INTRODUCTION TO GAME PROGRAMMING

Project Report

Submitted By

Karmesh Duggar

R142219014

500075888

Under the guidance of

Mr. PANKAJ BADONI

Department of Virtualisation, School of Computer Science and Engineering,

UNIVERSITY OF PETROLEUM AND ENERGY STUDIES,

INDEX

Introduction

First-person shooter (**FPS**) is a video game genre centered on gun and other weapon-based combat in a first-person perspective; that is, the player experiences the action through the eyes of the protagonist. The genre shares common traits with other shooter games, which in turn makes it fall under the heading action game.

First-person shooters are a type of shooter game featuring a first-person point of view with which the player sees the action through the eyes of the player character.

Game Overview

A Fps 3D Game made for mobile phones & tablets. It has been developed in Unity 3D using C# and as for the sprites, all of them are downloaded via internet and make in adobe photoshop. The prime motive of the game is to kill the enemies (Enemy soldier) and reach the final destination.. Player has to keep in mind that it did not get vision of the enemy and also too close to enemy as it take damage to player.

Game Mechanics

Game

Player has to complete four objectives and reach the final destination in order to complete the mission and it has a limited health which is shown in health bar and it get depleted if player got attack by enemy soldier and player has to shoot the enemy soldier using its two guns one is assault rifle and second one is sniper rifle.

Where as sniper can zoom in at a large range where as assault rifle can zoom in through short range.

And player has four objectives in which player has to collect some materials in the game using action keys and finally reach the final destination where game ends.

Player has given a ammo levels to clear the stage and some ammo box to refil the ammo is also there.

Player can use minimap to check where enemy is and take advantage of it in order to complete the stage.

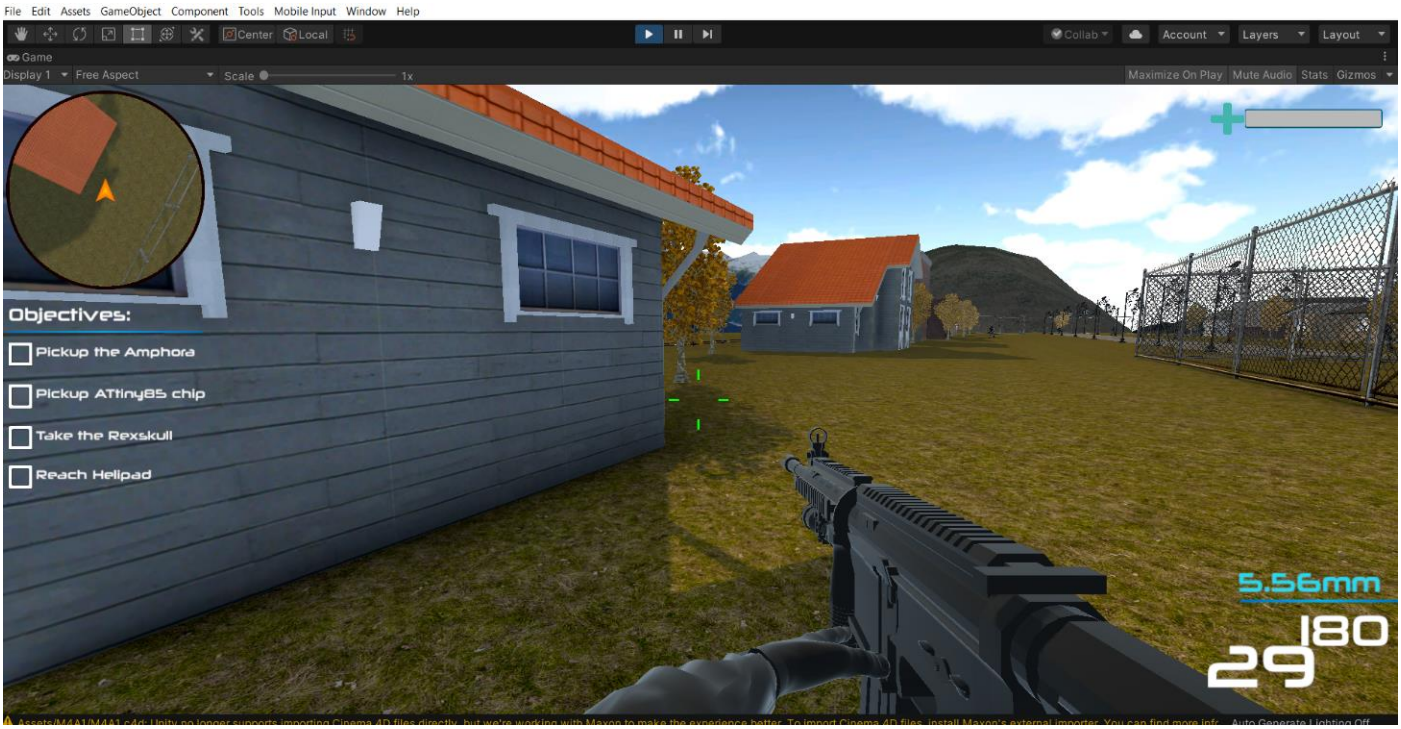
Assault rifle has lesser damage and sniper rifle is one shot one kill.

Controls

Player use mouse left mouse button to shoot and right mouse button to zoom in and out and to pickup the objects that it has to in order to complete the objectives player use key E and key Z.

Player can move forward,backward,right,left using W,S,D,A respectively and also using arrow keys. Player can also jump with space keys. And for sprint in order to run fast player can use combination of W and Shift key and also switch its weapons with mouse scrolling and Key X.

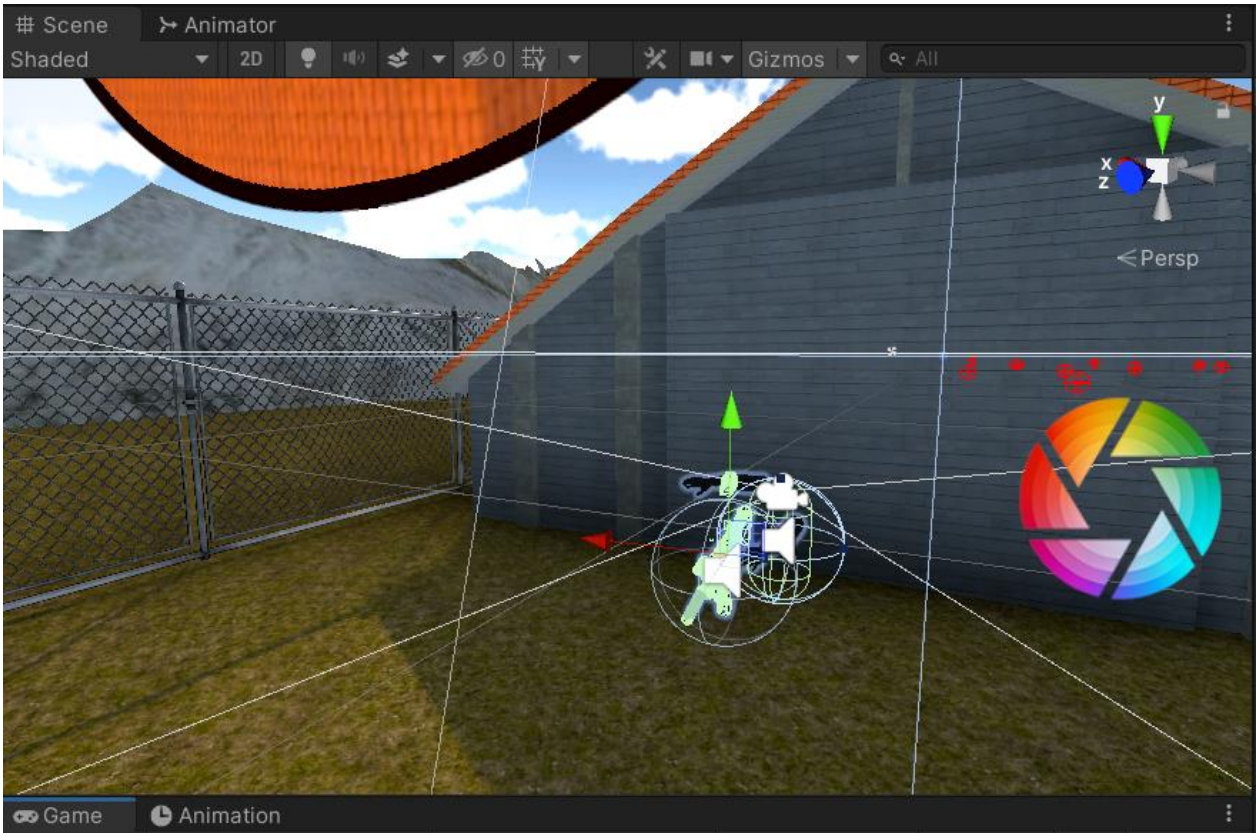
and All basic fps controls are same

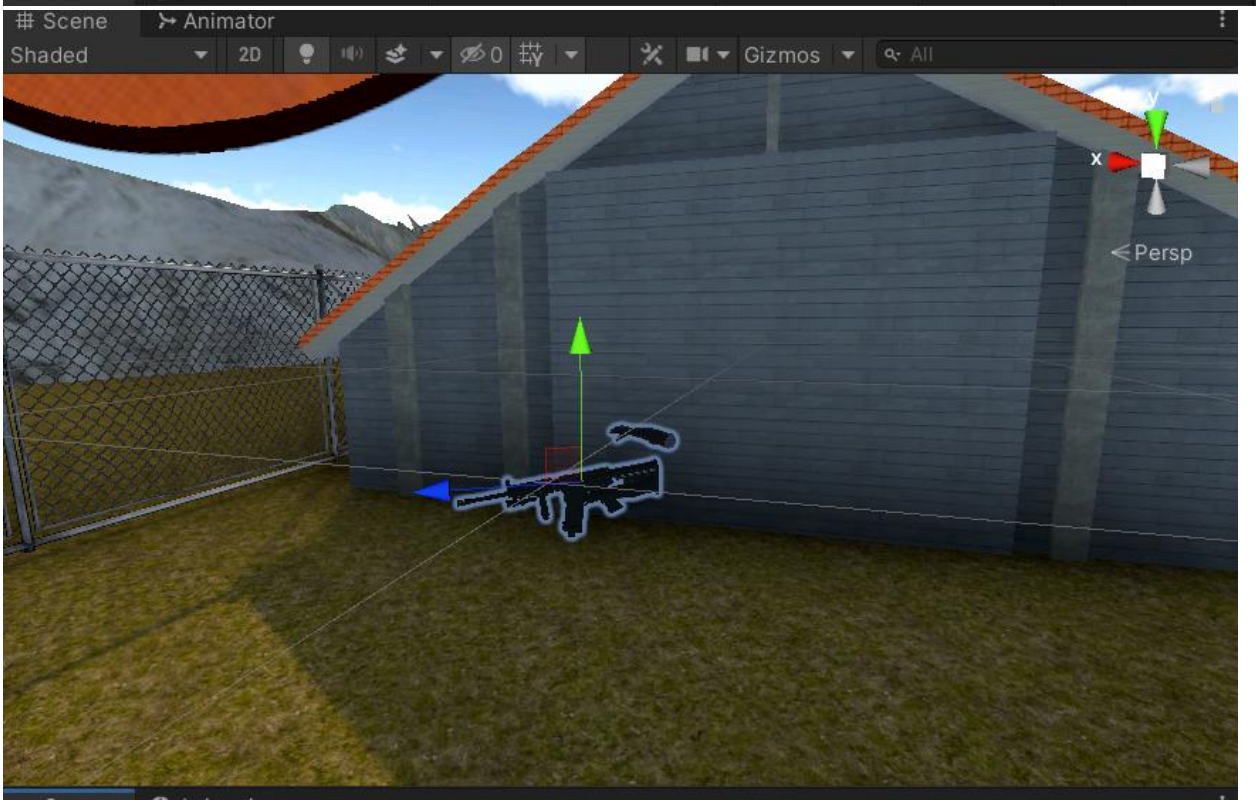


PLAYER MECHANICS :

Player can move around through arrow keys

And shoot through left mouse button and zoom in out its gun with right mouse button and use Key E and Key Z to interact with object in order to finish the game and complete the objectives and it can also sprint via shift and jump via space and also switch its weapons with mouse scrolling and Key X.





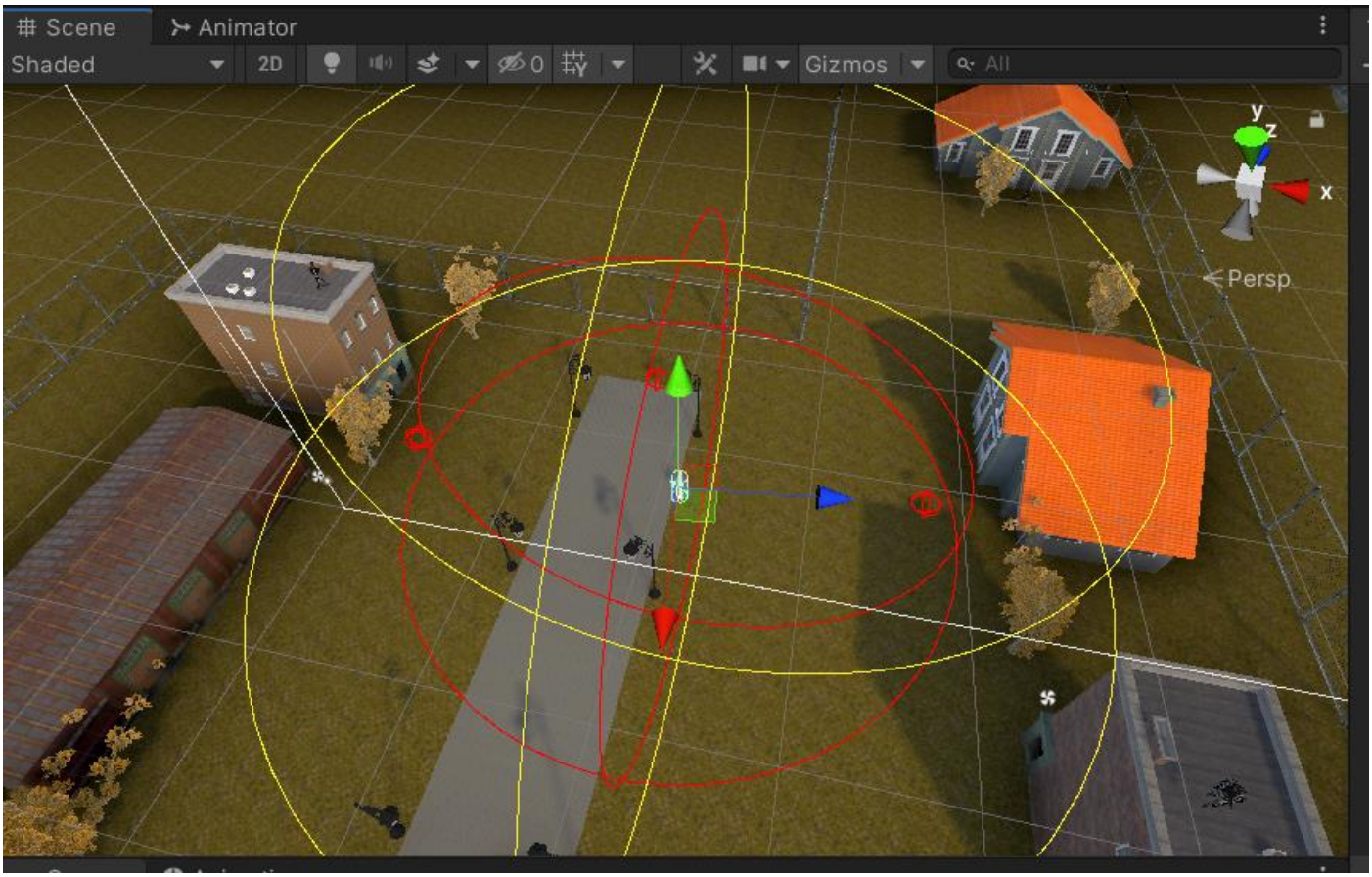
ENEMY MECHANICS :

For the enemy Movements,

Two types of enemy player one which is without patrolling and anthor one with patrol points and they move within the patrol points and as player come into different ranges they perform the actions accordingly through there AI code.

Enemy has a fixed attack range and spot range and as player comes in spot range and as well as player is in vision of the enemy enemy start chasing the player and as it come into attack range of the enemy it start attacking the player and start dealing damage to enemy.

Here red represent attack range of player and yellow represent chase range and three red sphere are the patrolling points.



And player also having a gun through using which it can shoot the player and decrease the health player and increase difficulty in the game.

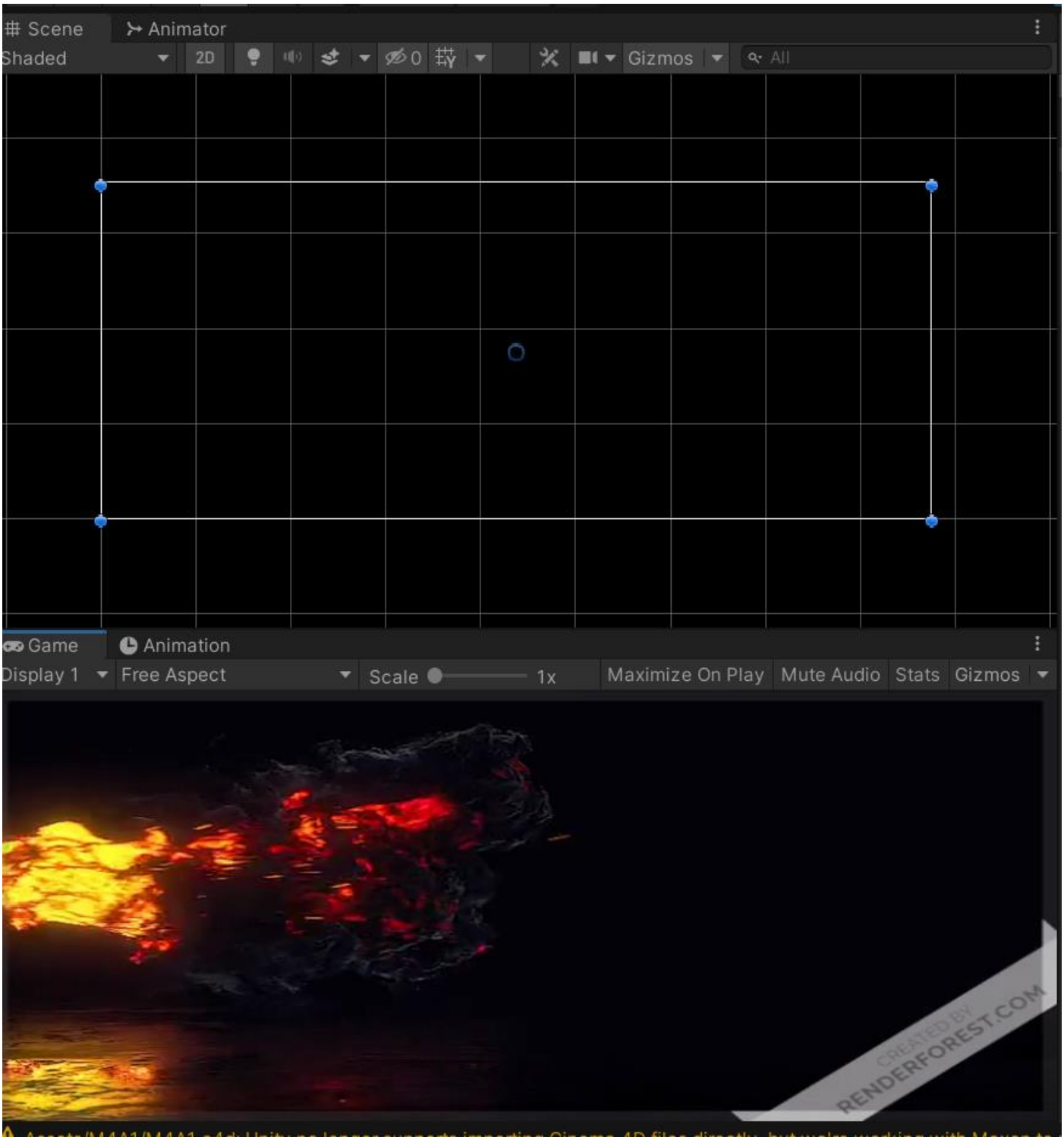
Game Design

User-Interface

In three scenes:

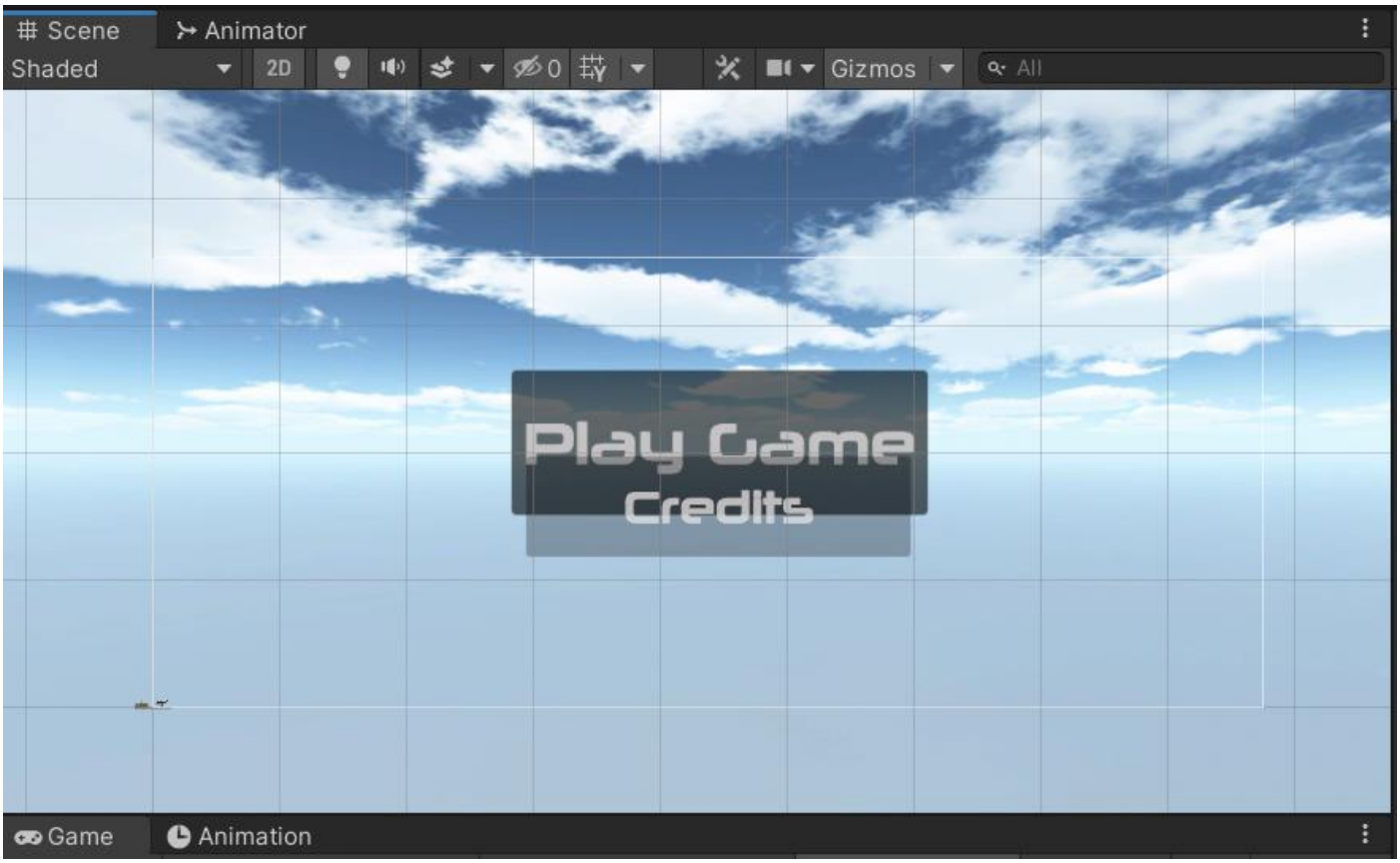
First one is Intro A video player is attached to canvas for intro of the

Game and publisher.

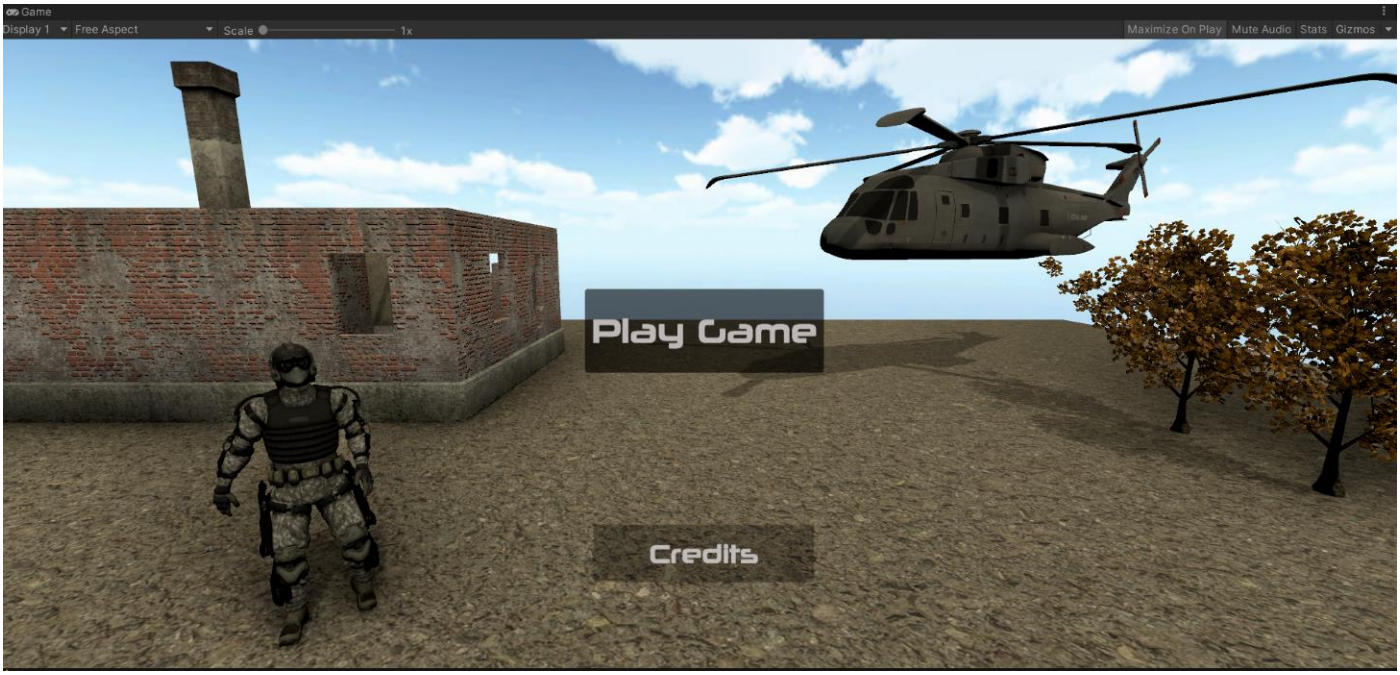


Second one is Game menu in which two buttons Play and credits

And as play button get clicked game starts that is third screen get load.

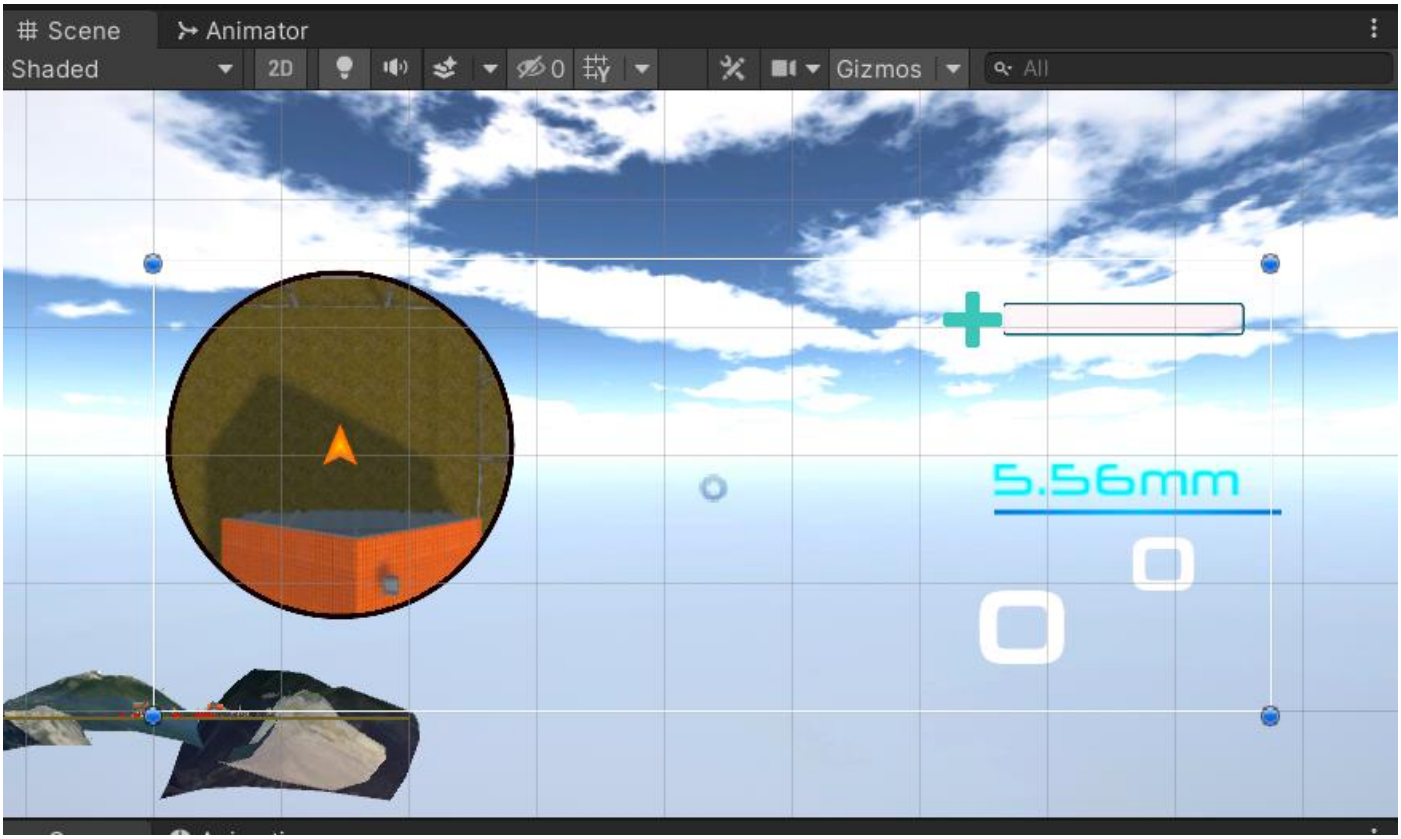


And the gameMenu looks:

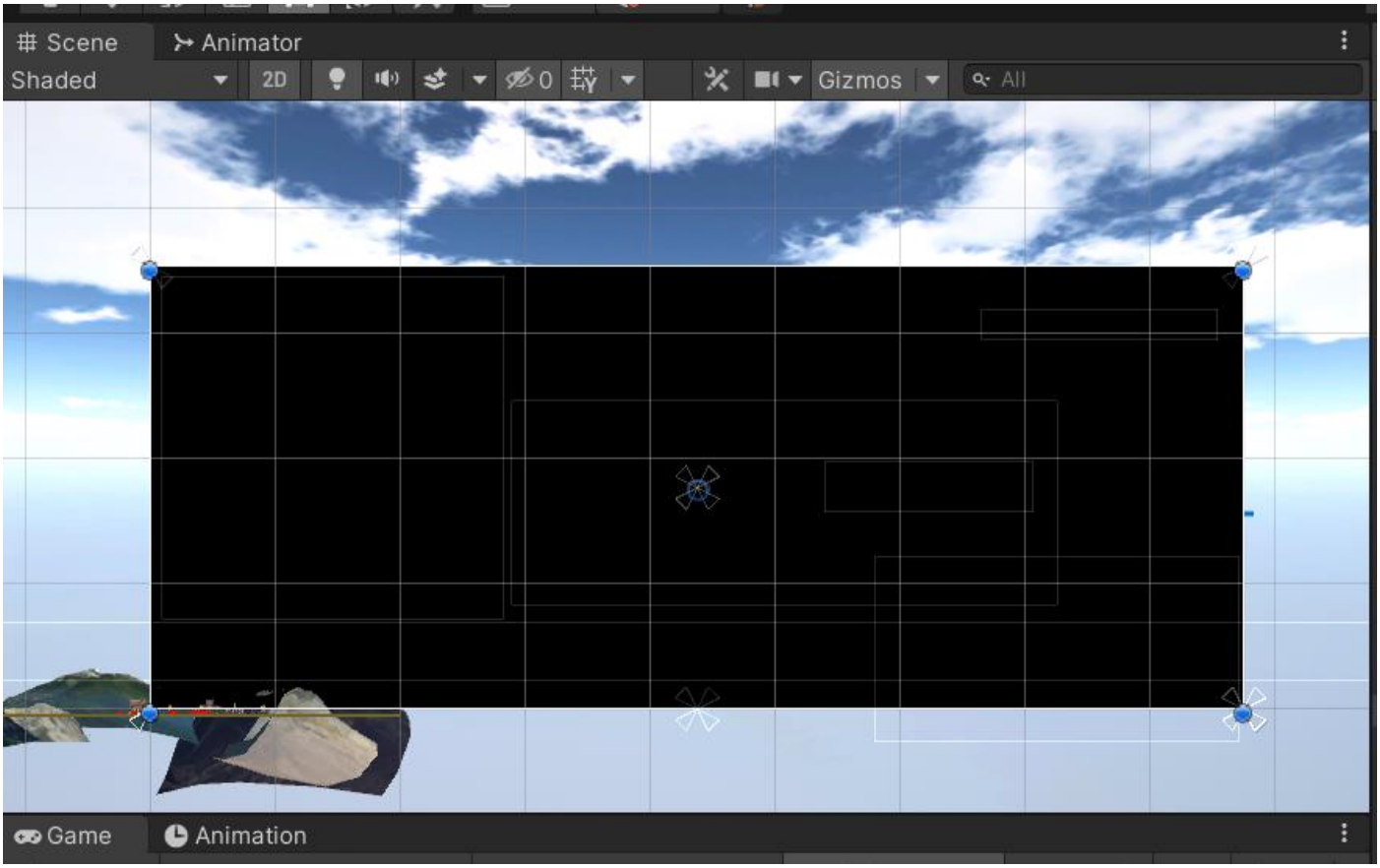


Third one:

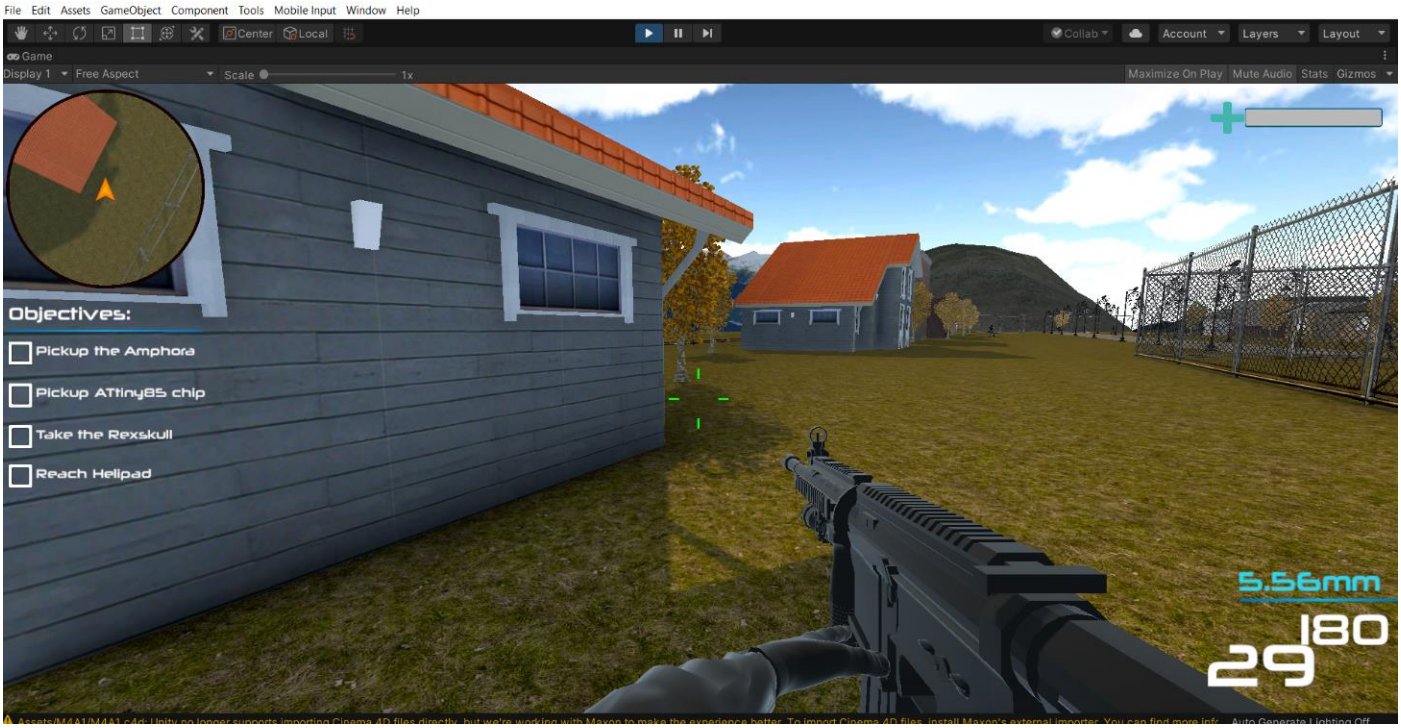
In user interface for first two seconds one fade in fade out screen black come in order to make game more realistic and player has Minimap at the top left and Health bar at the top right of the screen.And at the bottom right player has shown the ammo levels for gun player is holding and as player switch to other gun the ammo level for that gun is getting displayed and switched.



And for better gameplay this black screen animation for 2 seconds is there;



And the game looks like;

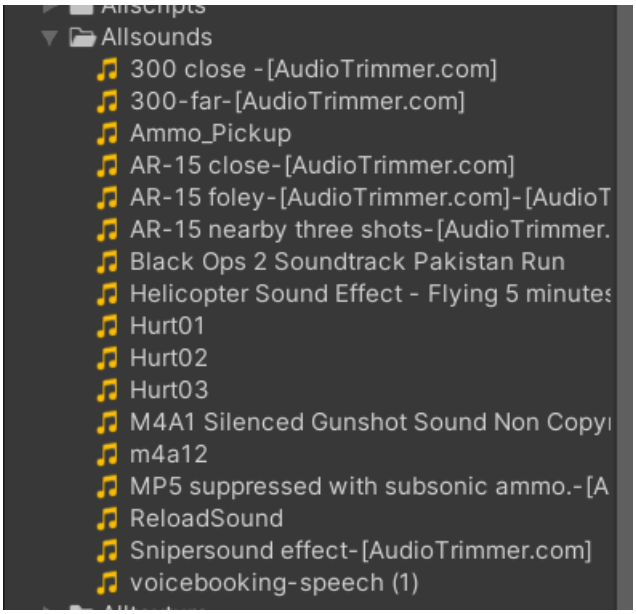


Sound:

The sound is used in first scene in intro of the game and in second scene helicopter sound used.

And in third scene:

A background sound,Two Gun sounds,Reload sound,Ammo pickup sound,Three Hurt sounds when player receive damage,Helicopter sound at the last of the stage (3D sound for limited distance),subtitle voice in starting of game.



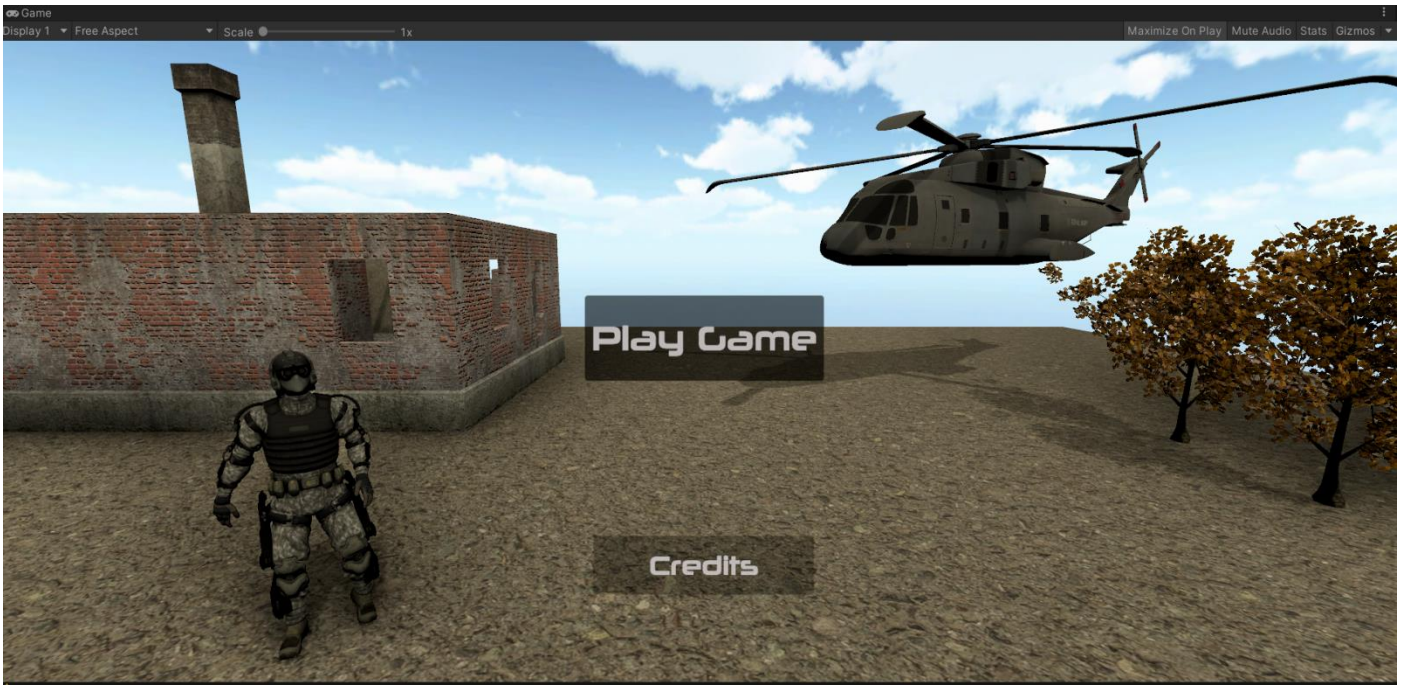
Scenes/Screens

In the game there are three scenes

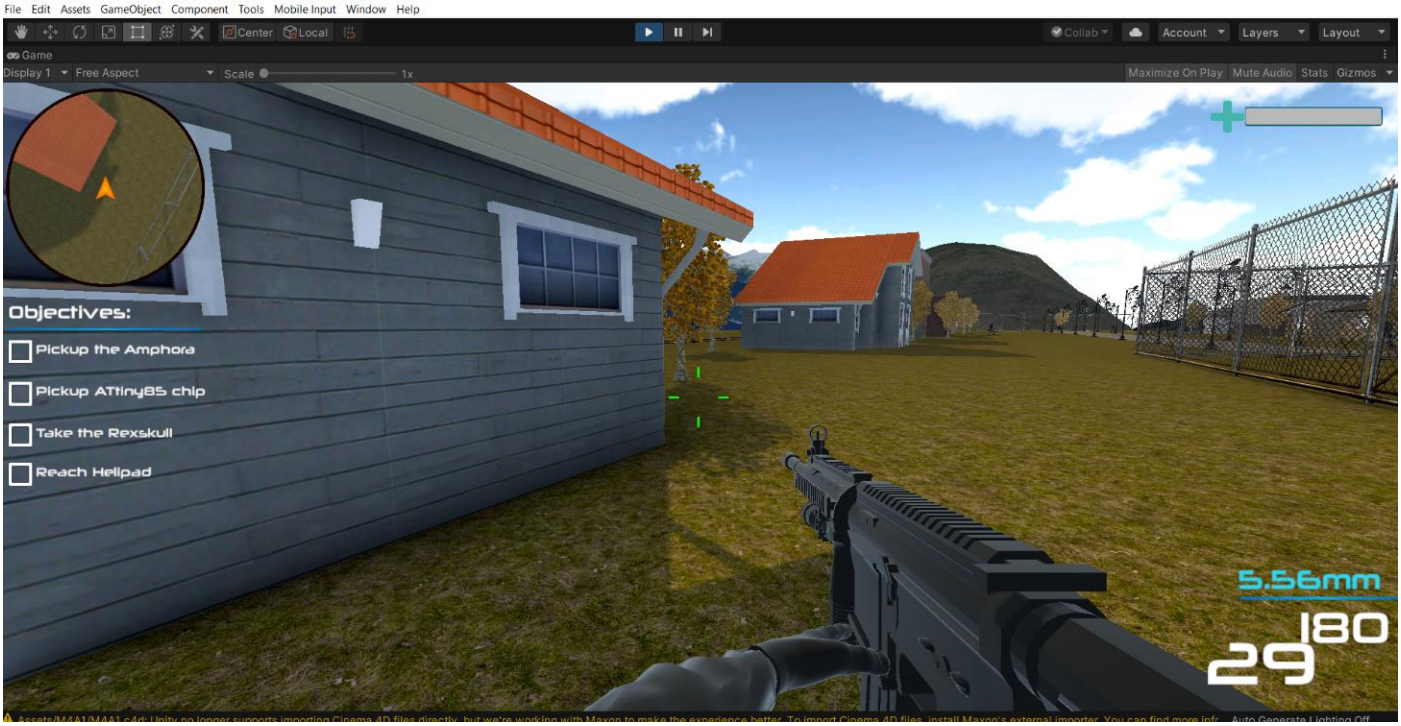
First scene is Intro A video player is attached to canvas for intro of the Game and publisher.

Second scene is Game menu in which two buttons Play and credits

And as play button get clicked game starts that is third screen get load. Which have animation for soldier standing and helicopter.



Third scene is our main game where player plays and enjoys the game.

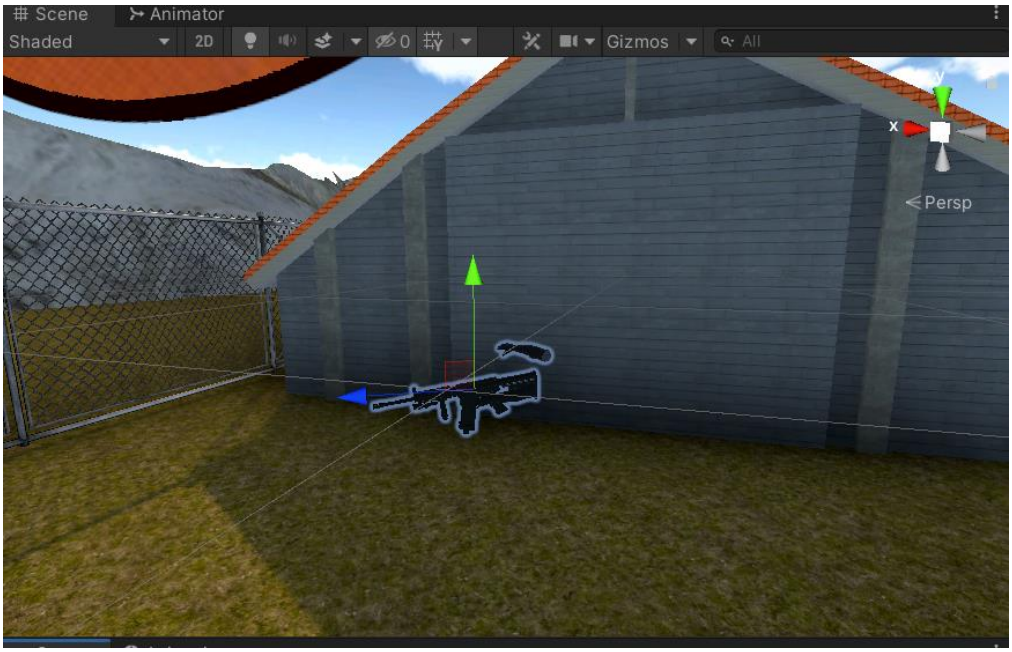


Assets and Game Objects

All the Assets and game objects are created in BLENDER and textured within the unity itself. And download via internet.

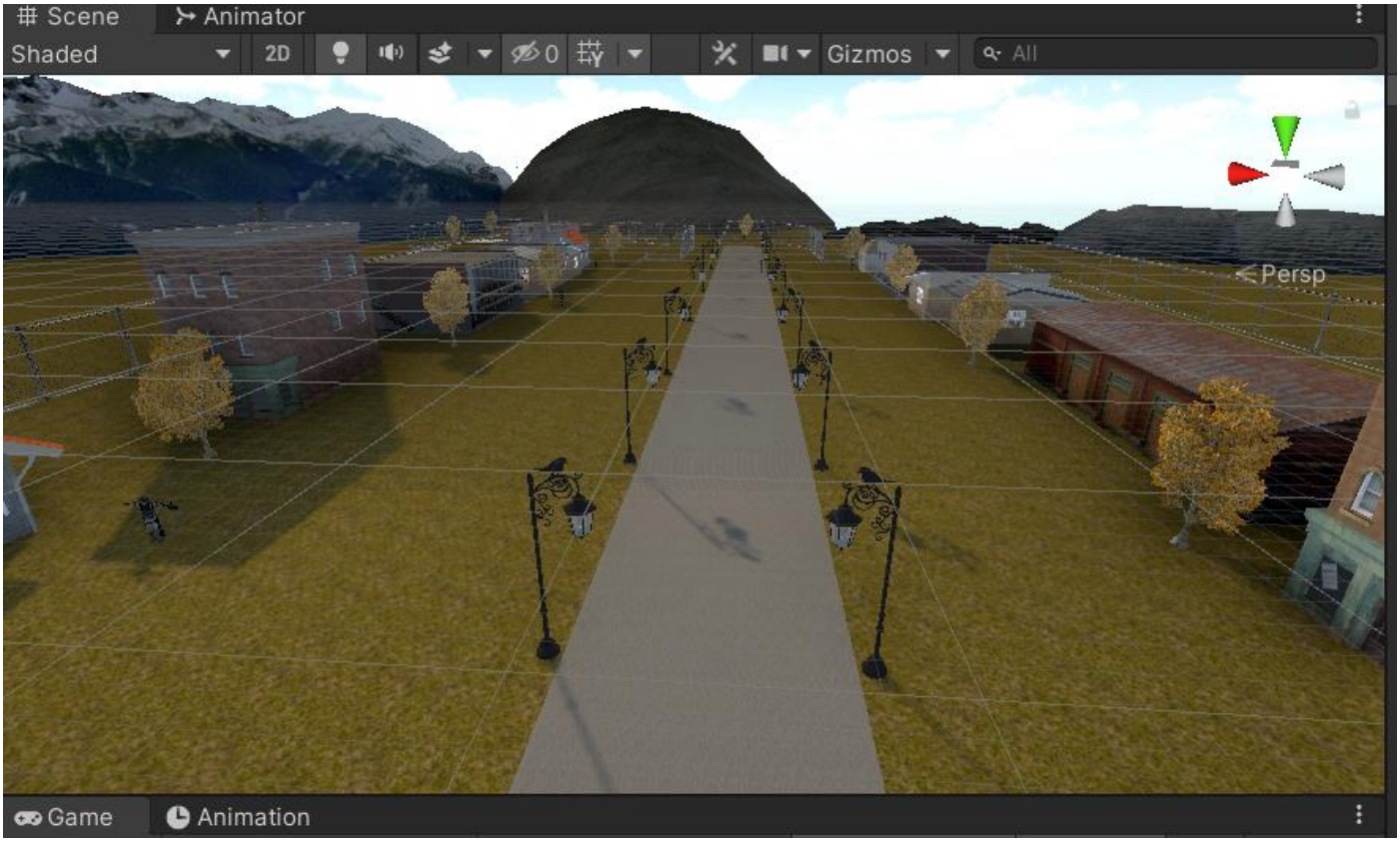
Some sprites:

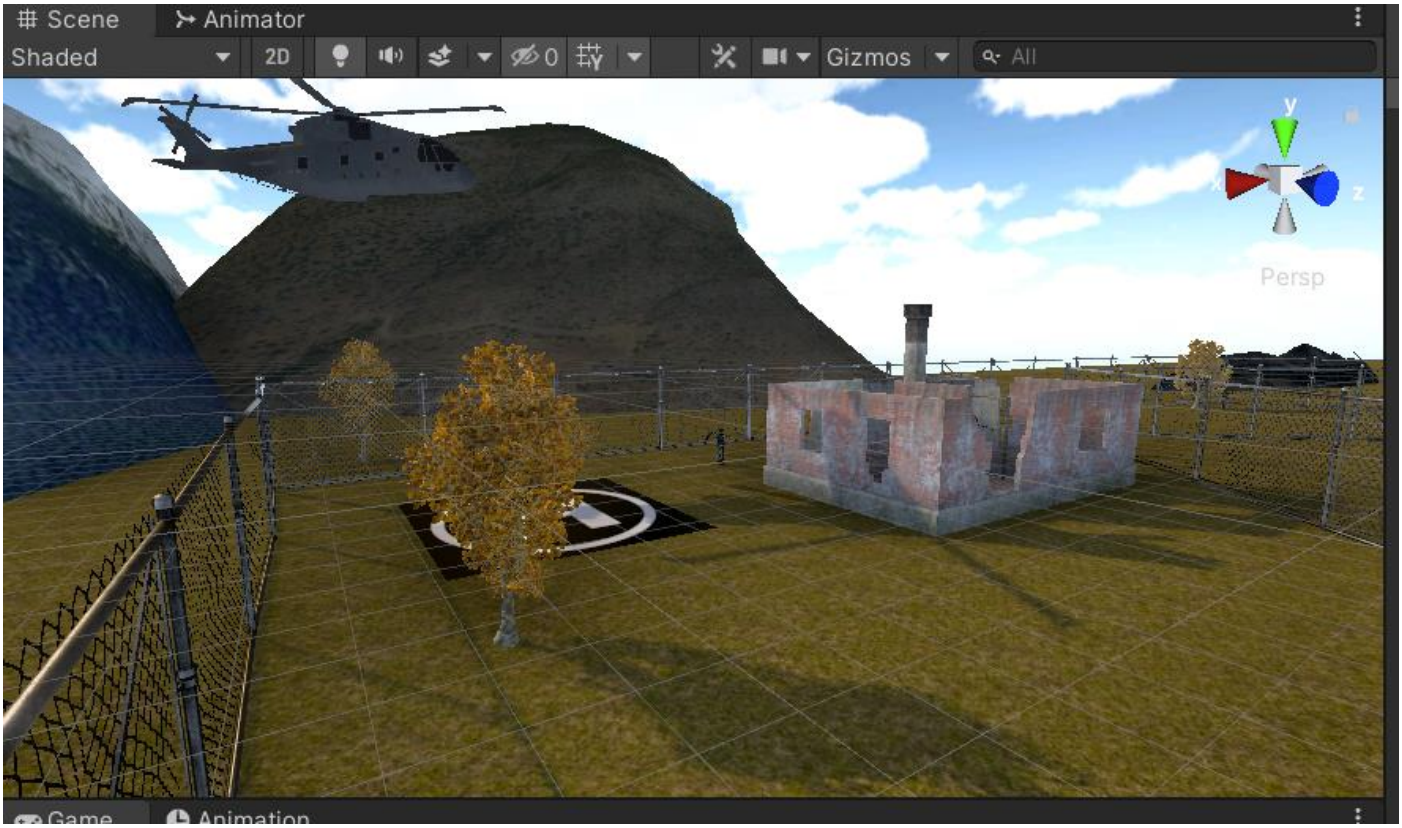
Main Character

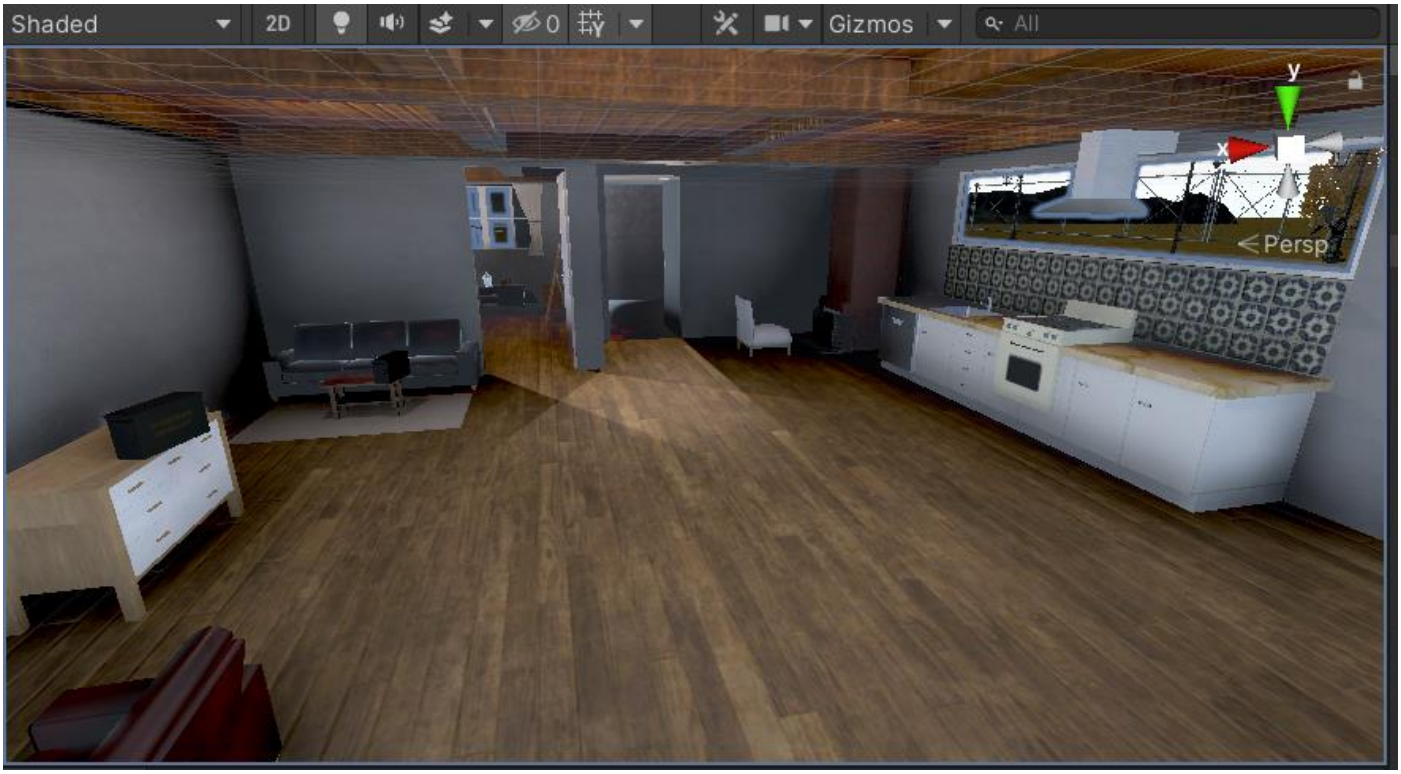


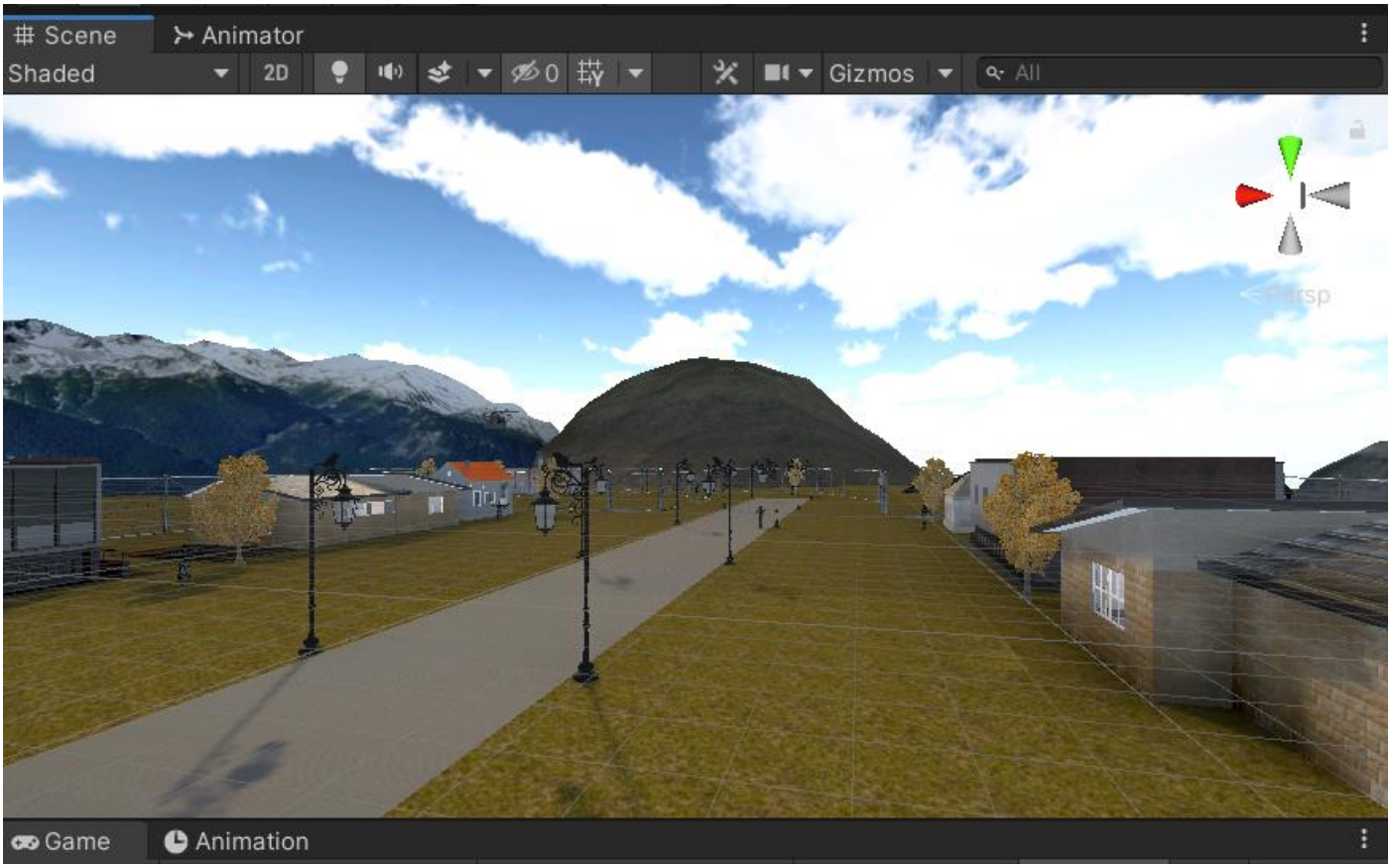
Enemy

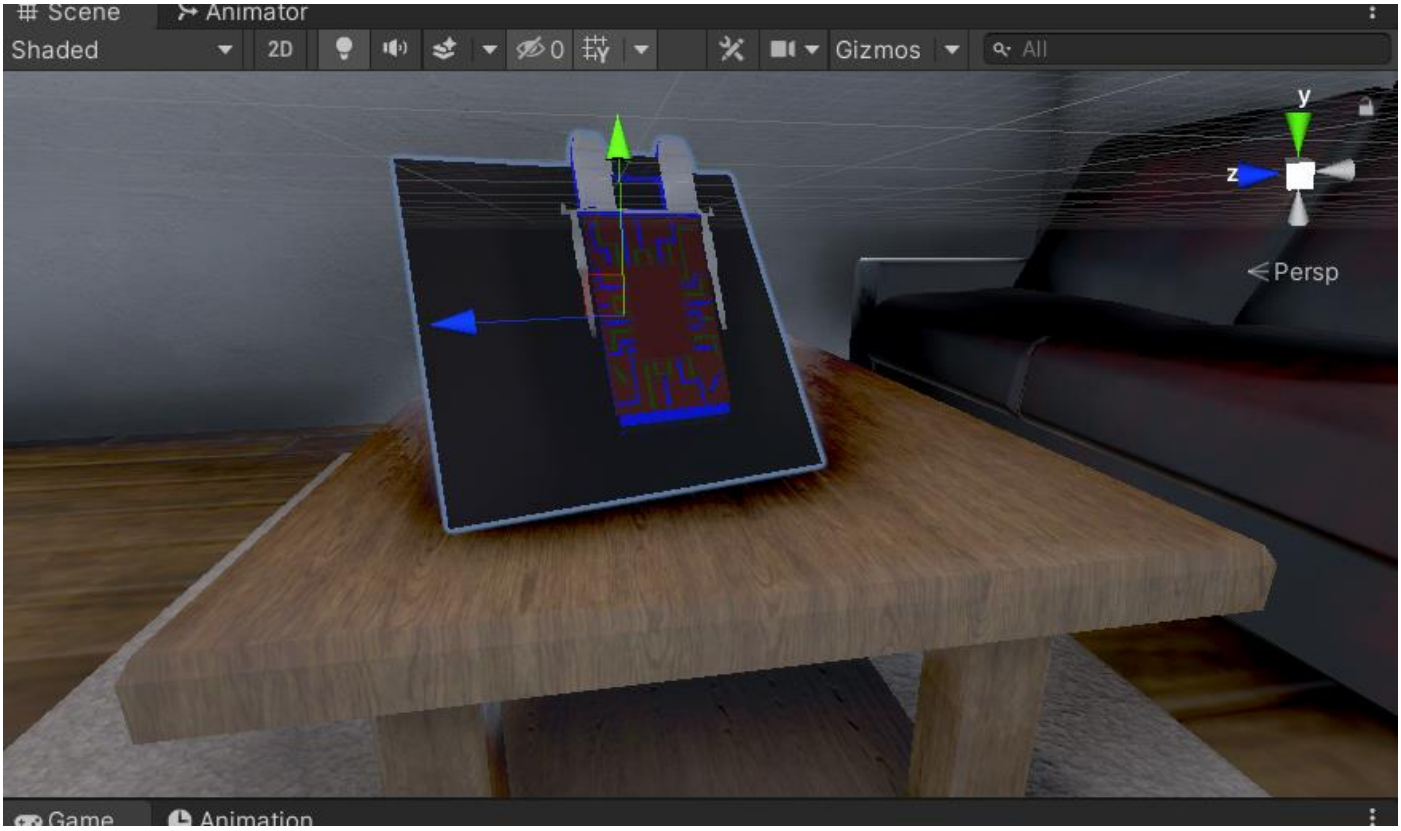
Background & Environment Items;

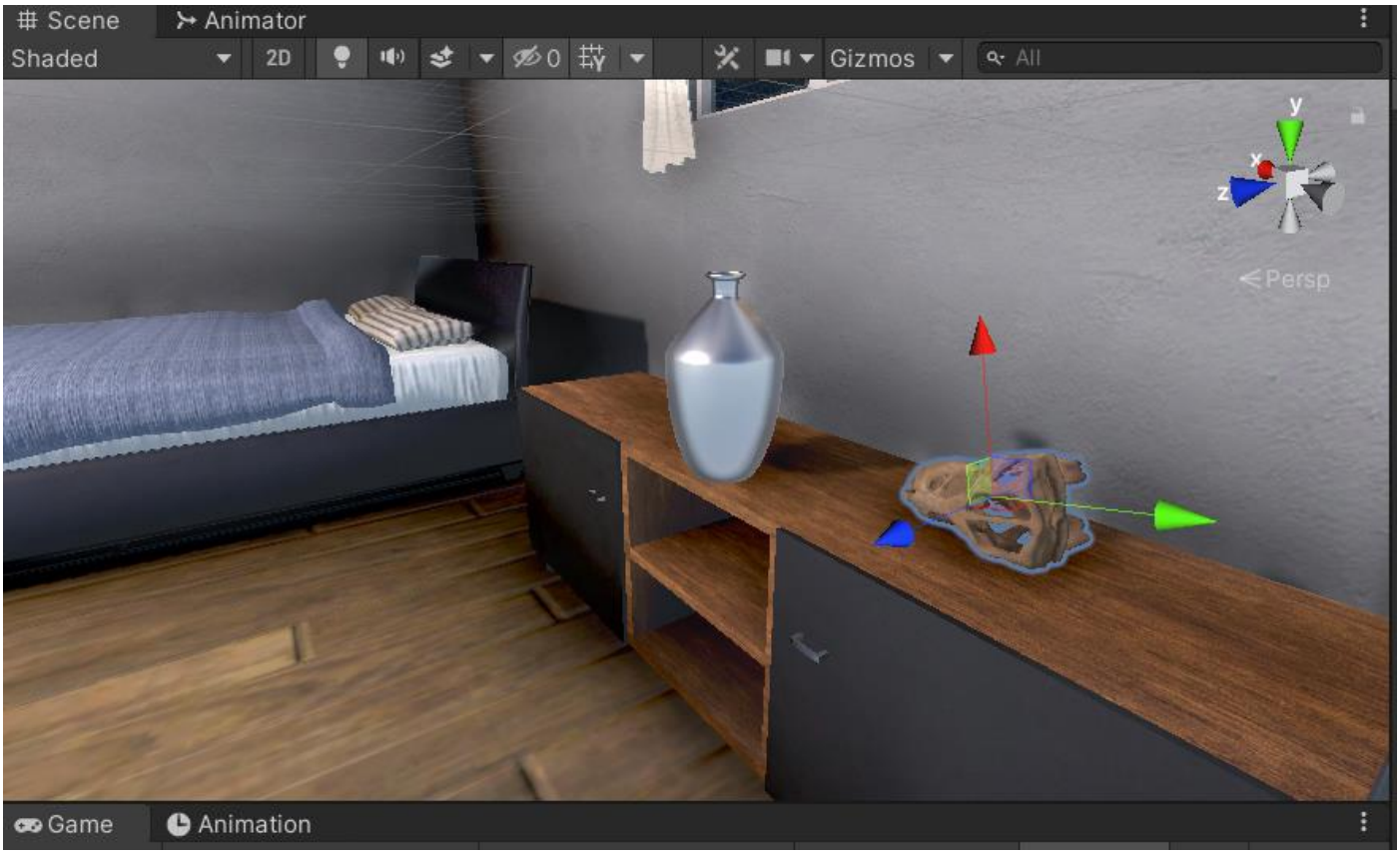

Scene
Animator
Shaded
2D
Gizmos
All
Persp
Game
Animation

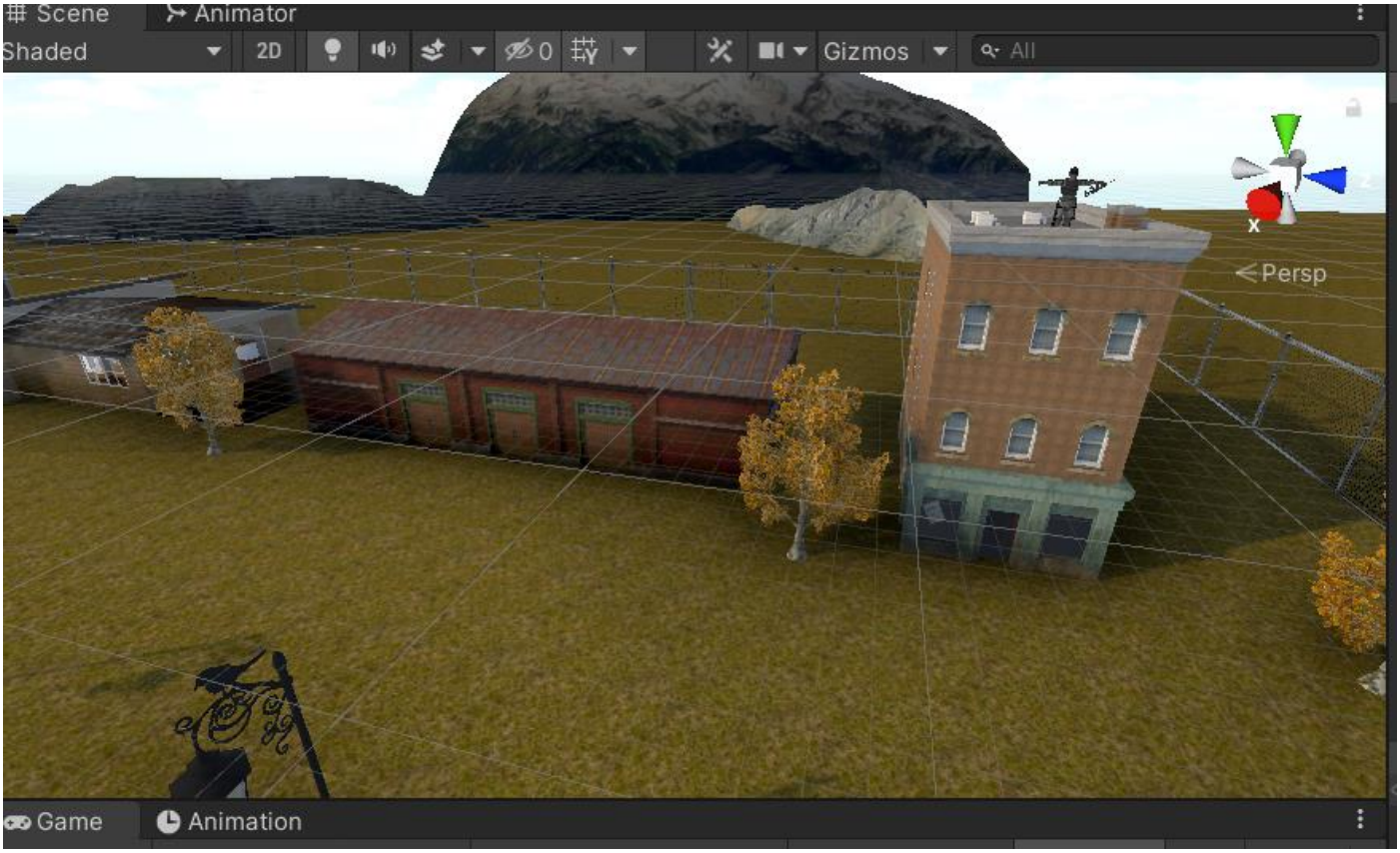

Scene
Animator
Shaded
2D
Gizmos
All
y
x
z
Persp
Game
Animation


Shaded
2D
Gizmos
All
y
x
Persp


Scene
Animator
Shaded
2D
Gizmos
All
Persp
Game
Animation


Scene
Animator
Shaded
2D
Gizmos
All
y
z
Persp
Game
Animation


Scene
Animator
Shaded
2D
Gizmos
All
y
x
z
Persp
Game
Animation

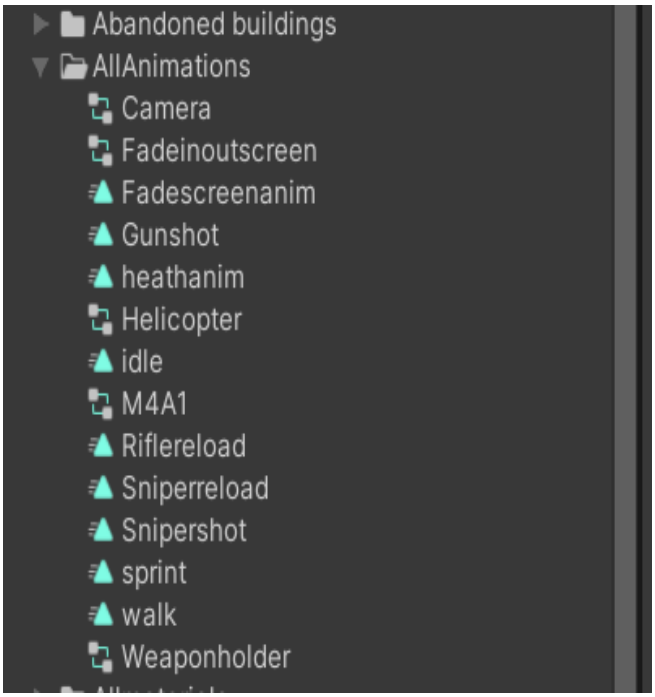

Scene
Animator
Shaded
2D
Gizmos
All
Persp
x
Game
Animation

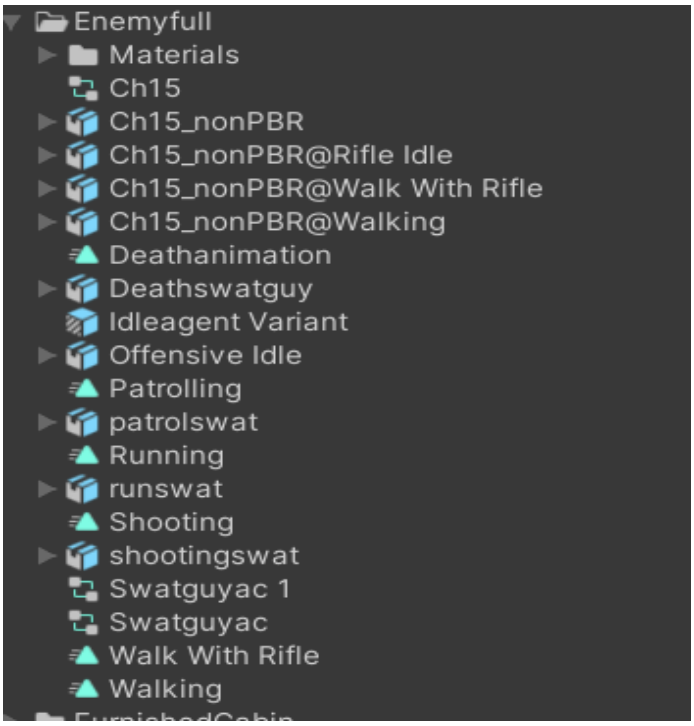
Animation

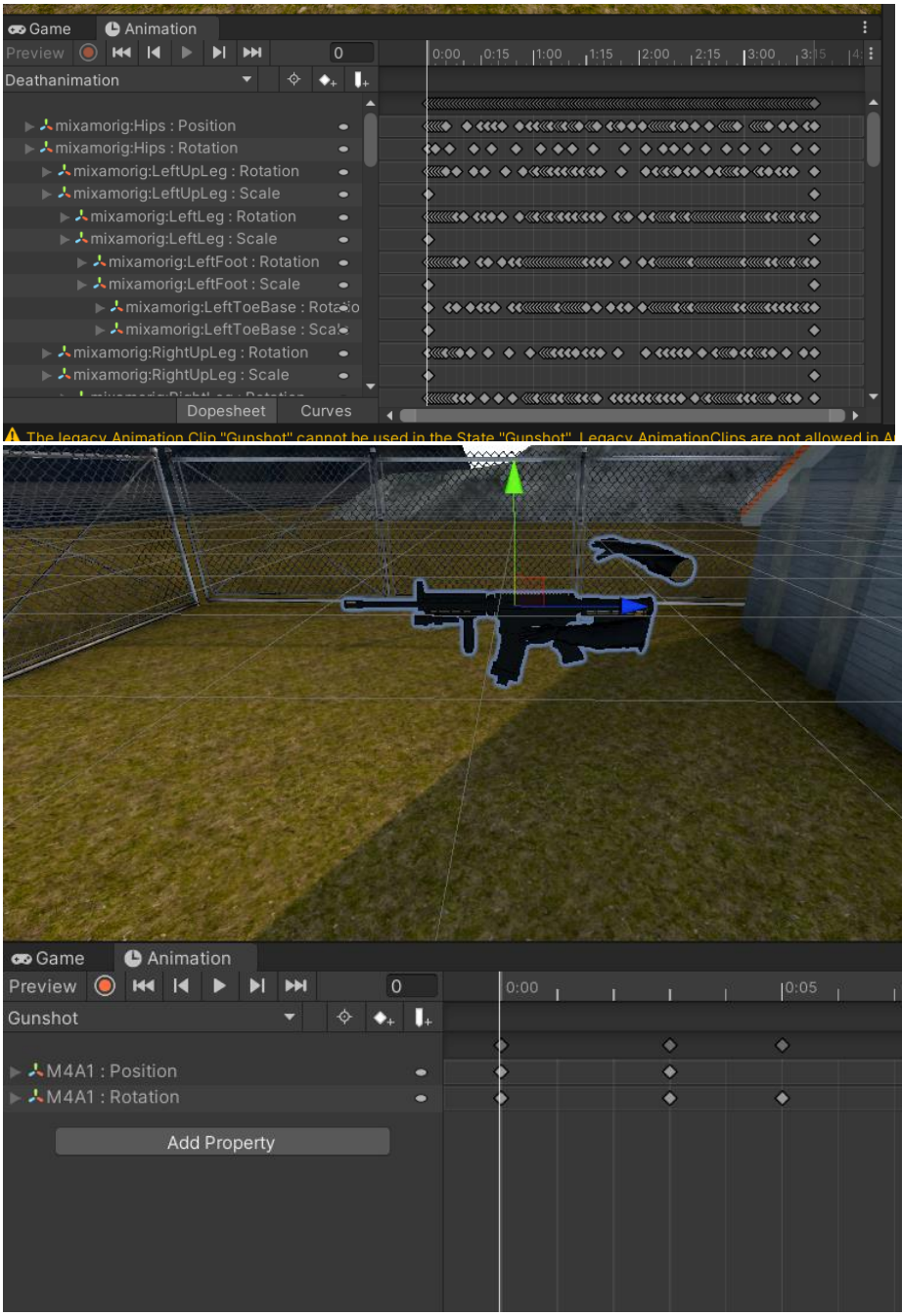
For the animation, we created most of the animation part in the unity animation editor and some animation download via internet.

Animations are used in reloading of both guns ,shooting animation of both guns ,Enemy player patrol,idle,walk,run,And die animation.and also one animation used for screen fade in fade out for 2 seconds in starting.

And for main player Idle , Walk,Sprint animation are used which shown on weapons when player perform different actions.



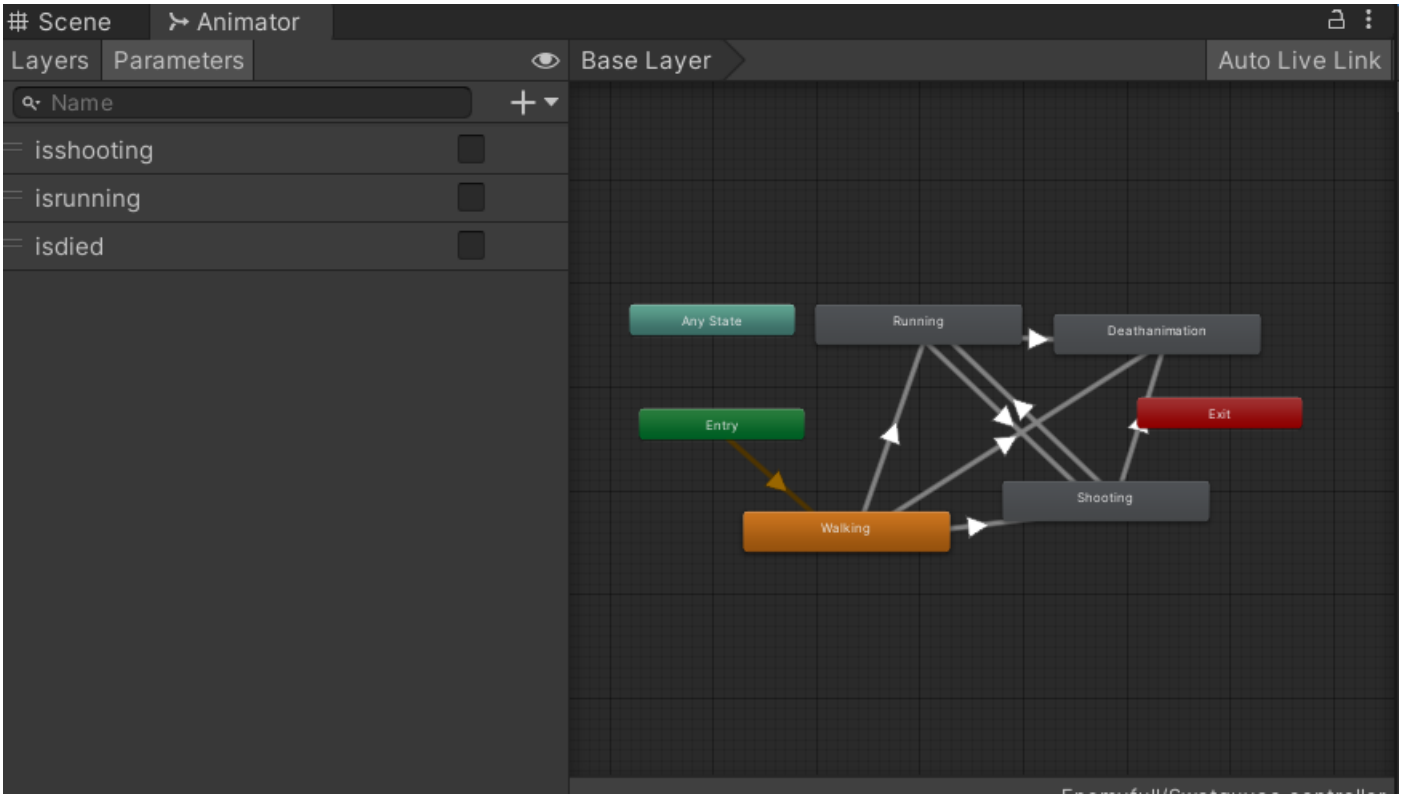




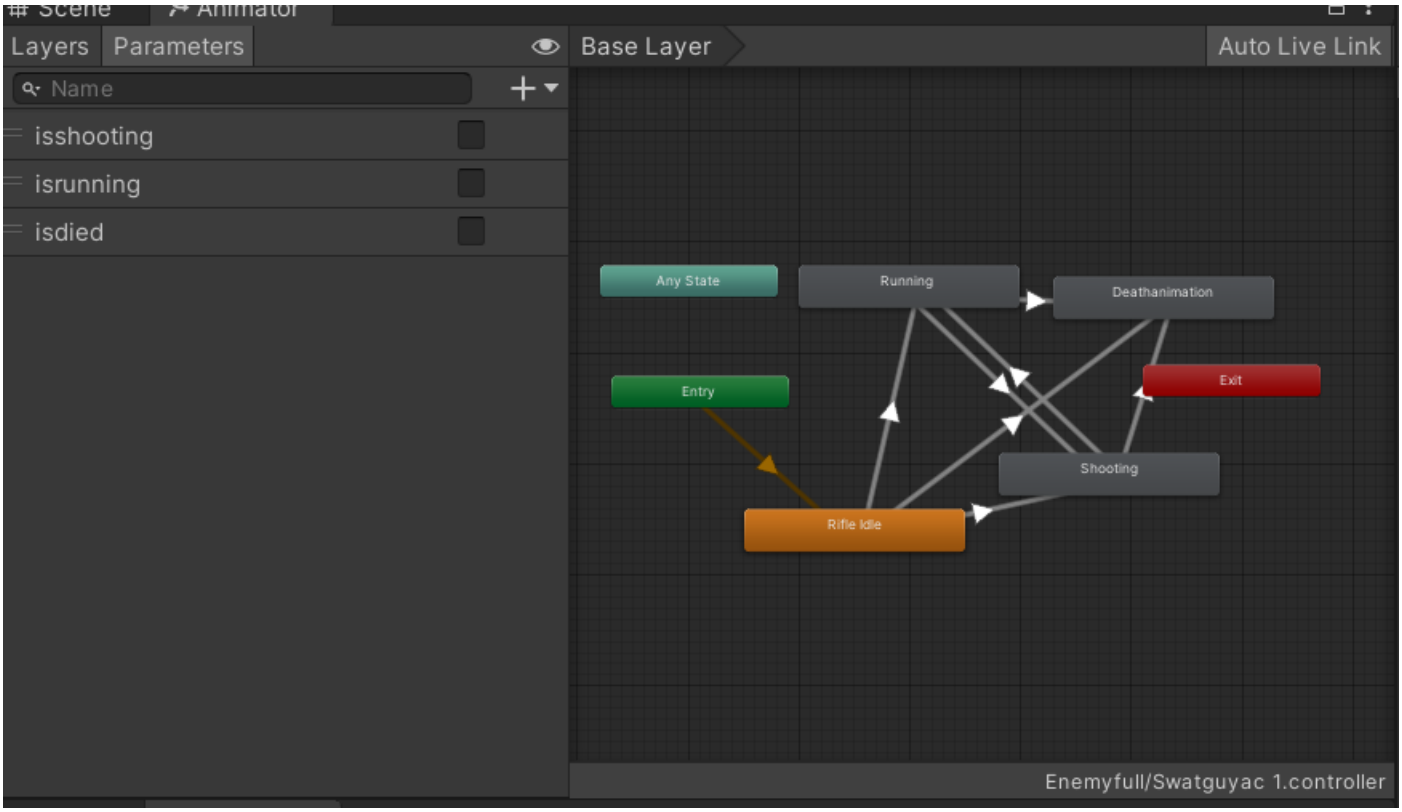
And after that, we use a component called ***ANIMATOR*** to apply trigger to each animation. So that each animation will play after the execution of a specific function not before that.

Enemy Animator

The one which is patrolling;

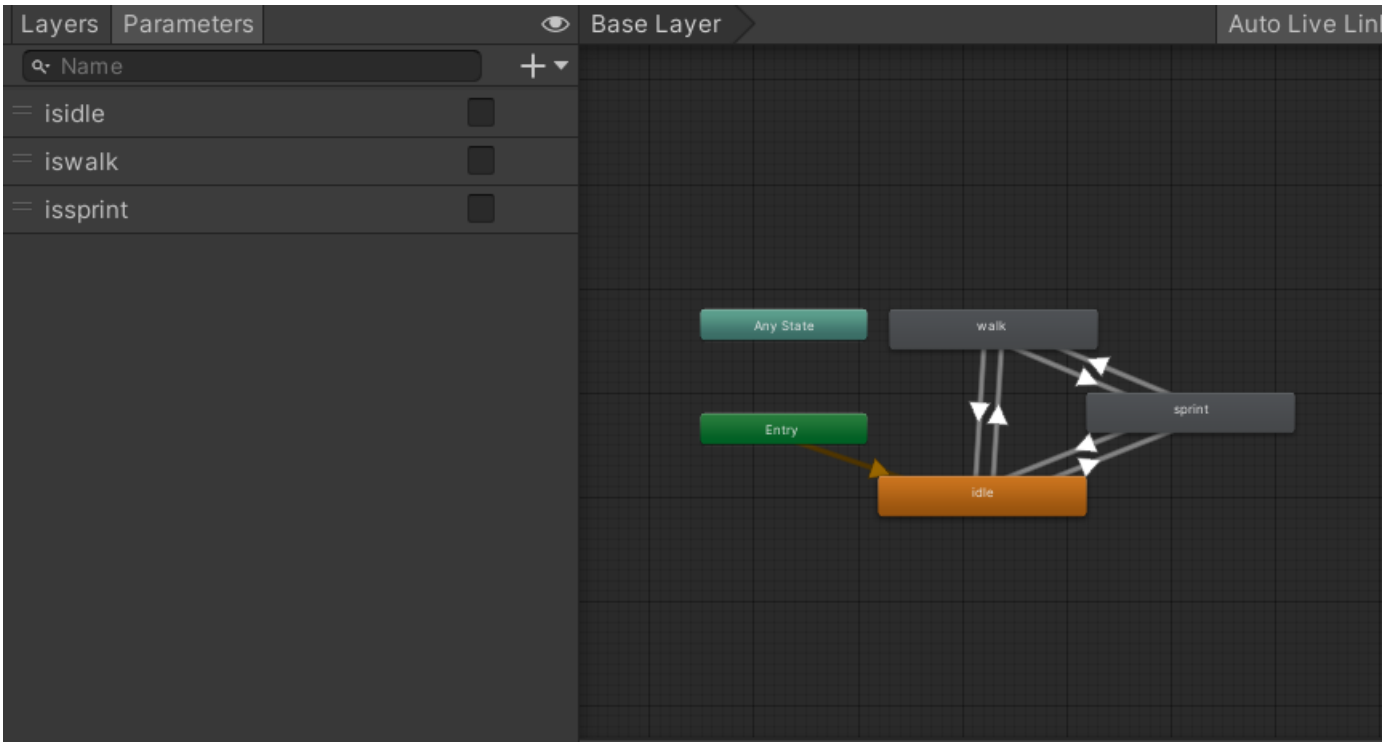


The one which is not patrolling;

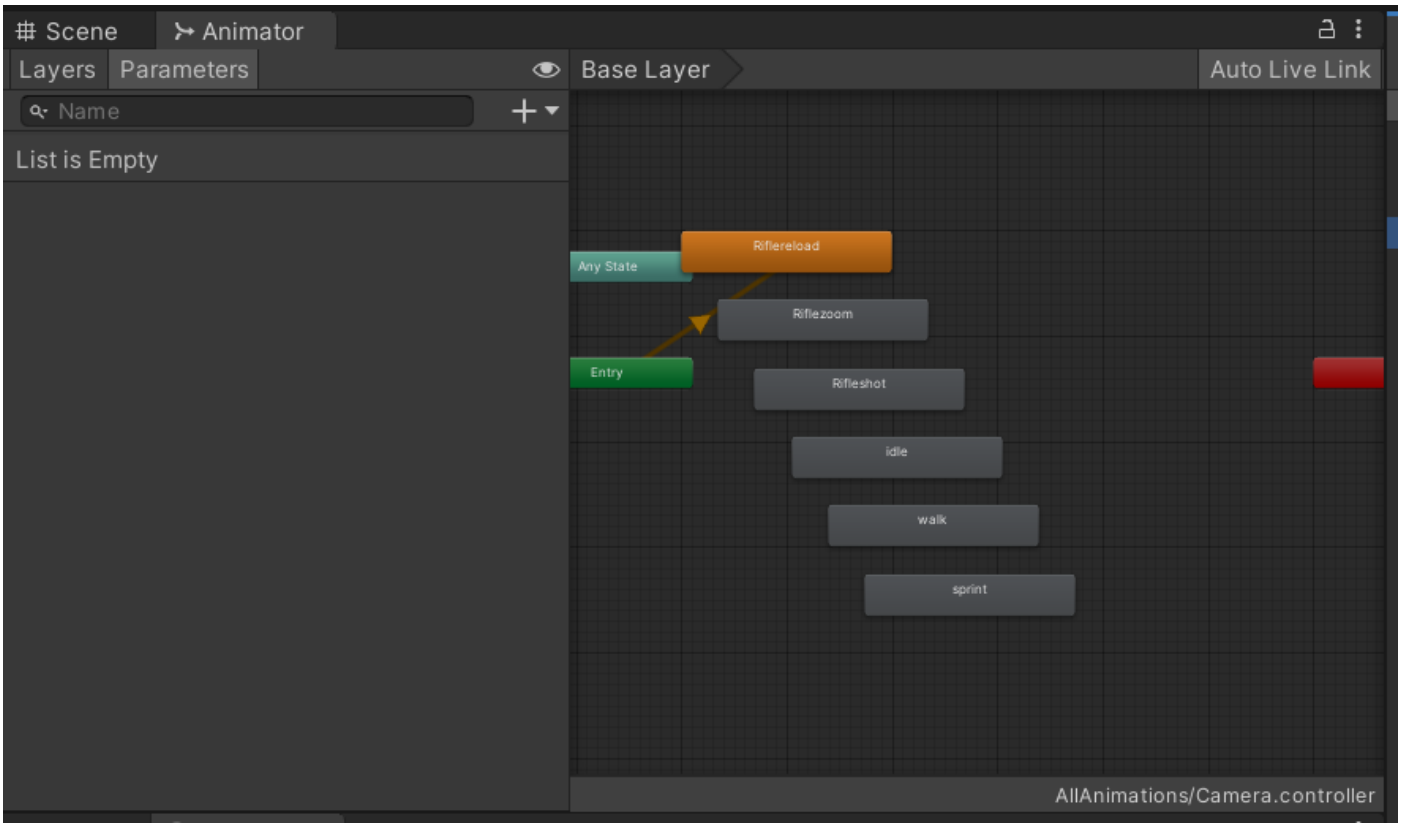


Player Animator:

For idle,walk,sprint;



For gun animations zoom reload;



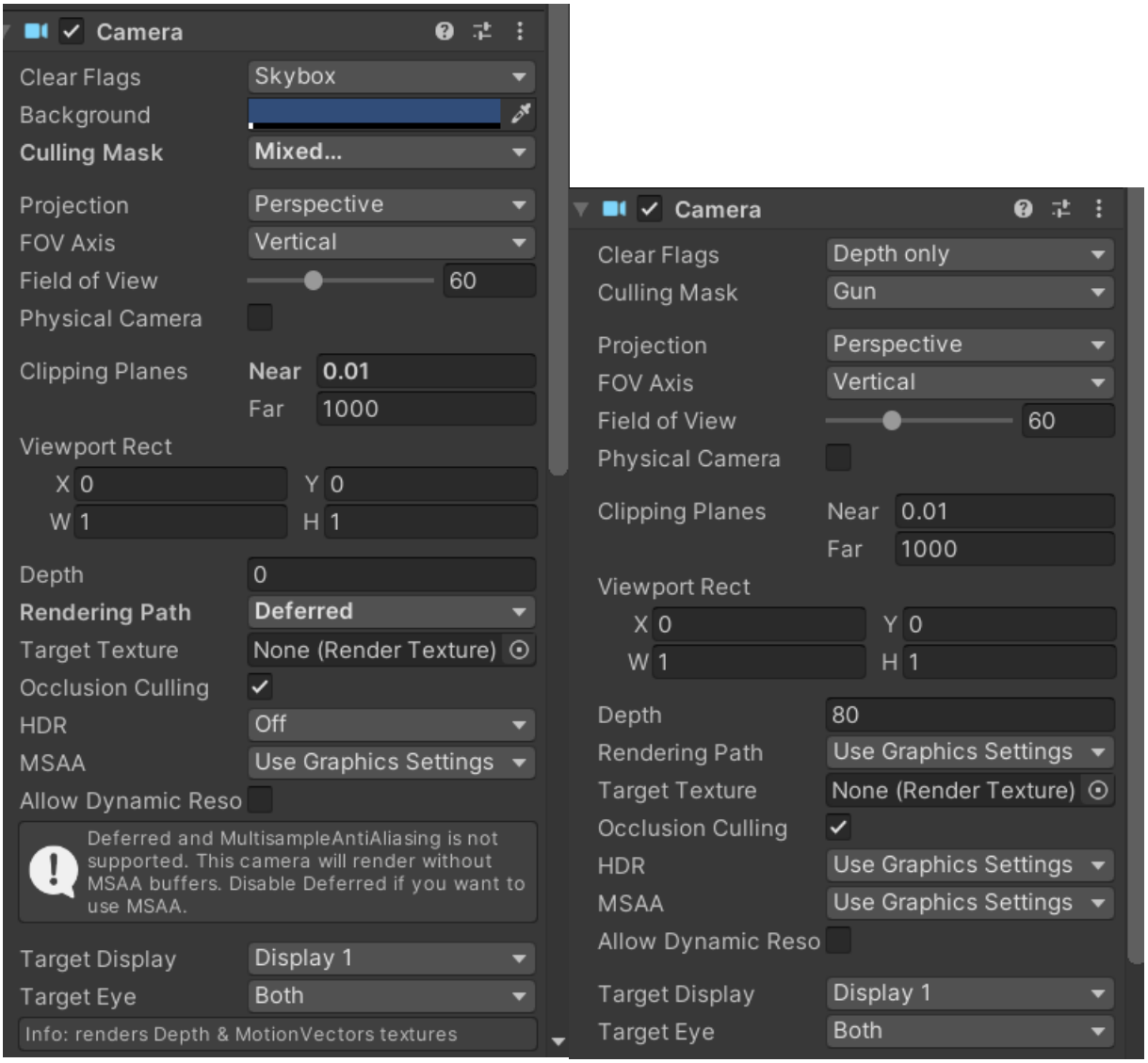
Camera

In first scene we have one camera which displays the video

In second scene we have one camera which displays the menu.

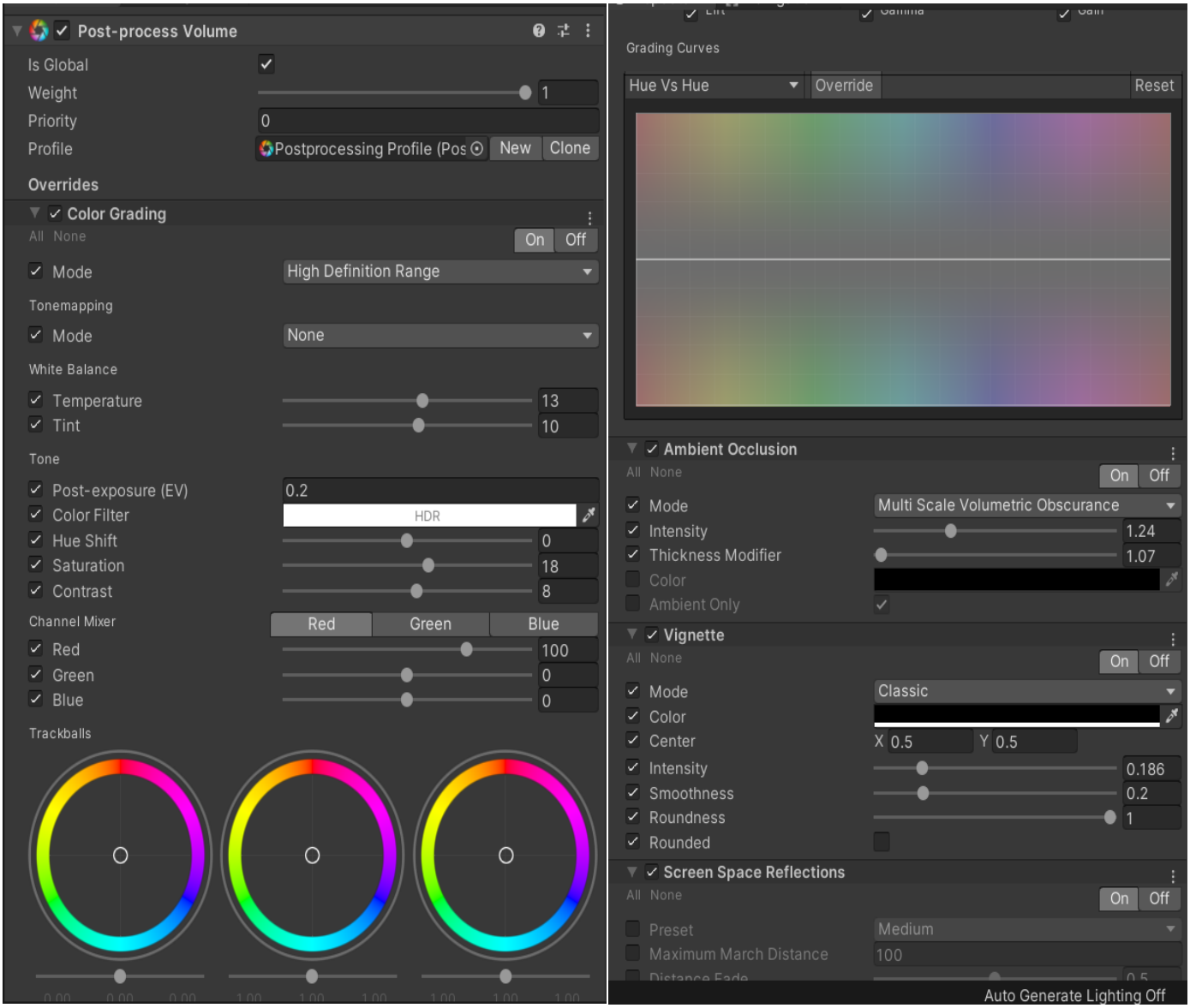
In third scene;

We have three cameras first which render the Whole scene except gun which player is holding and second one render only the gun so that problem of clipping get removed. And third camera is our minimap.



Topic : 8

Post Processing



Scripts

Total 28 scripts are used;

The index of the script are as follows;

For player movement by default unity player movement FPScontroller script used and for crosshair as well.

Script no.1:Used for ammo pickup;

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

public class Ammopickup : MonoBehaviour
{
    public AudioSource AmmoPickupSound;
    void OnTriggerEnter(Collider other)
    {
        AmmoPickupSound.Play();
        if (Globalammo.LoadedAmmo == 0)
            Globalammo.LoadedAmmo += 30;
        else
        Globalammo.CurrentAmmo += 30;
        this.gameObject.SetActive(false);
    }
}
```

Script no.2:Enemy script that is attached on every enemy player;

```
using UnityEngine;
using UnityEngine.AI;

public class EnemyAiTutorial : MonoBehaviour
{
    public NavMeshAgent agent;

    public Transform player;

    public LayerMask whatIsGround, whatIsPlayer;

    public float health;

    public Animator enemyanimator;

    public AudioSource audio;
    //Attacking
    public float timeBetweenAttacks;
    bool alreadyAttacked;

    //States
    public float sightRange, attackRange;
    public bool playerInSightRange, playerInAttackRange;
    public bool playerinvision=false;

    //PATROLLSTATE
    public Patrolling patrolling;

    //attacking
    public Playerhealth playerhealth;
    public int damageamount;
    private void Awake()
```

```
    {
        player = GameObject.FindWithTag("Player").transform;
        agent = GetComponent<NavMeshAgent>();
        enemyanimator = GetComponent<Animator>();
    }

    private void Update()
    {
        //Check for sight and attack range
        if (health > 0)
        {
            playerInSightRange = Physics.CheckSphere(transform.position, sightRange, whatIsPlayer);
            playerInAttackRange = Physics.CheckSphere(transform.position, attackRange,
whatIsPlayer);
            RaycastHit hit;
            Vector3 playerdirection = player.position - transform.position;
            if (Physics.Raycast(transform.position, playerdirection, out hit))
            {
                if (hit.transform.tag == "Player")
                {
                    playerinvision = true;

                }
                else
                {
                    playerinvision = false;
                }
            }
            if (!playerInSightRange && !playerInAttackRange)
                Patroling();
            if (playerInSightRange && !playerInAttackRange)
            {

                enemyanimator.SetBool("isshooting", false);

                enemyanimator.SetBool("isrunning", true);
                ChasePlayer();
                patrolling.enabled = false;
```

```
        }
        if (playerInAttackRange && playerInSightRange && playerinvision == true)
        {

            enemyanimator.SetBool("isrunning", false);
            enemyanimator.SetBool("isshooting", true);
            AttackPlayer();
            patrolling.enabled = false;
        }
    }
}

private void Patroling()
{
    //patrolling;
    patrolling.enabled = true;
}



private void ChasePlayer()
{

    agent.SetDestination(player.position);
    agent.speed = 6;
}

private void AttackPlayer()
{
    //Make sure enemy doesn't move
    agent.SetDestination(transform.position);

    transform.LookAt(player);
    Vector3 playerdirection = player.position - transform.position;
    if (!alreadyAttacked)
    {
        ///Attack code here
        RaycastHit Shot;
        if (Physics.Raycast(transform.position, playerdirection, out Shot))
```

```
        {
            if (Shot.transform.tag == "Player")
            {
                alreadyAttacked = true;

                Invoke(nameof(ResetAttack), timeBetweenAttacks);
                //enemydamage
                int prob = Random.Range(1, 10);
               // if(prob==Random.Range(1,8))
                playerhealth.TakeDamage(damageamount);
                if(health>0)
                audio.Play();
            }

        }
        ///End of attack code
    }
}
private void ResetAttack()
{
    alreadyAttacked = false;
}

public void TakeDamage(int damage)
{
    health -= damage;
    playerInSightRange = true;
    playerInAttackRange = true;

    if (health <= 0)
    {
        patrolling.enabled = false;

        enemyanimator.SetBool("isshooting", false);
        enemyanimator.SetBool("isrunning", false);
        enemyanimator.SetBool("isdied", true);

        Invoke(nameof(DestroyEnemy), 4f);
```

```
        }
    }
    private void DestroyEnemy()
    {
        Destroy(gameObject);
    }

    private void OnDrawGizmosSelected()
    {
        Gizmos.color = Color.red;
        Gizmos.DrawWireSphere(transform.position, attackRange);
        Gizmos.color = Color.yellow;
        Gizmos.DrawWireSphere(transform.position, sightRange);
    }
}
```

Script no.3 ;Enemy movement check whether player is in vision or not;

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.AI;
public class Enemymovement : MonoBehaviour
{
    public float fov = 120f;

    public Transform Target;

    public bool inSight;

    public float awakedistance=80f;
    // Update is called once per frame
    public bool AwareofPlayer = false;
    public bool playerinvision = false;
    public NavMeshAgent enemyagent;

    public float shootdistance;
    public Patrolling patrolofsc;
    Animator enemyanimator;

    public GameObject forshoot;
    public bool itisshooting=false;
    private void Start()
    {
        enemyanimator = GetComponent<Animator>();

    }
    void Update()
    {
        drawray();
        float PlayerDistance = Vector3.Distance(Target.position, transform.position);
        Vector3 playerdirection = Target.position - transform.position ;
        float playerAngle = Vector3.Angle(transform.position, playerdirection);
```

```
    if(playerAngle<=fov/2)
    {
        inSight = true;
    //    Debug.Log("Playerinsight");
    }
    else
    {
        inSight = false;
    }
    if (inSight == true && PlayerDistance <= awakedistance&&playerinvision==true)
        AwareofPlayer = true;

    if(PlayerDistance<=shootdistance)
    {

        forshoot.GetComponent<Enemyshooting>().Shoot();

    }
    if (AwareofPlayer == true&&itisshooting==false)
    {
        enemyagent.SetDestination(Target.position);
        enemyanimator.SetBool("isrunning", true);
        enemyagent.speed = 6f;
        patrolofsc.enabled = false;
    }




}
void drawray()
{
    Vector3 playerdirection = Target.position - transform.position;
    RaycastHit hit;
    if(Physics.Raycast(transform.position,playerdirection,out hit))
    {
        if(hit.transform.tag=="Player")
        {
            playerinvision = true;
```

```
            }
            else
            {
                playerinvision = false;
            }
        }
    }

}
```

Script no.4;Used for deducting the enemy health and killing him;

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

public class Enemysc : MonoBehaviour
{
    // Start is called before the first frame update
    public int EnemyHealth = 10;
    Animator enemyanimator;
    void DeductPoints(int DamageAmount)
    {
        EnemyHealth -= DamageAmount;
    }
    private void Start()
    {
        enemyanimator = GetComponent<Animator>();
    }

    void Update()
    {
        if (EnemyHealth <= 0)
        {
            enemyanimator.SetBool("isdied", true);
        }
    }
}
```

Script no.5;Used for enemy shooting animation;

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.AI;
public class Enemyshooting : MonoBehaviour
{
    // Start is called before the first frame update
    public Animator enemyanimator;
    public Enemymovement checkforshooting;
  public  void Shoot()
    {

             RaycastHit Shot;
        if (Physics.Raycast(transform.position, transform.TransformDirection(Vector3.forward), out
Shot))
        {
            if (Shot.transform.tag == "Player")
            {
                enemyanimator.SetBool("isshooting", true);

                checkforshooting.itisshooting = true;
                //enemydamage
            }
            else
            {
                checkforshooting.itisshooting = false;
               enemyanimator.SetBool("isrunning", true);
            }

        }


    }
}
```

Script no.6;Used for calculation ammo levels;

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

public class Globalammo : MonoBehaviour
{
    // Start is called before the first frame update
    public static int CurrentAmmo=180;
    public int InternalAmmo;
    public GameObject AmmoDisplay;

    public static int LoadedAmmo=30;
    public int InternalLoaded;
    public GameObject LoadedDisplay;
    public Weaponswitch objectforws;

    public static int CurrentAmmosni = 10;
    public int InternalAmmosni;



    public static int LoadedAmmosni = 5;
    public int InternalLoadedsni;

    void Update()
    {
        if (objectforws.selectedweapon == 0)
        {
            InternalAmmo = CurrentAmmo;
            InternalLoaded = LoadedAmmo;
            AmmoDisplay.GetComponent<Text>().text = "" + InternalAmmo;
            LoadedDisplay.GetComponent<Text>().text = "" + LoadedAmmo;
        }

        if(objectforws.selectedweapon==1)
        {
            InternalAmmosni = CurrentAmmosni;
```

```
            InternalLoadedsni = LoadedAmmosni;
            AmmoDisplay.GetComponent<Text>().text = "" + InternalAmmosni;
            LoadedDisplay.GetComponent<Text>().text = "" + LoadedAmmosni;
        }

    }
}
```

Script no.7;Used for Gunfire Assault rifle;

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

public class Gunfire : MonoBehaviour
{
    public GameObject Flash;
    private float nextfire = 0.0f;
    public float firerate = 0.5f;
    public IEnumerator MuzzleOff()
    {

        yield return new WaitForSeconds(0.15f);
        Flash.SetActive(false);
    }

    void Update()
    {
        fire();
    }
    void fire()
    {
        if(Input.GetButton("Fire1")&&Time.time>nextfire)
        {
            nextfire = Time.time + firerate;
            Forcefire();
        }
    }
    void Forcefire()
    {
        if (Input.GetButton("Fire1") && Globalammo.LoadedAmmo > 0)
        {
            AudioSource gunsound = GetComponent<AudioSource>();
            gunsound.Play();
            Flash.SetActive(true);
            StartCoroutine(MuzzleOff());
            GetComponent<Animation>().Play("Gunshot");
```

```
            Globalammo.LoadedAmmo -= 1;
        }


    }
}
```

Script no.8;Used for damaging the enemy

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

public class Handgundamage : MonoBehaviour
{
    // Start is called before the first frame update
    public int DamageAmount = 5;
    public float TargetDistance;
    public float AllowedRange = 15.0f;
    private float nextfire = 0.0f;
    public float firerate = 0.2f;
    public Camera playercamera;
    public GameObject blood;
   // public GameObject impact;
    void Update()
    {
        icall();
    }
    void icall()
    {
        if (Input.GetButton("Fire1") && Time.time > nextfire)
        {
            nextfire = Time.time + firerate;
            raycast();
        }
    }
    void raycast()
    {
        if (Input.GetButton("Fire1") && Globalammo.LoadedAmmo > 0)
        {
            RaycastHit Shot;
            if (Physics.Raycast(playercamera.transform.position, playercamera.transform.forward, out
Shot))
            {
                TargetDistance = Shot.distance;
```

```
                if (TargetDistance < AllowedRange)
                {
                    Shot.transform.SendMessage("TakeDamage", DamageAmount,
SendMessageOptions.DontRequireReceiver);
                    if(Shot.transform.tag=="Enemy")
                    {
                        Instantiate(blood, Shot.point, Quaternion.LookRotation(Shot.normal));
                    }
                }
               // Instantiate(impact, Shot.point, Quaternion.LookRotation(Shot.normal));//enemy
damage;
            }
        }
    }
}
```

Script no.9;Used for health bar

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
public class Healthbarscript : MonoBehaviour
{

    public Slider slider;
    public Gradient gradient;
    public Image fill;

    public void SetHealth(int health)
    {
        slider.value = health;
       fill.color= gradient.Evaluate(slider.normalizedValue);
    }
    public void MaxHealth(int health)
    {
        slider.maxValue = health;
        slider.value = health;
        fill.color = gradient.Evaluate(1f);
    }



}
```

Script no.10;Used for helicopter fans;

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
public class Healthbarscript : MonoBehaviour
{

    public Slider slider;
    public Gradient gradient;
    public Image fill;

    public void SetHealth(int health)
    {
        slider.value = health;
       fill.color= gradient.Evaluate(slider.normalizedValue);
    }
    public void MaxHealth(int health)
    {
        slider.maxValue = health;
        slider.value = health;
        fill.color = gradient.Evaluate(1f);
    }



}
```

Script no.11;Used for main menu

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;
public class Mainmenu : MonoBehaviour
{
    public void PlayGame()
    {
        SceneManager.LoadScene("Maingame");
    }

    //use for quiting of game
    public void QuitGame()
    {
        Application.Quit();
    }
}
```

Script no.12;Used for objectives completion

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
public class Objectivecompletionamphora : MonoBehaviour
{
    public Camera cam;
    float Distancefromtarget;
    public GameObject TextDisplay;
    public GameObject obj1;
    public GameObject obj2;
    // Start is called before the first frame update
    void Start()
    {

    }

    // Update is called once per frame
    void Update()
    {
        RaycastHit Shot;
        if (Physics.Raycast(transform.position, cam.transform.forward, out Shot))
        {
                Distancefromtarget = Shot.distance;
                if (Input.GetButton("Action2"))
                {
                    if (Distancefromtarget <= 2)
                    {
                        StartCoroutine(TakeNineMil());
                    }
                }

        }

    }
    void OnMouseOver()
    {
```

```
        if (Distancefromtarget <= 2)
        {
            TextDisplay.GetComponent<Text>().text = "Take the Amphora and skull";
        }
    }

    void OnMouseExit()
    {
        TextDisplay.GetComponent<Text>().text = "";
    }

    IEnumerator TakeNineMil()
    {
        obj1.SetActive(true);
        obj2.SetActive(true);
        transform.position = new Vector3(0, -1000, 0);

        yield return new WaitForSeconds(0.1f);
    }

}
```

Script no.13;Objective completion UI;

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
public class Objectivescompletion : MonoBehaviour
{

    public Camera cam;
    float Distancefromtarget;
    public GameObject TextDisplay;
    public GameObject obj1;
    // Start is called before the first frame update
    void Start()
    {

    }

    // Update is called once per frame
    void Update()
    {
        RaycastHit Shot;
        if(Physics.Raycast(transform.position,cam.transform.forward , out Shot))
        {
            Distancefromtarget = Shot.distance;
        }
        if (Input.GetButtonDown("Action"))
        {
            if (Distancefromtarget <= 2)
            {
                StartCoroutine(TakeNineMil());
            }
        }
    }
    void OnMouseOver()
    {
        if (Distancefromtarget <= 2)
        {
```

```
            TextDisplay.GetComponent<Text>().text = "Take the Chip";
        }
    }

    void OnMouseExit()
    {
        TextDisplay.GetComponent<Text>().text = "";
    }

    IEnumerator TakeNineMil()
    {
        obj1.SetActive(true);

        transform.position = new Vector3(0, -1000, 0);

        yield return new WaitForSeconds(0.1f);
    }


}
```

Script no.14;Used for enemy patrolling

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




public class Patrolling : MonoBehaviour
{
    [SerializeField]
    bool patrolwaiting;

    [SerializeField]
    float totalwaittime = 0f;



    [SerializeField]
    float switchprobablity=0.3f;

    [SerializeField]
    List<Waypoint> patrolPoints;

    NavMeshAgent navmeshagent;
    int currentpatrolindex;
    bool travelling;
    bool waiting;
    bool patrolforward;
    float waittimer;

    public void Start()
    {
        navmeshagent = this.GetComponent<NavMeshAgent>();
        currentpatrolindex = 0;
        SetDestination();
```

```
}
public void Update()
{

    if(travelling && navmeshagent.remainingDistance<=1f)
    {
        travelling = false;
        if(patrolwaiting)
        {
            waiting = true;
            waittimer = 0f;
        }
        else
        {
            ChangePatrolpoints();
            SetDestination();
        }
    }
    if(waiting)
    {
        waittimer += Time.deltaTime;
        if(waittimer>=totalwaittime)
        {
            waiting = false;
            ChangePatrolpoints();
            SetDestination();
        }
    }
}
public void SetDestination()
{
    if(patrolPoints!=null)
    {
        Vector3 targetvector = patrolPoints[currentpatrolindex].transform.position;
        navmeshagent.SetDestination(targetvector);
        travelling = true;
    }
}
```

```
    public void ChangePatrolpoints()
    {
        if(UnityEngine.Random.Range(0f,1f)<=switchprobablity)
        {
            patrolforward = !patrolforward;
        }
        if(patrolforward)
        {
            currentpatrolindex = (currentpatrolindex + 1) % patrolPoints.Count;

            //here i can use we are doing currentpetrolindex++
            //or if(currentpatrolindex>=patrolpoints.count)
            //then currentpatrolindex=0;

        }
        else
        {

            /*here what we are doing currentpatrolindex--;
             * if(curreentpatrolindex<0) then currentpatrolindex=patrolpoints.count-1;
             *
                    */
            if(--currentpatrolindex<0)
            {
                currentpatrolindex = patrolPoints.Count - 1;
            }
        }

    }

}
```

Script no.15;Used for game over and player health,sounds

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

public class Playerhealth : MonoBehaviour
{
    public int maxhealth = 100;
    public int currenthealth;
    public int painsound;
    public AudioSource hurt1;
    public AudioSource hurt2;
    public AudioSource hurt3;

    public Healthbarscript barreference;
     void Start()
    {
        currenthealth = maxhealth;
        barreference.MaxHealth(currenthealth);
    }

    // Update is called once per frame
    void Update()
    {
        if(currenthealth<=0)
        {
            //GAMEOVERSCENE
          Application.Quit();
        }

    }

    public void TakeDamage(int damage)
    {
        currenthealth -= damage;
        int hurt = Random.Range(0, 2);
```

```
        if (hurt == 0)
            hurt1.Play();
            if (hurt == 1)
                hurt2.Play();
        if (hurt == 2)
            hurt3.Play();

        barreference.SetHealth(currenthealth);
    }
}
```

Script no.16;Player idle breath walk sprint animation controller

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

public class Playerwalkrunidlecontroller : MonoBehaviour
{
    // Start is called before the first frame update

    public Animator playeranim;
    void Start()
    {

    }

    // Update is called once per frame
    void Update()
    {
        float x = Input.GetAxis("Horizontal");
        float y = Input.GetAxis("Vertical");
        if (Input.GetKey(KeyCode.LeftShift))
        {
            playeranim.SetBool("issprint", true);
            playeranim.SetBool("iswalk", false);
            playeranim.SetBool("isidle", false);
        }
        else if(x==0f&&y==0f)
        {
            playeranim.SetBool("isidle", true);
            playeranim.SetBool("issprint", false);
            playeranim.SetBool("iswalk", false);
        }
        else
        {
            playeranim.SetBool("issprint", false);
            playeranim.SetBool("iswalk", true);
```

```
            playeranim.SetBool("isidle", false);
        }
    }
}
```

Script no.17;Sniper raycasting

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

public class Raycastsniper : MonoBehaviour
{
    public int DamageAmount = 30;
    public float TargetDistance;
    public float AllowedRange = 15.0f;
    public bool firedone = false;
    public Camera playercamera;
    public GameObject blood;
    public IEnumerator MuzzleOff()
    {

        yield return new WaitForSeconds(2f);
        firedone = false;

    }

    void Update()
    {
        if (Input.GetButtonDown("Fire1") && firedone == false && Globalammo.LoadedAmmosni > 0)
        {
            firedone = true;
           // AudioSource gunsound = GetComponent<AudioSource>();
           // gunsound.Play();
           // GetComponent<Animation>().Play("Snipershot");
            RaycastHit Shot;
            if (Physics.Raycast(playercamera.transform.position, playercamera.transform.forward, out
Shot))
            {
                TargetDistance = Shot.distance;
                if (TargetDistance < AllowedRange)
                {
                    Shot.transform.SendMessage("TakeDamage", DamageAmount,
SendMessageOptions.DontRequireReceiver);
```

```
                    if(Shot.transform.tag=="Enemy")

                    Instantiate(blood, Shot.point, Quaternion.LookRotation(Shot.normal));

                    //  Shot.transform.SendMessage("TakeDamage", DamageAmount,
SendMessageOptions.DontRequireReceiver);

                }

            }

            StartCoroutine(MuzzleOff());


            //Globalammo.LoadedAmmosni -= 1;

        }

    }


}
```

Script no.18;Sniper Rifle damge with firerate +

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

public class Rifledamage : MonoBehaviour
{
    // Start is called before the first frame update
    public int DamageAmount = 5;
    public float TargetDistance;
    public float AllowedRange = 15.0f;
    private float nextfire = 0.0f;
    public float firerate = 1f;
    public Riflefire checkfirerifle;
    void Update()
    {
        icall();
    }
    void icall()
    {
        if (Input.GetButtonDown("Fire1") && Time.time > nextfire)
        {
            nextfire = Time.time + firerate;
            raycast();
        }
    }
    void raycast()
    {
        if (Input.GetButtonDown("Fire1") && Globalammo.LoadedAmmo > 0 )
        {
            RaycastHit Shot;
            if (Physics.Raycast(transform.position, transform.TransformDirection(Vector3.forward),
out Shot))
            {
                TargetDistance = Shot.distance;
                if (TargetDistance < AllowedRange)
                {
```

```
                    Shot.transform.SendMessage("TakeDamage", DamageAmount,
SendMessageOptions.DontRequireReceiver);

                }

            }

        }

    }

}
```

Script no.19;Sniper rifle ammo deduction and animation playing

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

public class Riflefire : MonoBehaviour
{
    public int DamageAmount = 5;
    public float TargetDistance;
    public float AllowedRange = 15.0f;
    public bool firedone = false;

    public IEnumerator MuzzleOff()
    {

        yield return new WaitForSeconds(2f);
        firedone = false;

    }

    void Update()
    {
        if (Input.GetButtonDown("Fire1")&&firedone==false&&Globalammo.LoadedAmmosni>0)
        {
            firedone = true;
            AudioSource gunsound = GetComponent<AudioSource>();
            gunsound.Play();
           GetComponent<Animation>().Play("Snipershot");
           /* RaycastHit Shot;
            if (Physics.Raycast(transform.position, transform.TransformDirection(Vector3.forward),
out Shot))
            {
                TargetDistance = Shot.distance;
                if (TargetDistance < AllowedRange)
                {
                    Shot.transform.SendMessage("DeductPoints", DamageAmount,
SendMessageOptions.DontRequireReceiver);
```

```
                }
            }*/
            StartCoroutine(MuzzleOff());


             Globalammo.LoadedAmmosni -= 1;
        }
    }

}
```

Script no.20;Assault rifle reloading

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

public class Riflereloadsc : MonoBehaviour
{
    public AudioSource ReloadSound;
   // public GameObject CrossObject;
    public GameObject MechanicsObject;
    public int ClipCount;
    public int ReserveCount;
    public int ReloadAvailable;
    public Gunfire GunComponent;

    void Start()
    {
        GunComponent = GetComponent<Gunfire>();
    }

    void Update()
    {
        ClipCount = Globalammo.LoadedAmmo;
        ReserveCount = Globalammo.CurrentAmmo;

        if (ReserveCount == 0)
        {
            ReloadAvailable = 0;
        }
        else
        {
            ReloadAvailable = 30 - ClipCount;
        }

        if (Input.GetButtonDown("Reload"))
        {
            if (ReloadAvailable >= 1)
            {
```

```
                if (ReserveCount <= ReloadAvailable)
                {
                    Globalammo.LoadedAmmo += ReserveCount;
                    Globalammo.CurrentAmmo -= ReserveCount;
                    ActionReload();
                }
                else
                {
                    Globalammo.LoadedAmmo += ReloadAvailable;
                    Globalammo.CurrentAmmo -= ReloadAvailable;
                    ActionReload();
                }
            }
            StartCoroutine(EnableScripts());

        }
    }

    public IEnumerator EnableScripts()
    {
       yield return new WaitForSeconds(1.1f);
        GetComponent<Gunfire>().enabled = true;
     //   CrossObject.SetActive(true);
        MechanicsObject.SetActive(true);
    }

    void ActionReload()
    {
        GetComponent<Gunfire>().enabled = false;
      //  CrossObject.SetActive(false);
        MechanicsObject.SetActive(false);
        ReloadSound.Play();
        GetComponent<Animation>().Play("Riflereload");
    }
}
```

Script no.21;Assault rifle zooming

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

public class Riflezoom : MonoBehaviour
{
    // Start is called before the first frame update

    public GameObject Playercam;
    public SimpleCrosshair Obj;
    // Update is called once per frame
    void Update()
    {
        if (Input.GetMouseButtonDown(1))
        {
            Playercam.GetComponent<Camera>().fieldOfView = 45;
            Obj.SetSize(20, true);
            Obj.SetGap(5, true);
            Obj.SetThickness(1, true);
        }
        if (Input.GetMouseButtonUp(1))
        {
            Playercam.GetComponent<Camera>().fieldOfView = 60;
            Obj.SetSize(14, true);
            Obj.SetGap(27, true);
            Obj.SetThickness(3, true);
        }
    }
}
```

Script no.22;Sniper rifle zoom change camera ui and clipping in sniper zooming

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

public class Scopechange : MonoBehaviour
{
    // Start is called before the first frame update
    void Start()
    {

    }
    public GameObject Playercam;
    public GameObject sniperobjecttex;
    //public GameObject orignalcurs;
    private bool zoomingin = false;
    public float fieldofview = 60f;
    public Camera clipping;
    // Update is called once per frame
    void Update()
    {
        if(Input.GetMouseButtonDown(1))
        {
            if (zoomingin == false)
            {
                clipping.cullingMask= 1 << LayerMask.NameToLayer("Nothing");  //use for clipping
purposes
                sniperobjecttex.SetActive(true);
                //   orignalcurs.SetActive(false);

                StartCoroutine(Zoomingcam());
                zoomingin = true;
            }
        }
        if(Input.GetMouseButtonUp(1))
        {
            clipping.cullingMask = 1 << LayerMask.NameToLayer("Gun");
```

```
            StopAllCoroutines();
            fieldofview = 60;
            Playercam.GetComponent<Camera>().fieldOfView = fieldofview;
            sniperobjecttex.SetActive(false);

            //  orignalcurs.SetActive(true);
            zoomingin = false;

        }
    }
    IEnumerator Zoomingcam()
    {
        while(fieldofview>5)
        {
            Playercam.GetComponent<Camera>().fieldOfView = fieldofview;
            fieldofview -= 2f;  //decides speed of zooming that when we press right mouse button higher it is faster it zooms;
            yield return new WaitForSeconds(0.01f);
        }
    }
}
```

Script no.23;Skybox moving with a fixed speed

```
using System.Collections;

using System.Collections.Generic;

using UnityEngine;


public class Skymove : MonoBehaviour

{

    // Start is called before the first frame update

    public float rotatespeed = 0.5f;  // i can change it later if i want to thats why i make it public we can write 0.5 also.

    void Start()

    {


    }


    // Update is called once per frame

    void Update()

    {

        RenderSettings.skybox.SetFloat("_Rotation", Time.time*rotatespeed);

    }

}
```

Script no.24;Sniper reloading and animation playing

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

public class Sniperreloadsc : MonoBehaviour
{
    public AudioSource ReloadSound;
    // public GameObject CrossObject;
    public GameObject MechanicsObject;
    public int ClipCount;
    public int ReserveCount;
    public int ReloadAvailable;
    public Riflefire GunComponent;
    // Update is called once per frame
    void Update()
    {
        ClipCount = Globalammo.LoadedAmmosni;
        ReserveCount = Globalammo.CurrentAmmosni;

        if (ReserveCount == 0)
        {
            ReloadAvailable = 0;
        }
        else
        {
            ReloadAvailable = 5 - ClipCount;
        }
        if (Input.GetButtonDown("Reload"))
        {
            if (ReloadAvailable >= 1)
            {
                if (ReserveCount <= ReloadAvailable)
                {
                    Globalammo.LoadedAmmosni += ReserveCount;
                    Globalammo.CurrentAmmosni -= ReserveCount;
                   ActionReload();
                }
```

```
                else
                {
                    Globalammo.LoadedAmmosni += ReloadAvailable;
                    Globalammo.CurrentAmmosni -= ReloadAvailable;
                    ActionReload();
                }
            }


        }
        void ActionReload()
        {
            ReloadSound.Play();
            GetComponent<Animation>().Play("Sniperreload");
        }
    }
}
```

Script no.25;Splash screen to menu switch

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;
public class Splashscreentomenu : MonoBehaviour
{


     void Start()
    {
        StartCoroutine(Splashfinish());
    }
    IEnumerator Splashfinish()
    {
        yield return new WaitForSeconds(12f);
        SceneManager.LoadScene("Mainmenu");
    }
}
```

Script no.26;Subtitle and voice playing in starting of game;

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
public class Subvoice : MonoBehaviour
{
    public GameObject thesub;
    public AudioSource audiosub;
    IEnumerator Subtivoice()
    {
        yield return new WaitForSeconds(2f);
        audiosub.Play();
        thesub.GetComponent<Text>().text = "Listen john,Your task is to collect very rare artifacts and a chip, then extract through the helipad";
        yield return new WaitForSeconds(5f);
        thesub.GetComponent<Text>().text = "";
    }
    private void Start()
    {
        StartCoroutine(Subtivoice());
    }
}
```

Script no.27;Drawing and making the patrol points;

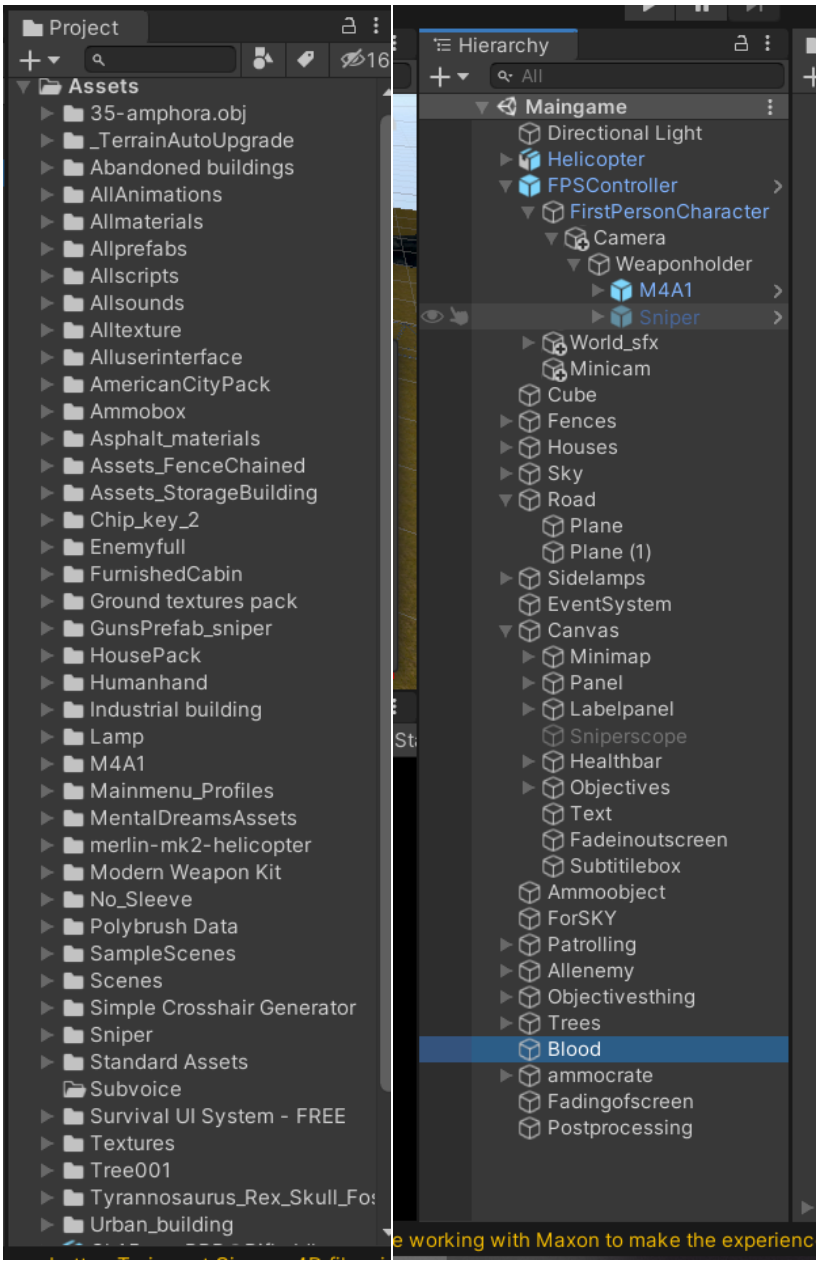
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Waypoint : MonoBehaviour
{
    // Start is called before the first frame update
    [SerializeField]
    protected float debugDrawradius = 1.0f;

    public virtual void OnDrawGizmos()
    {
        Gizmos.color = Color.red;
        Gizmos.DrawWireSphere(transform.position, debugDrawradius);
    }
}
```

Script no.28;Weapon switching between assault rifle and sniper rifle and ui also;

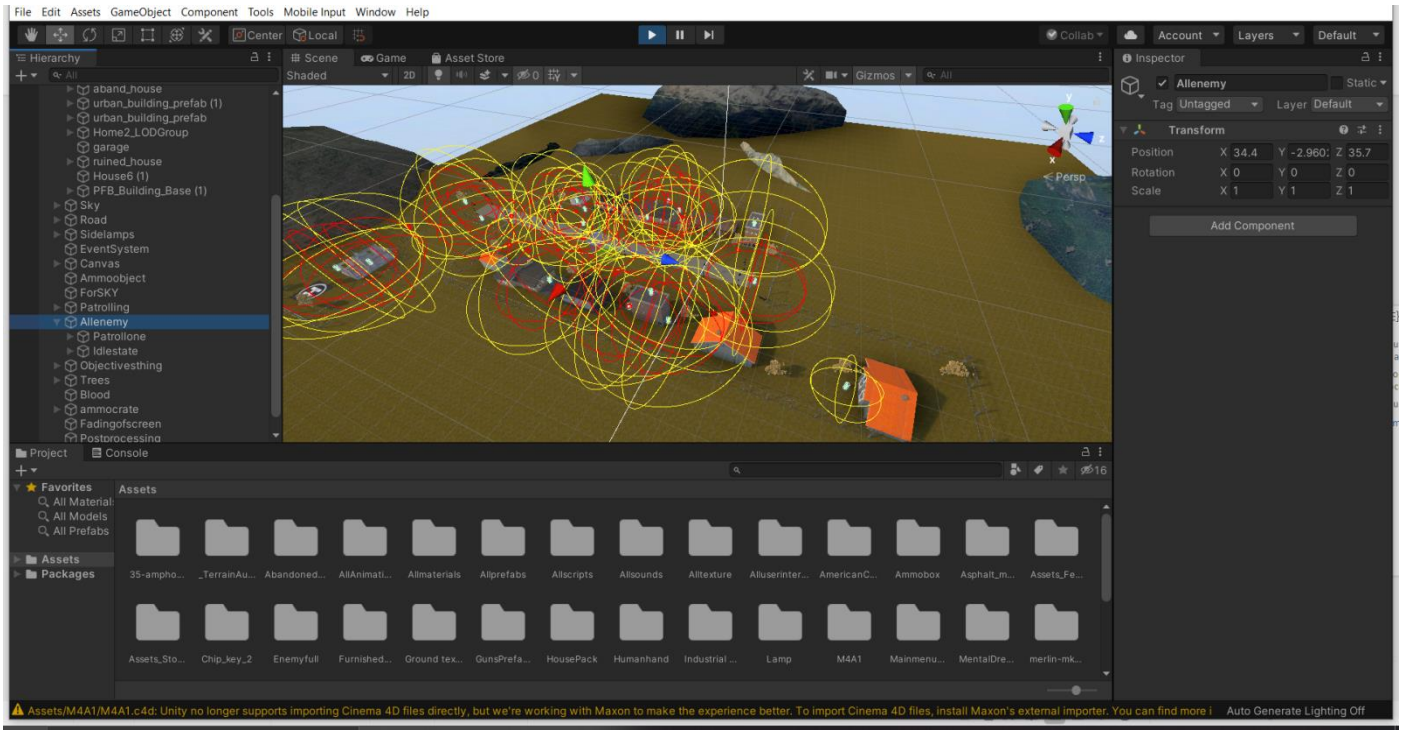
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
public class Weaponswitch : MonoBehaviour
{
    // Start is called before the first frame update
    public int selectedweapon = 0;
    public GameObject obj2;
    void Start()
    {
        Selectweapon();

    }

    // Update is called once per frame
    void Update()
    {

        int previousweaponslect = selectedweapon;
        if(Input.GetKeyDown(KeyCode.X))  //we scroll up >0;
        {
            if (selectedweapon >= transform.childCount - 1)
                selectedweapon = 0;
            else
            selectedweapon++;
        }
        if ( Input.GetKeyDown(KeyCode.Z))  //we scroll up >0;
        {
            if (selectedweapon <= transform.childCount - 1)
                selectedweapon = transform.childCount - 1;
            else
                selectedweapon-- ;
        }
        if (previousweaponslect != selectedweapon)
```

```
            Selectweapon();
    }
    void Selectweapon()
    {
        int i = 0;
        foreach(Transform weapon in transform)
        {
            if (i == selectedweapon)
                weapon.gameObject.SetActive(true);
            else
                weapon.gameObject.SetActive(false);
            i++;
        }
       if(selectedweapon==0)
            obj2.GetComponent<Text>().text = "" + 5.56f+"mm";
       else if(selectedweapon==1)
            obj2.GetComponent<Text>().text = "" + 7.62f + "mm";

    }
}
```

Project panel and hierarchy window;



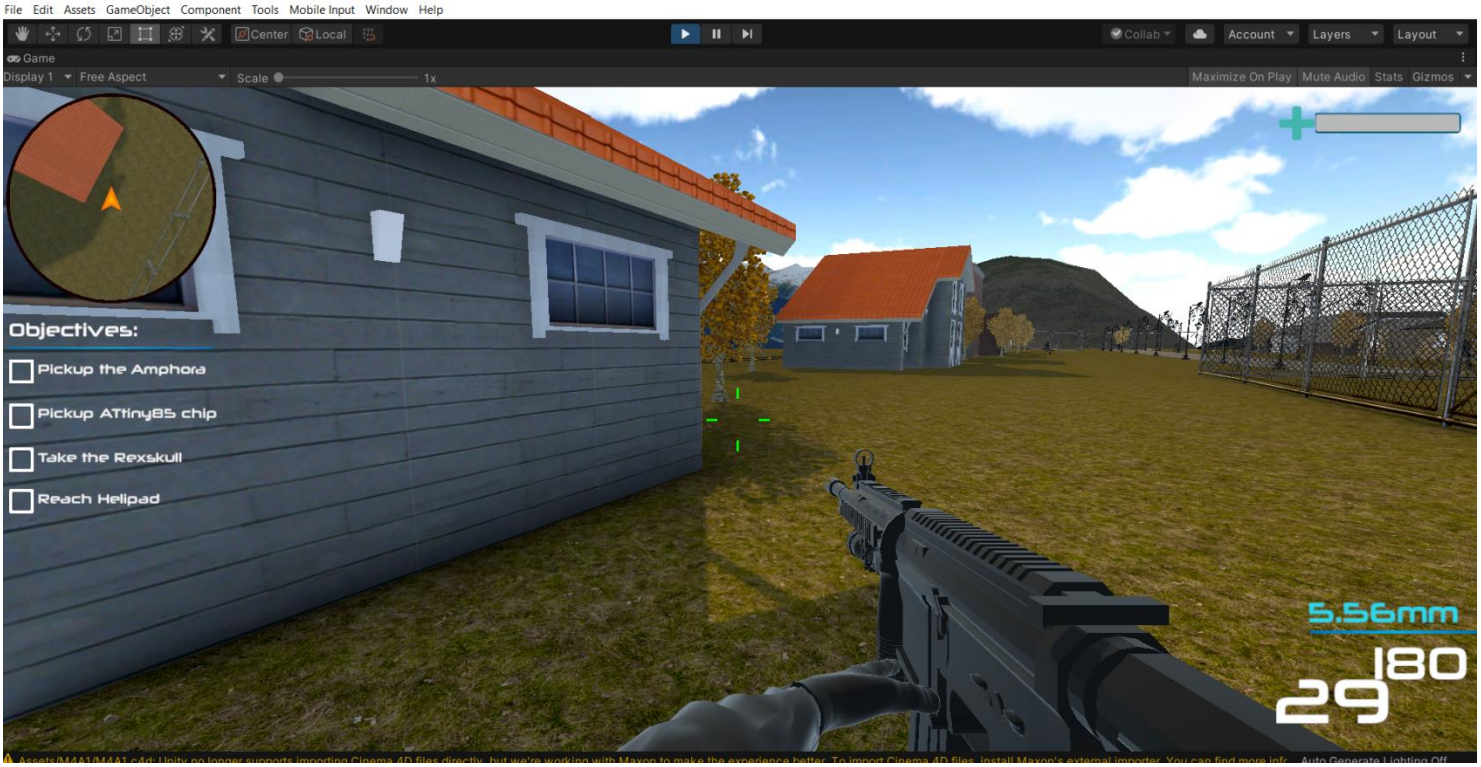
Scene view;


File Edit Assets GameObject Component Tools Mobile Input Window Help
Hierarchy
Scene
Game
Asset Store
Inspector
Alienemy
Transform
Position
Rotation
Scale
Add Component
Project
Console
Favorites
Assets
Packages
Persp
Assets/M4A1/M4A1.c4d: Unity no longer supports importing Cinema 4D files directly, but we're working with Maxon to make the experience better. To import Cinema 4D files, install Maxon's external importer. You can find more i
Auto Generate Lighting Off

Main Game;


Game
Display 1
Free Aspect
Scale
1x
Maximize On Play
Mute Audio
Stats
Gizmos
Objectives:
Pickup the Amphora
Pickup ATtiny85 chip
Take the Rexskull
Reach Helipad
7.62mm
4
5


Center
Local
Collab
Account
Layers
Layout
Game
Display 1
Free Aspect
Scale
1x
Maximize On Play
Mute Audio
Stats
Gizmos
Objectives:
Pickup the Amphora
Pickup ATtiny85 chip
Take the Rexskull
Reach Helipad
5.56mm
29
75



-* FINISH ***-**

References:

https://assetstore.unity.com/

https://free3d.com/

https://www.cgtrader.com/free-3d-models/

https://www.turbosquid.com/Search/3D-Models/free

https://www.renderforest.com/

https://www.mixamo.com/#/

https://www.freepnglogos.com/pics/scope