****

**T.C.**

**İSTANBUL SAĞLIK VE TEKNOLOJİ ÜNİVERSİTESİ**

**MÜHENDİSLİK VE DOĞA BİLİMLERİ FAKÜLTESİ**

**BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**

**BYM304 - Yazılım Mimarisi ve Tasarımı**

**2-D Platform Game**

**Emre ŞEN-210601004**

**Kutay Can BATUR-210601009**

**Beytullah ALKAN-210601027**

**ÖĞRETİM GÖREVLİSİ: Kemal Çağrı SERDAROĞLU**

**İçindekiler**

[1. Giriş 2](#_Toc168264434)

[2. Proje Tanımı ve Gereksinimleri 2](#_Toc168264435)

[2.1. Proje Tanımı 2](#_Toc168264436)

[2.2. Fonksiyonel Gereksinimler 2](#_Toc168264437)

[2.3. Fonksiyonel Olmayan Gereksinimler 3](#_Toc168264438)

[2.4. Sistem Gereksinimleri 3](#_Toc168264439)

[3. Yazılım Mimarisi ve Tasarımı 3](#_Toc168264440)

[3.1. Yazılım Mimarisi 3](#_Toc168264441)

[3.1.1. Mimari Tarz ve Model 3](#_Toc168264442)

[3.1.2. Sistem Bileşenleri ve İlişkileri 3](#_Toc168264443)

[3.2. Yazılım Tasarımı 4](#_Toc168264444)

[3.2.1. Yazılım Tasarım Desenleri 4](#_Toc168264445)

[3.2.2. Kullanılan UML Diyagramları 4](#_Toc168264446)

[4. Proje Uygulama Detayları 7](#_Toc168264447)

[4.1. Kullanılan Teknolojiler ve Araçlar 7](#_Toc168264448)

[4.2. Arayüz Tasarımı 8](#_Toc168264449)

[4.3. Yazılım Modülleri ve Fonksiyonları 9](#_Toc168264450)

[4.4. Kodlama Standartları ve Yöntemleri 10](#_Toc168264451)

[5. Sonuçlar ve Tartışma 10](#_Toc168264452)

[5.1. Proje Sonuçları 10](#_Toc168264453)

[5.2. Hedeflere Ulaşma Durumu 10](#_Toc168264454)

[5.3. Karşılaşılan Zorluklar ve Çözüm Yolları 10](#_Toc168264455)

[5.4. Proje Çıktıları ve Kazanımları 11](#_Toc168264456)

# 1. Giriş

Bu proje raporu, C# ve Unity kullanılarak geliştirilen 2D platform oyunu üzerine hazırlanan çalışmayı detaylandırmaktadır. Geliştirilen oyunda, oyuncuların çeşitli tuzakları aşarak ve meyveleri toplayarak beş farklı bölümde ilerlemesi sağlanmaktadır. Her bölümde farklı zorluk seviyeleri ve oyun mekanikleri sunularak oyunculara heyecan verici bir oyun deneyimi yaşatılması amaçlanmıştır. Bu raporda, projenin amacı, kapsamı, hedefleri ve gerçekleştirilme süreci detaylı bir şekilde ele alınacaktır.

Projenin amacı, oyunculara eğlenceli ve zorlayıcı bir oyun deneyimi sunarken, aynı zamanda oyun geliştirme sürecinde C# ve Unity gibi popüler yazılım araçlarını kullanarak yazılım geliştirme becerilerimizi pekiştirmektir. Oyun, oyuncuların reflekslerini ve stratejik düşünme yeteneklerini test ederken, aynı zamanda bizlere de yazılım mimarisi, tasarım desenleri ve oyun mekanikleri konusunda değerli bilgiler sunmaktadır.

# 2. Proje Tanımı ve Gereksinimleri

## 2.1. Proje Tanımı

Geliştirilen 2D platform oyunu, oyuncunun çeşitli zorluklarla dolu beş farklı bölümde ilerlemesini amaçlayan bir yapıdadır. Oyunun temel amacı, oyuncunun her bölümde yer alan meyveleri toplayarak bitiş noktasına ulaşmasıdır. Oyuncu, her bölümde karşısına çıkan tuzaklardan kaçarak ve engelleri aşarak ilerlemektedir. Bölümler ilerledikçe zorluk seviyesi artmakta ve oyuncunun refleksleri ile stratejik düşünme yeteneği test edilmektedir.

## 2.2. Fonksiyonel Gereksinimler

**Bölüm Yapısı:** Oyun, toplamda beş bölümden oluşmaktadır ve her bölümün kendine özgü zorlukları ve tuzakları bulunmaktadır.

**Tuzaklar ve Engeller:** Her bölümde oyuncunun karşısına çıkan çeşitli tuzaklar ve engeller yer almaktadır. Bu tuzaklar oyuncunun ilerlemesini zorlaştırmakta ve dikkat gerektirmektedir.

**Meyve Toplama:** Oyuncu, her bölümde yer alan meyveleri toplayarak puan kazanmakta ve bir sonraki bölüme geçiş yapabilmektedir.

**Bölüm Geçişi:** Oyuncu, bölüm sonuna ulaştığında bir sonraki bölüme geçmektedir. Tüm bölümler tamamlandığında oyun sona ermektedir.

**Kullanıcı Arayüzü:** Oyun, kullanıcı dostu bir arayüze sahip olmalı ve oyuncuların oyuna kolayca başlayabilmesi, ilerleyebilmesi ve skorlarını takip edebilmesi sağlanmalıdır.

## 2.3. Fonksiyonel Olmayan Gereksinimler

**Performans:** Oyun, düşük donanımlı cihazlarda bile akıcı bir şekilde çalışabilmelidir.

**Kullanıcı Deneyimi:** Oyun, oyuncuya akıcı ve keyifli bir deneyim sunmalıdır. Kullanıcı arayüzü ve kontroller, sezgisel ve kullanıcı dostu olmalıdır.

**Genişletilebilirlik:** Oyun, ileride eklenebilecek yeni bölümler ve özellikler için genişletilebilir bir yapıya sahip olmalıdır.3. Yazılım Mimarisi ve Tasarımı

## 2.4. Sistem Gereksinimleri

**Geliştirme Ortamı:** Unity oyun motoru ve C# programlama dili kullanılmaktadır.

**Hedef Platformlar:** Oyun, Windows, macOS ve Linux işletim sistemlerinde çalışacak şekilde tasarlanmıştır.

**Donanım Gereksinimleri:** Oyunun düşük donanım özelliklerine sahip bilgisayarlarda bile çalışabilmesi için optimize edilmiştir.

# 3. Yazılım Mimarisi ve Tasarımı

## 3.1. Yazılım Mimarisi

### 3.1.1. Mimari Tarz ve Model

Bu 2D oyun için katmanlı bir mimari tarzı kullanıldı. Bu, kodun daha organize ve yönetilebilir olmasını sağlayacak ve farklı bileşenler arasındaki bağımlılıkları azaltacaktır. Katmanlar şunlardır:

* **Sunum Katmanı:** Bu katman, oyunun grafik kullanıcı ara yüzünden (GUI) sorumludur. Karakterler, ortamlar ve diğer görsel öğeler gibi oyun nesnelerini temsil eden görsel öğeleri içerir.
* **İş Mantığı Katmanı:** Bu katman, oyunun mantığından sorumludur. Karakter hareketleri, düşman etkileşimleri, puanlama sistemi ve diğer oyun kuralları gibi oyunun nasıl çalıştığını yönetir.
* **Veri Erişim Katmanı:** Bu katman, oyun verileriyle etkileşimden sorumludur. Oyun dosyalarını (örneğin, seviye tasarımları, karakter istatistikleri) yükler ve kaydeder ve oyun sırasında ihtiyaç duyulan verileri iş mantığı katmanına sağlar.

### 3.1.2. Sistem Bileşenleri ve İlişkileri

Sistemin aşağıdaki ana bileşenleri olacaktır:

* **Oyuncu Karakteri:** Bu bileşen, oyuncunun kontrol ettiği karakteri temsil eder. Hareket etme, zıplama ve düşmanlarla etkileşim kurma gibi yeteneklere sahiptir.
* **Tuzaklar:** Bu bileşenler, oyuncunun ilerlemesini engelleyen statik nesnelerdir. Platformlar, duvarlar ve çukurlar gibi engeller olabilir.
* **Kirazlar:** Bu bileşenler, oyuncunun toplayabileceği ve puan kazanabileceği nesnelerdir.
* **Oyun Motoru:** Bu bileşen, oyunun genel akışından sorumludur. Seviyeleri yükler, oyuncu girişini işler, oyun nesnelerini günceller ve oyunu çizer.

Bu bileşenler, aşağıdaki gibi etkileşime girecektir:

* Oyuncu karakteri engellere çarpıştığında, hasar alabilir veya ölebilir.
* Oyuncu karakteri bir kiraz topladığında, puan kazanır.
* Oyuncu karakteri bir seviyenin sonuna ulaştığında, bir sonraki seviyeye geçer.
* Oyun yöneticisi, oyun nesnelerinin davranışlarını ve oyunun genel akışını yönetir.

## 3.2. Yazılım Tasarımı

### 3.2.1. Yazılım Tasarım Desenleri

### 3.2.2. Kullanılan UML Diyagramları

#### 3.2.2.1 Use-Case diyagramı

**Seçenek 1:** Oyun Başlat

* Aktör, oyun başlatma seçeneğini seçer.
* Sistem, oyunun yükleme işlemini başlatır.
* Oyun yüklendikten sonra, sistem oyunu başlatır.
* Oyuncu oyunu oynamaya başlar.

**Seçenek 2:** Oyun Yükle

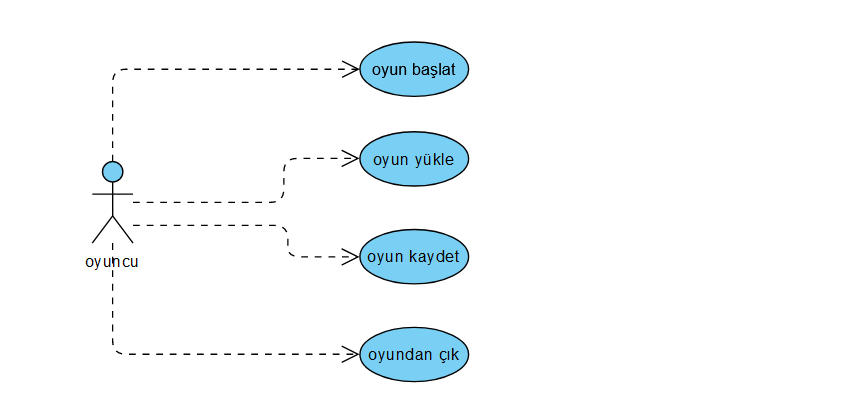
* Oyuncu, oyun kayıtlı yükleme seçeneğini seçer.
* Sistem, kayıtlı oyunun yükleme işlemini başlatır.
* Oyun yüklendikten sonra, sistem oyunu başlatır.
* Oyuncu oyunu oynamaya başlar.

**Seçenek 3:** Oyun Kaydet

* Oyuncu, oyun kaydetme seçeneğini seçer.
* Sistem, oyunun mevcut durumunu kaydeder.
* Oyuncu oyuna devam edebilir veya oyundan çıkabilir.

**Seçenek 4:** Oyundan Çık

* Oyuncu, oyundan çıkma seçeneğini seçer.
* Sistem, oyuncudan oyundan çıkmak isteyip istemediğini onaylar.
* Oyuncu çıkmayı onaylarsa, sistem oyunu kapatır ve oyuncuya ana menüye döner.



#### 3.2.2.2 Class Diyagramı

**PowerManager**, oyuncunun güçlendirmeleri almasını yönetir. Bu sınıf, oyuncunun güçlendirmeleri aldığı durumda **PlayerController** ile etkileşime geçer.

**TrapController**, oyuncunun tuzaklara yakalanmasını yönetir. Oyuncu bir tuzakla etkileşime girdiğinde, bu durum **PlayerController** üzerinden **TrapController** ile gerçekleşir.

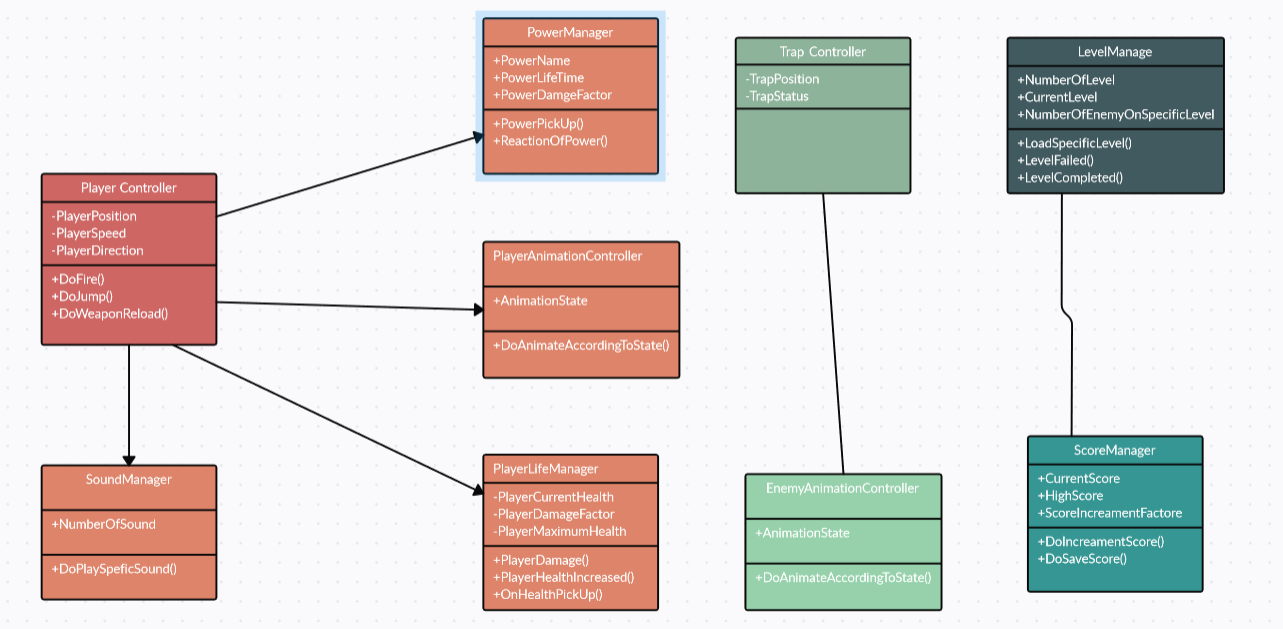
**LevelManager**, oyunun seviyelerini yönetir. Oyuncu seviyeler arasında ilerlediğinde, bu durum **PlayerController** üzerinden **LevelManager** ile gerçekleşir.

**ScoreManager**, oyuncunun puanını yönetir. Oyuncu puan kazandığında, bu durum **PlayerController** üzerinden **ScoreManager** ile gerçekleşir.

**SoundManager**, oyun seslerini yönetir. Oyuncunun eylemleri sırasında seslerin çalınması, yine **PlayerController** üzerinden **SoundManager** ile gerçekleşir.

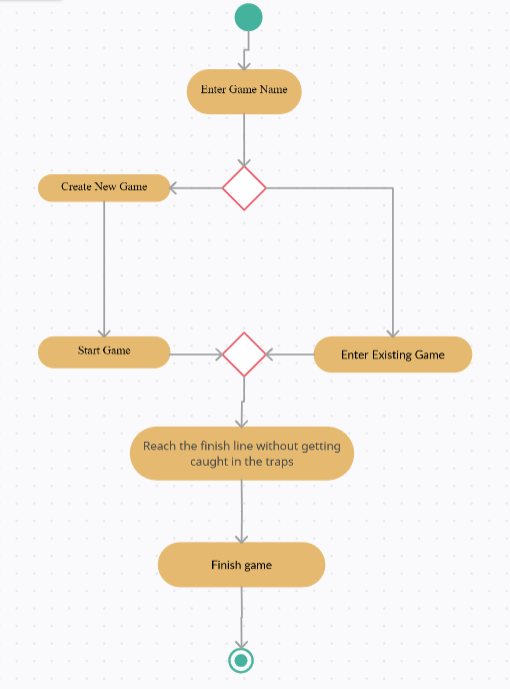
**PlayerLifeManager**, oyuncunun canını yönetir. Oyuncunun canı azaldığında veya arttığında, bu durum yine **PlayerController** üzerinden **PlayerLifeManager** ile gerçekleşir.

**EnemyAnimationController**, düşmanların animasyonlarını yönetir. Oyuncu bir düşmanla etkileşime girdiğinde, bu durum **PlayerController** üzerinden **EnemyAnimationController** ile gerçekleşir.



#### 3.2.2.3 Activity Diyagramı

Yeni bir oyun oluşturma ve oyunu tamamlayıp bitiş çizgisine ulaşma sürecini adım adım gösterir. İki ana dal, "Yeni Oyun Oluştur" ve "Var Olan Oyuna Gir" olarak ayrılır. Yeni bir oyun oluştururken, ilk olarak bir oyun adı seçilir. Son olarak, oyun başlatılır. Var olan bir oyuna girerken ise, öncelikle oyun adı seçilir ve ardından oyun başlatılır. Her iki dala da ortak olan adımlar arasında tuzaklardan kaçınma, bitiş çizgisine ulaşma ve oyunun bitirilmesi bulunur.



# 4. Proje Uygulama Detayları

## 4.1. Kullanılan Teknolojiler ve Araçlar

Proje geliştirme sürecinde kullanılan temel teknolojiler ve araçlar aşağıda detaylandırılmıştır:

**Unity:** Geliştirilen oyun Unity oyun motoru kullanılarak inşa edilmiştir. Unity, 2D ve 3D oyun geliştirme için güçlü bir platform sunar ve geniş bir araç seti ile oyun geliştiricilere esneklik sağlar.

**C#:** Oyunun mantığı ve işleyişi C# programlama dili kullanılarak kodlanmıştır. C#, Unity ile uyumlu, güçlü ve esnek bir programlama dilidir.

**Visual Studio:** Kodlama ve hata ayıklama işlemleri için Visual Studio entegre geliştirme ortamı (IDE) kullanılmıştır.

**Trello:** Proje yönetimi ve görev takibi için Trello kullanılmıştır. Bu araç, ekip içindeki iş bölümü ve proje ilerleyişinin takibi açısından oldukça faydalıdır.

## 4.2. Arayüz Tasarımı

Oyunun kullanıcı arayüzü (UI) tasarımı, oyuncunun kolayca oyun içi işlemleri yapabilmesi için kullanıcı dostu bir şekilde dizayn edilmiştir. Unity'nin UI araçları kullanılarak çeşitli menüler ve arayüz elemanları oluşturulmuştur.

metin, grafik, kırpıntı çizim, grafik tasarım içeren bir resim

Açıklama otomatik olarak oluşturuldu**Ana Menü:** Oyun başlatma, ayarlar ve çıkış seçeneklerini içerir.

**Oyun İçi Arayüz:** Oyuncunun sağlık durumu, topladığı meyve sayısı ve mevcut bölüm bilgilerini gösterir.

ekran görüntüsü, Çizgi film, çizgi film, Animasyon içeren bir resim

Açıklama otomatik olarak oluşturuldu**Skor Tablosu:** Oyuncuların en yüksek skorlarını ve liderlik tablosunu gösterir.

**Oyun Sonu Ekranı:** Oyuncunun bölümdeki performansını ve toplam skorunu gösterir.

metin, logo, yazı tipi, grafik içeren bir resim

Açıklama otomatik olarak oluşturuldu

## 4.3. Yazılım Modülleri ve Fonksiyonları

Oyun, çeşitli modüller ve fonksiyonlar kullanılarak modüler bir yapıda tasarlanmıştır. Bu modüller ve fonksiyonlar, oyun içi işlevleri gerçekleştirmek için bir arada çalışır.

**Oyun Yönetici Modülü (Game Manager):** Oyunun genel akışını kontrol eder, bölümleri yükler ve oyuncunun ilerlemesini takip eder.

**Karakter Kontrol Modülü (Player Controller):** Oyuncu karakterinin hareketleri, zıplama, koşma ve diğer etkileşimleri yönetir.

**Düşman ve Tuzak Modülü (Enemy and Trap Manager):** Bölümlerdeki düşmanların ve tuzakların davranışlarını kontrol eder.

**Ödül ve Puan Modülü (Reward and Score Manager):** Oyuncunun topladığı meyveleri ve kazandığı puanları hesaplar ve yönetir.

**Arayüz Modülü (UI Manager):** Kullanıcı arayüzü elemanlarını yönetir ve günceller.

**Ses ve Efekt Modülü (Audio Manager):** Oyun içi ses efektlerini ve müzikleri kontrol eder.

## 4.4. Kodlama Standartları ve Yöntemleri

Proje geliştirme sürecinde belirli kodlama standartları ve yöntemleri takip edilmiştir:

**Adlandırma Konvansiyonları:** Değişken, fonksiyon ve sınıf adlandırmaları için standart konvansiyonlar kullanılmıştır.

**Yorum Satırları ve Belgeler:** Kodun anlaşılabilirliğini artırmak için yeterli miktarda yorum satırı eklenmiştir. Özellikle karmaşık algoritmalar ve fonksiyonlar detaylı olarak açıklanmıştır.

**Modüler Yapı:** Kod, yeniden kullanılabilir ve yönetilebilir modüller halinde yazılmıştır.

**Versiyon Kontrol:** Proje dosyaları, Git versiyon kontrol sistemi kullanılarak yönetilmiştir. Bu sayede ekip içi iş birliği ve kod değişikliklerinin takibi sağlanmıştır.

# 5. Sonuçlar ve Tartışma

## 5.1. Proje Sonuçları

Projenin tamamlanmasıyla birlikte, belirlenen teknik hedeflere büyük ölçüde ulaşılmış ve proje, öngörülen zaman diliminde bitirilmiştir. Geliştirilen 2D platform oyunu, C# ve Unity kullanılarak inşa edilmiş olup, beş bölümden oluşan, çeşitli tuzaklar ve meyve toplama mekanikleri ile zenginleştirilmiştir. Projenin en önemli çıktılarından biri, yazılım mimarisi ve tasarımı konusundaki kapsamlı çalışmalarımızdır. Oyun, modüler ve genişletilebilir bir mimari ile tasarlanmış, bu da gelecekteki geliştirmelere olanak tanımaktadır.

## 5.2. Hedeflere Ulaşma Durumu

Projenin başlangıcında belirlenen hedefler, yazılım mimarisi ve tasarımı açısından büyük ölçüde gerçekleştirilmiş olup istenen oyuncu giriş ekranı hariç istenen seviyededir:

**Modüler Yapı:** Oyun, her biri belirli bir işlevi yerine getiren bağımsız modüller halinde tasarlanmıştır. Bu, kodun okunabilirliğini artırmış ve bakımını kolaylaştırmıştır.

**Genişletilebilirlik:** Projenin genişletilebilir bir yapıda olması hedeflenmiş ve bu hedefe başarılı bir şekilde ulaşılmıştır. Yeni bölümler ve özellikler eklemek için altyapı hazırlanmıştır.

**Performans Optimizasyonu:** Oyun, düşük donanımlı cihazlarda bile akıcı bir şekilde çalışacak şekilde optimize edilmiştir.

## 5.3. Karşılaşılan Zorluklar ve Çözüm Yolları

**Oyun Mekanikleri:** Oyuncu karakterinin hareketleri ve etkileşimleri, istenilen akıcılıkta değildi. Unity'nin fizik motoru daha etkin kullanılarak karakter hareketleri optimize edildi ve daha akıcı bir oyun deneyimi sağlandı.

**Ekip İçi Koordinasyon**: Ekip içi iletişim ve koordinasyon zaman zaman zorlayıcı oldu. Bu durumu aşmak için düzenli toplantılar ve proje yönetim araçları (Trello) kullanıldı.

## 5.4. Proje Çıktıları ve Kazanımları

Proje sonunda elde edilen çıktılar ve kazanımlar, yazılım mimarisi ve tasarımı açısından değerlidir:

**Tamamlanmış ve Çalışan Bir Oyun:** Oyun, belirlenen teknik gereksinimleri karşılayan ve oynanabilir bir durumda tamamlanmıştır. Yazılım mimarisi, modüler ve genişletilebilir bir yapıdadır.

**Gelişmiş Teknik Beceriler:** Proje sürecinde ekip üyeleri, C# ve Unity kullanarak yazılım geliştirme becerilerini pekiştirmiş ve yazılım mimarisi konusundaki bilgilerini artırmıştır.

**Modüler Yapı ve Genişletilebilirlik:** Oyun, ileride eklenebilecek yeni bölümler ve özellikler için genişletilebilir bir yapıda tasarlanmıştır. Bu, gelecekte yapılacak geliştirmeler için sağlam bir temel oluşturmuştur.

**Performans ve Optimizasyon:** Oyunun performansı optimize edilerek düşük donanımlı cihazlarda bile akıcı bir oyun deneyimi sağlanmıştır.

Bu sonuçlar ve kazanımlar, projenin başarılı bir şekilde tamamlandığını ve yazılım mimarisi ve tasarımı konusundaki hedeflere büyük ölçüde ulaşıldığını göstermektedir. Gelecekteki projelerde bu deneyimler ve kazanımlar önemli bir temel oluşturacaktır.