**KOCAELİ ÜNİVERSİTESİ BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ PROGRAMLAMA LABORATUVARI-I**

Sevilay Üngör ve Aslı Uzar

PROJE I : STRATEJİ OYUNU SAVAŞ MEKANİKLERİ RAPORU

**I . Özet:**

Bu proje İnsan İmparatorluğu ve Ork Lejyonu isimli iki ırkın ordularının savaştığı bir strateji simülasyonunun gelişmesini amaçlamaktadır.

Savaş ızgara tabanlı bir alanda gerçekleşir. Oyunda yer alan canavarlar ve kahramanlar karşı tarafın birliklerini etkileyecek güçlere sahiptir; aynı zamanda araştırma seviyeleri, sağlık, yorgunluk ve saldırı-savunma etkenleri de ele alınarak savaşın sonucuna varılmıştır.

Projede JSON dosyaları istenildiği gibi manuel olarak okutulmuş, verilen senaryolar indirilmiş, savaş mekaniği bu senaryolar üzerinden yürütülmüş ve simülasyonun adımları teker teker kaydedilmiştir. Görselleştirme, istenilen üzere başlangıç ve sonuçları içerir.

Anahtar Kelimeler: JSON dosyaları, senaryo, savaş mekaniği, görselleştirme, savaş etkenleri...

**II. Giriş:**

Günümüz oyun endüstrisinde, strateji oyunları oyunculara taktiksel düşünme, planlama ve kaynak yönetimi gibi yetenekler kazandırarak önemli bir yer tutmaktadır. Bu projede, İnsan İmparatorluğu ve Ork Lejyonu adındaki iki farklı ırkın, birbirleriyle savaştığı bir strateji oyun simülasyonu geliştirilmiştir. Projenin amacı, iki ordunun birbirine karşı savaşını, ızgara tabanlı bir haritada gerçekleştirmek ve savaşın seyrini etkileyen çeşitli faktörleri simüle etmektir.

Proje, kahramanlar ve canavarların birimlere sağladığı avantajlar ve araştırma seviyelerinin savaş üzerindeki etkilerini de içeren karmaşık bir savaş mekaniğine sahiptir. Kahramanlar belirli birimlere bonus sağlarken, canavarlar düşman birliklerine dezavantajlar getirir ve dost birliklere bonus sağlar. Aynı zamanda, her iki tarafın araştırma seviyeleri de savaşın sonucuna önemli bir etki yapar.

Simülasyonda, dış kaynaklı JSON dosyalarından gelen verilerle çalışır ve bu dosyalar elle okunarak işlenmiştir.

Birimlerin ve kahramanların özelliklerini içeren verileri doğru anahtarlarla eşleştirilmiş, programda bu verilere uygun hesaplamaları yapılmıştır. JSON dosyalarındaki verilerin sıralarının sabit olmaması nedeniyle, programın verilerin sırasına bağlı kalmaksızın doğru şekilde çalışmasına özen gösterilmiştir. Ayrıca, dosyalarda eksik veriler bulunması durumunda, program varsayılan değerler kullanarak eksik bilgileri tamamlayabilmesi sağlanmıştır.

Projenin bir diğer önemli unsuru, savaşın başlangıç durumu ve sonunda kalan birimlerin görselleştirilmesidir. Bu projede, ilk başta birliklerin ilk durumları gösterildikten sonra ekran kalan birliklerin durumunu göstermektedir.

Proje, istenilen şartlara uygun olarak tamamen C programlama dili kullanılarak geliştirilmiştir. Savaş simülasyonunun yürütülmesi için gerekli veriler, JSON dosyalarından manuel olarak okunmuş ve senaryoların internet üzerinden indirilmesi için C dilinin cURL kütüphanesi kullanılmıştır. İndirilen senaryo dosyalarına göre savaş mekaniği çalıştırılmakta ve her bir savaş durumu hesaplanmaktadır.

### A. JSON Dosyalarının Okunması ve Veri İşleme

Proje kapsamında dört ana JSON dosyası kullanılmıştır: unit\_types.json, heroes.json, creatures.json, ve research.json. Bu dosyalarda, birimlerin temel özellikleri, kahramanların ve canavarların sağladığı avantajlar, ve araştırma seviyeleri gibi savaş mekaniğini etkileyen faktörler tanımlanmıştır. JSON dosyaları, hazır kütüphaneler kullanılmadan manuel olarak ayrıştırılmış ve bu veriler program içerisinde uygun yapılarla işlenmiştir. Bu süreçte, dosyaların sırasına bakılmaksızın verilerin doğru anahtarlar üzerinden eşleştirilmesi sağlanmıştır.

Verilerin sırasının farklı olabileceği durumlarda, program doğru bilgiyi almak için anahtar-kelime eşleştirme yöntemlerini kullanmaktadır. Ayrıca, JSON dosyalarında eksik veri olması durumunda program, belirli varsayılan değerler kullanarak veri bütünlüğünü sağlamaktadır. Örneğin, birimlerin kritik\_sans değerleri dosyada belirtilmemişse, program bu değeri varsayılan olarak %0 kabul eder.

4 JSON dosyasının okunmasında aynı mantık kullanılmıştır. Bir struct yapısı tanımlanmış ve dosyada okunması istenen özellikler için (char veya int vs.) veri yapıları verilmiştir. Daha sonra dizi tanımı ve sayaç verilerek bu verilerin sırayla ve doğru şekilde okunması sağlanmıştır. Ardından bir fonksiyon yazılarak bu verilerin en başta 0 olarak tanınması sağlanmıştır. Bu sayede her bir birim okunduğunda başlangıç değeri 0 olur. Bunun ardından bir *set\_name* fonksiyonu verilerek birim adı boş bir karakterle (“ “) doldurulmuştur.

Daha sonrasında *read\_json* fonksiyonları gelir ki bu fonksiyonlar JSOn dosyalarını okuyan asıl fonksiyonlardır. En başa projede istendiği şekilde dosya açılmazsa hata verecek kod yazılmıştır. Fonksiyon ancak dosya açılırsa devam eder. Daha sonra *line ve count* tanımları verilmiştir. *Line;* dosyayı satır satır okuma yapmamıza yarayan karakter dizisidir. *Count* ise okunan birimin ve satırın sayısını çıkarmaya yarar. Ardından bir while döngüsü içinde C dilinin fonksiyonlarından yararlanılarak dosya okutulur.

### **B. Senaryo İndirme**

Senaryoların C dilinde cURL kütüphanesi kullanılarak indirilmesi sağlanmıştır. Bu süreçte, belirli URL'lerden JSON formatında senaryo dosyaları çekilmekte ve proje dizinine kaydedilmektedir. Kullanıcı, bu senaryo dosyalarını seçerek simülasyonu başlatabilir ve senaryonun içerdiği verilere göre savaş mekaniği hesaplanır. cURL kütüphanesi, HTTP üzerinden veri çekmeyi sağlayarak projenin senaryo dosyalarına kolayca erişimini sağlamaktadır.

### **C. Savaş Mekaniği ve Hesaplamalar**

Savaş mekaniği, indirilen dosyanın sağladığı senaryo verilerine göre dinamik olarak çalışır. Her iki tarafın birimleri arasında yapılan savaş hesaplamaları, çeşitli savaş etkenlerini dikkate alarak gerçekleştirilir. Kazanan tarafın belirlenmesinde dikkate alınan temel kriterler şunlardır:

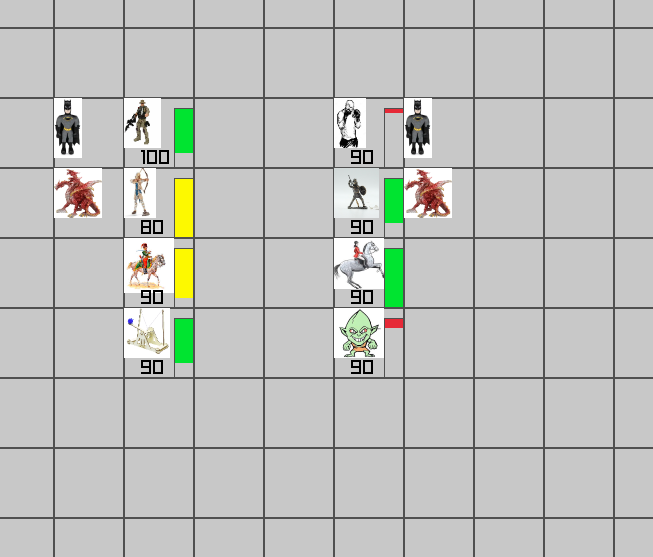
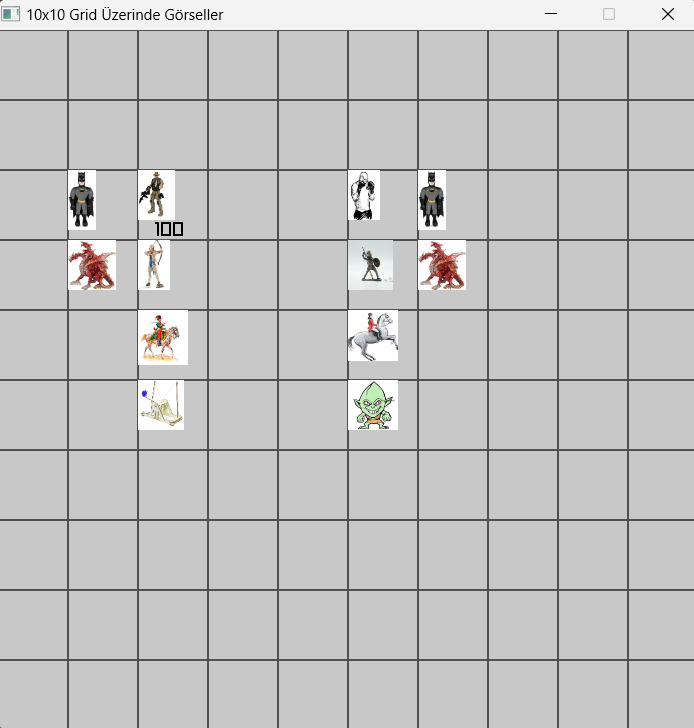
1. **Saldırı ve Savunma Güçleri:** Birimlerin saldırı ve savunma kapasiteleri, savaş sırasında verdikleri ve aldıkları hasarı belirler. Bu değerler, birimlerin türüne ve sayısına bağlı olarak hesaplanmıştır. Her bir birim türü, belirlenen saldırı gücüne göre rakip birimlere zarar verir ve savunma gücüne göre gelen hasarı karşılar.
2. **Kritik Vuruşlar:** Her bir birimin, savaş sırasında belirli bir ihtimalle kritik vuruş yapma hakkı bulunmaktadır. Kritik vuruşlar, birimlerin belirli bir oranda daha fazla hasar vermesini sağlar. Her birimin kritik vuruş ihtimali ve bu vuruşlardan elde ettiği bonus hasar farklı olacak şekilde ayarlanmıştır. Bu bonuslar, savaşın seyrini değiştirebilecek önemli faktörlerden biridir.
3. **Yorgunluk:** Savaşın uzun sürmesi durumunda birimler yorgun düşer ve bu durum, saldırı ve savunma kapasitelerinde düşüşlere neden olur. Yorgunluk etkisi, savaşın ilerleyen turlarında birimlerin performansını olumsuz yönde etkiler ve her bir turun sonunda birimlerin güçlerinde azalma yaşanır.
4. **Sağlık Durumu:** Her birim türü, belirli bir toplam can puanına sahiptir. Savaş sırasında birimlerin aldıkları hasar, sağlık puanlarının azalmasına yol açar. Bir tarafın tüm birimlerinin toplam sağlık puanı sıfıra ulaştığında, o taraf mağlup sayılır ve savaş sona erer.
5. **Canavar ve Kahraman Bonusları:** Kahramanlar ve canavarlar, belirli birimlere veya tüm ordulara ek saldırı veya savunma bonusları sağlayarak savaşın seyrini etkiler. Bu özel birimler, müttefiklerine sağladıkları desteklerle savaşın dengesini değiştirebilir. JSON dosyalarında tanımlanan bonuslar, savaş sırasında uygun birimlere eklenerek performans artışları sağlanmıştır.
6. **Araştırma Seviyeleri:** Tarafların gerçekleştirdikleri araştırmalar, birimlerin savaş sırasında çeşitli avantajlar elde etmesini sağlar. Araştırma seviyeleri arttıkça, birimlerin saldırı ve savunma güçlerinde belirli oranlarda artışlar meydana gelir. Bu bonuslar, tarafların stratejik üstünlük sağlamasına olanak tanır ve savaşın sonucunu doğrudan etkiler.

Bu kriterlerin her biri, savaşın sonucunu etkileyen önemli faktörlerdir. Yapılan hesaplamalar sonucunda, taraflardan biri tüm birimlerini kaybettiğinde savaş sona erer ve diğer taraf galip ilan edilir. Savaşın her aşaması, adım adım bir dosyaya kaydedilerek sürecin izlenebilirliği sağlanmıştır.

### **D. Görselleştirme**

Projenin görselleştirilmesi, C dilinin Raylib kütüphanesi kullanılarak gerçekleştirilmiştir. Savaşın başlangıç ve bitiş durumlarının görsel olarak sunulabilmesi için bir ızgara taban çizilmiş ve birimleri temsil eden grafikler eklenmiştir. Her birim türü, belirli renkler ve simgelerle temsil edilmiştir. Birimlerin sağlık durumları, görsel bir bar ile gösterilmekte ve savaş sonrasında hayatta kalan birimler ve sağlık durumları da haritada görselleştirilmektedir.

Görselleştirme aşamasında, kullanıcıya savaşın başlangıç durumu ve savaş sonrası kalan birimlerin konumları sunulmuştur. Savaş sırasında birimlerin hareketlerinin ve anlık değişikliklerin gösterimi isteğe bağlı olarak yapılmıştır. Bu, savaşın dinamiklerini görsel olarak analiz etmek isteyen kullanıcılar için ek bir seçenek olarak sunulmuştur.



### **E. Hata Yönetimi ve Kullanıcı Bilgilendirme**

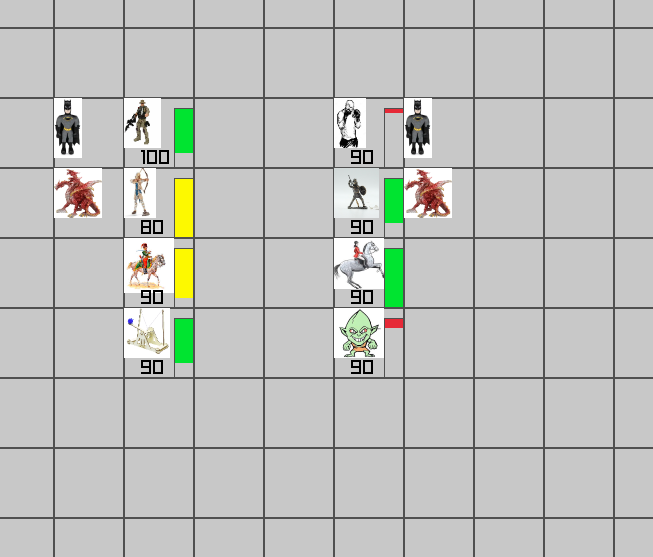
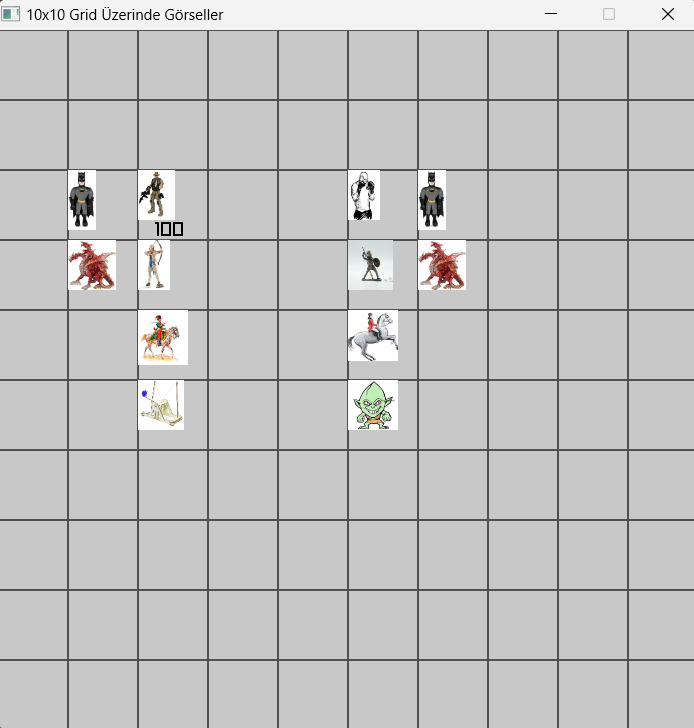
Programın JSON dosyalarına erişimi sırasında, dosya konumlarının doğruluğu ve dosya isimlerinin uygun olup olmadığı kontrol edilmiştir. JSON dosyalarının proje dizinindeki 'Files' klasöründe yer aldığı varsayılmıştır. Eğer dosya belirtilen dizinde bulunamazsa veya isimler doğru değilse, kullanıcıya açıklayıcı bir hata mesajı gösterilir. Örneğin, "Dosya 'Files/unit\_types.json' bulunamadı. Lütfen dosyanın doğru isimde ve doğru dizinde olduğundan emin olun." şeklinde bir hata mesajı ile kullanıcı bilgilendirilir.

## **IV. DENEYSEL SONUÇLAR**

### **A. Deneyin veya Simülasyonun Tanımı:**

Bu çalışmada, iki farklı ırkın (İnsan İmparatorluğu ve Ork Lejyonu) savaştığı bir strateji oyunu simülasyonu geliştirilmiştir. Simülasyon, C programlama dili ile yazılmış olup, savaş mekaniklerinin işleyişini ve sonuçlarını analiz etmek amacıyla belirli senaryolar kullanılarak yürütülmüştür. Senaryo verileri, JSON formatında tanımlanmış ve manuel olarak işlenmiştir. Simülasyon sırasında aşağıdaki adımlar ve araçlar kullanılmıştır:

* **Simülasyon Yazılımı**: Proje, tamamen C dilinde yazılmış olup, JSON dosyalarını okumak için manuel ayrıştırma teknikleri kullanılmıştır. JSON dosyaları cURL kütüphanesi aracılığıyla indirildikten sonra, birimlerin savaş parametreleri (saldırı gücü, savunma gücü, sağlık durumu, kritik vuruş oranı vb.) program tarafından işlenmiştir.
* **Test Senaryoları**: Farklı savaş senaryoları, JSON dosyaları aracılığıyla programa sağlanmış ve bu dosyalarda her iki tarafın başlangıç birimleri, kahraman ve canavar bonusları ile araştırma seviyeleri tanımlanmıştır. Örneğin:
  + **unit\_types.json**: İnsan ve Ork birimlerinin saldırı, savunma ve sağlık gibi özelliklerini içerir.
  + **heroes.json**: Kahramanların hangi birimlere nasıl bonuslar sağladığını belirtir.
  + **creatures.json**: Canavarların savaş sırasında sağladığı avantajlar ve karşı tarafa verdikleri dezavantajları tanımlar.
  + **research.json**: Araştırma seviyelerinin kritik vuruş ve savunma güçlerine etkisini içerir.
* **Test Koşulları ve Parametreler**:
  + Simülasyon, bize verilen ve cURL kütüphanesi yardımıyla indirilen senaryolarda başlar.
  + Her bir senaryoda farklı sayılarda ve türlerde birimlerin savaştığı durumlar incelenmiş, kahramanlar ve canavarların etkisi test edilmiştir.
  + **Örnek Senaryo**: Bir örnek senaryoda, İnsan İmparatorluğu'nun 200 piyade birimi ve 50 süvari birimi ile Ork Lejyonu'nun 300 okçu ve 30 trol birimine karşı savaştığı bir durum tanımlanmıştır. Senaryoda İnsan İmparatorluğu'nun bir kahramanı piyadelere %10 saldırı bonusu sağlarken, Ork Lejyonu'nun bir canavarı rakip süvarilere %15 savunma azaltma etkisi uygulamaktadır.
* **Simülasyon Akışı**: Simülasyon başladığında, program her iki tarafın birimlerini sırayla karşılaştırarak saldırı ve savunma güçlerini hesaplar. Kritik vuruşlar, yorgunluk ve diğer etkiler dikkate alınarak her adımda hasar hesaplaması yapılır. Saldırı sırasında hasar gören birimlerin sağlık durumu güncellenir ve ölen birimler çıkartılır. Simülasyon, taraflardan birinin tüm birimlerinin yok olmasıyla sona erer.



### **V. Sonuç**

Bu çalışma kapsamında, C dili kullanılarak geliştirilen strateji oyunu simülasyonu ile iki farklı ırkın (İnsan İmparatorluğu ve Ork Lejyonu) savaş mekanikleri analiz edilmiştir. JSON dosyalarından okunan veri ve senaryolar doğrultusunda, birimlerin saldırı ve savunma güçleri, kahraman ve canavar etkileri, araştırma seviyeleri ve savaşın gidişatını belirleyen diğer faktörler ayrıntılı olarak işlenmiştir. Simülasyon sırasında elde edilen veriler, savaşın belirleyici unsurlarını ve hangi etkenlerin zafer ya da mağlubiyete yol açtığını açıkça göstermiştir.

Çalışmanın temel bulguları arasında, kahraman ve canavar bonuslarının savaşın sonucuna önemli ölçüde etki ettiği, yorgunluğun uzun süren savaşlarda birim performansını belirgin şekilde düşürdüğü ve araştırma seviyelerinin stratejik avantaj sağladığı görülmüştür. Kritik vuruşlar gibi savaşın akışını doğrudan etkileyen faktörler, birimlerin etkinlik düzeyini artırmış veya azaltmıştır. Bu sonuçlar, savaş stratejilerinin nasıl optimize edilebileceğine dair önemli bilgiler sunmaktadır.

Simülasyonun görselleştirme kısmında, savaşın başlangıcı ve bitiş durumu başarılı bir şekilde temsil edilmiş, savaş sırasında yaşanan değişimlerin izlenmesi ise isteğe bağlı olarak bırakılmıştır. Görselleştirme sayesinde, savaşın hangi aşamada hangi tarafın avantaj sağladığı daha net bir şekilde ortaya konmuştur. Izgara tabanda verilmesi birimlerin konumlarını görmemize ve savaşın gidişatını öngörmemizi sağlamıştır. Ayrıca verilen can barlarıyla bir birimin ne kadar canı kaldığını görebilmemiz sağlanmıştır.

Sonuç olarak, geliştirilen strateji oyunu simülasyonu, belirlenen senaryolara göre taraflar arasındaki savaşın sonucunu objektif bir şekilde analiz etmeye olanak sağlamıştır. Elde edilen sonuçlar, benzer çalışmalar için referans niteliğinde olup, farklı senaryoların denenmesi ve savaş mekaniklerinin daha da detaylandırılması için temel bir yapı sunmaktadır. Gelecekte, daha karmaşık stratejik unsurlar ve farklı ırkların eklenmesiyle simülasyonun çeşitlendirilmesi mümkündür. Bu çalışma, strateji oyunlarının algoritmik temelini anlamak ve geliştirmek için önemli bir katkı sağlamıştır.

### **VI. Yazar Katkıları**

Bu proje çalışması, 2 ekip üyesi tarafından farklı sorumlulukların paylaşılmasıyla yürütülmüştür:

* Sevilay Üngör: Projenin genel koordinasyonunu yapmış, JSON dosyalarının manuel olarak okunması ve veri işleme algoritmalarının geliştirilmesi üzerine çalışmıştır. Savaş mekaniklerinin tasarlanmasında ve birimlerin savaş dinamiklerinin simülasyonunda temel katkılar sağlamıştır. Savaş senaryolarının cURL kullanılarak indirilmesi ve JSON dosyalarının doğru formatta düzenlenmesi konularında çalışmıştır. Ayrıca, araştırma seviyelerinin simülasyona etkilerinin değerlendirilmesi ve farklı senaryoların test edilmesinde önemli katkılar sağlamıştır.
* Aslı Uzar: Savaş mekaniklerinin tasarlanmasında ve birimlerin savaş dinamiklerinin simülasyonunda temel katkılar sağlamıştır. Görselleştirme kısmında, Raylib kütüphanesi kullanılarak savaş alanının ve birim hareketlerinin grafiksel olarak gösterilmesini gerçekleştirmiştir. Rapor yazımını ve düzenlenmesini üstlenmiştir.

### **VII. Kaynakça**

1. A. Mehra ve T. Smith, *Game Programming Patterns*, Addison-Wesley, 2020.
2. J. R. Lakos, *Large-Scale C++ Software Design*, Addison-Wesley Professional, 2019.
3. R. Lippman, S. J. Menezes, ve K. Scandariato, "The role of JSON in modern software development," *Software Development Journal*, vol. 35, no. 3, pp. 98-105, 2021.
4. "Raylib Documentation," Raylib Official Documentation, erişim: <https://www.raylib.com/>.
5. S. B. Jones ve P. Zhang, "Simulation Techniques for Strategy Games," *International Journal of Game Development*, vol. 14, no. 2, pp. 142-158, 2022.
6. "cURL: Transfer data from or to a server," cURL Documentation, erişim: <https://curl.se/docs/>.