



Beykent Üniversitesi
Yazılım Mühendisliği Bölümü
Yazılım Mühendisliği Tasarım Projesi

2023 – GÜZ
WitDark Dökümantasyonu

Whispers in the Dark (WitDark)

Grup Üyeleri
2003013062 - Beyzanur Yakut
2003013024 - Gizem Güven
2003013014 - Mustafa Örnek
2103013800 - Ömer Faruk Dinçaslan
2003013036 - Zeynep Ciplak

İÇİNDEKİLER

İÇİNDEKİLER.....	1
GÖREV DAĞILIM TABLOSU.....	3
I Projenin Amacı ve Hedefleri.....	4
1.1 Projenin Amacı.....	4
1.2 Projenin Hedefleri.....	4
2 Projenin Kapsamı.....	4
2.1 Mevcut Durum:.....	4
2.2 Projenin İçeriği.....	5
2.2.1 Oyun Arayüz Katmanı (Game Interface Katmanı):.....	6
2.2.2 Alana Özgü Katman (Domain Specific Katmanı):.....	6
2.2.3 Altyapı Katmanı (Infrastructure Katmanı):.....	6
2.2.4 Platform yazılımı katmanı :.....	6
2.2.5 Grafikler (Graphics):.....	6
2.2.6 Grafik Kullanıcı Arayüzü (GUI):.....	7
2.2.7 Yapay zeka (Artificial Intelligence):.....	7
2.2.8 Fizikler (Physics):.....	7
2.2.9 Sesler (Sound):.....	7
2.2.10 Rakip Ürün Karşılaştırması:.....	7
3 Proje Ağ Diyagramı.....	7
3.1 Projeye Ait En Erken Başlangıç ve En Erken Bitiş Zamanını Gösteren Ağ Diyagramı.....	8
4 Fonksiyonel Olmayan Gereksinimler.....	9
4.1 Uyumluluk (Compliance) Gereksinimleri:.....	10
4.2 Güvenlik (Safety) Gereksinimleri:.....	10
4.3 Emniyet (Security) Gereksinimleri:.....	10
4.5 Güvenilirlik (Reliability) Gereksinimleri:.....	10
4.6 Kullanılabilirlik (Usability) Gereksinimleri:.....	11
4.7 Sürdürülebilirlik (Sustainability) Gereksinimleri:.....	11
4.8 Gizlilik (Privacy) Gereksinimleri:.....	11
5 Tanımlamalar.....	11
II Gereksinimlerin Analizi.....	12
6 Uses Cases ve Tablo Açıklamaları.....	12
6.1 Menü Use Case.....	12
6.2 Oyuncu Karakteri Use Case.....	14
6.3 Oyuncunun NPC'ler ile Olan Etkileşimi Use Case.....	14
6.4 Düşman Davranışı ve Temel Mekanikleri Use Case.....	16
6.5 Oyuncunun Nesneler ile Etkileşimi Use Case.....	17
7 Fonksiyonel Gereksinimler.....	17
III Tasarım Dokümantasyonu.....	18
8 Tasarım Hedeflerinin Tanımlanması.....	18
8.1 Reliability (Güvenilirlik).....	19
8.2 Modifiability (Değiştirilebilirlik).....	19
8.3 Maintainability (Sürdürülebilirlik).....	19
8.4 Adaptability (Uyarlanabilirlik).....	19

8.5 Efficiency (Verimlilik).....	20
8.6 Portability (Taşınabilirlik).....	20
8.7 Traceability of Requirements (Gereksinimlerin İzlenebilirliği).....	20
8.8 Fault Tolerance (Hata Toleransı).....	20
8.9 Robustness (Sağlamlık).....	20
8.10 Well-defined Interfaces (İyi Tanımlanmış Arayüzleri).....	21
9 Önerilen Yazılım Mimarisi.....	21
10 Sınıf Diyagramları.....	23
11 Dinamik Model.....	24
11.1 Menü: Yeni Oyun ve Devam Et Sequence Diyagramı.....	24
11.2 Menü: Ayarlar Sequence Diyagramı.....	25
11.3 Menü: Oyundan Çık Sequence Diyagramı.....	27
Şekil 11.3: Menü kullanımına ait Sequence Diyagramı:Oyundan Çık Seçeneği.....	27
11.4 Oyuncu Mekanikleri: Savunma Sequence Diyagramı.....	28
11.5 Oyuncu Mekanikleri: Saldırma Sequence Diyagramı.....	29
11.6 Oyuncu Mekanikleri: Hareket Etme Sequence Diyagramı.....	30
11.7 Oyuncu Mekanikleri: Yaşama Sequence Diyagramı.....	31
11.8 Oyuncu - NPC Etkileşimi Sequence Diyagramı.....	32
11.9 Düşman Karakterinin Mekanikleri Sequence Diyagramı:.....	33
11.10 Oyuncu - Nesneler Etkileşimi: Kapı ve Tabela Sequence Diyagramı:.....	34
11.11 Oyuncu - Nesneler Etkileşimi:Envanter Sequence Diyagramı:.....	35
12 Kullanıcı Arayüzü.....	35
12.1 Oyun Menüsü Arayüzü.....	37
12.2 Envanter Menüsü Arayüzü.....	37
12.3 Oyun Arayüzü(HUD).....	38
IV Test Planları.....	40
13 Test Edilebilecek ve Test Edilemeyecek Özellikler.....	40
14 Test Cases.....	47
Ayrıntılı Test Durumları ve Sonuçları.....	47
V Sözlük.....	52
Referanslar.....	53

GÖREV DAĞILIM TABLOSU

	Numara	Ad Soyad	Konu Başlıklarları
1	2103013800	Ömer Faruk Dinçaslan	<ul style="list-style-type: none"> -3 Proje Ağ Diyagramı -6 Uses Cases ve Tablo Açıklamaları -10 Sınıf Diyagramları -12 Kullanıcı Arayüzü
2	2003013024	Gizem Güven	<ul style="list-style-type: none"> -2 Projenin Kapsamı -4 Fonksiyonel Olmayan Gereksinimler -8 Tasarım Dokümantasyonu -9 Önerilen Yazılım Mimarisi -13 Test EdileBILECEK ve Test Edilemeyecek Özellikler
3	2003013014	Mustafa Örnek	<ul style="list-style-type: none"> -1 Projenin Amacı ve Hedefleri -2 Projenin Kapsamı -4 Fonksiyonel Olmayan Gereksinimler -5 Tanımlamalar -8 Tasarım Hedeflerinin Tanımlanması -14 Test Cases
4	2003013062	Beyzanur Yakut	<ul style="list-style-type: none"> -6 Uses Cases ve Tablo Açıklamaları -10 Sınıf Diyagramları -11 Dinamik Model -14 Test Cases
5	2003013036	Zeynep Ciplak	<ul style="list-style-type: none"> -2 Projenin Kapsamı -Tablo 6 : Fonksiyonel Gereksinimlerin Tanımı -11 Dinamik Model -13 Test EdileBILECEK ve Test Edilemeyecek Özellikler -14 Test Cases

I Projenin Amacı ve Hedefleri

1.1 Projenin Amacı

WitDark, görme engelli bireylerin oyun sektöründeki etkileşimini güçlendirmek ve aynı zamanda sıradışı hikayesi ve kurgusuyla görme engeli olmayan bireyleri de kitlesine dahil ederken, bu engel ile ilgili insan faktörünün etkisini vurgulayarak empati geliştirmeyi hedefler. Ses ve haptik öğelerle zenginleştirilmiş deneyimi sayesinde WitDark, merkezine aldığı görme engelli kitleye oyun dünyasında etkileyici bir keşif ve deneyim sunmayı amaçlar.

1.2 Projenin Hedefleri

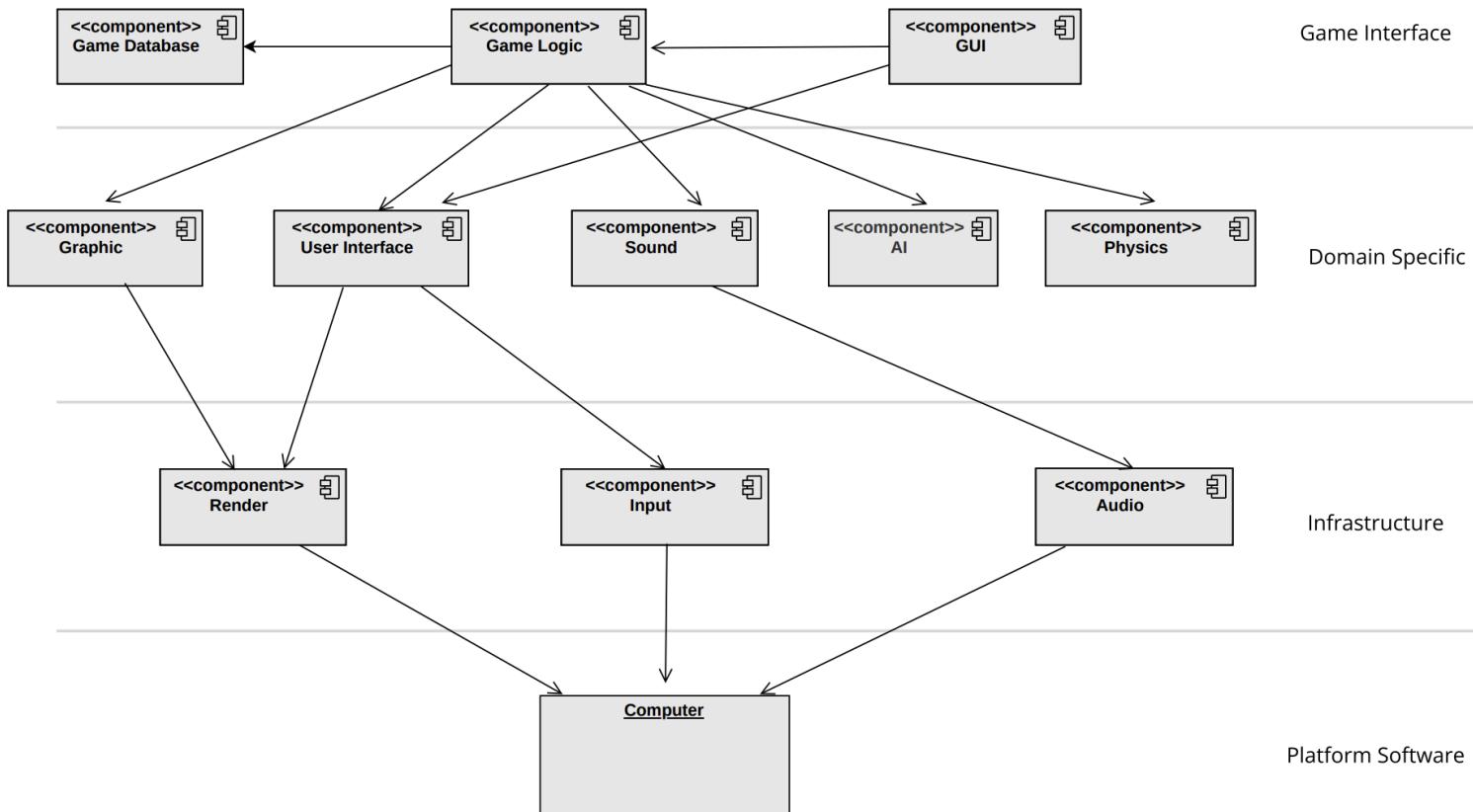
- Kullanıcı dostu arayüz tasarımlı ve kolay oynanabilirlik sağlanarak görme engelli bireylerin oyunu rahatlıkla kontrol etmelerini sağlamak
- Ses, haptik ve diğer dokunsal öğeleri içeren oyun mekanığının detaylı şekilde tasarlanması.
- Detaylı ses tasarımıyla oyunun, oyunculara etkileyici bir işitsel deneyim sunması
- Duyusal öğelerin kullanımıyla oyunun, dokunsal geri bildirimlerle zenginleştirilmesi
- Farklı zevklere hitap eden, farklı oyun türlerinin oyuna yerleştirilerek görme engelli oyunculara yönelik adaptasyonları.
- Görme engelli oyuncuların oyun içinde veya dışında topluluklara katılımlarını sağlamak.

2 Projenin Kapsamı

2.1 Mevcut Durum:

Günümüzde görme engelliler için yapılan oyunlar sınırlı sayıda yer almaktadır. Buna ek olarak oyun içeriğinin kısıtlı yapısı, sürekli aynı kodların tekrar edilerek farklı oyunlar arasındaki benzerlik yapıları gibi faktörler sebebiyle oyun konusunda yetersiz bir kütüphaneye sahiptirler. Bu projede her türden görme engeli olan bireylere hitap edecek şekilde bir oyun geliştirerek çeşitliliği artırmak, benzersiz içerikler kullanarak oynamaktan sıkıldıkları oyunların önüne geçmek ve farklı bir soluk getirmek amacıyla kullanıcılar yeni bir oyun sunuyoruz.

2.2 Projenin İçeriği



Sekil 1: Whispers in the Dark sistemine ait “component” diyagramı

2.2.1 Oyun Arayüz Katmanı (Game Interface Katmanı):

Mimarının en üst katmanı olup nesne ve bileşenlerden oluşur. Bu katman aynı zamanda game engine'i de kapsar. Oyun kodlanırken hangi oyun motoru kullanılacak gibi sorulara cevap verir. Oyuna özel nesneler bulunur. Oyun arayüz katmanı, oyun içeriği, oyun mantığı, nesneler database de depolanır. Database katmanındaki nesneler de bu katmanda yer alır.

2.2.2 Alana Özgü Katman (Domain Specific Katmanı):

Oyun geliştirme sürecinde, bir oyunun içeriği, mekaniği ve özel gereksinimleriyle ilgili özel kodları içeren katmanı ifade eder. Bu katman, belirli bir oyun konsepti, tema veya türüne özgü olan, oyunun temel mekanlığını ve özelliklerini belirleme ve özelleştirme yeteneği sağlar.

2.2.3 Altyapı Katmanı (Infrastructure Katmanı):

Bu katmandaki bileşenler potansiyel olarak bütün domain lerde tekrardan kullanılabilir, genel amaçlı hizmetlerin sağlanması gibi işlevleri vardır.

2.2.4 Platform yazılımı katmanı :

Eklediğiniz Başlıklar (Biçim > Paragraf stilleri), içindekiler tablosunda görünür.Bu katman oyunu desteklemek için getirilen standart parçalardan oluşur.

2.2.5 Grafikler (Graphics):

Oyunun görselleştirilmesi ile alakalı bir alt sistem aktöridür.

2.2.6 Grafik Kullanıcı Arayüzü (GUI):

Grafiksel kullanıcı arayüzü, elektronik cihazların simgeler, ikonlar ve diğer görsel grafikler yardımıyla kullanmasına yardımcı olması amacıyla geliştirilmiş tasarımlardır. GUI oyun arayüzü oluşturmaya yarayan işlevselligi sağlar.

2.2.7 Yapay zeka (Artificial Intelligence):

Non-player-character'lerdeki karar alma yeteneklerini ve stratejik davranışlarını simüle ederek oyunculara etkileşimli bir deneyim sunar. Örneğin, "saldırı stratejisi" işlevselligi. Bu sayede, NPC'ler oyun içinde akıllıca hareket ederek, oyunculara dinamik ve heyecanlı bir oyun atmosferi sunarlar.

2.2.8 Fizikler (Physics):

Oyun dünyasında nesnelerin hareketini, çarpışmalarını, çekim gücü gibi fiziksel etkileşimleri simüle eden yazılım bileşenidir.

2.2.9 Sesler (Sound):

Sesli betimlemeleri, oyun seslerini oluşturan ve düzenleyen kütüphane/aktör.

2.2.10 Rakip Ürün Karşılaştırması:

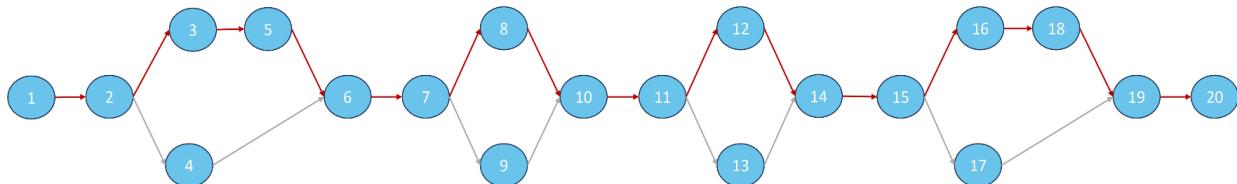
Görme engelli bireylere yönelik geliştirilen diğer oyunlar, tamamen kör bireylere yönelik olarak siyah ekranдан ibaretken WitDark, bütün görme engelli bireyleri hedefleyerek hikayeye yedirilmiş bir şekilde görsel tasarım geliştirmeye odaklanıyor. Kendine özgün ses tasarımıyla da diğer oyunlardan farklı olarak konfigüre edilmesi planlanıyor. Rakip ürünler çevrimiçi olarak sunucularla kontrol edilip çok oyunculu ve senaryo kısmında eksik bir sistem geliştirmişken WitDark; çevrimdışı, tek oyunculu ve senaryoya odaklı bir oyun sunmayı planlıyor.

3 Proje Ağ Diyagramı

	Aktivite Listesi	Öncelik İlişkisi	Süreç (gün)	En Erken		En Geç		Bolluk
				Başlangıç	Bitiş	Başlangıç	Bitiş	
1	Projeye Başlangıç	—	0	0	0	0	0	0
2	Veri Toplama	1	5	0	5	0	5	0
3	Hedef Kullanıcıları Belirleme	2	2	5	7	5	7	0
4	Kullanıcılarla Görüşme	2	4	5	9	6	10	1
5	Kullanıcı İhtiyaçlarını Tanımlama	3	3	7	10	7	10	0
6	Olurluk Çalışması	4, 5	6	10	16	10	16	0
7	Gidişat Raporunun Hazırlanması - 1	6	7	16	23	16	23	0
8	Yazılım Gereksinim Analizi	7	13	23	36	23	36	0
9	Sistem Gereksinim Analizi	7	8	23	31	28	36	5
10	Yazılım Mimarisinin Tasarlanması	8, 9	14	36	50	36	50	0
11	Gidişat Raporunun Hazırlanması - 2	10	7	50	57	50	57	0
12	Back-End Çalışması	11	21	57	78	57	78	0
13	Front-End Çalışması	11	14	57	71	64	78	7
14	Gidişat Raporunun Hazırlanması - 3	12, 13	7	78	85	78	85	0
15	Mevcut Sistemi İnceleme	14	5	85	90	85	90	0
16	Kullanıcı Deneyimine Sunma	15	7	90	97	90	97	0
17	Geri Dönütlerin Değerlendirilmesi	15	5	90	95	103	108	13
18	Projenin Revizesi	16	11	97	108	97	108	0
19	Gidişat Raporunun Hazırlanması - 4	17, 18	14	108	122	108	122	0
20	Projenin Bitişi	19	0	122	122	122	122	0

Kritik Yol: 1, 2, 3, 5, 6, 7, 8, 10, 11, 13, 14, 15, 16, 18, 19, 20

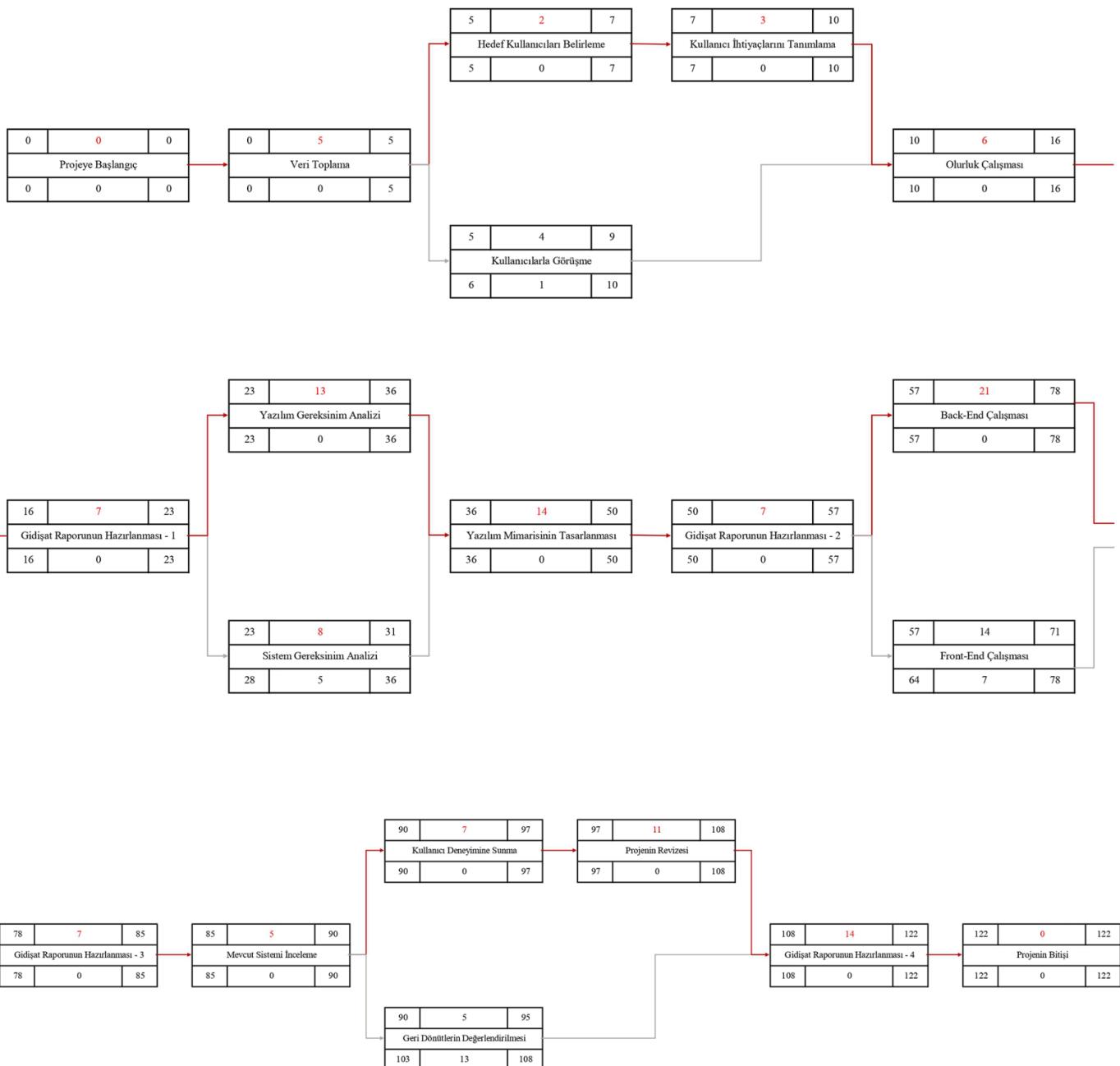
Şekil 2: WitDark Projesine ait Faaliyet Tablosu



Şekil 3: WitDark Projesine ait “Kritik Yol” gösterimi

3.1 Projeye Ait En Erken Başlangıç ve En Erken Bitiş Zamanını Gösteren Ağ Diyagramı

Kritik Yol kırmızı ile gösterilmiştir.



Şekil 4: WitDark Projesine ait Ağ Diyagramı

4 Fonksiyonel Olmayan Gereksinimler

4.1 Uyumluluk (Compliance) Gereksinimleri:

Yazılım sistemi veya uygulamanın ilgili yasal, düzenleyici ve endüstri standartlarına uygun olduğunu gösterir.

Oyun için kullanılacak ses faktörünün kullanıcı için en uygun şekilde kesinti olmadan, anlaşılır bir dıksiyon kullanılarak yapılmasına öncelik vermek amaçlanmıştır. Bu amaçla görme engelli kullanıcıların sesli betimleme uygulamalarındaki kullanım gereklilikleri dikkate alınarak ilgili yasal ve düzenleyici zorunlulukları yerine getirmek ve görme engelli kullanıcılar için uygunluk esas alınmıştır.

4.2 Güvenlik (Safety) Gereksinimleri:

Nitelik gereksinimleridir. Olumsuz durumlarla sonuçlanabilecek faktörlerin bildirimidir.

Oyunumuz için hitap ettiğimiz kullanıcılar adına ön planda tuttuğumuz ses faktörünün en düşük hata verme oranında olması için güvenilir kaynaklar kullanmayı ve bunları kendi ses sistemlerimizle oyuna dahil etmeye karar verdik. Karmaşık bir hikaye yapısına girmeden ses faktörünü oyunun akışına uygun şekilde katmak hem kullanıcıya akıcı bir deneyim sunarken hem de ilerlemeyi daha uygun hale getirecektir.

4.3 Emniyet (Security) Gereksinimleri:

Sisteme ait nesnelerin istenmeyen çevre koşullarına göre korunması ile ilgili kuralları belirler.

Kullanıcıdan kaynaklı oyun sırasında herhangi bir kesinti yaşanması veya oyundan beklenmedik şekilde atılmasına karşı oyunumuzun kullanıcı hikayede ilerledikçe otomatik olarak sistem tarafından kaydedilmesini tercih ettim. Bu sayede kullanıcı kaldığı yerden bir kayıp yaşamadan devam edebilecektir. Ve kullanıcıya daha iyi bir oyun deneyimi sunulmuş olacaktır.

4.5 Güvenilirlik (Reliability) Gereksinimleri:

Yazılımın işlevsellığının uzun süre devam etmesi beklenir. Yazılımın gerçekleştireceklerinde bazı istisnaların olması olasılığına rağmen doğru ve güçlü olarak sağlanmalıdır.

Oyunumuz için kullanıcı gereksinimlerini karşılaması ve yazılımın %99 hatasız çalışması hedeflenmektedir.

4.6 Kullanılabilirlik (Usability) Gereksinimleri:

Yazılım sistemi veya uygulamanın kullanımının kolay ve anlaşılır olmasına yönelik özelliklerdir. Bu özellikler, sistemin kullanımının basitliğini, anlaşılırlığını, erişilebilirliğini ve öğrenilebilirliğini içerebilir.

Kullanıcının oyunu daha rahat bir şekilde oynayabilmesi, kontrol bileşenlerini öğrenebilmesi adına oyunda hikayeyi ve mekanikleri destekleyecek şekilde eğitici bir mod eklenmesi tercih edilmiştir. Eklediğimiz eğitici modu ses tasarımı ile zenginleştirerek daha anlaşılabilir bir hale getirmek amaçlanmıştır. Bu sayede oyun hakkında bir bilgisi olmayan kullanıcıya daha rahat ve öğretici bir deneyim sunma hedeflenmiştir.

4.7 Sürdürülebilirlik (Sustainability) Gereksinimleri:

Yazılım, enerji verimliliği ve kaynak kullanımı açısından sürdürülebilir bir tasarıma sahip olmalıdır. Bu, çevresel etkileri azaltmayı ve uzun ömürlü bir çözüm sunmayı içermelidir.

Herhangi bir oyun güncellemesi için kullanıcıya önceden haber vermek ve güncelleme sırasında çevresel etkilerden zarar görmemek adına kullanıcı girişinin geçici süreliğine kapatılması amaçlanmıştır.

4.8 Gizlilik (Privacy) Gereksinimleri:

Bu gereksinimler, kullanıcıların mahremiyetini ve kişisel bilgilerini korumayı amaçlar. Gizlilik gereksinimleri genellikle veri güvenliği standartlarına ve ilgili yasal düzenlemelere uyum içinde olmalıdır.

Kullanıcının seçtiği karakter üzerinden ilerlemesinin kaydedildiği dosyaya bir başka kullanıcı tarafından erişilmesinin engellenmesi amaçlanmaktadır. Bu amaçla kullanıcı kendi hikayesinin değiştirilebilirliğini önlemış olacaktır.

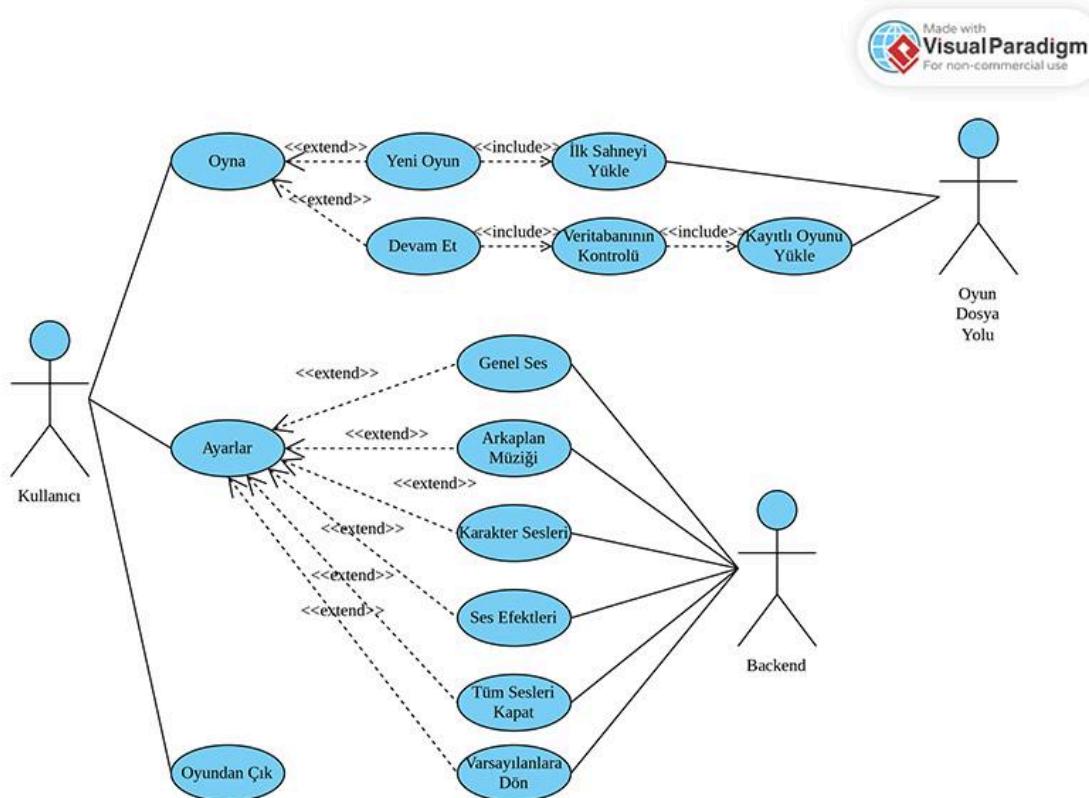
5 Tanımlamalar

1. *HP (Health Points)*: Sağlık Puanları (Karakterin sağlık değerini simgeler.)
2. *STAMINA*: Enerji (Karakterin sahip olduğu fiziksel enerjiyi ifade eder.)
3. *MANA*: Yetenek Gücü
4. *SPEED*: Hız
5. *DEF (Defense)*: Korunma
6. *DODGE (Deflect)*: Kaçınma/Savurma
7. *GUARD*: Engelleme
8. *ATK (Attack)*: Saldırı
9. *INV (Inventory)*: Envanter
10. *MAP*: Harita
11. *QUEST*: Görev
12. *MUSIC*: Müzik
13. *EXIT*: Çıkış
14. *KEY*: Anahtar
15. *WEAPON*: Silah/Teçhizat
16. *WAYPOINT*: Yer işaretleri
17. *TOP-DOWN*: Yukarıdan bakış
18. *SIDE-VIEW*: Yandan bakış
19. *NPC (Non-Player Character)*: Oyuncunun kontrol etmediği karakter

II Gereksinimlerin Analizi

6 Uses Cases ve Tablo Açıklamaları

6.1 Menü Use Case

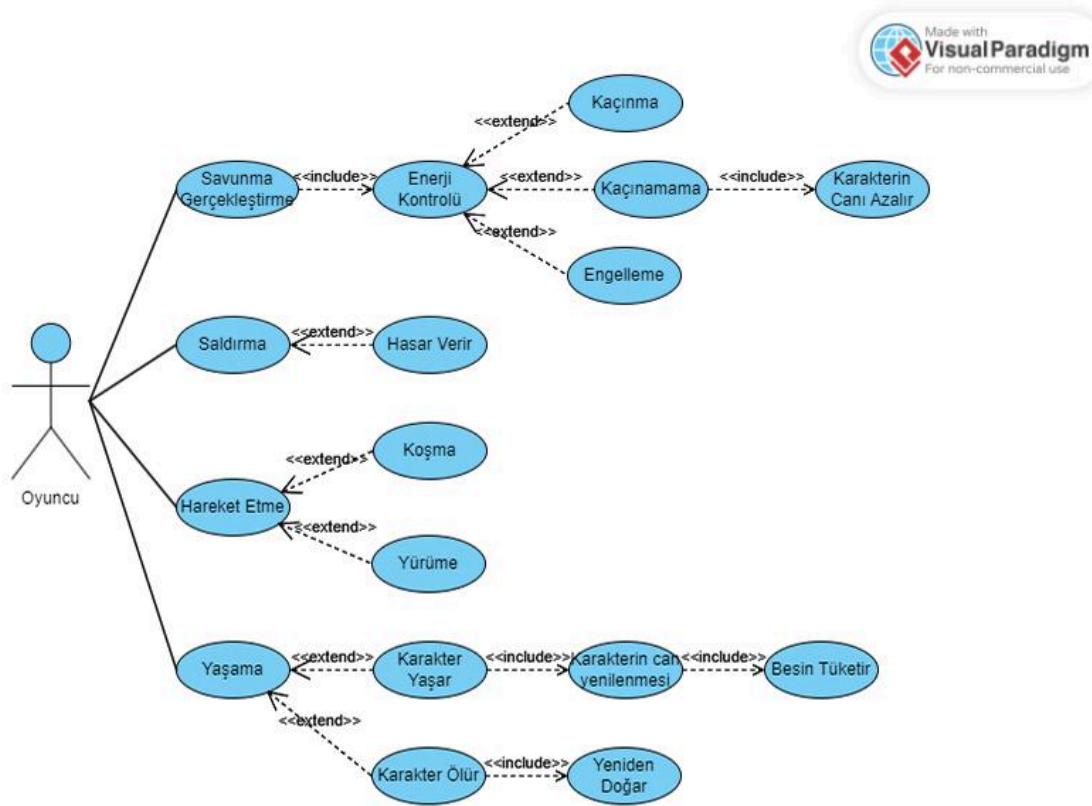


Şekil 5: Menü kullanımına ait Use Case Diyagramı

Tablo 1*: Menü kullanımına ait Use Case Diyagramı Açıklaması

Use Case Numarası	1
Use Case Adı	Menü
Aktörler	Kullanıcı, Veritabanı, Backend
Ön Koşullar	Oyuncunun oyunu başlattığı varsayıılır
Son Durum	Oyuncu "Menü" ekranını kullanarak istediği işlemi başarıyla gerçekleştirmiştir
Başarılı Senaryo	1- Kullanıcı "Oyna" butonuna tıklar ve devamında "Yeni Oyun" veya "Devam Et" seçeneklerinden birini seçer 2- Kullanıcı "Yeni Oyun" butonuna tıkladığında oyunun ilk sahnesi yüklenir 3- Kullanıcı "Devam Et" butonuna tıkladığında oyuna kaldığı yerden devam eder
Alternatif Senaryo	1- Kullanıcı "Ayarlar" butonuna tıklar 2- Kullanıcı değiştirmek istediği ayarları değiştirir 3- Kullanıcı menüye geri döner
Alternatif Senaryo	1- Kullanıcı "Oyundan Çık" butonuna tıklar 2- Oyun kapanır

6.2 Oyuncu Karakteri Use Case

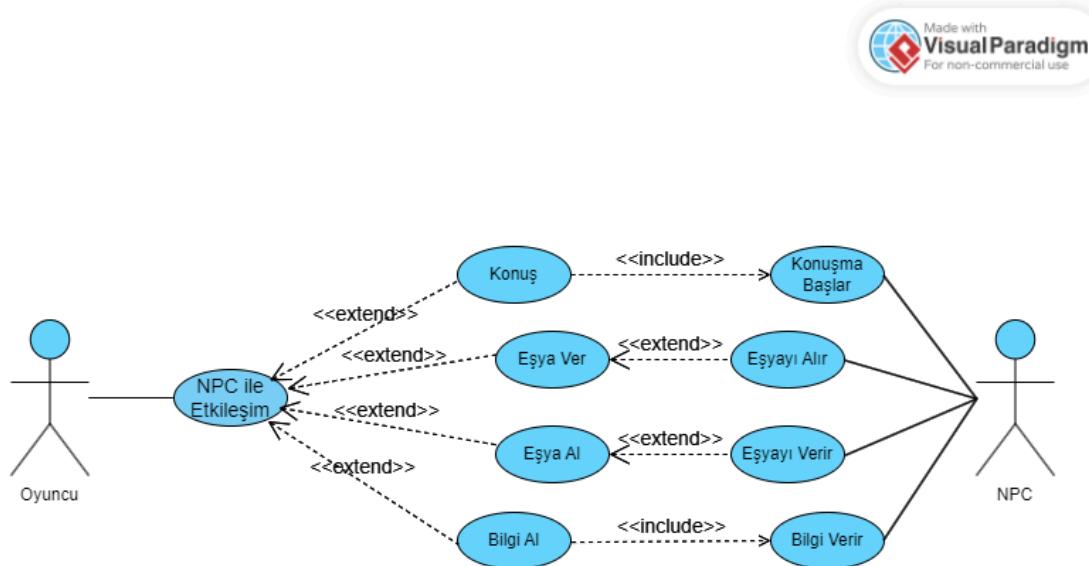


Şekil 6: Oyuncu Karakterinin Davranış ve Temel Mekaniklerine ait Use Case Diyagramı

Tablo 2*: Oyuncu Karakterinin Davranış ve Temel Mekaniklerine ait Use Case Diyagramı Açıklaması

Use Case Numarası	2
Use Case Adı	Oyuncu Karakteri
Aktörler	Oyuncu
Ön Koşullar	Oyuncu oyun içerisinde olduğu varsayıılır
Son Durum	Oyuncu karakterin ve oyunun temel dinamiklerini ve oynanış tarzını tanır
Başarılı Senaryo	1- Oyuncu "Hareket Etme" komutıyla karakteri istediği yönde yönlendirir, hızını belirler 2- Oyuncu "Saldırma" komutu ile bu aksiyonu gerçekleştirir 3- Oyuncu "Savunma" komutu ile ister saldırılardan kaçınır isterse saldırıcıları engeller, bunları da sahip olduğu enerji sayesinde sergiler
Alternatif Senaryo	1- Karakter canı azaldığı takdirde çeşitli besinler tüketerek kaybettiği sağlığını geri kazanabilir 2- Karakterin canı 0 olduğunda "Ölür" ve "Yeniden Doğar"

6.3 Oyuncunun NPC'ler ile Olan Etkileşimi Use Case

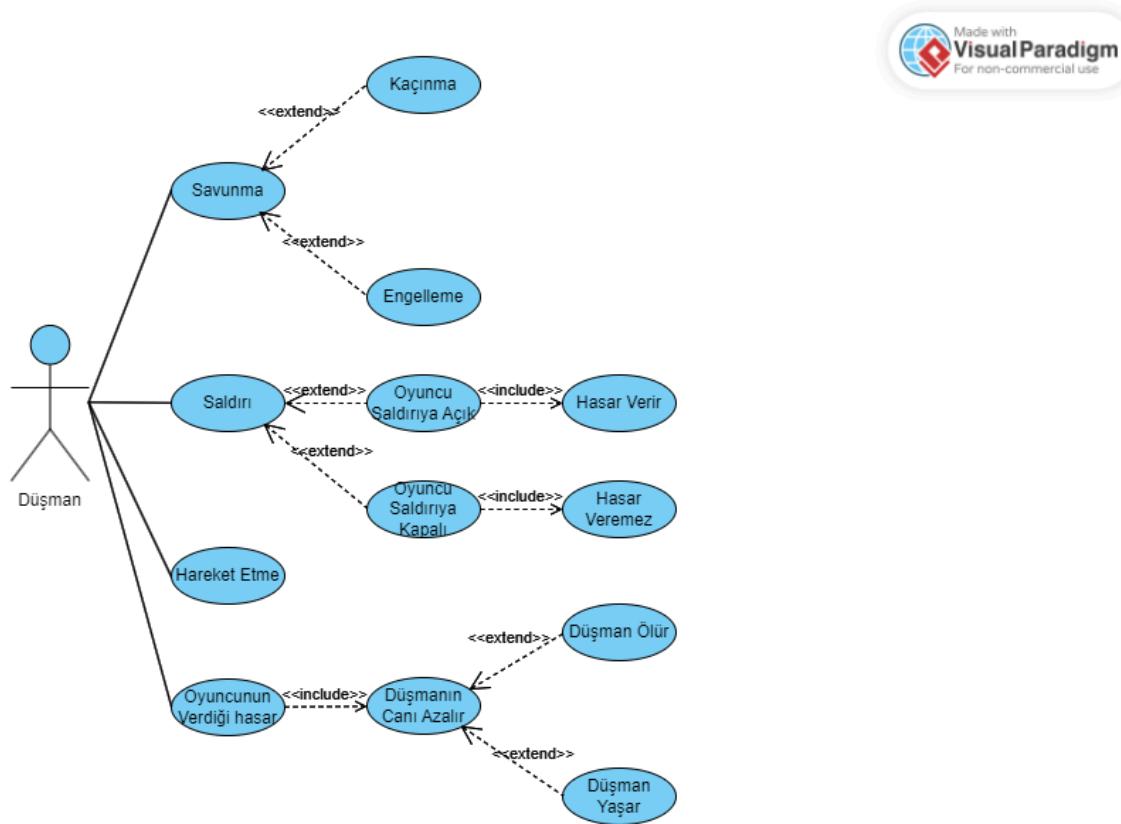


Şekil 7: Oyuncu Karakterinin Non-Player-Character ile Etkileşimine Ait Use Case Diyagramı

Tablo 3*: Oyuncu Karakterinin Non-Player-Character ile Etkileşimine Ait Use Case Diyagramı Açıklaması

Use Case Numarası	3
Use Case Adı	Oyuncunun NPC'ler ile Olan Etkileşimi
Aktörler	Oyuncu, NPC
Ön Koşullar	"Oyuncu", "NPC" ile etkileşimde bulunmalıdır
Son Durum	"Oyuncu", "NPC" ile başarılı bir etkileşimde bulunmuştur
Başarılı Senaryo	1- "Oyuncu", "NPC ile Etkileşim" komutuyla "Konuş", "Eşya Ver", "Eşya Al", "Bilgi Al" komutlarını başarıyla gerçekleştirir
Alternatif Senaryo	1- "Oyuncu", "Eşya Ver" komutunu kullandığında gerekli eşyaya sahip ise "Oyuncu" tarafından eşya verilir ve "NPC" tarafından eşya alınır
Alternatif Senaryo	1- "Oyuncu", "Eşya Al" komutunu kullandığında "NPC" gerekli eşyaya sahip ise "Oyuncu" tarafından eşya alınır ve "NPC" tarafından eşya verilir

6.4 Düşman Davranışı ve Temel Mekanikleri Use Case

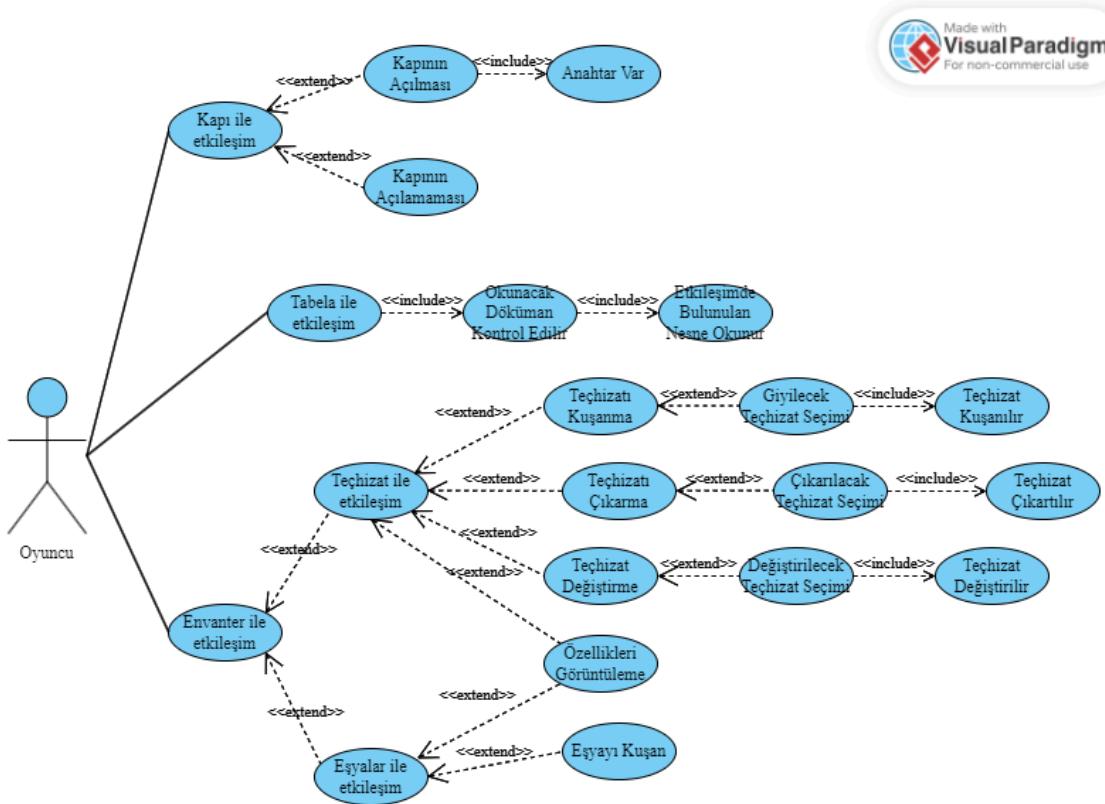


Şekil 8: Düşman Karakterinin Davranışı ve Temel Mekaniklerine Ait Use Case Diyagramı

Tablo 4*: Düşman Karakterinin Davranışı ve Temel Mekaniklerine Ait Use Case Diyagramı Açıklaması

Use Case Numarası	4
Use Case Adı	Düşman Davranışı ve Temel Mekanikleri
Aktörler	Düşman
Ön Koşullar	"Oyuncu", "Düşman" ile etkileşimde bulunacağı oyun sahnesinde olmalıdır
Son Durum	"Düşman" karakteri "Oyuncu" ile etkileşime girer
Başarılı Senaryo	1- "Düşman", "Hareket Etme" komutuyla ilerler 2- "Düşman", "Saldırma" komutu ile "Oyuncu"nun saldırıyı engellemeye veya kaçınma durumuna göre hasar verir 3- "Düşman", "Savunma" komutu ile saldırılardan kaçınır veya saldırıcıları engeller 4- "Oyuncu"nun verdiği hasar düşmanın can durumu ve saldırının gücüne göre "Düşman"ı öldürür veya canını azaltır
Alternatif Senaryo	-

6.5 Oyuncunun Nesneler ile Etkileşimi Use Case



Şekil 9: Oyuncunun Nesneler ile Etkileşimi'ne ait Use Case Diyagramı

Tablo 5*: Oyuncunun Nesneler ile Etkileşimi'ne ait Use Case Diyagramı Açıklaması

Use Case Numarası	5
Use Case Adı	Oyuncunun Nesneler ile Etkileşimi
Aktörler	Oyuncu
Ön Koşullar	Oyuncunun oyun esnasında herhangi bir nesne ile etkileşime girmek istediği varsayılar
Son Durum	Oyuncu ilgili nesneler ile etkileşime girer
Başarılı Senaryo	1- "Oyuncu", anahtara sahip olması durumunda etkileşime girdiği kapıyı açabilir 2- "Oyuncu", tabela ile etkileşime girdiğinde tabela üzerindeki yazı sesli olarak ifade edilir 3- "Oyuncu", envanter ile ilgili "Teçhizat Kuşanma", "Teçhizat Çıkarma", "Teçhizat Değiştirme", "Eşyayı Kuşan", "Özellikleri Görüntüleme" özelliklerini envanterindeki teçhizatlar ve eşyalar için kullanabilir
Alternatif Senaryo	-

Tablo 6 : Fonksiyonel Gereksinimlerin Tanımı

Section/Requirement ID	Gereksinim Tanımı (Requirement Definition)
FR1.0	Kullanıcı oyun yazılımını başlatır.
FR1.1.0	Kullanıcı oyna butonuna basabilir.
FR1.1.1	Kullanıcı yeni oyun başlatabilir ve ilk sahne yüklenir.
FR1.1.2	Kullanıcı devam et butonuna basabilir ve kayıtlı oyun yüklenir.
FR1.2.0	Kullanıcı ayarlar butonu ile oyun ayarlarına gidebilir ve genel ses , arkaplan müziği, karakter sesleri , ses efekleri , tüm sesleri kapat , varsayılanlara geri dön ayarlarını değiştirebilir.
FR1.3.0	Kullanıcı oyundan çıkış butonuna basarak oyundan çıkabilir.
FR2.0.0	Kullanıcı verilen durumlardan birini seçerek karakterini bir aksiyona sokar
FR2.1.0	Oyuncu karakteri kullanıcının aldığı bir komut durumu ile savunma, saldırıma, hareket etme, yaşama aksiyonlarından birini gerçekleştirir.
FR2.1.1	Oyuncu karakteri, düşman karakteri ile etkileşime girip ona zarar verebilir ya da ondan zarar alabilir.
FR2.2.0	Oyun yazılımindaki yapay zeka, belirli bir komut dizisini takip ederek belirlenmiş eylemlerden birini gerçekleştirebilir.
FR3.0.0	Düşman karakteri komut grubundan aldığı bir komut durumu ile savunma, saldırısı, hareket etme , oyuncunun verdiği hasar bu aksiyonlarından birini gerçekleştirebilir
FR3.1.0	Düşman karakteri , oyuncu karakteri ile etkileşime girip ona zarar verebilir ya da ondan zarar alabilir.
FR4.0.0	Oyuncu karakteri, oyuncu olmayan karakter ile girdiği etkileşimden bir bilgiye ya da bir nesneye (item) sahip olur.
FR5.0.0	Oyuncu karakteri kapı ile etkileşim, tabela ile etkileşim ,envanter ile etkileşime girerek ilgili nesnelerle etkileşimde bulunabilir.

III Tasarım Dokümantasyonu

8 Tasarım Hedeflerinin Tanımlanması

Engelsiz bir oyun deneyimi sunmak amacıyla oluşturulan gereksinimler göz önüne alındığında ve açık dünya türünde bir oyunda olabilecek tüm detayları birkaç farklı şekilde(sesli betimleme, haptik geribildirim) ele almak gerektiğinden tasarım hedefleri şekillendirilirken olabildiğince ekleme yapılmıştır.

8.1 Reliability (Güvenilirlik)

Ürünün belirli bir süre boyunca ve belirli koşullar altında sürekli olarak beklenen performansı gösterme yeteneğidir. Ürünün, minimum hata oranı ile çalışabilmesi için gerekli prosesler uygulandığında bileşenlerin uyumu ile sağlıklı bir proje geliştirme döngüsü oluşturmak, oyunculara sürekli ve tutarlı bir deneyim sunması amacıyla hata oranını düşürmek, oyun istikrarını ve dolayısıyla kullanıcı memnuniyetini artırmak hedeflenmiştir.

8.2 Modifiability (Değiştirilebilirlik)

Ürünün gelecekteki değişiklıklere, güncellemelere veya genişlemelere uygun olma yeteneğini ifade eder. Modüler bir tasarım benimseyerek, oyunun farklı bileşenlerinin bağımsız olarak geliştirilebilir ve değiştirilebilir olması hedeflenmiş, yüksek performanslı bir ürün elde etmek için amaçlanmıştır. Yeni bir oyun özelliği eklemek istendiğinde veya mevcut bir özellik güncellenmek istendiğinde, ilgili modül üzerinde çalışmak, geriye dönük uyumluluğu artırır ve geliştirme sürecini hızlandırır. Bu sayede, oyunun ömrü boyunca çeşitli yeniliklere hızlı bir şekilde adapte olabilir ve oyunculara güncel ve zengin bir deneyim sunabilir.

8.3 Maintainability (Sürdürülebilirlik)

İyi bir kod stili benimsemek, kodun belirli modüllerini belgeleyip yorumlayarak ve güçlü bir versiyon kontrol sistemi kullanarak yapılan değişiklikleri takip edip geri alabilme alanı oluşturmak ve kolaylıkla sürdürülebilir bir sistem oluşturmak hedeflenmiştir. Bu pratiğin benimsenmesi, geliştirme ekibinin etkili bir şekilde işbirliği yapmasını sağlamak, oyunun gelişimini ve başarı oranını desteklemek için önemlidir.

8.4 Adaptability (Uyarlanabilirlik)

Oyun mekanlığını ve içeriğini etkili bir şekilde güncelleyebilme yeteneği, yeni özellikleri uygularken veya oyunu güncellerken kullanıcı geri bildirimlerini sistematik bir şekilde değerlendirme, ve çoklu platform geliştirmesini başarılı bir şekilde uygulama amacıyla, Unity oyun motoru, bu hususlardaki farkındalığı göz önünde bulundurularak tercih edilmiştir.

Unity'nin uyarlanabilirlik yetenekleri, yeni özelliklerini uygularken veya oyunu güncellerken kullanıcı geri bildirimlerini değerlendirmeyi kolaylaştırır. Oyunun kullanıcılarından gelen geri bildirimler, geliştirme sürecinde önemli bir rol oynar. Unity'nin geniş kullanıcı tabanı ve topluluk desteği, bu geri bildirimleri daha sistemli bir şekilde değerlendirmeyi ve kullanıcı taleplerine hızlı bir şekilde yanıt vermeye sağlar. Bu da oyunun adaptasyon sürecini iyileştirir ve kullanıcıların taleplerine hızlı bir şekilde cevap verebilme esnekliğini artırır.

8.5 Efficiency (Verimlilik)

Verimlilik, bir görevi yerine getirirken yapılan hatalardan veya boş harcanılan materyaller, enerji, efor, para ve zamandan kaçınmak için sıkılıkla ölçülebilir bir yetenektir. Hızlı bir şekilde yanıt veren bir oyun tasarlayabilmek için giriş gecikmesini minimize etmek bu bağlamda hedeflenmiştir.

Oyun akışını düzenleyerek yüksek performanslı ses efektleri kullanmak, enerji verimliliği açısından optimize edilmiş bir oyun tasarlama hedeflenmiştir. Oyunculara, oyunun performansını kendi sistem özelliklerine göre özelleştirebilmeleri için geniş ayar seçenekleri sunulması hedeflenen bir başka verimlilik unsurudur.

8.6 Portability (Taşınabilirlik)

Bir bilgisayar programının düşük efor sarf edilerek farklı platformlarda çalıştırılabilmesidir. Geliştirme maliyetlerinin azaltılmasındaki temel konudur. Unity'nin çoklu platform desteği sayesinde, oyun hem masaüstü bilgisayarlar, hem de mobil cihazlar gibi çeşitli platformlarda sorunsuz bir şekilde çalışabilir.

8.7 Traceability of Requirements (Gereksinimlerin İzlenebilirliği)

Geliştirme yaşam döngüsünde bir gereksinimi ileriye doğru veya geriye doğru izleyebilme yeteneğidir. Bir oyunun belirli bir oyun tasarım dökümanındaki gereksinimlere uygunluğunu takip edebilen bir izleme sistemi ile kontrol edilebilir. Gereksinimlerin, proje döngüsünün sağlıklı bir şekilde gerçekleştirilebilmesi nezdinde bir gereksinim izleme sistemi kullanılarak kontrol sağlanacaktır. Araştırmalarımıza dayanarak kullanacağımız gereksinim izleme sistemi JIRA olarak seçilmiştir. JIRA, kullanıcı dostu arayüzü ve kapsamlı özelliklerini sayesinde birçok avantaj sunar.

8.8 Fault Tolerance (Hata Toleransı)

Bir sistemin, bir veya daha fazla bileşeninde arıza veya büyük bir işlev bozukluğu olması durumunda sistemin çalışmaya devam etmesini sağlayan esneklik özellikleidir. Çalışma kalitesi düşerse, küçük bir arızanın bile tam bir arızaya yol açabileceği safça hazırlanmış bir sistemle karşılaşıldığında, düşüş arızanın ciddiyeti ile orantılıdır. Hata toleransı özellikle yüksek kullanılabilirliğe sahip, görev açısından kritik ve hayatı önem taşıyan sistemlerde aranır. Oyuncunun oyun durumunu belirli aralıklarla veya önemli olaylardan önce kaydetmek

ve bu kayıtları kullanarak oyunu hata durumlarından kurtarmak mümkündür. Bu, oyuncunun tekrar bağlandığında son kayıtları yükleyerek oyunu devam ettirmesine izin verir.

8.9 Robustness (Sağlamlık)

Uygulama sırasında hatalarla ve hatalı girdilerle başa çıkabilme yeteneği. Projemizde hatayı minimumda tutmak asıl amacımız olsa da gerçekçilik açısından hiç hata olmaması mümkün olamadığından ürünün herhangi bir hata(bug) olduğunda çözülebilmesinin kolay olması açısından modüler olarak geliştirilecek olan sistemin herhangi bir durumdan etkilenmemesi amaçlanır.

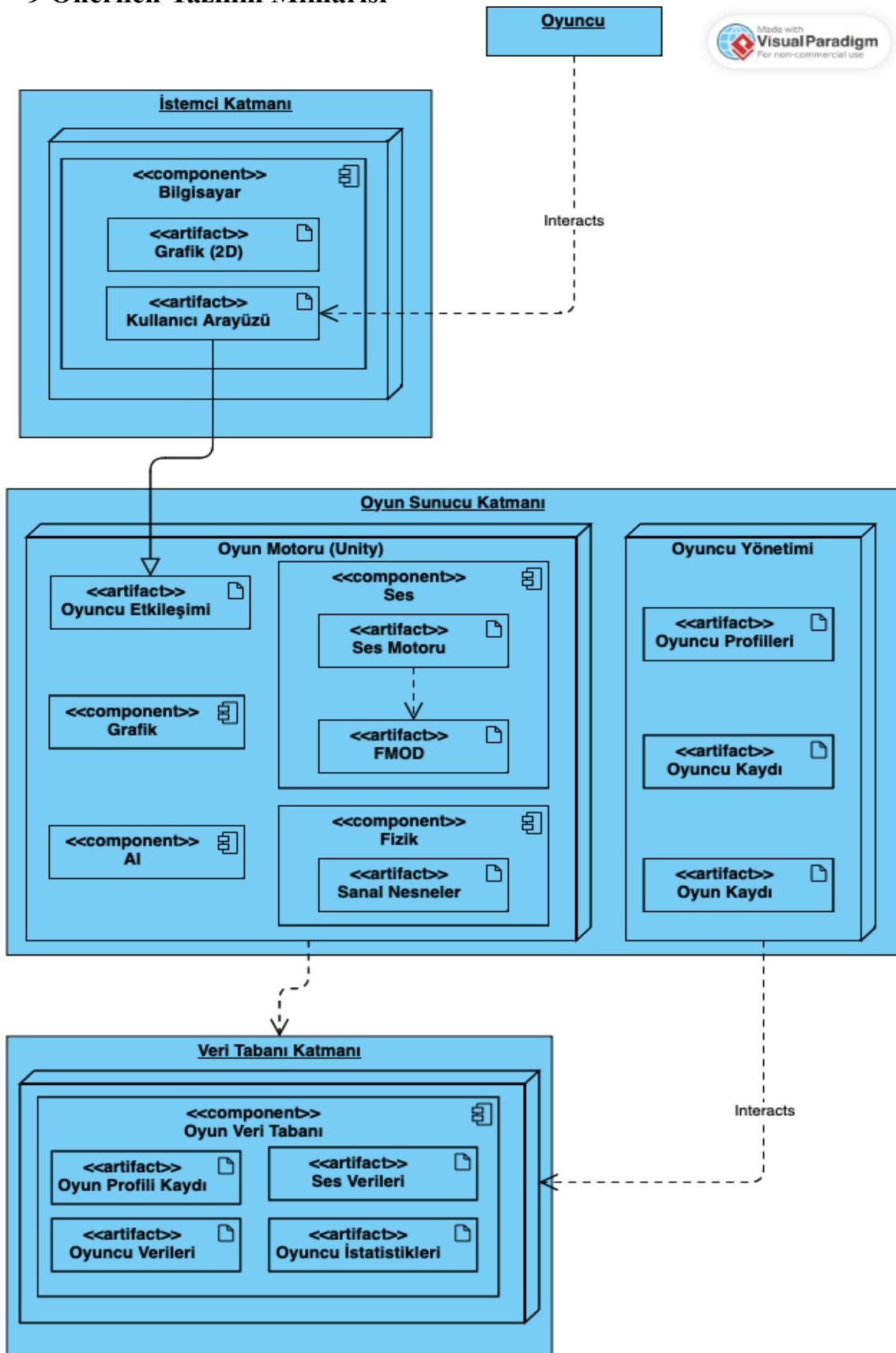
8.10 Well-defined Interfaces (İyi Tanımlanmış Arayüzler)

Proje detayları göz önüne alındığında kullanıcı dostu bir arayüz elde etmek büyük önem taşımaktadır. Bu kapsamda kullanılan kontrol cihazının, hareket odaklı, sesli betimlemeli ve haptik dokunuşları içeren bir arayüz ile gerçekleştirilmesi hedeflenmektedir.

Ana menü ve envanter menüsü gibi arayüzlerdeki kullanıcının arama alanını zorlaştırmayacak şekilde kontrolcünün hareketine odaklanılacaktır. Kontrolcü nereden başlıyorsa, o bölgede menünün seslendirilmesi, kullanıcının menüler arasında rahatça gezinebilmesini sağlar. Harita ve yol gösterme sistemleri gibi konum etkeni gerektiren sistemlerde ise sesli betimleme kullanımıyla ilgili farklılığa gidilecektir. Normalde rehberlik şeklinde ilerleyen sesli betimleme sistemi, alternatif olarak harita sisteminde tüm haritayı betimlemek ve seçilen kontrol tuşlarıyla yönlere göre açıklama yapılarak gerçekleştirilecektir.

Görev ve bilgi durumları, oyun içi iletişim durumları, karakterin can barı, enerji seviyesi, harita görev bilgileri gibi unsurlar, kullanıcıya bilgi aktarımında farklı arayüz tasarımlarını içerecektir. Görev ve bilgi durumlarında daha net ve odaklanmış bir iletişim, oyun içi iletişim durumlarında ise daha dinamik ve interaktif bir arayüz hedeflenmektedir. Oyun içindeki bilgiler, kullanıcının dikkatini dağıtmadan, anlaşılır bir şekilde sunulacak ve karakterin durumu hakkında net bilgiler sağlanacaktır.

9 Önerilen Yazılım Mimarisi



Sekil 9.1: WitDark Oyunu Deployment Diyagramı

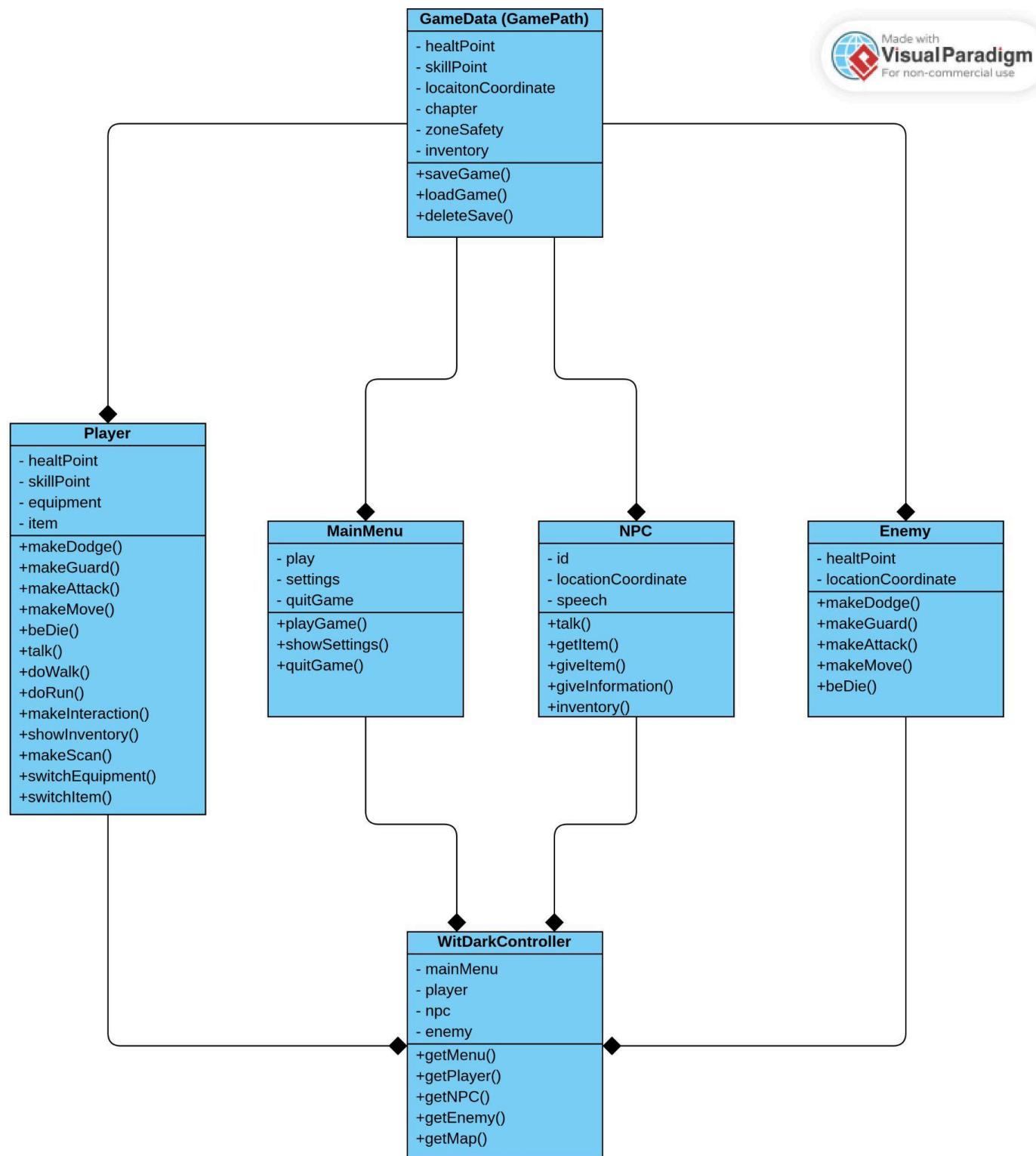
Katmanlı mimari, yazılım geliştirmenin bir yaklaşımıdır ve bir bilgisayar oyunu için bu yaklaşım, oyunun karmaşaklığını yönetmek, bakımını kolaylaştırmak ve geliştirme sürecini daha modüler hale getirmek için kullanılabilir. Katmanlı mimari kullanarak, bir bilgisayar oyununun karmaşaklığını yönetmek ve geliştirmek daha kolay olacaktır. Bu yaklaşım, büyük ve karmaşık oyun projelerinde ölçeklenebilirlik ve bakım kolaylığı sağlar. Bu sebeple ilerleyen aşamalarda bize kolaylık sağlamaası ve sistemin kolayca yeniden düzenlenmesi gibi unsurlara dikkat ederek projemiz için katmanlı mimariyi seçtik.

Diyagram istemci katmanı, oyun sunucu katmanı ve veritabanı katmanı olmak üzere 3 ana katmandan oluşmaktadır. İstemci katmanında oyuncu bilgisayar ile etkileşime girer. Grafik artifact'ı görsel öğeleri ve görsel efektleri işleyen bir bileşeni temsil etmektedir. Kullanıcı arayüzü ise kullanıcının etkileşimde bulunduğu kullanıcı arayüzü bileşenlerini içermektedir.

Oyun sunucu katmanı oyun motoru ve oyuncu yönetimi düğümlerini içermektedir. Oyun motoru bileşenleri grafik, yapay zeka, ses ve fiziktir. Ses bileşeninde ses motoru bulunmaktadır. Fizik bileşeni sanal nesneleri bulundurmaktadır. Oyuncu yönetimi bileşenleri ise oyuncu profilleri, oyuncu kaydı ve oyun kaydıdır.

Son olarak veritabanı katmanı, oyun veritabanını içermektedir. Burada bütün bilgiler, yazılımlar toplanmakta ve içinde tutulmaktadır. Oyun veritabanı, oyuncu verileri, oyun profili kaydı, ses verileri ve oyuncu istatistikleri bileşenlerini içerir. Veri tabanı katmanı diğer katmanlardan aldığı bilgilerle verileri depolar.

10 Sınıf Diyagramları

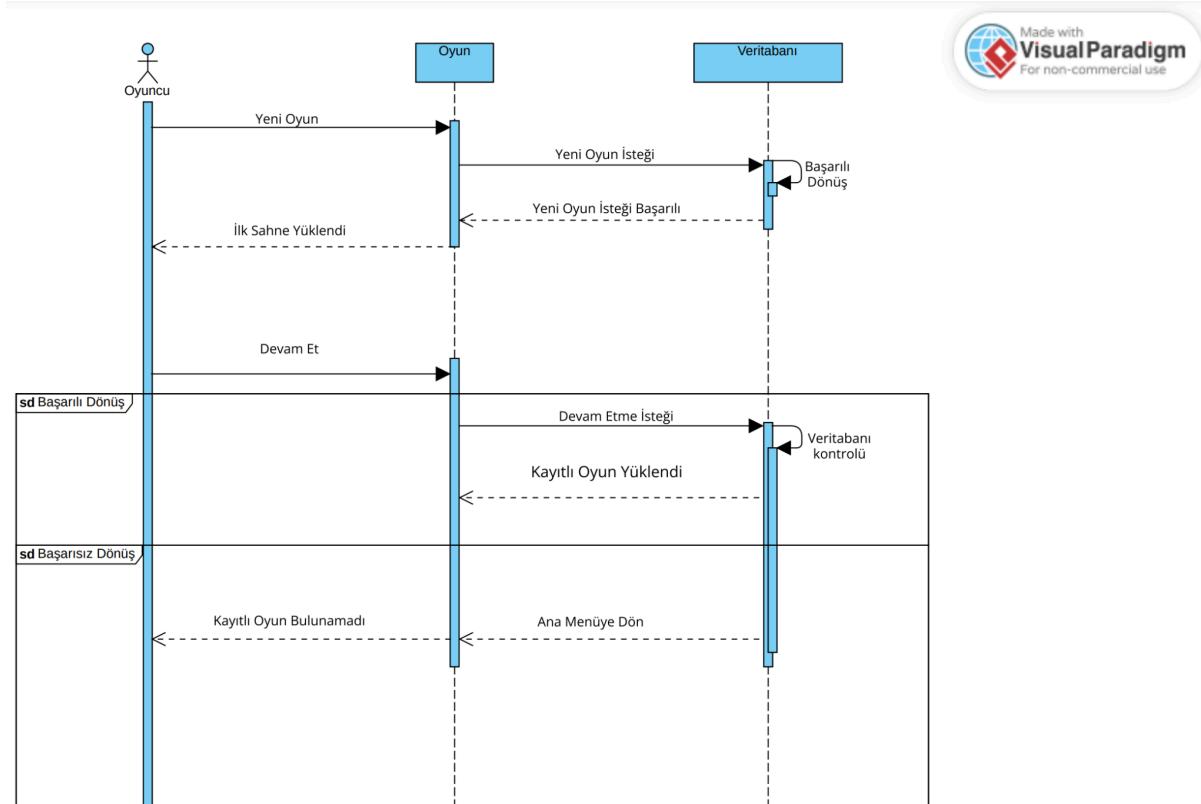


Şekil 10.1: Sınıf Diyagramı

Yukarıdaki sınıf diyagramında WitDark projesinin back-end(arka uç) kısmında kullanılan nesne tabanlı yaklaşım (Object-Oriented-Programming) gösterilmektedir. Bu kısım ele alındığında projenin büyük bir çoğunuğunda Unity oyun motoru kullanılması tasarlanmıştır.

11 Dinamik Model

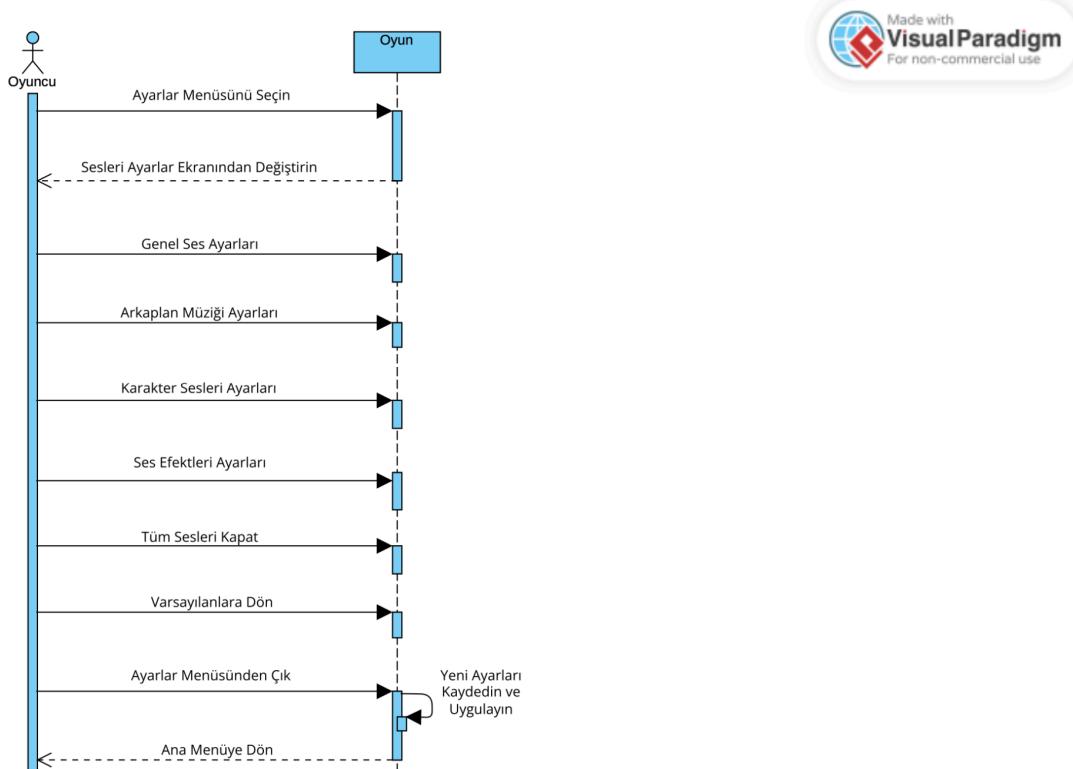
11.1 Menü: Yeni Oyun ve Devam Et Sequence Diyagramı



Şekil 11.1: Menü kullanımına ait Sequence Diyagramı: Yeni Oyun ve Devam Et Seçenekleri

Oyuncu, sisteme ilk giriş yaptığında ana menü ile karşılaşır. Ana menüdeki "Oyna" butonuna basılması durumunda, "Yeni Oyun" ve "Devam Et" seçenekleri görüntülenir. Eğer "Yeni Oyun" seçeneği seçilirse, veritabanı katmanına yeni oyun isteği gönderilir. Başarılı bir dönüş alındığında, ilk sahne yüklenir. Eğer "Devam Et" seçeneği seçilirse, veritabanı katmanına devam etme isteği gönderilir. Veritabanı kontrolü sonucunda başarılı bir dönüş alınırsa, kayıtlı oyun yüklenir. Başarısız bir dönüş durumunda ise ana menüye dönülür ve kullanıcıya "Kayıtlı oyun bulunamadı" bilgisi iletılır.

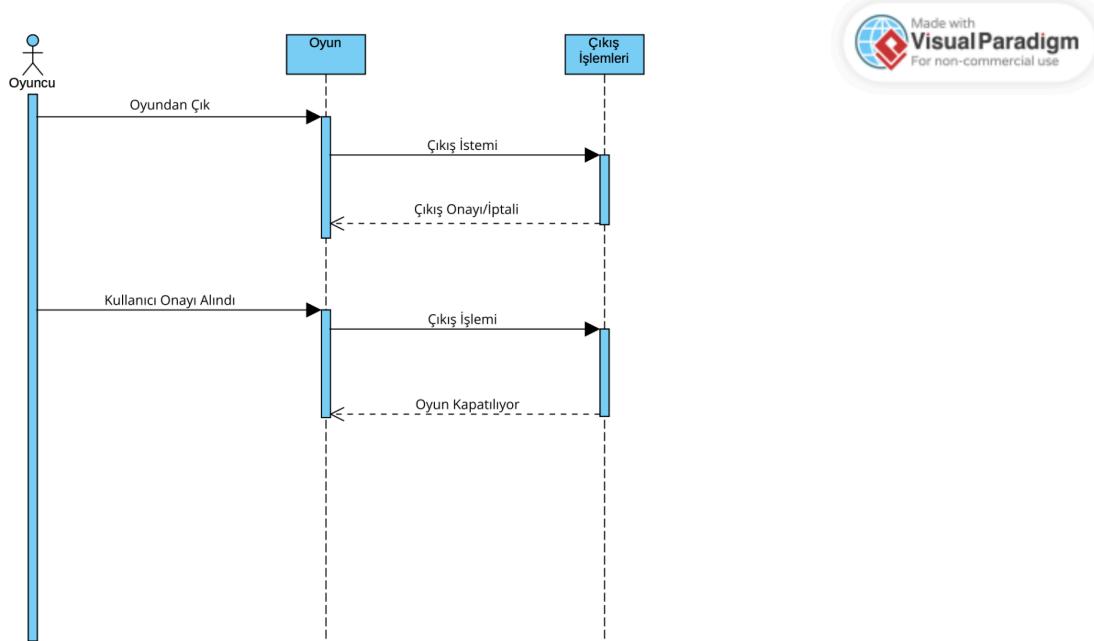
11.2 Menü: Ayarlar Sequence Diyagramı



Şekil 11.2: Menü kullanımına ait Sequence Diyagramı:Ayarlar

Kullanıcı, menüden "Ayarlar" butonuna tıkladıktan sonra, ekranın istediği ses ayarını seçer. Bu ekran içinde genel ses ayarları, arka plan müziği, karakter sesleri, ses efektleri, tüm sesleri kapat, ve varsayınlara dön seçenekleri bulunmaktadır. Kullanıcı, istediği ses ayarını seçtikten sonra, "Ayarları Kaydet ve Uygula" seçenekyle değişiklikleri onaylar. Bu durumda, yapılan değişiklikler oyun katmanına bildirilir. Başarılı bir şekilde ayarlar kaydedilip uygulandığında, kullanıcı ana menüye geri döner. Eğer kullanıcı "Ayarlar" menüsünden çıkmak isterse, yine oyun katmanına bir bildirim gönderir. Yapılan değişiklikler kaydedilir ve uygulanır, ardından ana menüe dönüş sağlanır.

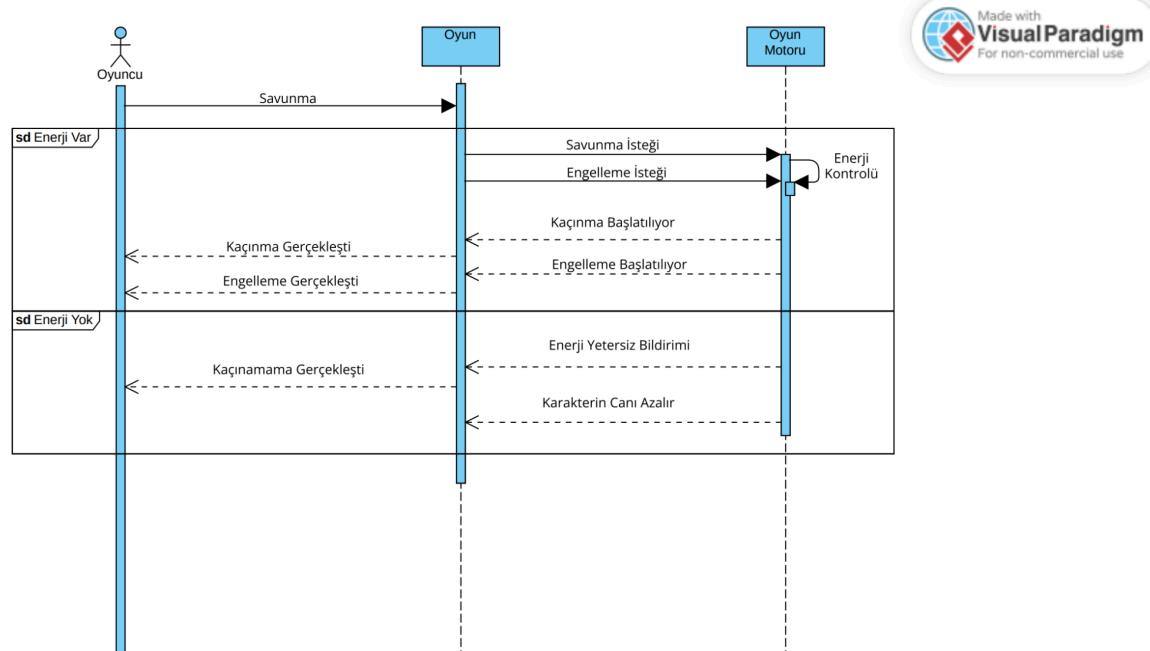
11.3 Menü: Oyundan Çık Sequence Diyagramı



Şekil 11.3: Menü kullanımına ait Sequence Diyagramı: Oyundan Çık Seçeneği

Oyuncu, ana menüdeyken "Oyundan Çık" butonuna bastığında, oyun katmanına çıkış bildirimi gönderilir. Çıkış işlemleri katmanına çıkış istemi iletilir. Çıkış işlemleri katmanından oyun katmanına çıkış onayı iptali bildirimi gönderilir. Oyuncu onayı alındıktan sonra, çıkış işlemi tamamlanarak oyun kapatılır.

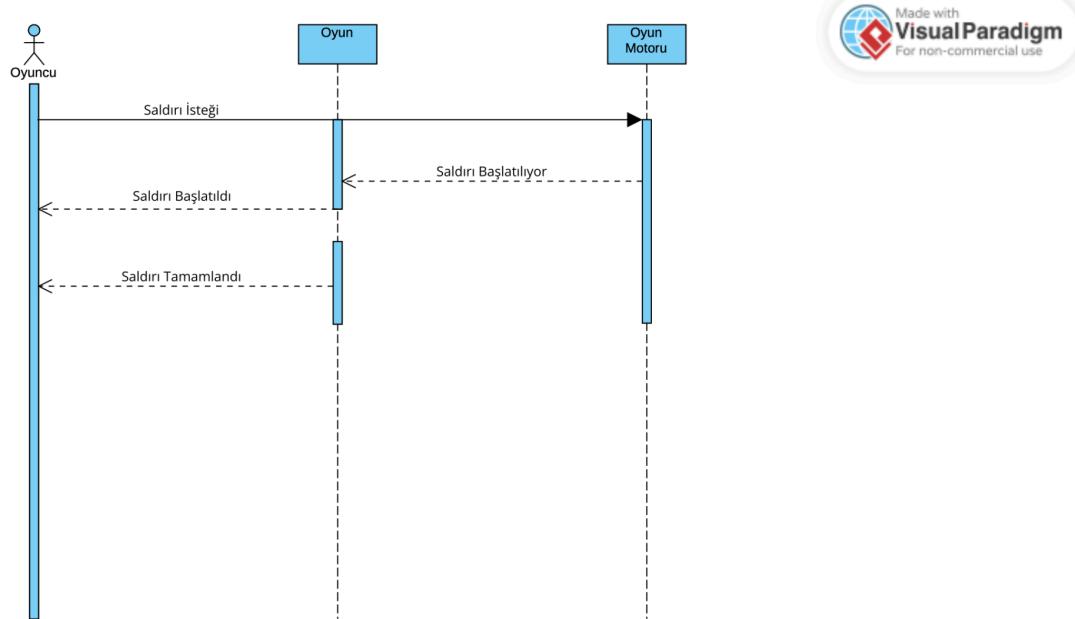
11.4 Oyuncu Mekanikleri: Savunma Sequence Diyagramı



Şekil 11.4: Oyuncu Karakterinin Davranış ve Temel Mekaniklerine ait Sequence Diyagramı:Savunma Gerçekleştirme

Oyuncu karakteri kendini savunmak istediğiinde enerji kontrolü yapılarak, yeterli enerji mevcut ise , karakter hızlı bir kaçınma manevrası gerçekleştirir. Yeterli enerji yoksa, kaçınma başarısız olur ve karakter kaçınmamış olur.Engelleme hareketi yapmak istediğiinde enerji kontrolü yapılır ve yeterli enerji varsa, karakter başarılı bir şekilde saldırıyı engeller.Yeterli enerji yoksa, engelleme hareketi gerçekleşmez ve karakterin canı azalır.

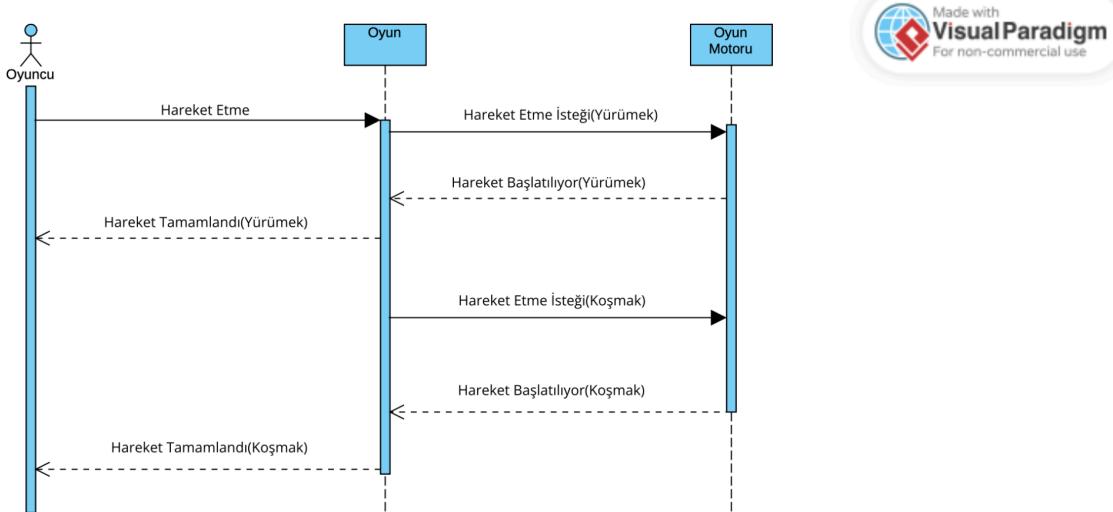
11.5 Oyuncu Mekanikleri: Saldırma Sequence Diyagramı



Şekil 11.5: Oyuncu Karakterinin Davranış ve Temel Mekaniklerine ait Sequence Diyagramı : Saldırma

Oyuncu, saldırısı eylemini gerçekleştirmek istediğiinde, bu istek oyun motoru katmanına iletilir. Oyun motoru, oyuncu tarafından iletilen saldırısı isteğini alır ve saldırıyı başlatır. Saldırı tamamlandığında oyuncuya saldırısının başarıyla gerçekleştirildiğine dair bir geri bildirim iletilir. Bu geri bildirim, oyuncunun eyleminin oyun dünyası üzerindeki etkilerini anlamasına yardımcı olur.

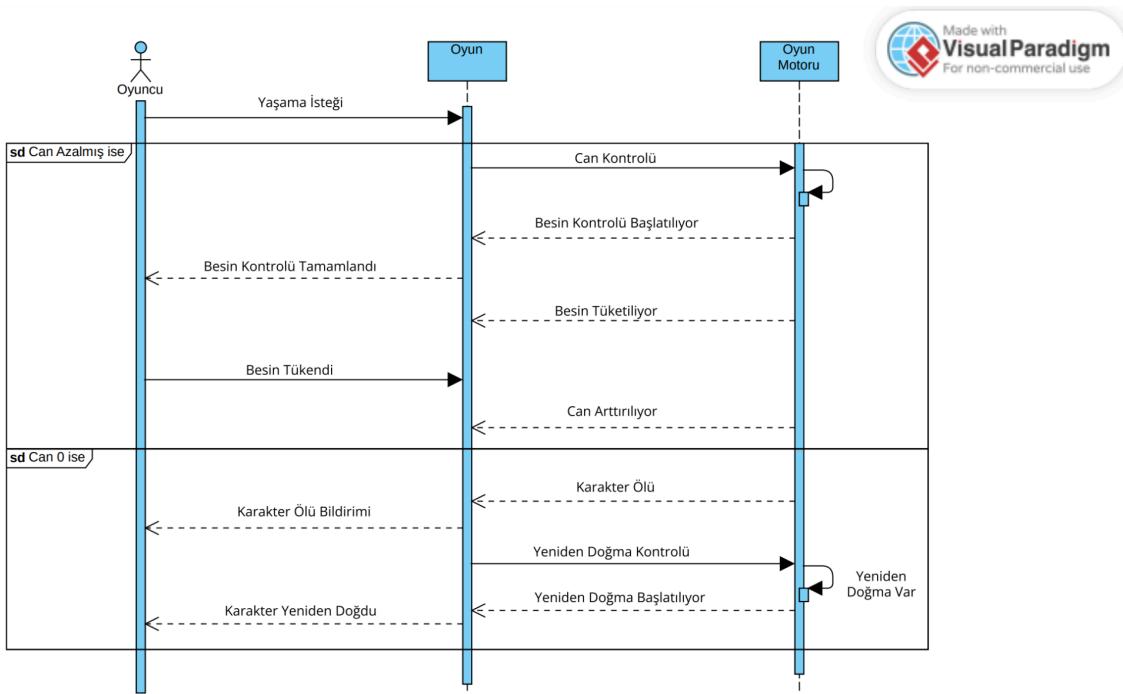
11.6 Oyuncu Mekanikleri: Hareket Etme Sequence Diyagramı



Şekil 11.6: Oyuncu Karakterinin Davranış ve Temel Mekaniklerine ait Sequence Diyagramı : Hareket Etme

Oyuncu hareket etme davranışını gerçekleştirmek istediğiinde, hareket etme isteğini oyun katmanına gönderir. Oyun motoru, oyuncunun yürüme isteğini alır ve yürüme hareketini başlatır. Oyuncuya "Hareket başlatıldı (yürüme)" geribildirimi ilettilir. Yürüme hareketi tamamlandığında, oyuncuya "Hareket tamamlandı (yürümek)" bildirimi gönderilir. Aynı şekilde, oyuncu koşma isteğini oyun katmanına gönderir. Oyun motoru, oyuncunun koşma isteğini alır ve koşma hareketini başlatır. Oyuncuya "Hareket başlatıldı (koşma)" geribildirimi ilettilir. Koşma hareketi tamamlandığında, oyuncuya "Hareket tamamlandı (koşmak)" bildirimi gönderilir.

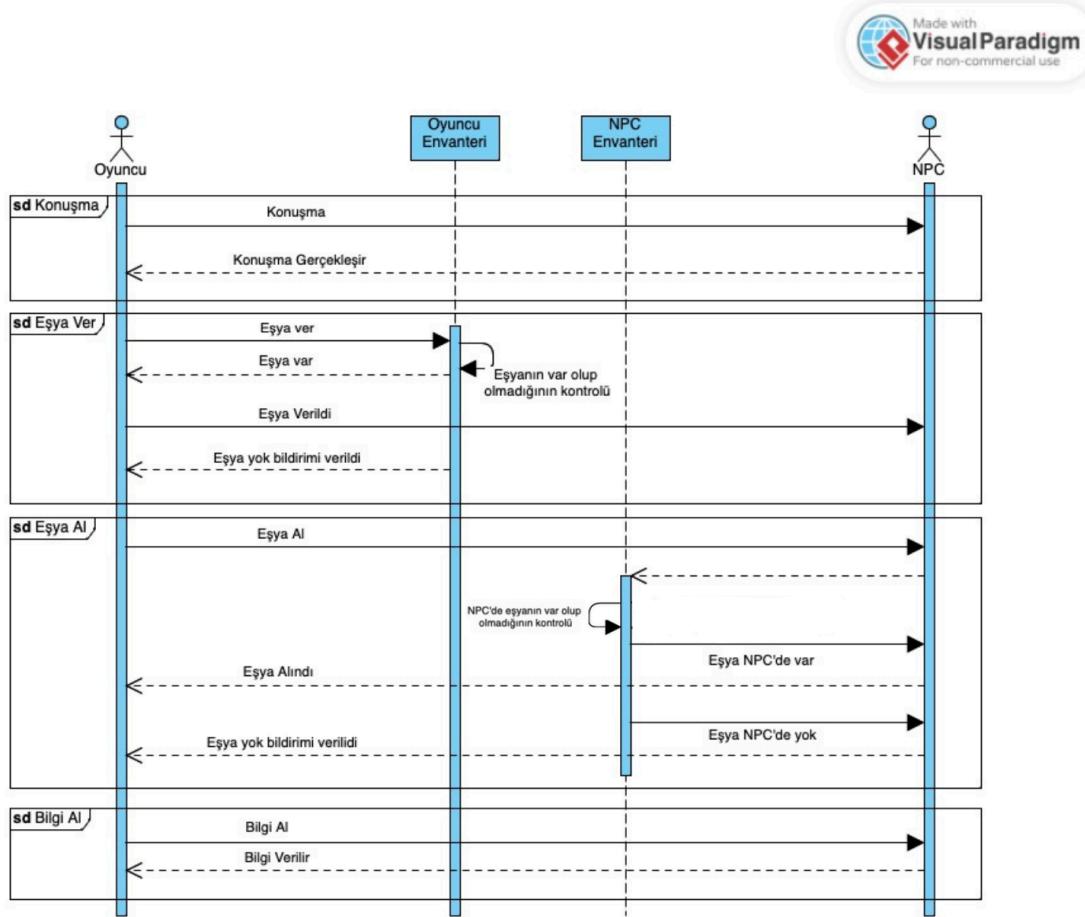
11.7 Oyuncu Mekanikleri: Yaşama Sequence Diyagramı



Şekil 11.7: Oyuncu Karakterinin Davranış ve Temel Mekaniklerine ait Sequence Diyagramı : Yaşama

Oyuncu, hayatı kalmak için oyun katmanına yaşama isteğini ileterek can kontrolünü başlatır. Oyun katmanı, can kontrolünü başlatmak üzere olduğunu oyun motoruna bildirir. Oyun motoru, oyuncunun mevcut can durumunu kontrol eder. Eğer can azalmışsa, besin kontrolü başlatılır ve oyuncuya "Besin kontrolü tamamlandı" geribildirimi gönderilir. Oyuncu, oyun katmanına besin tüketildi bildirimi gönderirse, can artışı gerçekleştirilir. Can kontrolü sırasında can 0'a eşitse, karakterin öldüğü bilgisi oyun motoru katmanından oyun katmanına ilettilir. Oyun katmanı, oyuncuya "Karakter ölü" geribildirimini ileterek karakterin olduğunu bildirir. Oyuncudan oyun katmanına yeniden doğma kontrol bildirimi gönderilmesi durumunda, kontrol sağlandığında yeniden doğma başlatılır ve oyuncuya "Yeniden doğma başlatıldı" geribildirimi ilettilir. Oyun katmanı, oyuncuya "Karakter yeniden doğdu" geribildirimi ileterek oyuncunun tekrar oyun dünyasına katıldığını bildirir.

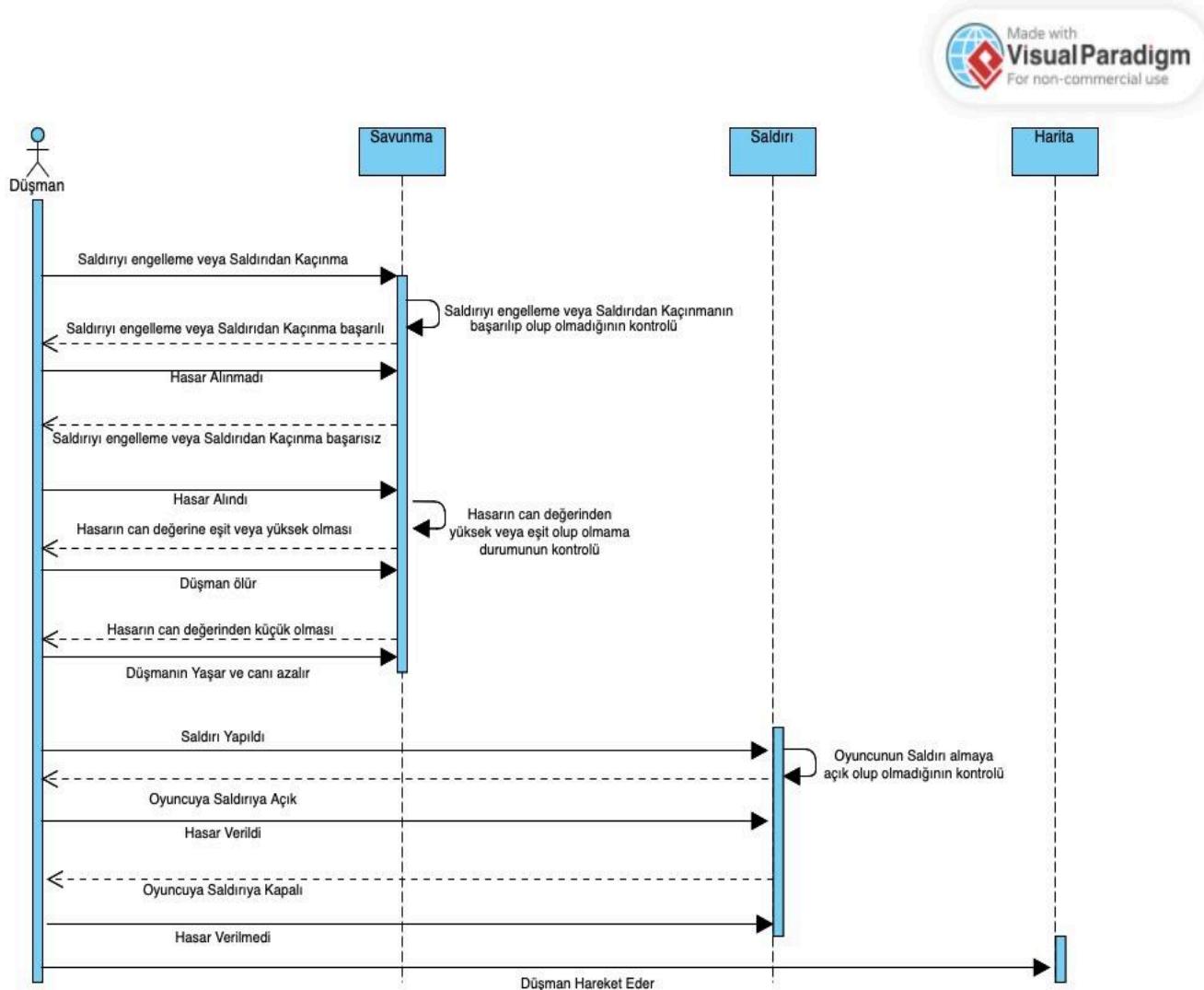
11.8 Oyuncu - NPC Etkileşimi Sequence Diyagramı



Şekil 11.8 : Oyuncu Karakterinin Non-Player-Character ile Etkileşimine Ait Sequence Diyagramı

Oyuncu NPC ile etkileşime girerek konuşma başlatır, konuşma gerçekleşir. Oyuncu, Oyuncu Envanteri ile etkileşime girer ve eşyanın var olması durumu kontrol edilir. Eşya var ise eşya verilir ve eşya yok ise oyuncuya eşya yok bildirimi verilir. Oyuncu, NPC ile etkileşime girer ve eşya alma talebi gönderir NPC envanterinde eşya var ise eşya verilir eşya yok ise oyuncuya eşya yok bildirimi verilir. Oyuncu, NPC ile etkileşime girerek bilgi alışverişi gerçekleştirir.

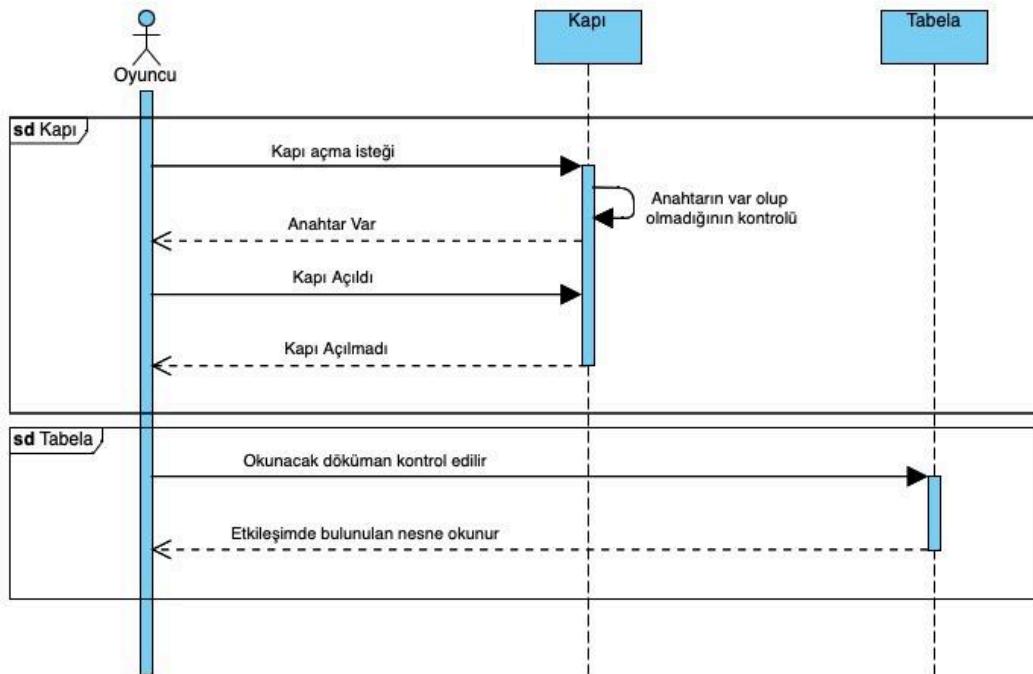
11.9 Düşman Karakterinin Mekanikleri Sequence Diyagramı



Şekil 11.9 : Düşman Karakterinin Davranışı ve Temel Mekaniklerine Ait Sequence Diyagramı

Düşman, Kaçınma veya Engelleme davranışları ile savunma gerçekleştirir. Saldırıyı engelleme veya saldırından kaçınma durumları başarılı ise oyuncuya Saldırıyı engelleme veya Saldırıdan kaçınma başarılı mesajı gönderilir. Düşman hasar aldığında hasarın can değerinden yüksek olup olmama durumu kontrol edilir hasar can değerine eşit veya can değerinden yüksek ise düşman ölü. Hasarın can değerinden düşük olması durumunda düşman yaşar ve canı azalır. Oyuncu saldırıyla açık ise Düşman Oyuncuya hasar verir. Oyuncu saldırıyla kapalı ise Düşman Oyuncuya hasar vermez. Düşman harita üzerinde hareket eder.

11.10 Oyuncu - Nesneler Etkileşimi: Kapı ve Tabela Sequence Diyagramı

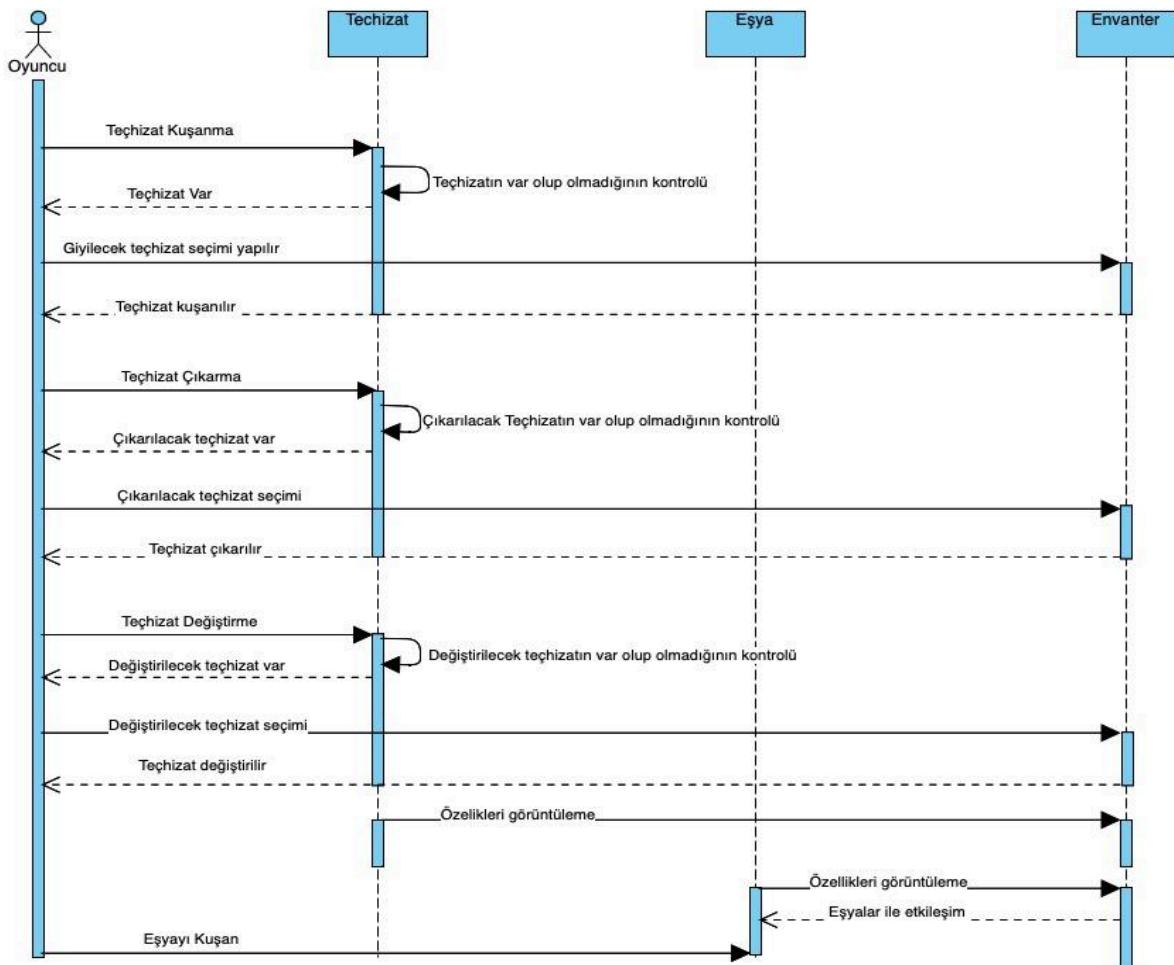


Şekil 11.10 : Oyuncunun Nesneler ile Etkileşimi’ne ait Sequence Diyagramı: Kapı ve Tabela ile Etkileşimi

Oyuncunun Kapı ile etkileşimi gerçekleşir. Anahtar var ise kapının açılması sağlanır. Aksi durumda anahtar yoktur ve kapı açılmaz bunun için oyuncuya kapı açılmadı bildirimi gönderilir. Oyuncunun Tabela ile etkileşimi gerçekleşir, okunacak doküman kontrol edilir, doküman var ise etkileşimde bulunulan nesnenin okunması gerçekleşir.

11.11 Oyuncu - Nesneler Etkileşimi:Envanter Sequence Diyagramı

Made with
Visual Paradigm
For non-commercial use



Şekil 11.11 : Oyuncunun Nesneler ile Etkileşimi’ne ait Sequence Diyagramı :Envanter ile Etkileşimi

Oyuncu Teçhizat ve Envanter ile etkileşime girer. Teçhizat var ise, Teçhizat seçimi yapılarak Teçhizat kuşanılır. Çıkarılacak Teçhizat var ise, Çıkarılacak Teçhizat seçimi yapılarak Teçhizat çıkarılır. Değiştirilecek Teçhizat var ise, Değiştirilecek Teçhizat seçimi yapılarak Teçhizat Değiştirilir. Teçhizat ile Eşya, Envanter ile etkileşime girerek Özellikleri görüntüler. Envanter Eşya ile etkileşime girer ve Oyuncu Eşyayı kuşanır.

12 Kullanıcı Arayüzü

Kullanıcı arayüzü (User Interface), bir cihazdaki insan-bilgisayar iletişimini ve etkileşimiğini sağlamaya yarayan noktayı tanımlar. Bu kimi zaman ekranları, kimi zaman klavyeyi, kimi zaman fareyi, kimi zaman da masaüstü görünümünü içerebilir. Bu noktada oluşturulan etkileşimin amacı hedef cihazın insanlar tarafından etkili bir şekilde çalıştırılması ve kontrol edilmesidir. Bu aynı zamanda kullanıcının herhangi bir uygulama veya web sitesi ile etkileşime girme yoludur. Günümüzde birçok işletmenin web uygulamalarına ve mobil uygulamalara giderek artan bağımlılığı, birçok şirketin bu alandaki geliştirmelere öncelik vermesine yol açmaktadır.

Bir site veya mobil uygulamada bulunan temel bileşenlerin, işlemlerin detaysız ve kabaca bir şekilde bir yerleştirme düzenine getirilerek içerisinde projenin kurgusunu, akışını ve işlevini hem müşteriye hem de yazılımcılara fikir sunma amacıyla yansıtma verilen ad Wireframe'dır.(Şema Tasarımı). Bu tasarımda oluşturulan Wireframe'ler(Şema tasarımları) Lo-Fi(Düşük Kalitede) veya Hi-Fi(Yüksek Kalitede) olabilir.

Lo-Fi(Low Fidelity) ve Hi-Fi(High Fidelity) terimleri kullanıcı arayüzünün kalitesini tanımlamakta kullanılır. Lo-Fi tasarımlar bizlere daha hızlı ve daha detaysız bir tasarım sunarken Hi-Fi tasarımlar ise, daha detaylı, projenin finaline daha sadık ve daha çok bilgi içeren bir tasarım sunar. Buradaki fikir, projede tasarım aşamasına gelindiğinde mümkün olduğu kadar zaman kurtarmaktır.

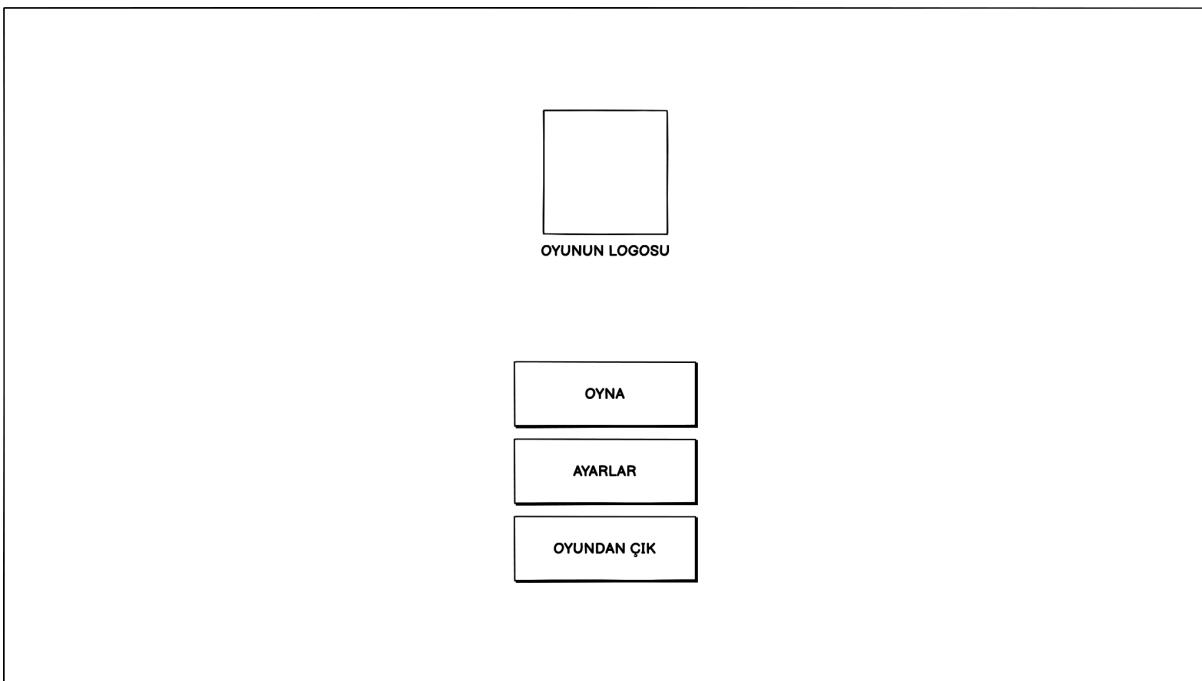
WitDark projesinde bir oyun yapmak hedeflenmiştir. Peki oyunlarda kullanılan kullanıcı arayüzü(User Interface) nedir? Basitçe söylemek gerekirse, oyun arayüzü tasarımı, kullanıcıların herhangi bir video oyununda amaçlanan eylem planını takip etmelerine yardımcı olacak görsel ipuçları oluşturmaya odaklanır. 'Açmak için X'e basın' gibi ekrandaki talimatları içerir.

Bu projenin gelişim aşamasında tasarlanan Lo-Fi ve Hi-Fi tasarımları ideal projeye ön ayak olan birer taslaktılar. Bu tasarımları yaparken oyundaki her bir kullanıcı arayüzünün ulaşmak istediğimiz kitleye uygun olmasına özen gösterdik. Bu sebeple tasarımlarımızda sadeliğin ve anlaşılabilirliğin yanı sıra kullanılabilirliğe de önem verdik.

WitDark projesinde tasarladığımız Lo-Fi(Low Fidelity) Wireframe'ler Balsamiq adlı prototipleme tool'u yardımıyla yapılmıştır. Balzamiq uygulaması herhangi bir uygulamada, kod yazılmadan önce, düşüncenizin veya konseptinizin dijital taslaklarını oluşturmayı amaçlar. Biz öncelikle Balzamiq yardımıyla ideal arayüzün iskelet yapısını oluşturduk.

Projenin Hi-Fi(High Fidelity) Wireframe tasarımları ise Photoshop adlı uygulama ile yapılmıştır. Adobe Photoshop piksel tabanlı Wireframe'ler oluşturmaya yardımcı olan bir uygulamadır. Kullanıcı arayüzü tasarımının bu aşamasında ideal ürüne en yakın arayüzleri elde ettik.

12.1 Oyun Menüsü Arayüzü



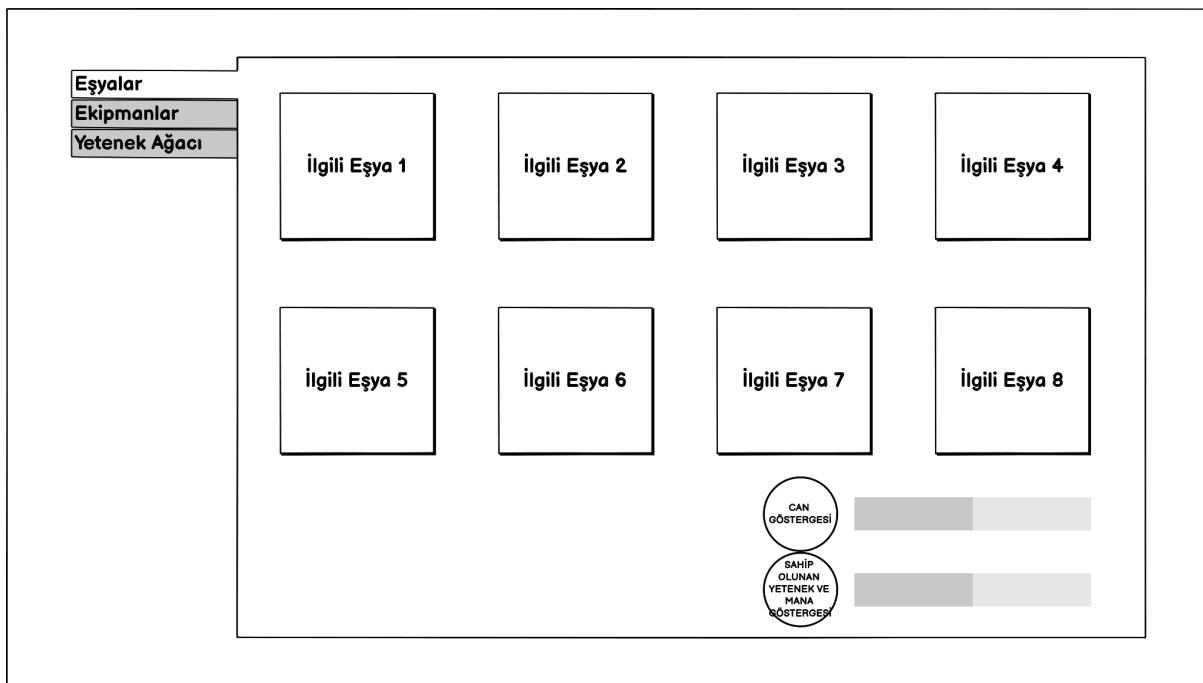
Şekil 12.1.1: Oyun Menüsü Arayüzü Lo-Fi Wireframe



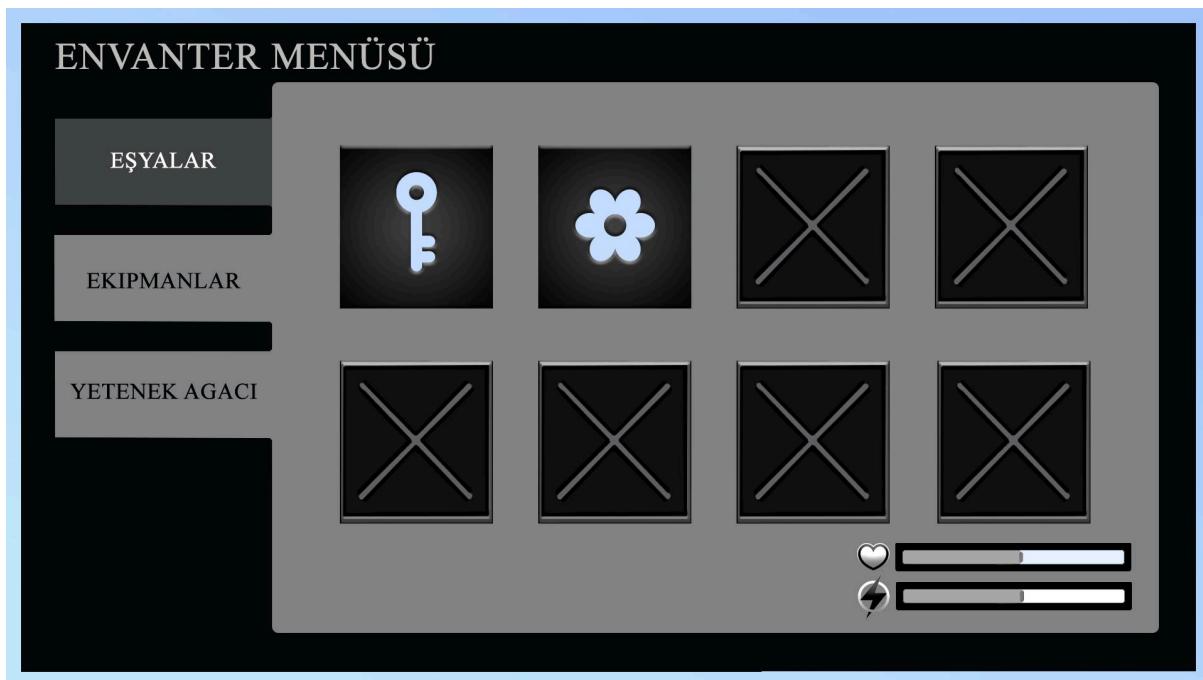
Şekil 12.1.2: Oyun Menüsü Arayüzü Hi-Fi Wireframe

Oyuncunun oyuna giriş yaptığı(oyuna başladığı) ekranıdır. Oyuncu isterse oyun esnasında bu menüye dönebilir veya hatta istenildiği takdirde, oyuncu bu menü aracılığı ile oyunun ayarlar kısmına yönlendirilir. Aynı şekilde oyuncu oyundan çıkış yapmak için bu menüyü kullanır.

12.2 Envanter Menüsü Arayüzü



Şekil 12.2.1: Envanter Menüsü Arayüzü Lo-Fi Wireframe



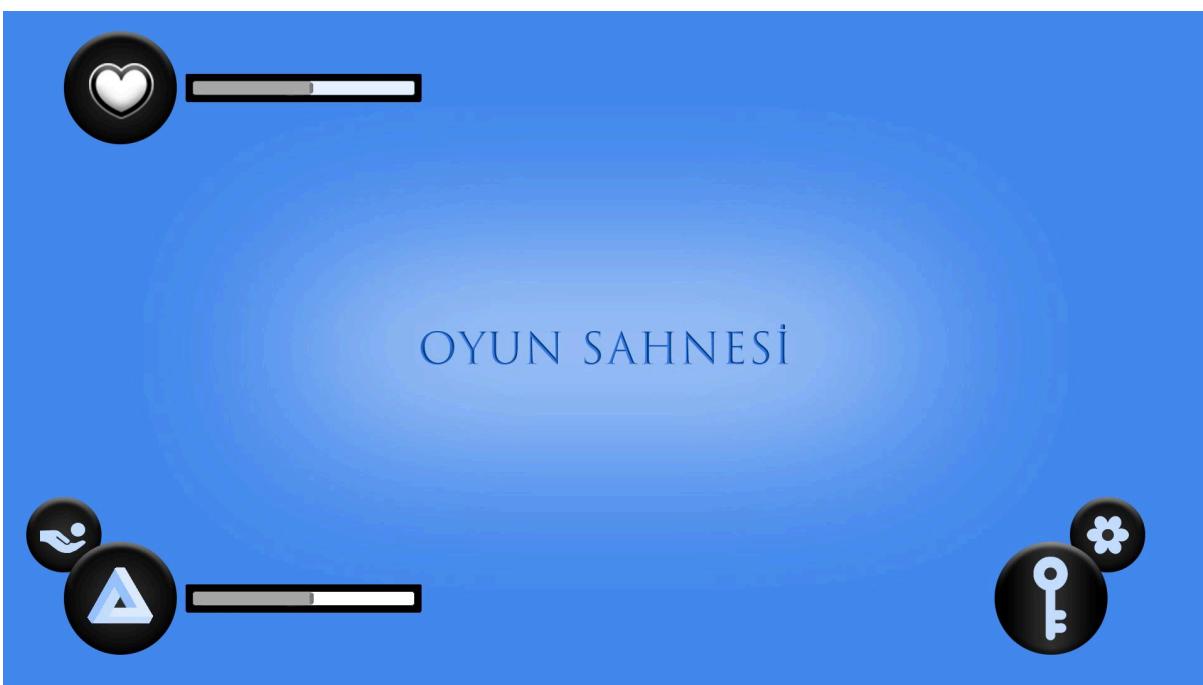
Şekil 12.2.2: Envanter Menüsü Arayüzü Hi-Fi Wireframe

Oyuncu oyuncunun herhangi bir kısmında çeşitli nedenlerden dolayı (örneğin; Hedef eşyaları incelemek, aktif eşyayı değiştirmek, ekipmanlarını incelemek, oyun karakterine ait yetenek ağacını incelemek...) bu menüyü kullanır. Bu menüye yönlendirilmek ve bu menüden çıkış yapmak için aynı varsayılan tuş kullanılmaktadır.

12.3 Oyun Arayüzü(HUD)



Şekil 12.3.1: Oyun Arayüzü(HUD) Lo-Fi Wireframe



Şekil 12.3.2: Oyun Arayüzü(HUD) Lo-Fi Wireframe

Oynanış esnasında ekrana aktarılan durum göstergesidir. Bu göstergeler karakterin canı, enerjisi ve güncel eşya gibi birtakım bilgileri içerir. Oyuncu atanan belli başlı tuşlar aracılığı ile de bu bilgilere erişebilir fakat bu göstergeleri tasarlarken düşüncemiz oyun projemizde hedeflediğimiz her bir kitleye yönelik bir oynanış kolaylığı sağlamaktır.

IV Test Planları

13 Test Edilebilecek ve Test Edilemeyecek Özellikler

Tablo 13.1*: WitDark Sisteminin Test Edilecek Alt Sistemleri

ID	Test Edilen Alt Sistemler	Test Edilen Senaryolar
1	Oyun Başlatma Sistemi	Ana Menü sistemi kullanılarak yeni bir oyun başlatma işlemi veya kayıtlı oyunun sorgulanması işlemi
2	Ayarlar Sistemi	Ayarları düzenleme ve yapılan değişikliklerin kaydedilip uygulandığını kontrol ederek "Ayarlar" menüsünden çıkış yapma işlemi
3	Oyun Ses Sistemi	Oyunda bulunan ses efektleri, müzik ve diyalogların kontrolleri
4	Girdi (Input) Sistemi	Oyuncu girdilerinin kontrolü
5	Karakter Savunma ve Saldırı Sistemi	Oyuncu Karakterinin savunma ve saldırma davranışlarının kullanılması
6	Oyuncu Hareket Sistemi	Oyuncu karakterinin yürüme ve koşma hareketlerinin gerçekleştirilmesi
7	Can Kontrol Sistemi	Oyuncu karakterinin ve düşman karakterin mevcut can durumunun kontrol edilmesi
8	NPC ile Etkileşim Sistemi	Oyuncu karakterinin NPC karakter ile etkileşiminin sonuçlarını kontrol etme işlemi
9	Düşman Davranış Sistemi	Düşman karakterin davranışlarını test etme işlemi
10	Oyuncu-Nesne Etkileşim Sistemi	Oyuncu karakterinin nesneler ile etkileşimi test etme işlemi
11	Yeniden Doğma Sistemi	Oyuncu karakterinin yeniden doğma işleminin başlatılmasını kontrol etme
12	Oyundan Çıkma Sistemi	Oyundan başarıyla çıktıığını kontrol etme işlemi
13	Arayüz Sistemi	Oyuncunun oyun sırasında gördüğü grafiksel öğelerin kontrolü

1. WitDark oyununda oyun başlatma sistemine uygulanan test senaryosudur. Oyuncu, WitDark oyununda "Yeni Oyun" veya "Devam et" seçeneklerinden birini tercih eder. Sonrasında oyunu başlatma işlemini gerçekleştirir. Oyun başlatma işlemi sırasında olası hatalar ve uygunsuz durumlar kara kutu testi ile belirlenir.

2. WitDark oyununda ayarlar sistemine uygulanan test senaryosudur. Oyuncu, WitDark oyununa giriş yaptıktan sonra "Ayarlar" butonuna tıklar. Ardından, ses ayarlarını düzenler ve yapılan değişikliklerin başarıyla kaydedilip uygulandığını kontrol eder. Son olarak, "Ayarlar" menüsünden çıkış yapma işlemi test edilir. Bu senaryo **gri kutu testi** kullanılarak gerçekleştirilir.
3. WitDark oyununda oyun ses sistemine uygulanan test senaryosudur. Ses efektlerinin şiddeti, oyun içerisinde farklı bölümlerdeki müziğin başlatılması, durdurulması ve seviyesinin ayarlanması, diyalog seslerinin netliği, ses tonu ve zamanlaması kontrol edilir. Bu senaryo **gri kutu testi** ile test edilir.
4. WitDark oyununda girdi (input) sistemine uygulanan test senaryosudur. Oyuncu karakterin girdiği bilgilerin etkilerini kontrol eder. Bu senaryo **gri kutu testi** kullanılarak gerçekleştirilir.
5. WitDark oyununda karakter savunma ve saldırı sistemine uygulanan test senaryosudur. Oyuncu karakterin savunma ve saldırı faaliyetlerini gerçekleştirmesine olanak sağlar. Bu senaryo **kara kutu testi** kullanılarak gerçekleştirilir.
6. WitDark oyununda oyuncu hareket sistemine uygulanan test senaryosudur. Oyuncu karakterinin yürüme ve koşma eylemlerini test eder. Bu senaryo **kara kutu testi** kullanılarak gerçekleştirilir.
7. WitDark oyununda can kontrol sistemine uygulanan test senaryosudur. Oyuncu ve düşman karakterin mevcut can durumunu kontrol ederek aldığı hasarı hesaplar. Bu senaryo **gri kutu testi** kullanılarak gerçekleştirilir.
8. WitDark oyununda NPC ile etkileşim sistemine uygulanan test senaryosudur. Oyuncu karakterin NPC karakter ile konuşma, eşya alma, eşya verme ve bilgi alma faaliyetlerini kontrol eder. Bu senaryo **kara kutu testi** kullanılarak gerçekleştirilir.
9. WitDark oyununda düşman davranış sistemine uygulanan test senaryosudur. Düşman karakterin savunma,saldırı,hareket etme,hasar verme ve hasar alma durumlarını kontrol eder.Bu senaryo **kara kutu testi** kullanılarak gerçekleştirilir.
10. WitDark oyununda oyuncu-nesne etkileşim sistemine uygulanan test senaryosudur. Oyuncu karakterin kapı,tabela,teçhizat,envanter nesneleri ile girdiği etkileşimi test eder.Bu senaryo **kara kutu testi** kullanılarak gerçekleştirilir.
11. WitDark oyununda yeniden doğma sistemine uygulanan test senaryosudur. Oyuncu karakterin canı 0 olduğunda gerçekleşen işlem yeniden doğma işlemidir. Bu senaryo **gri kutu testi** kullanılarak gerçekleştirilir.

12. WitDark oyununda oyundan çıkışa sisteme uygulanan test senaryosudur. Oyuncu oyundan çıkış butonuna bastığında işlemin başarılı bir şekilde gerçekleşip gerçekleşmediğini kontrol eder. Bu senaryo **kara kutu testi** kullanılarak gerçekleştirilir.

13. WitDark oyununda arayüz sistemine uygulanan test senaryosudur. Oyun sırasında grafiksel çıktıların beklenen şekilde olup olmadığını kontrol eder. Bu senaryo **kara kutu testi** kullanılarak gerçekleştirilir.

Tablo 13.2*: WitDark Sisteminin Test Edilmeyecek Alt Sistemleri

Test Edilemeyen Alt Sistemler	Test Edilemeyen Özellikler
Oyun Fizik Motoru Sistemi	Motorların tüm özellikleri ve durumları
Yapay Zeka (AI) Sistemi	Karmaşık oyun senaryosunda tüm olası durumlar
Çevresel Etkileşimler Sistemi	Oyuncunun beklenmedik davranışları veya rastgele olaylar
Oyun Performans Optimizasyonu Sistemi	Oyunun her türlü donanım konfigürasyonundaki performans sonucu farklılığı
Güvenlik ve Hile Engelleme Sistemi	Tüm olası saldırı senaryolarının önceden kestirilememesi
Arayüz Farklılık Sistemi	Arayüzün farklı platformlardaki değişkenleri

Oyunumuzda test edilemeyen alt sistemler şu şekildedir:

1. Oyun Fizik Motoru:

Oyunlar genellikle karmaşık fizik motorları içerir, bu motorlar nesnelerin fiziksel davranışlarını simüle eder. Ancak, bu motorların tüm özellikleri ve durumları tam anlamıyla test edilemeyebilir.

2. Yapay Zeka (AI):

Oyunlardaki yapay zeka genellikle belirli durumlar ve kararlar üzerine odaklanmıştır. Ancak, karmaşık oyun senaryolarında tüm olası durumları test etmek pratik olarak mümkün olmayabilir.

3. Çevresel Etkileşimler:

Oyun dünyasındaki her olası çevresel etkileşimi test etmek oldukça zorlu bir görevdir. Oyun ortamında ortaya çıkabilecek rastgele olaylar veya oyuncunun beklenmedik davranışları, tamamen önceden kestirilemez.

4. Oyun Performansı Optimizasyonu:

Oyun performansı genellikle donanım ve grafik optimizasyonlarına bağlı alt sistemlerle ilişkilidir. Ancak, oyunun her türlü donanım konfigürasyonunda kusursuz performansı garanti etmek her zaman mümkün olmayabilir.

5. Güvenlik ve Hile Engellemeye:

Oyun güvenliği ve hile engelleme önlemleri, her türlü saldırıyla karşı test edilmelidir, ancak tüm olası saldırısı senaryolarını önceden kestirmek zordur.

6. Arayüz Farklılık Sistemi:

Oyuncunun kullandığı kontrol cihazı türüne bağlı olarak yaşanabilecek girdi gecikmeleri, oyun deneyimini etkileyen kritik bir faktördür. Farklı kontrol cihazları, örneğin klavye, fare, denetleyici veya dokunmatik ekran, farklı tepki süreleri sunabilir, bu da oyunun performansını ve hassasiyetini etkiler.

Bu faktörler, bir oyun geliştirildiğinde, test sürecinde karşılaşılan önemli zorluklardır. Birçok senaryo test edilse de, oyunun genel karmaşıklığı sebebiyle, bazı alt sistemlerin tam kapsamlı testi pratik olarak mümkün olmayabilir. Bu durum, test aşamasında oyunumuz için belirli öncelikler belirleme ve kritik alanlara odaklanma amacıyla hareket etmemize sebep olmuştur.

Tablo 13.3*: Elle gerçekleştirilecek testlerin açıklaması

Elle Gerçekleştirilen Test	Test Edilen Özellik	Test Türü
Oyunu başlatmak için “YENİ OYUN” tuşuna bas	Kullanıcının yeni oyuna başlaması	Fonksiyonellik/ Kullanılabilirlik
Oyuna devam etmek için “OYUNA DEVAM ET” tuşuna bas	Kullanıcının kayıtlı oyuna devam etmesi	Fonksiyonellik/ Kullanılabilirlik
Oyundan çıkmak için “OYUNDAN ÇIK” tuşuna bas	Kullanıcının oyundan çıkışması	Fonksiyonellik/ Kullanılabilirlik
Karakteri hareket ettirmek için atanmış yön tuşlarına bas	Kullanıcının karakteri hareket ettirmesi	Fonksiyonellik/ Kullanılabilirlik
Karakterin savunma aksiyonu alması için atanmış olan savunma tuşlarına bas	Kullanıcının, karakterin savunmaya geçmesini sağlaması	Fonksiyonellik/ Kullanılabilirlik
Karakterin saldırı aksiyonu alması için atanmış olan saldırıcı tuşlarına bas	Kullanıcının, karakterin saldırıyla geçmesini sağlaması	Fonksiyonellik/ Kullanılabilirlik
Karakteri oyun içerisinde aşılmaması engellenmiş olarak atanmış hedeflere yönlendir	Karakterin oyun içerisindeki engel ile etkileşimi	Fonksiyonellik/ Kullanılabilirlik
Karakteri, kullanıcı olmayan düşmana yönlendir	Karakterin oyun içerisindeki düşman ile etkileşimi	Fonksiyonellik/ Kullanılabilirlik
Karakteri, kullanıcı olmayan başka bir karaktere yönlendir	Karakterin oyun içerisindeki başka karakter ile etkileşimi	Fonksiyonellik/ Kullanılabilirlik
Karakteri, kullanıcı olmayan karakterle konuşurstur	Karakterin, kullanıcı olmayan karakter ile konuşması	Fonksiyonellik/ Kullanılabilirlik

İşlemi gerçekleştirecek karakterle, kullanıcı olmayan bir karakter arasında iletişim kur	Karakterin, kullanıcı olmayan karakterle eşya takası gerçekleştirmesi	Fonksiyonellik/ Kullanılabilirlik
Oyun ayarları için “AYARLAR” tuşuna bas	Oyuncunun oyunu kendi tercihlerine göre ayarlaması	Fonksiyonellik/ Kullanılabilirlik
Oyun sesli betimlemesini açmak için atanmış tuşa bas	Oyun içinde sesli betimlemelerin devreye girmesi	Fonksiyonellik/ Kullanılabilirlik
Karakterin oyun içerisinde objeleri kullanmasını sağlayacak adımları izle	Karakterin objeler ile etkileşimi	Fonksiyonellik/ Kullanılabilirlik
Karakterin oyun içerisinde kullanabildiği obje var mı diye kontrol et	Objenin Karakterde olduğunun kontrolü	Fonksiyonellik/ Kullanılabilirlik
Karakterin canının azaldığını ses ile bildir	Karakter hasar alınca çıkan ses	Fonksiyonellik/ Kullanılabilirlik
Karakterin canının iyileştiğini ses ile bildir	Karakter iyileşince çıkan ses	Fonksiyonellik/ Kullanılabilirlik
Karakterin kaçınma aksiyonunu başarılı bir şekilde gerçekleştirdiğini ses ile bildir	Karakter kaçınma aksiyonu yapınca çıkan ses	Fonksiyonellik/ Kullanılabilirlik
Karakterin saldırısı aksiyonu yaptığı ses ile bildir	Karakter saldırısı yapınca çıkan ses	Fonksiyonellik/ Kullanılabilirlik
Karakter, kullanıcı olmayan karakter ile etkileşime girdiğinde konuşmanın gerçekleştiğini ses ile bildir	Kullanıcı olmayan karakter ile etkileşimde çıkan ses	Fonksiyonellik/ Kullanılabilirlik
Düşman karakterin canının azaldığını ses ile bildir	Düşman karakteri hasar alınca çıkan ses	Fonksiyonellik/ Kullanılabilirlik
Düşman karakterinin kendini belli etmesi için çıkardığı ses	Düşman karakterinin kendi sesi	Fonksiyonellik/ Kullanılabilirlik

Oyun bölgesi için belirlenen arka plan müziği	Oyun bölgesi müziği	Fonksiyonellik/ Kullanılabilirlik
Karakter hasar alınca arayüzdeki can göstergesinin azalması	Arayüzde karakterin canının azalması	Fonksiyonellik/ Kullanılabilirlik
Karakterin iyileştirme özelliğini kullanarak arayüzdeki can göstergesinin artması	Arayüzde karakterin kendini iyileştirmesi	Fonksiyonellik/ Kullanılabilirlik
Karakter kaçınma aksiyonunu gerçekleştirdiğinde enerji göstergesinin azalması	Arayüzde enerji göstergesinin azalması	Fonksiyonellik/ Kullanılabilirlik
Karakter belli bir süre bekledikten sonra enerji göstergesinin artması	Arayüzde enerji göstergesinin artması	Fonksiyonellik/ Kullanılabilirlik
Karakter can iksirini kullandığı zaman arayüzde iksiri azaldığı görünüyor mu	Arayüzde can iksiri göstergesinin azalması	Fonksiyonellik/ Kullanılabilirlik
Karakterin eşyaları arayüzde gösteriliyor mu	Arayüzde karakterin envanterindeki eşyaların görünmesi	Fonksiyonellik/ Kullanılabilirlik
Karakter, envanterindeki eşyalardan birini kullanınca envanterden gidiyor mu	Arayüzde karakterin envanterindeki eşyaların eksilmesi	Fonksiyonellik/ Kullanılabilirlik
Karakter kapı objesi ile etkileşime girdiğinde anahtarın olmadığı söylenmesi	Arayüzde anahtarın olmadığı söylenmesi	Fonksiyonellik/ Kullanılabilirlik
Karakter kapı objesi ile etkileşime girdiği zaman kapının açıldığının söylenmesi	Arayüzde anahtarın kapayı açığının söylenmesi	Fonksiyonellik/ Kullanılabilirlik
Karakterin çevresindeki nesnelerin söylenmesi için atanmış tuşa bas	Karaktere belirli mesafede olan nesnelerin sesli betimlenmesi	Fonksiyonellik/ Kullanılabilirlik
Karakterin çevresindeki kullanıcı olmayan karakterlerin söylenmesi için atanmış tuşa bas	Karaktere belirli mesafede olan kullanıcı olmayan karakterlerin sesli betimlenmesi	Fonksiyonellik/ Kullanılabilirlik
Haritayı görmek için atanmış tuşa bas	Arayüzde haritanın açılması	Fonksiyonellik/ Kullanılabilirlik

Karakterin sağ bakışı için atanmış tuşa bas	Karaktere gelen seslerin sağ taraftan sesli betimlenmesi	Fonksiyonellik/ Kullanılabilirlik
Karakterin sol bakışı için atanmış tuşa bas	Karaktere gelen seslerin sol taraftan sesli betimlenmesi	Fonksiyonellik/ Kullanılabilirlik

14 Test Cases

Ayrıntılı Test Durumları ve Sonuçları

Test Case ID: TC-01

Test Amaçları: Kullanıcının oyuna başarıyla giriş yapabilmesi kontrolü.

Test Prosedürleri ve Sonuçları: Kullanıcının Yeni Oyun veya Devam Et butonlarından birine tıklayarak sisteme giriş yapması.

Test Verisi: Veri, kayıtlı oyun verisi

Beklenen Sonuç: Kullanıcının Yeni Oyun ile başlangıç verisini, Devam Et ile kayıtlı oyun verisini onayladıktan sonra uygulamanın verileri yüklemesi ile oyuna giriş yapabilmesi

Gerçek Sonuç: Beklenen Sonuç ile aynı.

Hata Türü: Eksik veya yanlış veri / bağlantı hatası

Hata Önemi: Önemli

Test Case ID: TC-02

Test Amaçları: Kullanıcının Ayarlar Sistemi'ne girişi ve ayarları düzenlemesi, yapılan değişikliklerin başarıyla kaydedilip uygulandığının kontrolü.

Test Prosedürleri ve Sonuçları: Kullanıcının Ayarlar Sistemi'nde değişiklikler yapması ve bu değişikliklerin başarıyla kaydedilip uygulanması.

Test Verisi: Veri

Beklenen Sonuç: Kullanıcının Ayarlar Sistemi'ni güncellemesi.

Gerçek Sonuç: Beklenen Sonuç ile aynı

Hata Türü: Yapılan değişikliklerin kaydedilmemesi / Kaydedilen ayarların uygulanmaması

Hata Önemi: Önemli

Test Case ID: TC-03

Test Amaçları: Kullanıcının Ses Ayarı Sistemi'ne girişi ve ses efektleri, müzik ve diyalogların kontrolünün başarıyla kaydedilip uygulandığının kontrolü.

Test Prosedürleri ve Sonuçları: Kullanıcının Ses Ayarları Sistemi'nde değişiklikler yapması ve bu değişikliklerin başarıyla kaydedilip uygulanması.

Test Verisi: Veri

Beklenen Sonuç: Kullanıcının Ses Ayarları Sistemi'ni güncellemesi.

Gerçek Sonuç: Beklenen Sonuç ile aynı.

Hata Türü: Ayarların kaydedilmemesi / Kaydedilen ayarların uygulanmaması

Hata Önemi: Önemli; yapılan ses ayarlarının kaydedilip uygulanmaması, oyun deneyimini olumsuz etkiler.

Test Case ID: TC-04

Test Amaçları: Kullanıcının girdiği bilgilerin doğru şekilde işlenip oyun dünyasına etkisinin kontrolü.

Test Prosedürleri ve Sonuçları: Kullanıcının kontrol cihazı üzerindeki farklı tuşları kullanarak girdi girmesi.

Test Verisi: Oyuncu kontrol tuşları

Beklenen Sonuç: Kullanıcının girdilerinin girilmesi sonucunda oyunun kontrolünün başarıyla sağlanması.

Gerçek Sonuç: Beklenen Sonuç ile aynı

Hata Türü: Kontrol cihazının herhangi bir kısmının doğru çalışmaması / Eksik veya yanlış veri hatası

Hata Önemi: Önemli; girdi sisteminin doğru çalışmaması, oyuncunun istenilen şekilde etkileşimde bulunamamasına neden olabilir.

Test Case ID: TC-05

Test Amaçları: Oyuncu karakterin başarılı bir şekilde savunma ve saldırı faaliyetlerini gerçekleştirmesinin kontrolü

Test Prosedürleri ve Sonuçları: Oyuncunun savunma veya saldırısı yeteneklerini etkileyen kontrol tuşlarına basması sonucunda savunma veya saldırısı faaliyetlerinin gerçekleşmesi.

Test Verisi: Oyuncu kontrol tuşları

Beklenen Sonuç: Oyuncu karakterinin savunma veya saldırısı yeteneğini aktifleştirdiğinde başarılı bir şekilde savunma veya saldırısı yapabilmesi

Gerçek Sonuç: Beklenen Sonuç ile aynı.

Hata Türü: Kodlama hatası / Eksik veya yanlış veri

Hata Önemi: Kritik; oyunun temel oynanabilirliğini etkiler.

Test Case ID: TC-06

Test Amaçları: Oyuncu karakterin başarılı bir şekilde yürüme ve koşma hareketi yeteneklerinin kontrolü.

Test Prosedürleri ve Sonuçları: Oyuncu karakterin yürüme veya koşma tuşlarına basıldığında harekete geçmesi.

Test Verisi: Oyuncu kontrol tuşları

Beklenen Sonuç: Oyuncu karakterin yürüme ve koşma yeteneklerini başarılı bir şekilde kontrol edebilmesi.

Gerçek Sonuç: Beklenen Sonuç ile aynı.

Hata Türü: Fonksiyon hatası

Hata Önemi: Önemli; oyunun kullanıcı deneyimini etkiler.

Test Case ID: TCH-07

Test Amaçları: Kullanıcı tarafından oyuncu karakterin ve düşman karakterinin mevcut can durumunun kontrolü ve aldığı hasarın hesaplanması.

Test Prosedürleri ve Sonuçları: Kullanıcı tarafından oyuncu ve düşman karakterlerinin can durumu kontrol edilir ve aldığı hasar hesaplanır.

Test Verisi: Veri

Beklenen Sonuç: Oyuncu ve düşman karakteri hasar aldığından can durumlarının düşmesi ve hasar almadıklarında can durumlarının aynı kalması beklenir.

Gerçek Sonuç: Beklenen sonuç ile aynı.

Hata Türü: Eksik veya yanlış veri / Bağlantı hatası

Hata Önemi: Önemli

Test Case ID: TCH-08

Test Amaçları: Oyuncu karakterinin Non-Player-Character ile etkileşimi kontrol edilir.

Test Prosedürleri ve Sonuçları: Oyuncu karakteri ile NPC karakter arasında konuşma, eşya alma, eşya verme ve bilgi alma faaliyetlerinin gerçekleştirir.

Test Verisi: Veri

Beklenen Sonuç: Oyuncu karakterinin NPC karakter ile etkileşime girmesi ve istenilen faaliyetlerin gerçekleştirilmesi.

Gerçek Sonuç: Beklenen sonuç ile aynı.

Hata Türü: Eksik veya yanlış veri / Bağlantı hatası

Hata Önemi: Önemli

Test Case ID: TCH-09

Test Amaçları: Düşman karakterinin savunma, saldırısı, hareket etme, hasar verme ve hasar alma eylemlerini gerçekleştirdiğinin kontrolü.

Test Prosedürleri ve Sonuçları: Düşman karakterinin gerekli eylemleri gerçekleştirmesi ve bu eylemler sonucunda ortaya çıkması beklenen durumların gerçekleşmesi

Test Verisi: Veri

Beklenen Sonuç: Düşman karakterinin gerekli eylemleri gerçekleştirmesi ve bu eylemler sonucunda ortaya çıkması beklenen durumların gerçekleşmesi beklenir.

Gerçek Sonuç: Beklenen sonuç ile aynı.

Hata Türü: Eksik veya yanlış veri / Bağlantı hatası

Hata Önemi: Önemli

Test Case ID: TCH-10

Test Amaçları: Oyuncu karakterinin nesneler ile etkileşime girmesi test edilir.

Test Prosedürleri ve Sonuçları: Oyuncunun kapı, tabela, teçhizat ve envanter nesneler ile etkileşime girmesi kontrolü.

Test Verisi: Veri

Beklenen Sonuç: Oyuncunun nesneler ile etkileşime girmesi durumunda ortaya çıkacak durumların gerçekleşmesi.

Gerçek Sonuç: Beklenen sonuç ile aynı.

Hata Türü: Eksik veya yanlış veri / Bağlantı hatası

Hata Önemi: Önemli

Test Case ID: TCH-11

Test Amaçları: Oyuncu karakterinin yeniden doğması işleminin başlatılması kontrolü.

Test Prosedürleri ve Sonuçları: Oyuncu karakterinin canı 0 olduğunda yeniden doğması işleminin gerçekleşmesi.

Test Verisi: Veri

Beklenen Sonuç: Oyuncu karakterinin canı bittiği zaman oyuncunun yeniden doğması ve yeni oyunun başlaması beklenir.

Gerçek Sonuç: Beklenen sonuç ile aynı.

Hata Türü: Eksik veya yanlış veri / Bağlantı hatası

Hata Önemi: Önemli

Test Case ID: TCH-12

Test Amaçları: Kullanıcının oyundan başarıyla çıkışılmasını kontrol edilmesi.

Test Prosedürleri ve Sonuçları: Kullanıcının oyundan çıkmak istediği zaman Oyundan Çık butonuna bastığında bu işlemin başarıyla gerçekleşmesi.

Test Verisi: Veri

Beklenen Sonuç: Kullanıcının oyundan çıkmak istediği zaman oyundan çıkışılması.

Gerçek Sonuç: Beklenen sonuç ile aynı.

Hata Türü: Eksik veya yanlış veri / Bağlantı hatası

Hata Önemi: Önemli

Test Case ID: TC-13

Test Amaçları: Arayüz sistemindeki grafiksel çıktıların kontrolü.

Test Prosedürleri ve Sonuçları: Kullanıcının oyunu başlatması ve diğer grafiksel unsurları kontrol cihazı yardımıyla görüntülemesi

Test Verisi: Veri

Beklenen Sonuç: Oyun başladığında arayüzdeki grafik öğelerinin çıktılarının başarıyla görüntülenmesi

Gerçek Sonuç: Beklenen Sonuç ile aynı.

Hata Türü: Fonksiyon hatası / Ekran hatası

Hata Önemi: Önemli

V Sözlük

Ağ Diyagramı: Bir projedeki faaliyetleri, etkileşimleri ve sıralamaları görsel olarak temsil eden bir diyagramdır.

Beyaz Kutu Testi: Bir yazılım veya sistem bileşeninin iç yapısını değerlendirmek amacıyla yapılan bir test türüdür. Bu test, genellikle yazılımın kodunu inceleyerek ve iç mantığını anlayarak gerçekleştirilir.

Deployment Diyagramı: Yazılım ve donanım bileşenlerinin fizikalî dağılımını gösteren UML diyagramı. Sistemdeki bileşenlerin nerede çalıştığını ve nasıl bağlandığını gösterir.

Fonksiyonel Gereksinimler: Tasarlanan bir ürünün, bir çok dalda ne kadar yetkin olacağının belirlendiği ve hangi alanların daha fazla katkı olacağını belirleyen gereksinimlerdir.

Gereksinim Belirleme Sistemi: Bir projenin veya sistemin ihtiyaçlarını belirleme sürecini yöneten bir sistemdir. Bu süreç, kullanıcıların ve paydaşların bekłentilerini anlamak, gereksinimleri tanımlamak ve proje veya sistem için gerekli olan özelliklerini belirlemek için kullanılır.

Gri Kutu Testi: Siyah kutu testi (dış yapıya odaklanan) ve beyaz kutu testi (iç yapıya odaklanan) arasında bir orta noktada olan bir test türüdür. Gri kutu testi, hem yazılımın dış davranışlarını hem de iç yapısını değerlendirmeyi amaçlar.

Haptik Öğe: Dokunma duyusuyla etkileşime geçen nesne veya teknolojik öğedir. Haptik geribildirim, kullanıcıya dokunma veya titreşim gibi duyusal tepkilerle geri dönüş sağlar, genellikle dokunmatik ekranlar, sanal gerçeklik kumandaları veya akıllı telefonlarda kullanılır.

Hi-Fi (High Fidelity): Yüksek kalite oranını anlatmak için kullanılan terimdir.

Kara Kutu Testi: Bir yazılım veya sistem bileşenini sadece dışarıdan gözlemleyerek, iç yapısına odaklanmadan değerlendiren bir test türüdür. Bu test, yazılımın belirli girdi değerleriyle nasıl tepki verdiği ve beklenen çıktıları üretecek işlevselligini yerine getirip getirmedigini kontrol eder.

Kritik Yol: Bir proje veya görevin tamamlanması için gereken en uzun süreyi temsil eden ve proje bitiş tarihini etkileyen bir dizi görevin birleşimidir. Proje yönetimi ve zaman çizelgesi oluşturma süreçlerinde kullanılan bir kavramdır.

Lo-Fi (Low Fidelity): Düşük kalite oranını anlatmak için kullanılan terimdir.

Oyun Mekanığı: Bir video oyununun temel oynanış öğeleri ve kurallarını ifade eder. Oyun mekanığı, oyuncuların oyun dünyasıyla etkileşimde bulunmalarını, görevleri tamamlamalarını ve oyun içinde ilerlemelerini sağlar. Bu, oyunun temel dinamikleri, kuralları, stratejileri ve olaylarını içerir.

Oyun Motoru: Video oyunlarının geliştirilmesi için kullanılan yazılım ortamıdır.

Performans Optimizasyonu: Bir sistem, uygulama veya sürecin çalışma verimliliğini ve hızını artırmak amacıyla yapılan çeşitli geliştirmeleri içeren bir süreçtir.

Prosedür: Genellikle belirli bir süreci veya işlemi düzenlemek veya yönetmek amacıyla belirlenmiş adımlar ve kurallar içeren bir talimat veya düzenlemeyi ifade eder. Prosedürler, bir kuruluş içindeki belirli görevleri yerine getirme, standartları sürdürme veya belirli bir amaç için adım adım önerileri uygulama amacını taşır. Bu terim, iş süreçleri, güvenlik protokoller, kalite kontrol adımları gibi çeşitli bağamlarda kullanılabilir.

Sequence Diyagramı: Nesneler arasındaki etkileşimleri ve zaman sıralamasını gösteren UML diyagramı. Sistemdeki olayları ve mesajları zaman sırasında gösterir, işleyişi anlamak için kullanılır.

Ses Motoru: Ses sinyallerini üreten veya işleyen bir yazılımı ifade eder.

Sistem Testi: Bir bilgisayar sisteminin genel performansını ve işlevsellliğini değerlendiren test sürecidir.

Sınıf Diyagramı: Unified Modeling Language (UML) adı verilen bir modelleme dilinde kullanılan bir diyagram türüdür. Sınıf diyagramları, bir sistemde bulunan sınıfları, bu sınıflar arasındaki ilişkileri, sınıfların özelliklerini (alanlar) ve davranışlarını (metotlar) gösterir. Bu diyagramlar, bir yazılım uygulamasının tasarımını ve yapısını görselleştirmek amacıyla kullanılır.

Sanal Nesneler: Yazılım tarafından oluşturulan, dijital ortamda var olan ve kullanıcılar tarafından etkileşimde bulunulabilen nesnelerdir.

Use Case Diyagramı: Kullanıcının bir fonksiyon aracılığı ile sistem üzerinde gerçekleştirmek istedikleri işlemlerin diyagram halidir.

Veritabanı: Veri depolamak, düzenlemek ve yönetmek için kullanılan yapılandırılmış depolama sistemi. Bilgiyi tablolar, sütunlar ve satırlar şeklinde düzenleyerek etkili bir şekilde depolar ve erişimini sağlar.

Referanslar

1. https://en.wikipedia.org/wiki/Non-functional_requirement
2. <https://blindhelp.net/>
3. <https://samtupy.com/>
4. <https://www.agarchive.net/pages/devs/eurofly.html>
5. [Ideal Modeling & Diagramming Tool for Agile Team Collaboration \(visual-paradigm.com\)](https://visual-paradigm.com/Ideal%20Modeling%20&%20Diagramming%20Tool%20for%20Agile%20Team%20Collaboration)
6. <https://mwgame.net/>
7. <https://mwgame.net/download>
8. <https://caniplaythat.com/2020/06/18/the-last-of-us-2-review-blind-accessibility/>
9. <https://univera-ng.blogspot.com/2010/04/uml-ve-modelleme-bolum-10-compo-nent-ve.html>
10. <https://staff.emu.edu.tr/duygucelik/Documents/BLGM412/UML%20ve%20Modellem-e.pdf>
11. <https://agilebusinessthink.wordpress.com/2017/11/23/fonksiyonel-ve-fonksiyonel-olmayan-gereksinim-nedir/#:~:text=Fonksiyonel%20gereksinim%2C%20bir%20sistemin%20neyi,%E2%80%9Cmust%20do%E2%80%9D%20ile%20kullan%C4%B1l%C4%B1r>