;

        private NavMeshAgent navMeshAgent;
        private Transform targetedEnemy;
        private Ray shootRay;
        private RaycastHit shootHit;
        private bool walking;
        private bool enemyClicked;
        private float nextFire;

        // Use this for initialization
        void Awake()
        {
            anim = GetComponent<Animator>();
            navMeshAgent = GetComponent<NavMeshAgent>();

        }

        // Update is called once per frame
        void Update()
        {
            var line = this.GetComponent<LineRenderer>();
            Ray ray = Camera.main.ScreenPointToRay(Input.mousePosition);
            RaycastHit hit;

            // Line Renderer
            if (navMeshAgent.path != null)
            {
                if (line == null)
                {
                    line = this.gameObject.AddComponent<LineRenderer>();
                    line.material = new Material(Shader.Find("Sprites/Default")) { color = Color.yellow };
                    line.SetWidth(0.1f, 0.1f);
                    line.SetColors(Color.yellow, Color.yellow);
                }

                var path = navMeshAgent.path;

                line.SetVertexCount(path.corners.Length);

                for (int i = 0; i < path.corners.Length; i++)
                {
```

```
                line.SetPosition(i, path.corners[i]);
            }
        }

        if (Input.GetButtonDown("Fire2"))
        {
            if (Physics.Raycast(ray, out hit, 100))
            {
                if (hit.collider.tag.Contains("Enemy"))
                {
                    targetedEnemy = hit.transform;
                    enemyClicked = true;
                    line.material = new Material(Shader.Find("Sprites/Default")) { color = Color.red };
                } else
                {
                    walking = true;
                    enemyClicked = false;
                    navMeshAgent.destination = hit.point;
                    navMeshAgent.Resume();
                    line.material = new Material(Shader.Find("Sprites/Default")) { color = Color.yellow };
                }
            }
        }

        if (enemyClicked)
        {
            MoveAndShoot();
        }

        if (navMeshAgent.remainingDistance <= navMeshAgent.stoppingDistance)
        {
            if (!navMeshAgent.hasPath || Mathf.Abs(navMeshAgent.velocity.sqrMagnitude) <
float.Epsilon)
            {
                walking = false;
            }

        } else
        {
            walking = true;
        }

        anim.SetBool("IsWalking", walking);
    }

    private void MoveAndShoot()
    {
        if (targetedEnemy == null)
        {
            return;
        }

        navMeshAgent.destination = targetedEnemy.position;
        if (navMeshAgent.remainingDistance >= shootDistance)
        {
            navMeshAgent.Resume();
            walking = true;
        }
```

```
            if (navMeshAgent.remainingDistance <= shootDistance)
            {
                transform.LookAt(targetedEnemy);
                Vector3 dirToShoot = targetedEnemy.transform.position - transform.position;
                if (Time.time > nextFire)
                {
                    nextFire = Time.time + shootRate;
                    shootingScript.Shoot(dirToShoot);
                }

                navMeshAgent.Stop();
                walking = false;
            }
        }
    }
}
```

HighScoreManager

```
using UnityEngine;
using UnityEngine.UI;

using System.Collections;

public class HighScoreManager : MonoBehaviour {
    public static int highScore;

    Text text;

            // Use this for initialization
            void Awake () {
        text = GetComponent<Text>();
        highScore = PlayerPrefs.GetInt("highScore");
            }

            // Update is called once per frame
            void Update () {
        if (ScoreManager.score > highScore) highScore = ScoreManager.score;

        text.text = "High Score: " + highScore;
            }
}
```

StartManager

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

```
public class StartManager : MonoBehaviour {
    private StartManager startManager;

    PauseManager pauseManager;
    Canvas canvas;

    private void Awake()
    {
        startManager = GetComponent<StartManager>();
        pauseManager = GetComponent<PauseManager>();
        canvas = GetComponent<Canvas>();
    }

    // Use this for initialization
    void Start () {
        canvas.enabled = true;
            }

            // Update is called once per frame
            void Update () {
        pauseManager.Pause();
    }
}
```

PlayerShooting

```
using UnityEngine;

public class PlayerShooting : MonoBehaviour
{
    public int damagePerShot = 20;
    public float timeBetweenBullets = 0.15f;
    public float range = 100f;


    float timer;
    Ray shootRay;
    RaycastHit shootHit;
    int shootableMask;
    ParticleSystem gunParticles;
    LineRenderer gunLine;
    AudioSource gunAudio;
    Light gunLight;
    float effectsDisplayTime = 0.2f;


    void Awake ()
    {
        shootableMask = LayerMask.GetMask ("Shootable");
        gunParticles = GetComponent<ParticleSystem> ();
        gunLine = GetComponent <LineRenderer> ();
        gunAudio = GetComponent<AudioSource> ();
        gunLight = GetComponent<Light> ();
    }


    void Update ()
    {
        timer += Time.deltaTime;

                        if(Input.GetButton ("Fire1") && timer >= timeBetweenBullets && Time.timeScale !=
0)
        {
            Shoot ();
        }

        if(timer >= timeBetweenBullets * effectsDisplayTime)
        {
            DisableEffects ();
        }
    }


    public void DisableEffects ()
    {
        gunLine.enabled = false;
        gunLight.enabled = false;
    }

    void Shoot ()
    {
        timer = 0f;

        gunAudio.Play ();

        gunLight.enabled = true;

        gunParticles.Stop ();
        gunParticles.Play ();

        gunLine.enabled = true;
        gunLine.SetPosition (0, transform.position);

        shootRay.origin = transform.position;
        shootRay.direction = transform.forward;

        if(Physics.Raycast (shootRay, out shootHit, range, shootableMask))
        {
            EnemyHealth enemyHealth = shootHit.collider.GetComponent <EnemyHealth> ();
            if(enemyHealth != null)
            {
                enemyHealth.TakeDamage (damagePerShot, shootHit.point);
            }
            gunLine.SetPosition (1, shootHit.point);
        }
        else
        {
            gunLine.SetPosition (1, shootRay.origin + shootRay.direction * range);
        }
    }
}
```

PlayerHealth

```
using UnityEngine;
using UnityEngine.UI;
using System.Collections;
using UnityEngine.SceneManagement;


public class PlayerHealth : MonoBehaviour
{
    public int startingHealth = 100;
    public int currentHealth;
    public Slider healthSlider;
    public Image damageImage;
    public AudioClip deathClip;
    public float flashSpeed = 5f;
    public Color flashColour = new Color(1f, 0f, 0f, 0.1f);


    Animator anim;
    AudioSource playerAudio;
    PlayerMovement playerMovement;
    //PlayerShooting playerShooting;
    bool isDead;
    bool damaged;


    void Awake ()
    {
        anim = GetComponent <Animator> ();
        playerAudio = GetComponent <AudioSource> ();
        playerMovement = GetComponent <PlayerMovement> ();
        //playerShooting = GetComponentInChildren <PlayerShooting> ();
        currentHealth = startingHealth;
    }


    void Update ()
    {
        if(damaged)
        {
            damageImage.color = flashColour;
        }
        else
        {
            damageImage.color = Color.Lerp (damageImage.color, Color.clear, flashSpeed *
Time.deltaTime);
        }
        damaged = false;
    }


    public void TakeDamage (int amount)
    {
        damaged = true;

        currentHealth -= amount;

        healthSlider.value = currentHealth;

        playerAudio.Play ();
        if(currentHealth <= 0 && !isDead)
        {
            Death ();
        }
    }


    void Death ()
    {
        isDead = true;

        //playerShooting.DisableEffects ();

        anim.SetTrigger ("Die");

        playerAudio.clip = deathClip;
        playerAudio.Play ();

        playerMovement.enabled = false;
        //playerShooting.enabled = false;
    }


    public void RestartLevel ()
    {
        SceneManager.LoadScene (0);
    }
}
```

EnemyHealth

```
using UnityEngine;

public class EnemyHealth : MonoBehaviour
{
    public int startingHealth = 100;
    public int currentHealth;
    public float sinkSpeed = 2.5f;
    public int scoreValue = 10;
    public AudioClip deathClip;


    Animator anim;
    AudioSource enemyAudio;
    ParticleSystem hitParticles;
    CapsuleCollider capsuleCollider;
    bool isDead;
    bool isSinking;


    void Awake ()
    {
        anim = GetComponent <Animator> ();
        enemyAudio = GetComponent <AudioSource> ();
        hitParticles = GetComponentInChildren <ParticleSystem> ();
        capsuleCollider = GetComponent <CapsuleCollider> ();
        currentHealth = startingHealth;
    }


    void Update ()
    {
        if(isSinking)
        {
            transform.Translate (-Vector3.up * sinkSpeed * Time.deltaTime);
        }
    }


    public void TakeDamage (int amount, Vector3 hitPoint)
    {
        if(isDead)
            return;

        enemyAudio.Play ();

        currentHealth -= amount;

        hitParticles.transform.position = hitPoint;
        hitParticles.Play();

        if(currentHealth <= 0)
        {
            Death ();
        }
    }


    void Death ()
    {
        isDead = true;

        capsuleCollider.isTrigger = true;

        anim.SetTrigger ("Dead");

        enemyAudio.clip = deathClip;
        enemyAudio.Play ();
    }


    public void StartSinking ()
    {
        GetComponent <NavMeshAgent> ().enabled = false;
        GetComponent <Rigidbody> ().isKinematic = true;
        isSinking = true;
        //ScoreManager.score += scoreValue;
        Destroy (gameObject, 2f);
    }
}
```

EnemyAttack

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

```
public class EnemyAttack : MonoBehaviour
{
    public float timeBetweenAttacks = 0.5f;
    public int attackDamage = 10;


    Animator anim;
    GameObject player;
    PlayerHealth playerHealth;
    //EnemyHealth enemyHealth;
    bool playerInRange;
    float timer;


    void Awake ()
    {
        player = GameObject.FindGameObjectWithTag ("Player");
        playerHealth = player.GetComponent <PlayerHealth> ();
        //enemyHealth = GetComponent<EnemyHealth>();
        anim = GetComponent <Animator> ();
    }


    void OnTriggerEnter (Collider other)
    {
        if(other.gameObject == player)
        {
            playerInRange = true;
        }
    }


    void OnTriggerExit (Collider other)
    {
        if(other.gameObject == player)
        {
            playerInRange = false;
        }
    }


    void Update ()
    {
        timer += Time.deltaTime;

        if(timer >= timeBetweenAttacks && playerInRange/* && enemyHealth.currentHealth > 0*/)
        {
            Attack ();
        }

        if(playerHealth.currentHealth <= 0)
        {
            anim.SetTrigger ("PlayerDead");
        }
    }
    void Attack ()
    {
        timer = 0f;

        if(playerHealth.currentHealth > 0)
        {
            playerHealth.TakeDamage (attackDamage);
        }
    }
}
```

EnemyManager

```
using UnityEngine;

public class EnemyManager : MonoBehaviour
{
    public PlayerHealth playerHealth;
    public GameObject enemy;
    public float spawnTime = 3f;
    public Transform[] spawnPoints;


    void Start ()
    {
        InvokeRepeating ("Spawn", spawnTime, spawnTime);
    }


    void Spawn ()
    {
        if(playerHealth.currentHealth <= 0f)
        {
            return;
        }

        int spawnPointIndex = Random.Range (0, spawnPoints.Length);

        Instantiate (enemy, spawnPoints[spawnPointIndex].position, 
spawnPoints[spawnPointIndex].rotation);
    }
}
```

GameOverManager

```
using UnityEngine;
```

```
public class GameOverManager : MonoBehaviour
{
    public PlayerHealth playerHealth;


    Animator anim;


    void Awake()
    {
        anim = GetComponent<Animator>();
    }


    void Update()
    {
        if (playerHealth.currentHealth <= 0)
        {
            anim.SetTrigger("GameOver");
        }
    }
}
```

ScoreManager

```
using UnityEngine;
using UnityEngine.UI;
using System.Collections;

public class ScoreManager : MonoBehaviour
{
    public static int score;


    Text text;


    void Awake ()
    {
        text = GetComponent <Text> ();
        score = 0;
    }


    void Update ()
    {
        text.text = "Score: " + score;
    }
}
```

MixLevels

```
using UnityEngine;
using System.Collections;
using UnityEngine.Audio;

public class MixLevels : MonoBehaviour {

        public AudioMixer masterMixer;

        public void SetSfxLvl(float sfxLvl)
        {
                masterMixer.SetFloat("sfxVol", sfxLvl);
        }

        public void SetMusicLvl (float musicLvl)
        {
                masterMixer.SetFloat ("musicVol", musicLvl);
        }
}
```

CameraFollow

```
using UnityEngine;
using System.Collections;

namespace CompleteProject
{
    public class CameraFollow : MonoBehaviour
    {
        public Transform target;            // The position that that camera will be following.
        public float smoothing = 5f;        // The speed with which the camera will be following.


        Vector3 offset;                     // The initial offset from the target.


        void Start ()
        {
            // Calculate the initial offset.
            offset = transform.position - target.position;
        }

        void FixedUpdate ()
        {
            // Create a postion the camera is aiming for based on the offset from the target.
            Vector3 targetCamPos = target.position + offset;

            // Smoothly interpolate between the camera's current position and it's target position.
            transform.position = Vector3.Lerp (transform.position, targetCamPos, smoothing *
Time.deltaTime);
        }
    }
}
```