T.C.
FIRAT ÜNİVERSİTESİ
YAZILIM MÜHENDİSLİĞİ BÖLÜMÜ

UNİTY ORTMAINDA PCG VE AES ALGORİTMALARI KULLANAN FLAPPY BEARD OYUNU

Baraa KALAAJI – Yousif AL-EZZI

NİSAN - 2022

T.C.
FIRAT ÜNİVERSİTESİ
YAZILIM MÜHENDİSLİĞİ BÖLÜMÜ

Başlığı:	Unity Ortmaında PCG ve AES Algoritmaları Kullanan Flappy Beard Oyunu
Yazarı:	Baraa KALAAJI, Yousif AL-EZZI
Proje Danışmanı:	Doç. Dr. Fatih ÖZKAYNAK
Teslim Tarihi:	29.Nisan.2022

İÇİNDEKİLER

Sayfa

ÖZET

Unity Ortamında PCG Ve AES Algoritmaları Kullanan Flappy Beard Oyunu

Baraa KALAAJI, , Yousif AL-EZZI

FIRAT ÜNİVERSİTESİ
Yazılım Mühendisliği Bölümü

Siber güvenlik kavramı teknolojik çağında yaşayan insanların hayatlarını şekillendiren ve şirketlerin yapısından ayrılmaz bir parça haline gelmiştir. Bunun için yüksek verimlilik ve güvenlik standardlarına sahip olan PCG ve AES algoritmaları Unity gibi oyun geliştirme ortamında gerçekleştirmek çok yüksek bir önem taşımaktadır.

Projede PCG algoritması kullanarak yapılacak random sayı üretmesi, random sayı üretimi yapan varsayılan Random kütüphanesine göre daha kaliteli ve çok daha güvenli yapabileceği var sayılmıştır

Unity projeleri kapsamında yer alabilecek kullanıcı verileri (isim, soyad, skor, v.b.) AES (Advanced Encryption Standart) algoritması kullanarak yüksek güvenlilirlik ve güvenlik standartları göz önünde bulundurarak saklanabileceğini de var sayılmıştır.

C# tabanlı Unity oyun geliştirme platformu içinde algoritmaların bellek ve alan performansı analiz edilmiştir. Unity, orta ve az karmaşık yapılara sahip olan RNG (Random Number Generator) ile random sayı üretmesi, bellek ve alan açısından yüksek performans ile yapabilmesi gözlenmiştir.

Proje, Unity ortmaında PCG ve AES algoritmaları diğer RNG'ler ve şifreleme algoritmaları ile mukayese ederek yüksek performans ile çalışabileceği tespit edilmiştir.

Anahtar Kelimeler: Random Sayı Üreteci, PCG Algoritması, AES Algoritması, Güvenli Oyun Geliştirme

ŞEKİLLER LİSTESİ

Sayfa

1. GİRİŞ

Bir çok toplantı için toplantının refah düzeyi o toplantıdaki bulunan insanların bilinç seviyesi ile doğru orantılıdır. Özellikle son elli yıllık süreçte hayatımızın ayrılmaz bir parçası haline gelen yeni yazılımsal ve donanımsal teknolojileri, siber zorbalık, dolandırıcılık ve kimlik hırsızlıklarına yol açmıştır.

Dünya genelinde oyuncu sayısının 2020 itibariyle 2,6 milyara ulaşmıştır. Akademik çalışmalara ve Pazar analizlerine göre bu rakamın 2021'de 2,7 milyar olacağı öngörülmektedir. Aynı zamanda dünya genelinde E-Sporu takip eden ve "E-Spor hayranı" olarak nitelendirilen kitlenin 2020 itibariyle 518 milyona ulaştığı ve bu rakamın 2021'de 580 milyona ulaşmasının beklenmektedir. [01]

Türkiye'de ise International Data Corporation'ın (IDC) raporuna göre, kişisel bilgisayar satışlarının 2018'in ilk yarısında bir önceki yılın aynı dönemine kıyasla yüzde 9,3 artmıştır ve farklı elektronik platformlarda oyun oynayan bireylerin sayısı 30 milyondan fazla olmuştur. [02]

Yukarıda belirlenen her iki gerçek göz önünde bulundurarak oyun geliştirme dünyasında güvenlik ve bilgi koruma standardları ve yöntemleri sürekli olarak geliştirilmesi gerektiğini anlaşılmaktadır.

Proje, oyunların hızlı random sayı üretimine ihtiyacı göz önünde bulundurarak en verimli ve güvenli RNG keşfetmeye hedeflemektedir. Aynı zamanda oyunların içinde kullanılabilecek kullanıcıların verileri AES algoritması ve 3-Tier yazılım modeli gerçekleştirerek oyun geliştiricilere güvenli bir geliştirme yöntemi tasarlamaktadır.

2. GEREKSİNİM TASARIMI VE ANALİZİ

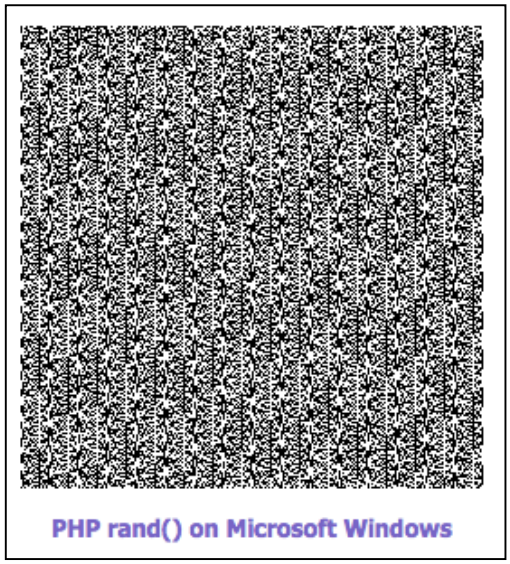
Yazılım mühendisliği projelerinin başlangıç noktası olarak bilinen Gereksinim Tasarımı ve Analizi aşaması yazılım projenin en önemli aşamasıdır. Bu aşamada yapılan araştırma ve analiz çalışmaları projenin potansiyel başarısı ve gerçek hayatta etkinliği hakkında ön fikir oluşmaktadır.

2.1. Mevcut Sistem İncelemesi

Mevcut halde random sayılar üretebilen birden fazla metot ve algoritma bulunmaktadır. Bu araçların ve metotların bazılarına analiz ve inceleme yaparak bu proje kapsamında kullanılacak algoritma ve yöntemler ile mukayese edilmesini sağlanacaktır.

2.1.1. Random Kütüphanesi

Java, C#, PHP, v.b. programlama dillerinde rastgele sayı üretmek için kullanılan varsayılan kütüphanedir. Random kütüphanesi CPU saatin tick değeri alarak rastgeleliği sağlamaktadır. Bu değere kolay bir şekilde erişilebilir olduğundan Random kütüphanesi yeterli güvenlik seviyesi karşılamaz ancak hızlı ve verimli bir şekilde sayı üretme özelliğine sahip olması genel kullanımlar için ideal olması sağlamaktadır.



Şekil.2.1 Random Kütüphanesinin Örütüleri

2.1.2. XOR Shift

Shift-register üreteçleri olarak da adlandırılan XOR Shift rasgele sayı üreteçleri, George Marsaglia tarafından keşfedilen bir pseudo rasgele sayı üreteci sınıfıdır. Bu üreteçler, aşırı seyrek polinomlar kullanmadan yazılımda özellikle verimli bir uygulamaya izin veren doğrusal geri

besleme kaydırmalı yazmaçların (LFSR'ler) bir alt kümesidir. Kendilerinin bit-kaydırılmış versiyonuna sahip bir sayının dışlayıcı veya tekrar tekrar alarak sıralarındaki bir sonraki sayıyı üretirler. Bu, onları modern bilgisayar mimarilerinde son derece verimli bir şekilde yürütmelerini sağlar, ancak bir donanım uygulamasında verimliliğe fayda sağlamaz. Tüm LFSR'ler gibi, uzun bir süre elde etmek için parametrelerin çok dikkatli seçilmesi gerekmektedir.

XOR Shift algoritmasının farklı versiyonlar ile farklı güvenlik seviyeleri temin etmektedir ancak XOR Shift algoritması ile üretilen random sayıların diğer RNG'ler ile tahmin etmesi daha kolay olması ve üretimin daha yavaş olması XOR Shift algoritmasının oyun uygulamaları için uygun olmadığını göstermektedir. Aynı zamanda bu yöntemin biraz yavaş olması, random sayı üretiminden sonra gerçekleştirilecek şifrelemeye bir engel oluşturacaktır.

2.1.3. Mevcut Sistemin Değerlendirilmesi

Mevcut sistem incelemesin bölümünde görüldüğü gibi mevut halde bulunan bazı RNG'ler sınırlıdır ve negatif yönlere de sahiptir. Bunun için bu çalışma kapsamında önerilen sistem bu negatif yönleri çözmeye hedeflemektedir.

2.2. Önerilen Sistem Analizi

Mevcut sistem ve araçları incelendiği zaman, sistemin eksikleri gözükmektedir. Bunun için bu çalışma kapsamında önerilen sistemin analizini yaparak mevcut sistemdeki bulunan dezavantajları nasıl ortadan kaldıracağı gösterilecektir.

2.2.1. Permuted Congruential Generator (PCG)

PCG, rastgele sayı üretecinin istatistiksel özelliklerini iyileştirmek hedefiyle çıkış permütasyon (Output Permutation Function [03]) işlevi uygulayan, 2014 yılında geliştirilen bir psödo-rastgele sayı üretme algoritmasıdır. Küçük, kolay uygulanabilirliği ve kısa çalışma zamanı ile mükemmel istatistiksel performans elde etmektedir.

PCG algoritmanın kolay uygulanabilirlik özelliği, oyun geliştiriciler için kullanılabilir ve güvenilir bir yöntem haline yol açmaktadır. Buna ek olarak Unity gibi oyun geliştirme platformun kullandığı C# programlama dilinde basit ve soyut implementasyonu, Unity kullanıcılar için varsayılan Random kütüphanesi yerine sürekli ve kolay bir şekilde kullanılabilen bir alternatif haline getirmektedir.

PCG algoritmasının kolay uygulanabilirliği ve diğer bazı RNG'ler ile karşılaştırarak hızlı random sayı üretme kabiliyeti, güvenlik seviyesine olumsuz bir etki oluşmamaktadır ve PCG algoritması, XORShift 32/64, RanQ ve Minstd (LCG) gibi bazı ünlü RNG'ler ile güvenlik

seviyesinde mukayese edildiği zaman PCG algoritmasının ürettiği random sayıların kalitesi daha yüksek ve tahmin edilme oranı daha düşük olduğunu görülmektedir.

2.2.2. Advanced Encryption System (AES)

AES (Advanced Encryption Standard; Gelişmiş Şifreleme Standardı), elektronik verinin şifrelenmesi için sunulan bir standarttır. Amerikan hükûmeti tarafından kabul edilen AES, uluslararası alanda da defacto şifreleme (kripto) standardı olarak kullanılmaktadır. DES'in (Data Encryption Standard - Veri Şifreleme Standardı) yerini almıştır. AES ile tanımlanan şifreleme algoritması, hem şifreleme hem de şifreli metni çözmede kullanılan anahtarların birbiriyle ilişkili olduğu, simetrik-anahtarlı bir algoritmadır. AES için şifreleme ve şifre çözme anahtarları aynıdır.

AES, Değiştirme-Karıştırma (Substitution-Permutation) olarak bilinen tasarım temeline dayanır. Öncülü DES ise Feistel yapıda tasarlanmış bir algoritmadır.

AES'in hem yazılım hem de donanım performansı yüksektir. 128-bit girdi bloğu, 128, 192 ve 256 bit anahtar uzunluğuna sahiptir. AES'in temel alındığı Rijndael ise 128-256 bit arasında 32'nin katı olan girdi blok uzunluklarını ve 128 bitten uzun anahtar uzunluklarını desteklemektedir. Dolayısıyla, standartlaşma sürecinde anahtar ve girdi blok uzunluklarında kısıtlamaya gidilmiştir.

AES algoritmasının yüksek güvenilirliği ve güvenliği, programlama dillerini geliştiricileri tarafından varsayılan kütüphaneler şeklinde programlama dillerine eklemelerine neden olmuştur. C# tabanlı Unity platformunda ise *System.Security.Cryptography* namespace altında *AesManaged* sınıfında yer almaktadır. *AesManaged* sınıfı kullanarak AES algoritmasında kullanılan Şifre, Metin ve IV belirlenebilmektedir. *AesManaged* sınıfın kolay kullanabilmesi ve küçük kod boyutu ve çok yüksek güvenlik seviyesinden dolayı şifreleme için ideal bir çözüm sunmaktadır.Sistem Tasarımı

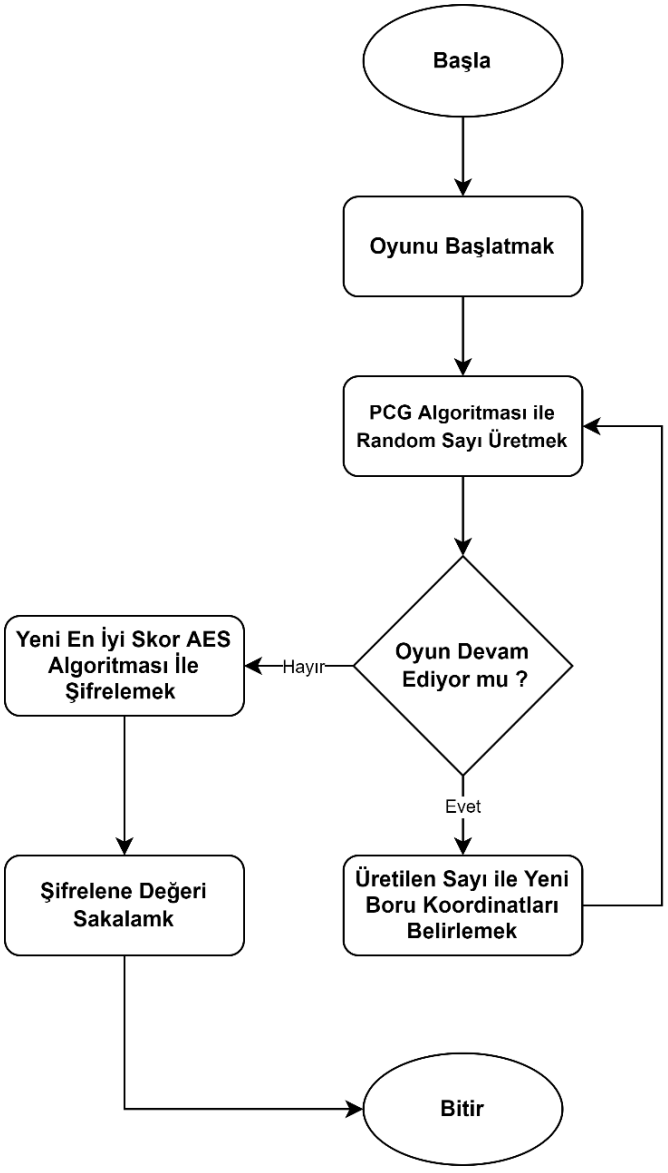
2.3. Genel Tasarım Bilgileri

2.3.1. İşlevsel Belirtimler

İşlevsel belirtimlerin tasarımı, sistemin ve yazılımın temel görevini tanımlayarak belirlenmektedir. Sistem, oyun geliştirme dünyasında yüksek güvenlik seviyesi sağlayabilen ve kullanıcıların dijital varlıklarını ve verilerini korumak için etkin ve hızlı bir yöntem sunmaktadır. Bu yöntem PCG random sayı üreteci ve AES şifreleme protokolu kullanarak tasarlanmaktadır.

2.3.2. Sistem Mimarisi

Şekil.2.2 Sistem Mimarisi



Şekil.2.3 Sistem Mimarisi – Akış Şeması

2.3.3. Testler

Genel hatlarıyla test iki aşamada gerçekleştirilmiştir.

Kod Aşaması: Yöntem, projeden ve oyun geliştirme platformundan ayrı olarak abstract kod halinde test edilmiştir.

Proje Kullanım Aşaması: Ayrı olarak test edilen kod parçaları (PCG ve AES kodları) Unity platformunda gerçek bir oyun üzerinde test edilmiştir.

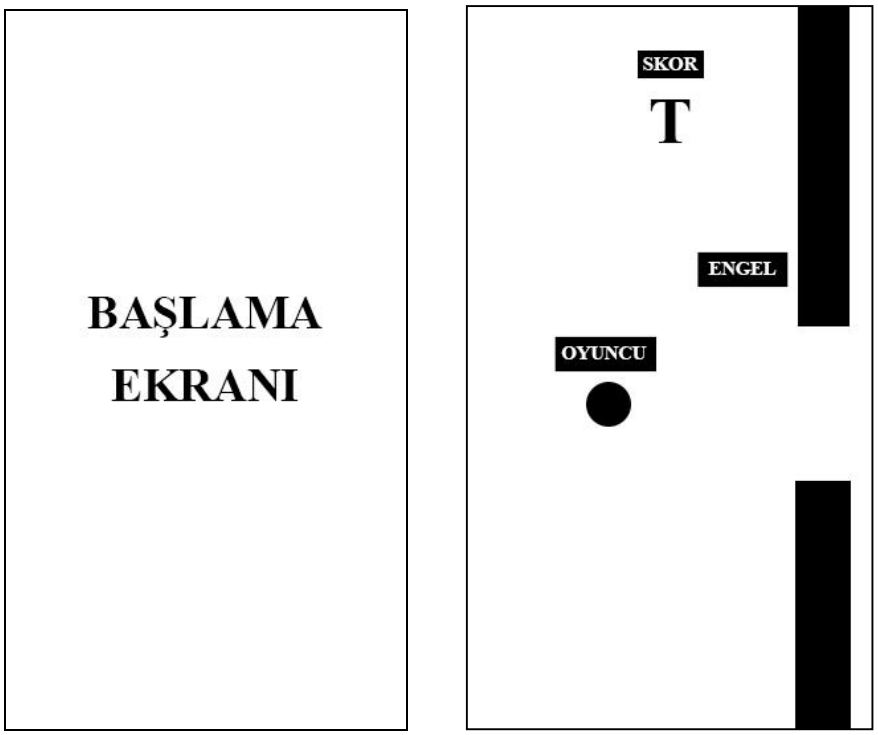
2.4. Süreç Tasarımı

Süreç tasarımı yapar iken uygulamanın içerisinde kullanılacak farklı modüller detaylı bir şekilde yazılmaktadır. Ancak bu çalışma kapsamında gerçekleştirilecek proje içerisinde kullanıcı ve basit veri tabanı modülü hariç yönetici modülü v.b. gibi farklı bir modül bulunmadığından dolayı sadece bu iki modül incelenmektedir.

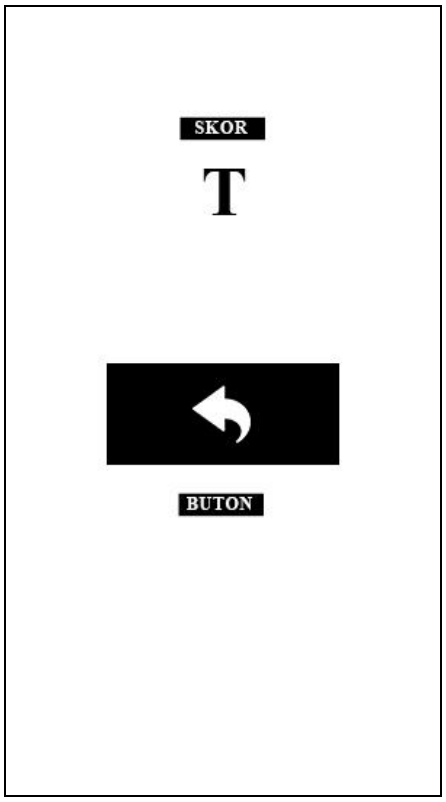
2.4.1. İşlev

Kullanıcının sisteme müdahale edebileceği ekrana erişmesi ve uygulamayı kullanabilmesi için aşması gereken bir modüldür.

2.4.2. Kullanıcı Arabirimi



Şekil.2.4 Kullanıcı Arabirimi - 1 ve 2

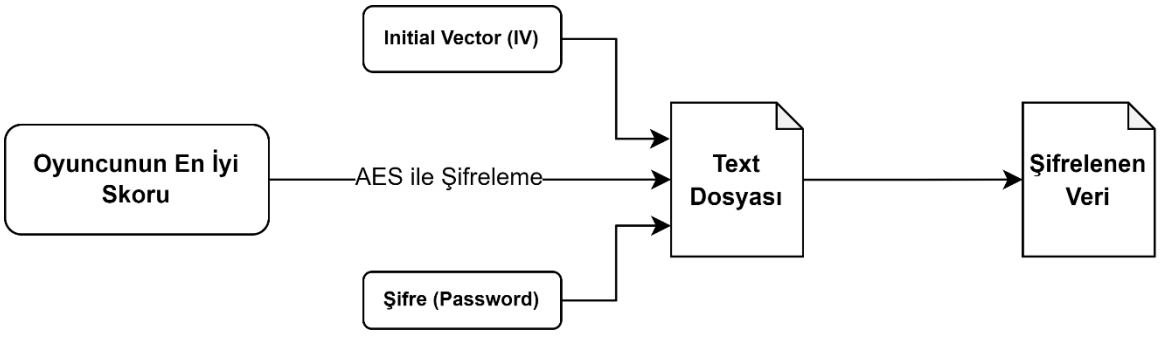


Şekil.2.5 Kullanıcı Arabirimi - 3

2.4.3. Veritabanı Arabirimi

Kullanıcının verilerini (En iyi skor) güvenli bir şekilde saklayabileceği için takip etmesi gereken bir modüldür.

/



Şekil.2.6 Veritabanı Arabirimi

3. SİSTEM GERÇEKLEŞTİRİMİ

Gerçekleştirim çalışması, tasarım sonucu üretilen süreç bilgisayar ortamında çalışan yazılım biçimine dönüştürülmesi çalışmalarını içermektedir.

3.1. Yazılım Geliştirme Ortamları

3.1.1. Programlama Dilleri Ve Frameworklar

C#: C#; Microsoft tarafından .NET Teknolojisi için geliştirilen modern bir programlama dilidir. Sözdizimi C-like (C benzeri) bir deneyim sunar. Microsoft tarafından geliştirilmiş olsa da ECMA ve ISO standartları altına alınmıştır Nesne yönelimli programlama kavramının gelişmesine katkıda bulunan aktif programlama dillerinden biridir. C#, .NET orta seviyeli programlama dillerindendir. Yani hem makine diline hem de insan algısına eşit seviyededir. Buradaki orta ifadesi dilin gücünü değil makine dili ile günlük konuşma diline olan mesafesini göstermektedir. [04]

Unity: Unity, öncelikli olarak bilgisayarlar, konsollar ve mobil cihazlar için video oyunları ve simülasyonları geliştirmek için kullanılan ve Unity Technologies tarafından geliştirilen çapraz platform bir oyun motorudur. İlk kez yalnızca Apple'ın 2005'teki Worldwide Developers Conference'da OS X için ilan edildi, bu tarihten itibaren 27 platformu hedeflemek üzere genişletildi. Unity oyun motoru; film sektörü, otomotiv sektörü, mimari, mühendislik ve inşaat gibi video oyunları dışındaki farklı endüstriler tarafından da benimsenmiş ve kullanılmaktadır. [05]

3.1.2. Yazılım Geliştirme Ortamı

Unity platformu kullanabilmek için C#'ı destekleyen bir geliştirme ortamı hazırlanması ve kullanması gerekemektedir. Bu proje kapsamında geliştirme ortamı olarak Microsoft tarafından geliştirilen güçlü Visual Studio IDE kullanılmaktadır.

3.2. Kod Yapılandırması

3.2.1. Üç Katmanlı Mimari (Three-Tier Architecture)

Üç katmanlı mimari, sunum katmanı, uygulama katmanı ve veri katmanından oluşan modüler bir istemci-sunucu mimarisidir. Veri Katmanı (Database Tier) bilgileri depolar, Uygulama Katmanı (Business Tier) mantığı işler ve Sunum Katmanı (Presentation Layer) diğer iki katman ile iletişim kuran bir grafik kullanıcı arabirimidir (GUI). Üç katmanlı mimari, işlevlerin yapılarını

ayrılaştırarak performans ve güvenlik arttırabilen mantıksal bir yapıdır. Üç katmanlı mimari, projenin herbir kısmı güvenli bir şekilde oluşturabilmek amacıyla kodun tasarımında kullanılmıştır.

3.2.2. Açıklama Satırları

Açıklama satırları kodun yazılması, test ve bakım edilmesi kolaylaştırdığından dolayı bu projede düzgün ve düzenli bir şekilde yazılmasına özen gösterilmiştir. Açıklama satırları karmaşık ve programın akışına büyük etkisi her satırın ve metodun başında yazılmaktadır.

3.2.3. Anlamlı İsimlendirme

Sistem kodlamasının genel yapısında kullanılan değişkenlerin ve metotların isimlerini, kodun içindeki işlevini açık bir şekilde gösterecek şeklinde ve büyük küçük harflerin duyarlılığına dikkat ederek yazılmaktadır.

3.2.4. Kod Biçimlemesi

Kod biçimi tasarlanır iken kodun anlaşılabilirliği ve kullanılabilirliğini artmaya hedeflenmektedir. Bunun için kod biçimlemesinde 4 temel kısım bulunmamaktadır:

- **Base:** Kodun temel kaynaklarını içeren kısımdır. Bu kısım içinde kodun içinde kullanılan renk, yazı tipi, yazı boyutu, yazı kalınlığı, v.b. bulunmaktadır. Aynı zamanda program çalışırken uygulamanın farklı sayfalarına yönlendirme sistemi de bulunmaktadır.
- **Models:** Yapay zeka modeli kullanıldığı kısımdır. Bu kısım içinde yapay zeka modelin ürettiği çıktıları elde edilmekte ve kullanıma hazır olmaktadır.
- **Pages:** Uygulama içinde kullanılan farklı sayfaların tasarımın yapıldığı kısımdır. Bu kısım içinde her bir sayfanın widget yapısı bulunmaktadır.
- **Services:** TensorFlow ve metinden konuşmaya (TTS) kütüphaneleri gibi kullanıldığı kısımdır.

3.3. Olağandışı Durum Tanımları

Olağan dışı durum, bir programın çalışmasının, geçersiz ya da yanlış veri oluşumu ya da başka nedenlerle istenmeyen bir biçimde sonlanmasına neden olan durum olarak tanımlanmaktadır. Olağandışı gelişen durumlarda try-catch blokları devreye girmekte ve program kırılmadan çalışmasına devam edebilecek şekilde kod yazmaya özen gösterilmektedir.

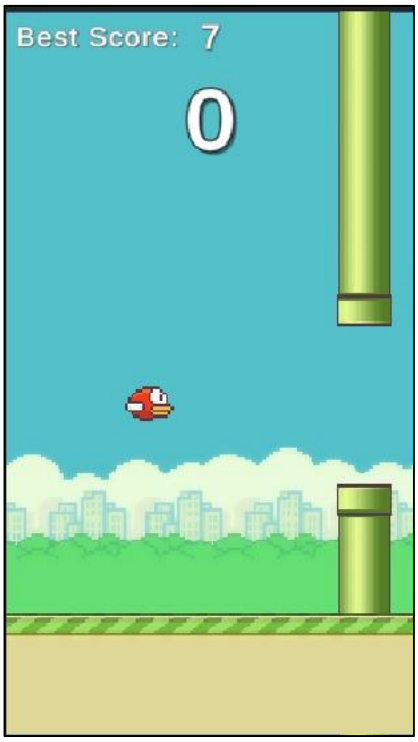
4. BULGULAR VE TARTIŞMA

4.1. Bulgular

Proje kapsamındaki yapılan uygulamayı çalıştırdıktan sonra splash screen ekranı gösterilmektedir.

Şekil.4.1 Kullanıcı Arayüzü - 1

Kullanıcı splash screen'in herhangi bir noktasında dokulduğu zaman uygulama çalışmaya başlamaktadır. Projenin çalışması bu tez kapsamında göstermek için ekranın görüntüsü alınacak ve kullanıcı arayüzün (UI) tüm ekranları gösterilecektir.



Şekil.4.2 Kullanıcı Arayüzü - 2

Şekil.4.3 Kullanıcı Arayüzü - 3

4.2. Tartışma

Projenin gerçekleştirmesinde analizi yapılan gereksinimleri göz önünde bulundurarak yapılmıştır. Bunun için dahili şifreleme ve random sayı üretme yapıların tasarımına bakılırsa oyun geliştirme sektöründe basit ve hızlı bir şekilde kullanabilmeleri için tasarlanmıştır.

Örnek olarak gerçekleştirilen Flappy Beard oyununda PCG, AES ve dosya okuma/yazma işlemleri gerçekleştirebilmek hedefiyle kolay yeniden kullanılabilen sınıflar tasarlanmıştır. Tasarlanmış bu sınıflar Unity platformunda gerçekleştirilmesi istenilen herhangi bir oyun içinde adil rastgelelik ve güvenlik kavramları sağlayabilmek amacıyla yer alabilmektedir.

Proje kapsamında tasarlanan şifreleme ve random sayı üretme yöntemi Unity platformu içinde kullanılmaya hazırlanmıştır. Unity hariç Unreal Engine gibi diğer oyun geliştirme platformalarında da kullanılabilir olmasına rağmen önceden hazır olan şifreleme kütüphaneleri bulunmadığından gerçekleştirmesi biraz daha zor hale gelebilmektedir. Ancak yöntemin sağladığı güvenlik ve güvenilirlik seviyesi farklı oyun geliştirme platformları arasında değişmemektedir.

5. SONUÇLAR

Bu çalışmada görüldüğü gibi, PCG random sayı üretme algoritması oyun uygulamalarında etkin ve performanslı bir şekilde çalışma kabiliyeti bulunmaktadır. Aynı zamanda AES şifreleme algoritması kullanarak oyun uygulamalarında yer alabilen kullanıcı dijital varlıklarını güvenli ve güvenilir bir şekilde saklanabilmektedir. AES ve PCG algoritmaların her ikisi hızlı performans ve çalışma imkanına sahip olduğundan oyun geliştirme sektöründe etkin ve önemli bir yer alabilmektedir. Bu proje kapsamında tasarlanan PCG, AES yöntemi oyun geliştirme geleceğini tasarlayabilecek ve geliştirebilecek benzer uygulamalara yol açmaktadır.

KAYNAKLAR

[01] https://zeo.org/tr/kaynaklar/e-kitaplar/q1-gaming-sektor-raporu

[02] https://www.aa.com.tr/tr/turkiye/turkiyede-30-milyondan-fazla-kisi-oyun-oynuyor/1216357

[03] https://en.wikipedia.org/wiki/Permutation

[04] https://tr.wikipedia.org/wiki/C_Sharp

[05] https://tr.wikipedia.org/wiki/Unity_(oyun_motoru)