

**T. C.**

**GAZİ ÜNİVERSİTESİ**

**TEKNOLOJİ FAKÜLTESİ**

**BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**

**2022-2023 GÜZ DÖNEMİ**

**OYUN PROGRAMLAMA DERSİ PROJE SUNUM RAPORU**

**3 BOYUTLU ZOMBİ DÜŞMAN ÖLDÜRME OYUNU – İNANILMAZ ZOMBİ OYUNU**

**Hazırlayan**

**Dovletyar Sultanov (191816401)**

**Aralık, 2022**

**İÇİNDEKİLER**

**GİRİŞ......................................................................................................3**

**KULLANILAN TEKNOLOJİLER.....................................................3**

**SİSTEM GEREKSİNİMLERİ.............................................................3**

**KULLANIM SENARYOLARI VE FONKSİYONLAR....................4**

**Kullanım Senaryosu.........................................................................4**

**Oyun...................................................................................................7**

**Kullanılan Fonksiyonlar...................................................................9**

**KODLAR................................................................................................10**

**KAYNAK................................................................................................15**

**PROJENİN GİTHUB LİNKİ...............................................................16**

**GİRİŞ**

Oyun sektörü son yıllarda büyük bir hızla gelişmiştir. Özellikle pandemi döneminde insanların evlerinde daha fazla vakit geçirme ihtiyacı ve artan mobil cihazların yaygınlaşmasıyla birlikte oyun oynama oranı da çok ciddi şekilde artmıştır. Bu durum, oyun sektörünün büyümesine ve yeni oyunların geliştirilmesine yönelik yatırımların artmasına neden olmuştur. Ayrıca, sanal ve artırılmış gerçeklik teknolojilerinin de oyunların içine entegre edilmesiyle birlikte oyunların gerçekçiliği ve inanılmazlığı da artmıştır. Bu teknolojiler, oyunların daha da gerçekçi ve etkileşimli hale gelmesine yardımcı olmuştur.

Bu proje, insanların sevdiği oyunlardan biri olan düşman zombiyi öldürerek hayatta kalma oyunudur. Eski bir terk edilmiş şehirde oyuncu kendini ve zombileri bulacaktır. Zombiler rastgele daha önce belirlenmiş noktalarda spawn edilmektedir(ortaya çıkmaktadır). Oyuncu ise bu zorlu şartlar altında hayatta kalmaya çalışacaktır. Oyun şimdilik basit düzeyde bir oyundur, ama ileride yeni sahneler, zombi ve silah türleri eklenerek daha da geliştirilecektir.

**KULLANILAN TEKNOLOJİLER**

Bu projede oyun motoru olarak Unity 2021.3.14f1 ve programlama dili olarak ise C# kullanılmıştır. Bunun sebeplerinden bazıları:

Unity oyun motoru, oyun geliştiriciler için birçok kolaylık sağlar ve çoklu platform desteği sunar. Bu sayede, bir oyunu bir kerede birden fazla platformda çalıştırabiliriz. Ayrıca, Unity'nin çok sayıda dokümanı ve aktif bir topluluğu vardır, bu da oyun geliştirme sürecinde yardım ve destek bulmanızı kolaylaştırır. Unity ayrıca, C#, Javascript gibi birçok programlama dilini destekler, bu da oyun geliştiricilere daha fazla esneklik sağlar.

C# ise, oyun sektöründe sıkça kullanılan ve popüler bir dil olmasının yanı sıra, ayrıca güçlü bir dil olması nedeniyle de tercih edilebilir. C#, nesne yönelimli programlama özelliklerine sahiptir ve oyun geliştirme için uygun bir dil olmasının yanı sıra, genel amaçlı uygulamaların geliştirilmesinde de kullanılabilir. C#, ayrıca Microsoft tarafından desteklenen ve güncellenen bir dildir.

**SİSTEM GEREKSİNİMLERİ**

İşletim sistemi: Windows 7 SP - Windows 8/8.1 – Windows 10 – Windows 11 (x64) (tabiki çıktı alınacağı zaman diğer işletim sistemleri içinde uygun olan çıktılar alınabilir)

İşlemci: İntel Core i5-3470 3.2 GHz veya AMD A6-6400K 3.9 GHz

RAM: 6 GB

Ekran kartı: NVIDIA GeForce GT 730 veya AMD Radeon RX550 (2GB veya yükseği)

**KULLANIM SENARYOLARI VE FONKSİYONLAR**

**Kullanım Senaryosu**

Oyun bir terk edilmiş şehirden ve belirli noktalarda rastgele oluşan zombilerden oluşmaktadır. Oyuncu elindeki silahı ile zombileri öldürerek hayatta kalmaya, zombiler ise oyuncuyu ağzı ise ısırarak öldürmeye çalışırlar. Oyuncu zombileri öldürdükdükçe seviyesi artar. Seviye arttıkça ise zombilerin belirlenmiş noktalarda spawn edilme sayısı artar. Yani seviye yükseldikçe zombileri hızlı hızlı naşan alarak mouse sağ tuşu ile ateş açarak öldürmemiz lazım. Oyuncu zombiyi doğru vurarsa zombi öler ve oyundan kaybolur (Destroy() edilir). Ancak oyuncu zombinin saldırısı karşısında başarısız olursa oyuncu yaralanır. Oyuncu zombiden her darbe aldığında can değeri azalır. Can değeri biterse oyun biter ve EndScreen açılır. Sahnede oyuncunu hangi seviyeye kadar hayatta kalabildiğini gösteren tekst, yeniden başlat butonu ve ana menüye dön butonu yer almaktadır. Oyun geliştirilirken bazı hazır assets kullanılmıştır.

Oyuncu oyunun sol kenarında Raunt(Seviye) sayısı görebilir ve böylece kendinin hangi sevieye geldiğini, oyunun ne kadar zor seviyeye geldiğini öğrenebilir.

Graphical user interface, text, application

Description automatically generatedGraphical user interface

Description automatically generated with low confidence

**Şekil 1** Seviyeyi gösteren text objesi

Oyuncu bu haritada gezerek gelen zombilere silahıyla ateş açarak kendini koruyabilir.





**Şekil 2** Oyun haritası

A picture containing text, sign

Description automatically generatedGraphical user interface

Description automatically generatedZombi tarafından saldırıya uğrayan oyuncunun canı, zombinin her saldırısında azalır. Oyuncu ne kadar canı kaldığını oyun ekranının sağ alt kısmında görebilir.

Graphical user interface

Description automatically generated

**Şekil 3** Oyuncunun can göstergesi

Eğer oyuncu oyun kaybederse yani canı 0 a eşit olursa oyun biter ve EndScreen açılır.

**Şekil 4** Oyuncunun kaybettiği ve EndScreen nin açıldığı an

Zombie’ ler oyuncuya saldırmak için oyuncuyu AI ve NavMesh yardımıyla bulurlar.

NavMesh bir oyun içinde hareket eden karakterlerin geçebilecekleri alanları ve bu alanlar arasında nasıl hareket edeceklerini belirleyen bir yapıdır. NavMesh bir oyun içinde hareket eden bir karakterin engelleri aşmasını ve belirli bir noktaya gitmesini sağlar. Bu şekilde karakterlerin doğal bir şekilde hareket ederek oyunun daha da gerçekçi gözükmesi sağlanmış olur. Ama yapay zekanın görmediği bir kör nokta vardır ki, oda aşağıdaki resimde görebileceğiniz gibi kamyonun yan tarafı ve arka tarafıdır, oyunun daha zorlayıcı seviyelerine zombie’ lerden birazda olsa korunabilmemize imkan verir. NavMesh alanları soluk renkli alanlardır, kör alanlar daha renkli alanlardır.

A screenshot of a video game

Description automatically generated

**Şekil 5** NavMesh ve kör alanlar

Graphical user interface, application, website

Description automatically generated

**Şekil 6** Oyunun Ana Menüsü

**Oyun**

Oyuncu ileriye, geriye, sağa, sola veya yanlara bakarak haritayı gezebilir. İleri gitmek için klavyeden w tuşuna, geriye gitmek için s tuşuna, sağa gitmek için d ve sola hareket etmek için a tuşuna tıklaması gerekmektedir. Ayrıca etrafa dönmesi için mouse u sağa veya sola oynatmalıdır. Ateş açmak için ise mouse un sağ tuşuna basması yeterlidir.

**A video game showing a video game

Description automatically generated with low confidence**

**Şekil 7** Oyun Haritasının Keşfi

Oyuncu NevMesh haritası alanında iken Zombie saldırır. Yani Zombie objesinin saldırı animasyonu aktif olur.

**A picture containing text, outdoor

Description automatically generated**

**Şekil 8** Duşman Zombie’ nin saldırı anı

Oyuncu NevMesh alanının dışında iken Zombie’ler oyuncuyu göremezler.



**Şekil 9** Oyuncu kör noktada

Yukarıdaki resimde bir gizmos görülmektedir. Bu gizmos oyuncunun yerde olup olmadığını kontrol ederek zıplama hareketinin doğru çalışması için kullanıldı.

**Kullanılan Fonksiyonlar**

Oyunda kullanılan bazı fonksiyonlar aşağıdaki gibidir.

**StartGame():** Oyunun çalışmaya başlamasından sonra Awake fonksiyonundan hemen sonra çalışmaya başlar.

**ExitGame():** Oyundan çıkış için kullanılır.

**Update():** Oyun çalışmaya başladığında, Star fonksiyonu çalıştıktan sonra sürekli çalışır.

**Shoot():** Oyunda karakterin ateş açabilmesini sağlar.

**EndGame():** Karakter öldüğünde oyun bitti ekranını açar.

**GoToMainMenu():** Ana menüye geçiş yapmamızı sağlar.

**Input:** Kullanıcıdan giriş alma veya oyuncunun klavyeden tuş kontrolü için kullanılır.

**LoadScene:** Sahneleri yüklemek için kullanılır.

**GetActiveScene:** Sahneler arası geçişlerde kullanılır, aktif sahneyi alır. Örneğin oyunun bitmesiyle bitiş sahnesinin gelmesi.

**RotateAround:** Nesneleri döndürmek için kullanılır. Örneğin güneşin dönüşü.

**OnCollisionEnter:** Objelerin birbiriyle çarpışmaya başlaması ile çağrılır. Örneğin zombini oyuncuyu yaralamaya başlaması.

**Instantiate:** Bir objeyi oluşturmak için kullanılır. Örneğin Zombie.

**Destroy:** Bir objeyi yok etmek için kullanılır, örneğin zombi veya oyuncu ölmesi.

**Find:** Bir GameObject’i bulmak için kullanılır. Örneğin zombinin haritada oyuncuyu bulması.

**Distance:** İki değer arasındaki mesafeyi döndürür. Örneğin oyuncu ile zombi arasındaki mesafe.

**Vector3:** X,Y,Z eksenlerinde konumlandırma, yön değiştirme gibi işlemlerde kullanılır.

**KODLAR**

**Oyunu yönetdiğimiz script**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

using UnityEngine.SceneManagement;

public class GameManager : MonoBehaviour

{

public int enemiesAlive = 0;

public int round = 0;

public GameObject[] spawnPoints;

public GameObject enemyPrefab;

public Text roundNumber;

public Text roundsSurvived;

public GameObject endScreen;

// Start is called before the first frame update

void Start()

{

}

// Update is called once per frame

void Update()

{

if (enemiesAlive == 0)

{

round++;

NextWave(round);

roundNumber.text = "Raunt: " + round.ToString();

}

}

public void NextWave(int round)

{

for (var x = 0; x < round; x++)

{

GameObject spawnPoint = spawnPoints[Random.Range(0, spawnPoints.Length)];

GameObject enemySpawned = Instantiate(enemyPrefab, spawnPoint.transform.position, Quaternion.identity);

enemySpawned.GetComponent<EnemyManager>().gameManager = GetComponent<GameManager>();

enemiesAlive++;

}

}

public void GoToMainMenu()

{

SceneManager.LoadScene(0);

}

public void Restart()

{

Time.timeScale = 1;

SceneManager.LoadScene(1);

}

public void EndGame()

{

Time.timeScale = 0;

Cursor.lockState = CursorLockMode.None;

endScreen.SetActive(true);

roundsSurvived.text = round.ToString();

}

}

**Ana Menüdeki Start, Exit tuşlarını işlevsel hale getiren script**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.SceneManagement;

public class MainMenuManager : MonoBehaviour

{

public void StartGame()

{

Time.timeScale = 1;

SceneManager.LoadScene(1);

}

public void ExitGame()

{

//Quit the game

Application.Quit();

}

}

**Mouse ile etrafa bakabildiğimiz script**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class MouseLook : MonoBehaviour

{

public float mouseSens = 1000f;

public Transform playerBody;

float xRotation = 0f;

// Start is called before the first frame update

void Start()

{

Cursor.lockState = CursorLockMode.Locked;

}

// Update is called once per frame

void Update()

{

float mouseX = Input.GetAxis("Mouse X") \* mouseSens \* Time.deltaTime;

float mouseY = Input.GetAxis("Mouse Y") \* mouseSens \* Time.deltaTime;

xRotation -= mouseY;

xRotation = Mathf.Clamp(xRotation, -90f, 90f);

transform.localRotation = Quaternion.Euler(xRotation, 0f, 0f);

playerBody.Rotate(Vector3.up \* mouseX);

}

}

**Karakterimizi yönetdiğimiz script**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

using UnityEngine.SceneManagement;

public class PlayerManager : MonoBehaviour

{

public float health = 100;

public Text healthText;

public GameManager gameManager;

public void Hit(float damage)

{

health -= damage;

healthText.text = health.ToString() + " Sağlık";

if (health <= 0)

{

gameManager.EndGame();

}

}

// Start is called before the first frame update

void Start()

{

}

// Update is called once per frame

void Update()

{

}

}

**Karakterimizi sağa, sola, ileriye, geriye haraket etdirdiğimiz script**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class PlayerMovement : MonoBehaviour

{

public CharacterController controller;

public float speed = 12f;

Vector3 velocity;

public float gravity = -9.81f;

public bool isGrounded;

public Transform groundCheck;

public float groundDistance = 0.4f;

public LayerMask groundMask;

public float jumpHeight = 2f;

// Start is called before the first frame update

void Start()

{

}

// Update is called once per frame

void Update()

{

isGrounded = Physics.CheckSphere(groundCheck.position, groundDistance, groundMask);

if(isGrounded && velocity.y < 0)

{

velocity.y = -2f;

}

float x = Input.GetAxis("Horizontal");

float z = Input.GetAxis("Vertical");

Vector3 move = transform.right \* x + transform.forward \* z;

controller.Move(move \* speed \* Time.deltaTime);

velocity.y += gravity \* Time.deltaTime;

controller.Move(velocity \* Time.deltaTime);

if (Input.GetButtonDown("Jump") && isGrounded)

{

velocity.y = Mathf.Sqrt(jumpHeight \* -2 \* gravity);

}

}

}

**Karakterimizi ateş açmasına olanak sağlayan script**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class WeaponManager : MonoBehaviour

{

public GameObject playerCam;

public float range = 100f;

public float damage = 25f;

public Animator playerAnimator;

// Start is called before the first frame update

void Start()

{

}

// Update is called once per frame

void Update()

{

if (playerAnimator.GetBool("isShooting"))

{

playerAnimator.SetBool("isShooting", false);

}

if (Input.GetButtonDown("Fire1"))

{

//Debug.Log("Test Ateş");

Shoot();

}

}

void Shoot()

{

playerAnimator.SetBool("isShooting", true);

RaycastHit hit;

if(Physics.Raycast(playerCam.transform.position, transform.forward, out hit, range))

{

EnemyManager enemyManager = hit.transform.GetComponent<EnemyManager>();

if (enemyManager !=null)

{

enemyManager.Hit(damage);

}

}

}

}

**Duşman Zombie’ mizi hareket etdirdiğimiz, yok etdiğimiz script**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.AI;

public class EnemyManager : MonoBehaviour

{

public GameObject player;

public Animator enemyAnimator;

public float damage = 20f;

public float health = 100f;

public GameManager gameManager;

public void Hit(float damage)

{

health -= damage;

if(health <= 0)

{

gameManager.enemiesAlive--;

//Destroy Duşman (Zombie)

Destroy(gameObject);

}

}

// Start is called before the first frame update

void Start()

{

player = GameObject.FindGameObjectWithTag("Player");

}

// Update is called once per frame

void Update()

{

GetComponent<NavMeshAgent>().destination = player.transform.position;

if (GetComponent<NavMeshAgent>().velocity.magnitude > 1)

{

enemyAnimator.SetBool("isRunning", true);

}

else

{

enemyAnimator.SetBool("isRunning", false);

}

}

private void OnCollisionEnter(Collision collision)

{

if (collision.gameObject == player)

{

player.GetComponent<PlayerManager>().Hit(damage);

}

}

}

**KAYNAKLAR**

* [Learn game development w/ Unity | Courses & tutorials in game design, VR, AR, & Real-time 3D | Unity Learn](https://learn.unity.com/)
* [Free Natural Environments - Asset Store (unity.com)](https://assetstore.unity.com/lists/free-natural-environments-11561)
* [Unity - Manual: Unity User Manual 2021.3 (LTS) (unity3d.com)](https://docs.unity3d.com/Manual/index.html)
* [[Unity FPS Tutorial 1] First Person Shooter in UNITY - YouTube](https://www.youtube.com/watch?v=hbM5Iy_0sYU&ab_channel=FuelledByCaffeine)
* [(8) Unity Basic Weapon System Tutorial - YouTube](https://www.youtube.com/watch?v=kXbQMhwj5Uc&ab_channel=Plai)

**PROJENİN GITHUB LİNKİ**

[Dovletyar/The-Incredible-Zombie-Game-in-Unity (github.com)](https://github.com/Dovletyar/The-Incredible-Zombie-Game-in-Unity)