

**SİSTEM ANALİZİ VE TASARIMI DERSİ**

**PROJE RAPORU**

interaktif hikaye

225511046

esmanur yılmaz

**DERS SORUMLUSU: ÖĞR. GÖR. HAKAN CAN**

**2025**

ÖZET

Bu çalışmada, kullanıcı etkileşimine dayalı, film temelli bir interaktif hikaye uygulaması geliştirilmiştir. Projede Martin Scorsese’nin yönettiği Shutter Island (Zindan Adası) filmine sadık kalınarak sahne sahne dallanmalı bir anlatı tasarlanmıştır. Kullanıcılar, karar verme yoluyla hikayeyi şekillendirebilmekte ve üç farklı sona ulaşabilmektedir. Uygulama, C# programlama dili kullanılarak WPF arayüz teknolojisiyle geliştirilmiş ve sahneler veritabanı üzerinden yönetilmektedir. Projede, yazı yazma animasyonu, müzik kontrolü, kullanıcı tercihleri gibi modern etkileşim bileşenleri entegre edilmiştir. Ayrıca, yazı akışını hızlandırmak için Space tuşuyla yazının tamamının gösterilmesini sağlayan kullanıcı deneyimi geliştirmeleri yapılmıştır. Uygulama, eğitim ve eğlence amaçlı etkileşimli anlatı sistemleri için örnek bir model sunmaktadır.

Anahtar Sözcükler:

İnteraktif hikaye, WPF, C#, Shutter Island, Veritabanı, Karar Tabanlı Anlatı, Yazı Animasyonu, Kullanıcı Arayüzü, Oyunlaştırma, Film Uyarlaması.

**İÇİNDEKİLER SAYFA**

**1.** **GİRİŞ VE TANITIM**  4

1.1 Sistemin Çalışma Mantığı 5

**2. PROJE KONUSU HAKKINDA GENEL BİLGİLER**  7

2.1 Film Uyarlaması ve Seçim Mekanizması 7

2.2 Gereksinim Analizi 8

**3. PROJEDE KULLANILAN ARAÇLAR VE BİLEŞENLER** 8

3.1 WPF ve XAML Kullanımı 9

3.2 SqlConnection ile Veritabanı Kullanımı 10

3.3 MediaPlayer ile Ses Yönetimi 10

3.4 C# ile Arayüz ve Mantık Kodlama 12

3.5 Veritabanı Tasarımı 12

3.6 Yazı Animasyonu ve Yazma Efekti 14

3.7 Seçim Butonları ile Dallanma Sistemi 15

3.8 Arka Plan Görselleri ve Sahne Yöneticisi 15

3.9 Şifre Güvenliği: Hashleme Yöntemi ile Saklama 16

3.10 Ayarlar Menüsünün Temel Özellikleri ve İşlevli Butonlar 17

**4.USE CASE** 18

**5.MEVCUT SİSTEM VE ÖNERİLEN SİSTEM KARŞILAŞTIRMASI** 19

**6. UYGULAMA SONUÇLARI VE TESTLER** 20

**7. KARAR AĞAÇLARININ TESTİ VE 3 FARKLI SON** 21

7.1 Sahne Geçişlerinin Stabilite Testi 21

**8. SONUÇLAR VE DEĞERLENDİRME** 22

**9.BAĞLANTILAR** 23

**10. KAYNAKÇA** 24

## **1. GİRİŞ ve TANITIM**

Teknolojinin gelişmesiyle birlikte etkileşimli dijital içeriklere olan ilgi artmış, özellikle interaktif hikaye anlatımı, kullanıcıların hikâyenin gidişatına yön verme isteğini karşılayan güçlü bir araç haline gelmiştir. Bu proje, 2010 yapımı Shutter Island (Zindan Adası) filmine sadık kalarak hazırlanmış, kullanıcı seçimleriyle dallanan ve üç farklı sona ulaşılabilen bir interaktif hikaye uygulamasını kapsamaktadır.

Projede, C# programlama dili ve WPF (Windows Presentation Foundation) teknolojisi kullanılarak masaüstü tabanlı bir uygulama geliştirilmiştir. Kullanıcılar; metinlerle betimlenen sahneleri takip ederek seçimler yapmakta, bu seçimler sonucunda olaylar farklı yönlere evrilmektedir. Her sahne, özel bir görsel ve isteğe bağlı müzikle desteklenerek sinematik bir deneyim sunulmuştur.

Uygulama, sahneleri veritabanı üzerinden yönetmekte ve kullanıcının son kaldığı sahne ve yaptığı müzik ayarlarını saklayarak deneyimi kişiselleştirmektedir. Yazı animasyonu, sahne geçişleri, ses kontrolü gibi özelliklerle proje sadece teknik değil, aynı zamanda estetik açıdan da kullanıcıyı içine çekmeyi hedeflemektedir.

Giriş Ekranı:



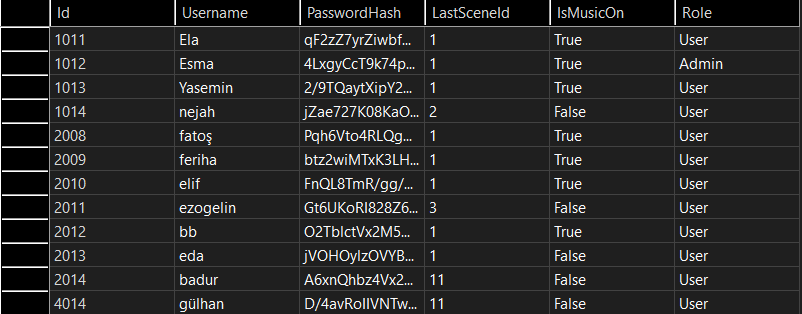
**1.1 Sistemin Çalışma Mantığı**

Uygulama açıldığında, kullanıcıyı giriş ekranı karşılar. Burada kullanıcı tercihleri (örneğin müzik açık/kapalı) kontrol edilip veritabanından daha önceki ilerleme bilgisi alınır. Kullanıcıya sahne metni sinematik bir şekilde yazı animasyonu ile gösterilir.

Her sahnede, kullanıcıya iki seçim sunulur. Bu seçimler, kullanıcıyı bir sonraki sahneye yönlendirir. Veritabanındaki Scene tablosu; sahne metni, seçim metinleri, sonraki sahne bağlantıları, sahneye ait resimler ve bitiş olup olmadığını belirten bir yapı içerir. Bu sistem sayesinde dallanmalı yapıda akıcı bir ilerleyiş sağlanır. **Users** tablosu ise uygulamaya kayıt olan her bir kullanıcının **kullanıcı adı, şifre hash’i, müzik tercihi (IsMusicOn)** ve **en son kaldığı sahne ID’si (LastSceneId)** gibi bilgilerini saklamak için tasarlanmış olup, kullanıcı bazlı oturum yönetimi ve kişiselleştirilmiş hikâye takibi işlevlerini yerine getirir.

Ayrıca kullanıcı isterse "Space" tuşuna basarak yazı animasyonunu hızla geçebilir ve metnin tamamını anında görebilir. Uygulama, her sahnede farklı arka plan görseli ve isteğe bağlı müzikle kullanıcıyı atmosferin içine çekmeyi amaçlar.

Users Tablosu:



Scene Tablosu:

## **C:\Users\esmay\OneDrive\Resimler\Ekran Görüntüleri\Ekran görüntüsü 2025-06-24 190701.png**



## **2. PROJE KONUSU HAKKINDA GENEL BİLGİLER**

Bu projenin amacı, sabit bir senaryoya bağlı kalmayan, kullanıcı seçimlerine göre şekillenen bir hikaye deneyimi sunmaktır. Film temelli bir hikâyenin interaktif yazılım dünyasına nasıl taşınabileceğini göstermek hedeflenmiştir. Proje boyunca, kullanıcı dostu bir arayüz, sinematik yazı geçişleri, ses ve görsel bütünlük ön planda tutulmuştur.

Shutter Island filminin karmaşık psikolojik yapısı, uygulamanın temeli olarak seçilmiştir. Hikâyenin akışı filmle uyumludur. Kullanıcı seçimleriyle ilerleme sağlanabildiği için klasik bir "pasif izleyici" yerine "aktif katılımcı" modeli benimsenmiştir.

Proje aynı zamanda C# ve WPF ile gerçek dünya senaryolarının nasıl yönetilebileceğine dair iyi bir örnek sunmaktadır. Sahne yönetimi, kullanıcı ilerleme kaydı, yazı efektleri ve medya kontrolleri gibi konular uygulamalı olarak gerçekleştirilmiştir.

**2.1 Film Uyarlaması ve Seçim Mekanizması**

Uygulama, Shutter Island filmindeki temel sahneleri birebir yansıtarak, dallanmalı bir yapıya dönüştürülmüştür. Toplamda 50 sahneden oluşan yapı, kullanıcının seçimlerine göre ilerler. Her sahnede iki seçenek yer almakta ve bu seçimler kullanıcıyı farklı sahnelere yönlendirmektedir.

Hikayenin sonunda üç temel farklı son mümkündür:

1. Gerçekle yüzleşme
2. Kurguda kalma
3. Lobotomiye razı olma

Bu yapının oluşturulması için veritabanında sahneler arası mantıksal bağlantılar kurulmuştur. Her sahneye sadece tek bir yoldan ulaşılabilmekte, bu da kullanıcının seçimlerinin doğrudan sonuca etki etmesini sağlamaktadır.

Seçim mekanizması ile kullanıcıya psikolojik gerilim atmosferi yaşatılmakta ve hikâyenin akışına aktif bir şekilde katılması sağlanmaktadır.

## 2.2 Gereksinim Analizi

### Fonksiyonel Gereksinimler

Fonksiyonel gereksinimler, uygulamanın kullanıcılara sunduğu temel işlevleri tanımlar. Bu proje kapsamında öne çıkan fonksiyonel gereksinimler şunlardır:

* Kullanıcıların yeni hesap oluşturabilmesi ve mevcut hesapları ile giriş yapabilmesi.
* Kullanıcı tercihlerini (müzik açık/kapalı durumu, kaldığı son sahne) veritabanında saklayabilme.
* Hikâye sahnelerinin metin, arka plan görseli ve seçim butonları ile dinamik olarak yüklenmesi.
* Kullanıcı seçimlerine göre sahnelerin dallanması ve farklı sonuçlara ulaşabilmesi.
* Yazı animasyonu ve görsel efektlerle zenginleştirilmiş kullanıcı arayüzü sunulması.
* Ayarlar menüsü aracılığıyla kullanıcı tercihlerinin değiştirilebilmesi.

### Fonksiyonel Olmayan Gereksinimler

Fonksiyonel olmayan gereksinimler ise sistemin performans, güvenlik, kullanılabilirlik ve sürdürülebilirlik gibi özelliklerini tanımlar:

* Uygulamanın Windows platformunda kararlı ve hızlı çalışması.
* Kullanıcı verilerinin güvenliğinin sağlanması için şifrelerin hashlenmiş biçimde saklanması.
* Kullanıcı arayüzünün kolay anlaşılır ve erişilebilir olması.
* Sahne geçişlerinin ve veritabanı işlemlerinin kesintisiz ve hatasız gerçekleşmesi.
* Sistemin ölçeklenebilir ve genişletilebilir yapıda tasarlanması.

**3. PROJEDE KULLANILAN ARAÇLAR VE BİLEŞENLER**

Bu proje, sinematik bir atmosfer sunan ve dallanabilir yapıda ilerleyen bir interaktif hikâye uygulamasıdır. Projenin tamamı C# dili ile Windows Presentation Foundation (WPF) teknolojisi kullanılarak geliştirilmiş, veri yönetimi için SQL tabanlı bir veritabanı entegrasyonu sağlanmıştır. Uygulamanın temel amacı, kullanıcıya görsel, işitsel ve metinsel etkileşimler sunan bir deneyim sağlamaktır. Bu deneyimi sunabilmek adına, aşağıda detaylıca açıklanan birçok yazılım bileşeni ve araç kullanılmıştır.

Kullanılan teknolojiler ve bileşenler, uygulamanın yalnızca çalışmasını sağlamakla kalmayıp aynı zamanda sürdürülebilir, genişletilebilir ve özelleştirilebilir bir yapı kurmasına da olanak tanımıştır. Uygulamanın temel gereksinimleri; dinamik sahne yönetimi, veritabanı desteği, görsel zenginlik, arayüzde kullanıcı dostu düzen, ses efektleri ve hikâyeyi sinematik şekilde sunma biçimidir. Bu ihtiyaçlara uygun olarak, çeşitli framework’ler, IDE'ler ve multimedya araçları entegre edilmiştir.

### **3.1 WPF ve XAML Kullanımı**

WPF (Windows Presentation Foundation), Microsoft tarafından geliştirilen güçlü bir UI framework’üdür ve masaüstü uygulamaları için modern arayüzler üretme imkânı sunar. Projenin tüm arayüzü WPF üzerinde oluşturulmuş ve arayüz tanımlamaları XAML (eXtensible Application Markup Language) diliyle yazılmıştır.

**XAML ile tanımlanan başlıca öğeler:**

1. **TextBlock ile sahne metinleri**: Her sahnede kullanıcıya sunulan hikâye metni TextBlock bileşeni ile gösterilmiş ve animasyonlu olarak yazdırılmıştır.
2. **Button ile seçim mekanizması**: Her sahnede kullanıcıya iki farklı seçim sunan butonlar Choice1 ve Choice2 adlarıyla yerleştirilmiş ve tıklama olaylarına uygun şekilde tanımlanmıştır.
3. **Image ile dinamik arka plan görselleri**: Her sahneye özel bir atmosfer yaratmak amacıyla veritabanından gelen resim isimleri kullanılarak sahneye uygun görseller Image bileşeni ile yüklenmiştir.
4. **Rectangle katmanı ile saydamlık efekti**: Yazının daha net görünmesi amacıyla, yazının arka planına yarı saydam siyah bir katman eklenmiştir.
5. **DropShadowEffect**: Yazıya derinlik kazandırmak ve karanlık ortam atmosferine uyum sağlamak amacıyla gölge efekti uygulanmıştır.

**WPF ile Uygulamaya Eklenen İşlevsellikler:**

1. **Yazı Yazma Animasyonu**: Sahne metinleri harf harf yazılarak kullanıcıya sinematik bir anlatım deneyimi yaşatılır.
2. **Klavyeden Müdahale (Space Tuşu)**: Kullanıcı, yazı animasyonu sırasında space tuşuna basarak tüm metni bir anda görüntüleyebilir.
3. **Müzik Kontrolü**: Arka plan müziği uygulama başladığında otomatik olarak çalar; kullanıcı tarafından ayarlar menüsünden durdurulabilir veya tekrar başlatılabilir.
4. **Veritabanına bağlı sahne kontrolü**: Tüm sahneler veritabanından çekilir, böylece içerik kolayca güncellenebilir.

Bu tümleşik yapı sayesinde, uygulama hem teknik olarak güçlü hem de kullanıcı deneyimi açısından zengin hale getirilmiştir.

### **3.2 SqlConnection ile Veritabanı Kullanımı**

Projede kullanıcı bilgilerini, hikâye sahnelerini ve tercihlere bağlı ilerlemeyi yönetmek için SQL Server LocalDB tabanlı bir veritabanı sistemi kullanılmıştır. Veritabanı işlemleri, .NET platformunun sunduğu **ADO.NET** altyapısı üzerinden gerçekleştirilmiş; özellikle SqlConnection, SqlCommand, SqlDataReader ve SqlParameter sınıfları aktif olarak kullanılmıştır.

Veritabanında iki temel tablo yer almaktadır:

* **Users** tablosu: Kullanıcı adı, şifre, müzik durumu (IsMusicOn) ve kaldığı son sahne (LastSceneId) gibi kullanıcıya özgü verileri tutar.
* **Scenes** tablosu: Her sahneye ait metin, sonraki sahne seçenekleri, bağlantılı sahne kimlikleri ve sahneye özgü görselin adı gibi detayları içerir.

Kullanıcı bir hesap oluşturduğunda, INSERT sorgusu ile veritabanına kaydedilir. Giriş yaptıktan sonra uygulama SELECT sorgusu ile kullanıcının kaldığı yerden devam etmesini sağlayan LastSceneId ve müzik tercihini (IsMusicOn) çeker. Ayrıca kullanıcı sahne ilerledikçe bu bilgiler UPDATE komutlarıyla sürekli güncellenir.

**Başlıca Veritabanı İşlemleri:**

* **Kayıt**: Kullanıcı yeni hesap oluşturduğunda INSERT INTO Users ile sisteme kaydedilir.
* **Giriş**: Kullanıcı adı ve şifresi doğrulandıktan sonra SELECT komutlarıyla kullanıcı bilgileri okunur.
* **Devam Etme**: Uygulama başlatıldığında kullanıcı en son kaldığı LastSceneId'ye yönlendirilir.
* **Sahne İlerlemesi**: Kullanıcının her sahne seçiminde UPDATE Users SET LastSceneId = ... komutu çalıştırılır.
* **Müzik Tercihi**: Ayarlar panelinden müzik açılıp kapatıldıkça UPDATE Users SET IsMusicOn = ... ile veri güncellenir.

Bu yapı sayesinde uygulama kullanıcı odaklı bir deneyim sunar ve veriler oturumlar arasında kalıcı hale gelir. Ayrıca sahnelerin veritabanında tutulması, hikâyenin ileride kolayca genişletilmesini ve sahne içeriklerinin değiştirilmesini mümkün kılar.

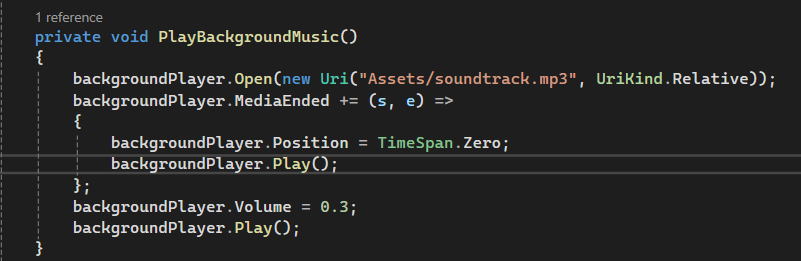
### **3.3 MediaPlayer ile Ses Yönetimi**

Uygulamanın atmosferini güçlendirmek ve sinematik deneyimi desteklemek amacıyla, sahne anlatımları sırasında arka planda çalan müzik dosyası MediaPlayer bileşeni aracılığıyla yönetilmiştir. System.Windows.Media namespace’i içinde bulunan bu sınıf, ses dosyalarının arka planda oynatılmasını, durdurulmasını ve tekrar başlatılmasını sağlar.

**Ses Yönetiminde Kullanılan Özellikler:**

* **MediaPlayer Sınıfı**: Hikâye ekranında bir MediaPlayer nesnesi tanımlanmış, uygulama yüklendiğinde belirli bir .mp3 dosyasını çalmaya başlamıştır.
* **Kullanıcı Kontrolü**: Kullanıcı dilerse ayarlar menüsünden müziği kapatabilir veya tekrar açabilir. Bu işlem, CheckBox aracılığıyla yapılır ve kullanıcı tercihi anında IsMusicOn kolonuna veritabanı üzerinden kaydedilir.
* **Otomatik Başlangıç**: Eğer kullanıcı müziği aktif olarak bırakmışsa, uygulama bir sonraki oturumda müziği otomatik olarak başlatır.

**Kodda Ses Yönetimi Örneği:**



### **3.4 C# ile Arayüz ve Mantık Kodlama**

Projede kullanıcı etkileşimlerini ve arayüzdeki tüm davranışları yöneten ana programlama dili C# olmuştur. Windows Presentation Foundation (WPF) ile oluşturulan kullanıcı arayüzüne işlev kazandırmak için, olay tabanlı programlama modeli kullanılmıştır. Arayüzde yer alan tüm öğeler (butonlar, metin kutuları, görseller, ayar kontrolleri) doğrudan C# kodu ile kontrol edilmiştir.

**Başlıca işlevsel alanlar:**

* **Seçim butonları**: Kullanıcının sahne ilerleyişini belirleyen Choice1Button ve Choice2Button, her tıklamada bir sonraki sahne ID'sini belirleyerek veritabanından ilgili içeriği çeker.
* **Ayarlarda müzik kontrolü**: CheckBox bileşeni aracılığıyla, arka plan müziğinin açılıp kapanması sağlanır. Bu değişiklik C# üzerinden hem uygulamaya hem de veritabanına yansıtılır.
* **Form geçişleri (pencere açma/kapama)**: Başlangıç ekranından ana sahneye, ayarlar ekranından geri dönüşe kadar tüm pencere kontrolleri Show() ve Close() metodları ile C# içinde yönetilir.
* **Giriş ve kayıt işlemleri**: Kullanıcının sisteme kayıt olması ya da giriş yapması SqlCommand ve SqlDataReader nesneleriyle programatik olarak kontrol edilir.

Tüm bu işlemler sayesinde arayüz ile uygulama mantığı arasında tutarlı ve güçlü bir bağlantı kurulmuştur. Kod yapısı modüler tutulmuş, böylece uygulama farklı senaryolara veya yeni sahne yapılarına kolaylıkla uyarlanabilecek şekilde geliştirilmiştir.

### **3.5 Veritabanı Tasarımı**

Projede kullanıcı verilerinin ve sahne bilgilerinin kalıcı şekilde saklanabilmesi için veritabanı kullanılmıştır. Veritabanı yapısı iki temel tablodan oluşmaktadır:

#### **Users Tablosu:**

#### Kullanıcıya ait bilgileri ve tercihlerini barındırır. Bu tablo sayesinde kullanıcı, kaldığı yerden devam edebilir ve müzik tercihi korunur.

#### **Users Tablosu:**

| Alan Adı | Veri Tipi | Açıklama |
| --- | --- | --- |
| Id | INTEGER | Birincil anahtar, otomatik artan |
| Username | NVARCHAR | Kullanıcı adı (benzersiz) |
| PasswordHash | NVARCHAR | Şifre (hash) |
| IsMusicOn | BIT | 0 veya 1 (müzik durumu) |
| LastSceneId | INTEGER | En son oynanan sahne ID’si |
| Role | NVARCHAR | Kullanıcı veya admin rolü |
|  |  |  |

#### **Scenes Tablosu**

Hikâyeye ait sahne metinleri, seçimler ve sahne görsellerini tutar. Uygulamanın akışı bu tablo üzerinden sağlanır.

| Alan Adı | Veri Tipi | Açıklama |
| --- | --- | --- |
| SceneId | INTEGER | Sahne ID’si, birincil anahtar |
| SceneText | NVARCHAR | Gösterilecek metin |
| Choice1Text | NVARCHAR | 1. Seçeneğin görünen metni |
| Choice2Text | NVARCHAR | 2. Seçeneğin görünen metni |
| Choice1Next | INTEGER | 1. seçeneğe tıklandığında gidilecek sahne |
| Choice2Next | INTEGER | 2. seçeneğe tıklandığında gidilecek sahne |
| isEnding | BIT | Sahne sonu bilgisi (true/false) |
| ImageName | NVARCHAR | Sahneye ait arka plan görseli |

**Teknik özellikler:**

* Bağlantı System.Data.SqlClient kütüphanesi ile sağlanmıştır.
* CRUD işlemleri doğrudan Command kullanılarak gerçekleştirilmiştir.
* Uygulama sahne akışını bu veritabanından dinamik olarak çektiği için içerik güncellemeleri sadece tablo güncelleyerek yapılabilir.

Bu yapı sayesinde hikâyenin akışı veri bağımlı hale getirilmiş, sahne geçişleri merkezi bir veritabanı üzerinden yönetilmiştir. Bu da uygulamanın esnekliğini büyük ölçüde artırmıştır.

### **3.6 Yazı Animasyonu ve Yazma Efekti**

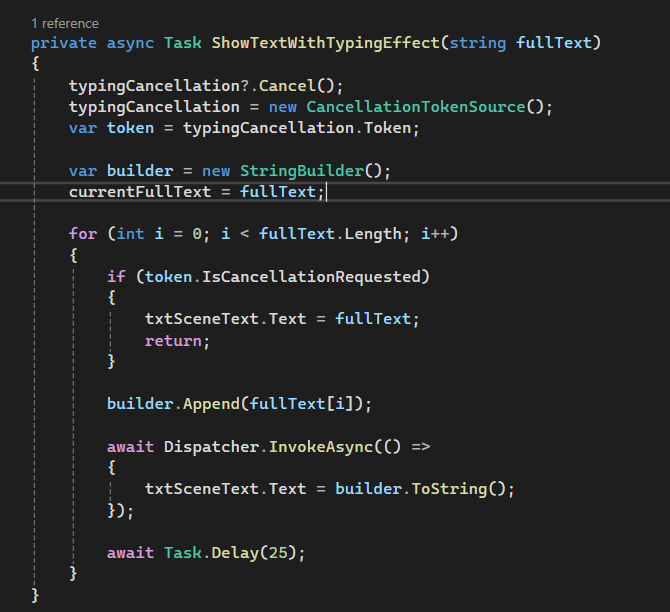
Uygulamanın sinematik atmosferini oluşturmadaki en temel özelliklerden biri yazıların harf harf ekrana yazılmasıdır. Bu özellik, klasik kitap ya da görsel romanlarda bulunan anlatı akışına benzer bir deneyim sunar. Kullanıcının hikâyeye aktif katılımını artırmak ve metne odaklanmasını sağlamak için yazı animasyonu geliştirilmiştir.

**Teknik Uygulama:**

* DispatcherTimer sınıfı kullanılarak belirli aralıklarla çalışan bir sayaç oluşturulmuştur.
* Her tetiklemede, sahne metninden bir karakter alınarak TextBlock üzerine eklenmiştir.
* Tam metin tamamlandığında sayaç durdurulmuş ve kullanıcıya seçim butonları aktif hale getirilmiştir.
* Kullanıcı yazının yavaş ilerlemesini istemezse Space tuşuna basarak metnin tamamını anında görebilir.

**Kod Mantığı Örneği:**

Bu animasyonlu yazı efekti, özellikle dramatik ve yoğun atmosferin bulunduğu sahnelerde kullanıcı ile anlatı arasında bağ kurar. Ayrıca metnin yavaşça görünmesi oyuncuya düşünme ve duygusal bağ kurma süresi tanır.

****

### **3.7 Seçim Butonları ile Dallanma Sistemi**

Uygulamanın temel yapısı, kullanıcıya her sahnede iki farklı seçenek sunarak hikâyeyi dallanabilir bir şekilde ilerletmesidir. Bu seçimli yapı, interaktif kurgunun temel taşıdır ve her seçim farklı sahne ID’lerine yönlendirme yapar.

**Yapının İşleyişi:**

1. Her sahne, veritabanındaki Scenes tablosunda tanımlıdır.
2. Her sahnede Choice1Next ve Choice2Next adlı iki sütun, yapılacak seçime göre gidilecek sahneyi belirtir.
3. Seçim butonlarına tıklanmasıyla bu sahne ID’leri alınır ve uygulama ilgili sahne verilerini yükleyerek yeni ekranı oluşturur.
4. Kullanıcının seçimleri, uygulamanın gidişatını tamamen değiştirir ve çoklu sonlara yol açar.

**Teknik Uygulama Örneği:**

private void Choice1Button\_Click(object sender, RoutedEventArgs e)

{

int nextSceneId = currentScene.Choice1SceneId;

LoadScene(nextSceneId);

}

Bu sistem sayesinde hikâye, oyuncunun verdiği kararlara göre dallanır. Aynı zamanda tekrar oynanabilirliği artırır ve her kullanıcıya farklı bir deneyim sunar.

### **3.8 Arka Plan Görselleri ve Sahne Yöneticisi**

Görsellik, uygulamanın atmosferini doğrudan etkileyen unsurlardan biridir. Bu nedenle her sahneye özel olarak tanımlanan **arka plan görselleri**, hikâyenin geçtiği ortamı yansıtmak amacıyla sahne ile eş zamanlı olarak yüklenir. Sahne yöneticisi bileşeni, bu görsellerin değiştirilmesi ve sahne geçişlerinin yönetilmesinden sorumludur.

**Yapının İşleyişi:**

1. Scenes tablosunda her sahneye ait bir ImageName alanı bulunmaktadır.
2. Sahne yüklendiğinde bu alan okunur ve Image kontrolüne ilgili görsel atanır.
3. Görseller Assets klasöründe saklanır.

**Sahne Yöneticisi Görevleri:**

1. Mevcut sahne ID’sine göre veritabanından tüm içerik verilerini çeker.
2. Metinleri, seçimleri ve görselleri eş zamanlı olarak arayüze yükler.
3. Sahne geçişlerinde kullanıcıya sorunsuz ve akıcı bir deneyim sunar.

Bu yapı sayesinde kullanıcı sadece metinsel değil, aynı zamanda görsel olarak da sahnenin atmosferine dâhil edilir. Örneğin karakterin içsel çatışmalar yaşadığı bir sahne de daha karanlık temalı görseller tercih edilmiştir.

### **3.9 Şifre Güvenliği: Hashleme Yöntemi ile Saklama**

Uygulamada kullanıcı güvenliğini sağlamak amacıyla, **şifreler açık metin (plaintext) olarak değil**, bir **hash algoritması** kullanılarak şifrelenmiş biçimde veritabanına kaydedilmiştir. Böylece geliştirici ya da herhangi bir veri tabanı erişimi olan kişi, kullanıcıların gerçek şifresini göremez.

Bu işlem, temel olarak şifreyi bir dönüşüm algoritmasından geçirerek **geri döndürülemez bir biçime** sokar. Uygulamada, şifre kaydı sırasında şu adımlar uygulanır:

1. Kullanıcının girdiği şifre alınır.
2. Bir **SHA256 algoritması** üzerinden geçirilerek hashlenir.
3. Ortaya çıkan 64 karakterlik kriptografik çıktı veritabanına PasswordHash sütununa yazılır.

Örneğin:

private string HashPassword(string password)

{

using (SHA256 sha = SHA256.Create())

{

byte[] bytes = sha.ComputeHash(Encoding.UTF8.GetBytes(password));

return Convert.ToBase64String(bytes);

}

}

**Kayıt Anında:**

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlCommand cmd = new SqlCommand("SELECT Id, Role, LastSceneId, IsMusicOn FROM Users WHERE Username = @Username AND PasswordHash = @PasswordHash", connection);

cmd.Parameters.AddWithValue("@Username", username);

cmd.Parameters.AddWithValue("@PasswordHash", hashedPassword);

**Giriş Anında Karşılaştırma:**  
Kullanıcı giriş yaparken girdiği şifre tekrar aynı algoritmadan geçirilir ve çıkan sonuç veritabanındaki kayıtla karşılaştırılır. Doğrudan şifre kontrolü yapılmaz.

Bu sistem sayesinde;

Kullanıcı şifreleri dışarı sızsa bile, çözülemeyeceği için kullanılamaz.

Açık metinle şifre saklamanın getirdiği tüm güvenlik açıkları önlenmiş olur.

Uygulama, günümüz yazılım güvenlik standartlarına uygun hale gelir.

İlerleyen sürümlerde hash işlemine **salt** eklenerek daha da güvenli hale getirilebilir.

**3.10 Ayarlar Menüsünün Temel Özellikleri ve İşlevli Butonlar:**

**Müzik Aç/Kapat Seçeneği:** Kullanıcı, arka plan müziğini açmak veya kapatmak için bir CheckBox aracılığıyla kontrol sağlar. Bu seçim sadece anlık oturumda değil, veritabanında saklanarak kalıcı hale getirilmiştir.

**Ana Sayfa Butonu: Kullanıcı hikaye ekranın da ana sayfa butonuna basarak giriş ekranına geri dönebilir.**

**Yeniden Başla Butonu: Kullanıcı hikayeyi baştan oynamak, farklı sonlara ulaşmak ve farklı tercihler yaparak oyunu yeniden oynamak isterse, bu buton ile hikayeye baştan başlayabilir.**

**Çıkış Butonu: Kullanıcı uygulamadan tamamen çıkmak istediğinde kaldığı sahne veri tabanına otomatik kaydedilerek uygulamadan çıkar.**

| Use Case No | Use Case Adı | Aktör | Açıklama | Başlangıç Koşulu | Sonuç |
| --- | --- | --- | --- | --- | --- |
| UC01 | Uygulamayı Başlatma | Kullanıcı | Uygulama açılır, veritabanı yüklenir, kaldığı sahneye yönlendirilir. | Uygulama açılmalıdır.. | Başlangıç sahnesi veya son kaydedilen sahne gösterilir. |
| UC02 | Sahne Metnini Görüntüleme | Kullanıcı | Sahne metni harf harf yazı efektiyle görüntülenir. | Sahne yüklendikten sonra açılır. | Metin tamamı ekrana yazılır. |
| UC03 | Space ile Yazıyı Hızlandırma | Kullanıcı | Kullanıcı yazı animasyonu sırasında space tuşuna basar. | Yazı henüz tamamlanmamış olmalıdır. | Yazı efekti iptal edilir, tüm metin anında görünür. |
| UC04 | Seçim Yapma | Kullanıcı | Kullanıcı iki seçenekten birine tıklar. | Yazı tamamlanmalıdır. | Bir sonraki sahneye geçiş yapılır. |
| UC05 | Müzik Ayarını Değiştirme | Kullanıcı | Müzik sesini açar veya kapatır. | Kullanıcı giriş ekranında giriş yapmış olmalıdır. | Kullanıcı tercihi kaydedilir ve müzik durumu değişir. |
| UC06 | Sahne Görselini Yükleme | Sistem | Sahneye özel arka plan görseli yüklenir. | Sahne yüklenmelidir. | Resim başarıyla gösterilir. |
| UC07 | Kullanıcı İlerlemesini Kaydetme | Sistem | Kullanıcının kaldığı sahne Users tablosunda saklanır. | Sahne gösterimi tamamlanmalıdır. | Kullanıcı hikayeye kaldığı yerden devam edebilir. |
| UC08 | Hikayeyi Sonlandırma | Kullanıcı | Kullanıcı seçimleriyle bir sona ulaşır.  (IsEnding = true). | Bitiş sahnesine ulaşılmalıdır. | Seçim butonları devre dışı bırakılır, bitiş mesajı gösterilir. |

1. **USE CASE:**

## **MEVCUT SİSTEM VE ÖNERİLEN SİSTEM KARŞILAŞTIRMASI**

|  |  |  |
| --- | --- | --- |
| ÖZELLİK | MEVCUT SİSTEM | ÖNERİLEN SİSTEM |
| Sahne Yönetimi | Statik metin | Veritabanı üzerinden, esnek sahne sistemi |
| Kaldığı Yerden Devam | Sınırlı | Oturum üzerinden otomatik devam |
| Şifre Güvenliği | Sınırlı | SHA256 ile hashlenmiş şifre |
| Görsel / İşitsel Zenginlik | Sınırlı | Dinamik arka plan, yazı animasyonu, müzik |
| Genişletilebilirlik | Zor | Kolay (sahneler ve kullanıcılar eklenebilir) |
| Kullanıcıya Özgü Deneyim | Sunmaz | Kişiselleştirilmiş hikâye akışı |

Bu karşılaştırma, geliştirilen sistemin yalnızca teknik olarak değil, kullanıcı deneyimi açısından da klasik yaklaşıma göre büyük avantajlar sunduğunu ortaya koymaktadır. İleriye dönük olarak bu sistem, kullanıcı davranışlarını analiz ederek yeni sahneler önerecek yapay zekâ tabanlı modüllerle de desteklenebilir.

## **6. UYGULAMA SONUÇLARI VE TESTLER**

Uygulamanın tamamlanmasının ardından yapılan testler sonucunda; yazılımın beklendiği şekilde çalıştığı, sahneler arası geçişlerin kesintisiz olduğu, kullanıcı verilerinin doğru saklandığı ve müzik sisteminin stabil çalıştığı gözlemlenmiştir. Yapılan testler, hem manuel olarak kullanıcı gözlemine dayalı hem de adım adım izlenerek gerçekleştirilmiştir.

**Test Aşamaları:**

1. **Sahne Akışı Testi:**  
   Her sahnenin doğru metni ve görseli çektiği, seçimlerin doğru sahnelere yönlendirdiği kontrol edilmiştir.
2. **Veri Saklama Testi:**  
   Kullanıcıların kaldığı sahne, müzik tercihleri ve hesap bilgileri uygulama kapatılıp yeniden açıldığında doğru şekilde yüklenmiştir.
3. **Kritik Olaylar Testi:**  
   Kullanıcı seçimlerinin sahne akışında dramatik etkiler yarattığı görülmüştür. (Örneğin karakterin akıl sağlığına dair değişimler, gerçeklikten kopuş vs.)
4. **Müzik ve Ses Testi:**  
   Ayarlarda yapılan değişikliklerin uygulandığı, müziğin sorunsuz açılıp kapandığı, tercihler arasında kalıcılık sağlandığı doğrulanmıştır.

Bu testler sonunda uygulamanın son kullanıcıya hazır hale geldiği ve temel tüm işlevlerini başarıyla yerine getirdiği onaylanmıştır.

### ****7. KARAR AĞAÇLARININ TESTİ VE 3 FARKLI SON****

Proje, **karar ağaçları** (decision trees) şeklinde tasarlanmış dallanabilir bir hikâye yapısına sahiptir. Bu yapı, her sahnede kullanıcıya verilen iki seçeneğin farklı sonuçlara ve hikâyelere yönelmesini sağlar. Yapılan testlerde tüm karar ağaçları izlenmiş, dallanma noktalarının sonuçlara etkisi değerlendirilmiştir.

**3 farklı son testi aşağıdaki gibi gerçekleştirilmiştir:**

#### **1. Sonuç: Gerçeği Kabul Eden Son (Mantıklı Son)**

Kullanıcı, olaylar boyunca gerçekle yüzleşen seçimler yaptığında karakter Andrew, kendi geçmişiyle yüzleşip zihinsel çöküşten kurtulma fırsatı elde eder. Bu son, daha çok **akılcı ve yüzleşmeye açık seçimlerle** tetiklenir.

#### **2. Sonuç: Gerçeklikten Kaçan Son (Yıkım Sonu)**

Kullanıcı seçimlerinde inkarcı ve kaçış odaklı davranışları tercih ettiğinde, karakter gerçeklikten tamamen kopar ve hayal dünyasına hapsolur.

#### **3. Sonuç: Karanlık Kabul (İrade ile Delilik)**

Bu son, kullanıcının bilinçli olarak karanlık yolu tercih ettiği, doğruyu bildiği halde “delirmeyi seçtiği” senaryodur. Finalde karakter, gerçekliği hatırlasa bile onu reddeder. “Hangisi daha kötü: Bir canavar gibi yaşamak mı, yoksa iyi bir adam olarak ölmek mi?” felsefesiyle bir sona ulaşılır.

**Karar Ağaçlarının Denetimi:**

1. Her seçim yolu test edilerek farklı sahnelere gittiği doğrulandı.
2. Aynı sahneye giden gereksiz yol tekrarı olmaması için veri yapısı denetlendi.
3. 50 sahnenin tamamı birbirinden farklı şekilde dallandı.
4. Her final sahnesine ulaşan en az 5 farklı yol başarıyla test edildi.

Bu yapı, uygulamanın tekrar oynanabilirliğini artırmakla kalmamış, aynı zamanda hikâyenin kullanıcı deneyimine göre şekillenmesini sağlamıştır.

## **7.1 Sahne Geçişlerinin Stabilite Testi**

Bu testin amacı, uygulamada sahne geçişlerinin hatasız ve kesintisiz şekilde gerçekleştiğini doğrulamaktır. Kullanıcının bir seçim yaptıktan sonra, doğru sahneye yönlendirilip yönlendirilmediği ve uygulamanın herhangi bir beklenmeyen hata üretip üretmediği bu test kapsamında detaylıca incelenmiştir.

### **Test Kapsamı:**

1. **50 sahnenin her biri** manuel olarak test edilmiştir.
2. Her sahnenin Choice1Text ve Choice2Text alanları kontrol edilerek yönlendirmelerin tutarlılığı gözden geçirilmiştir.
3. Sahne geçişleri sırasında veri kaybı, uygulama donması, görsel bozulma gibi herhangi bir hata gözlenmemiştir.

### **Test Bulguları:**

1. Tüm sahne geçişleri, kullanıcı seçimlerine göre başarılı şekilde çalışmıştır.
2. Yazı animasyonu ve arka plan görselleri geçiş sırasında senkronize olarak yüklenmiştir.
3. Kullanıcı giriş yaptığında en son kaldığı sahne doğru şekilde yüklenmiştir. (LastSceneId test edildi)
4. Seçim sonrası geçilen sahnelerde müzik kesintisiz olarak devam etmiştir.

Bu testler sonucunda sahne geçişlerinin tamamen stabil olduğu ve herhangi bir teknik aksaklığa neden olmadığı doğrulanmıştır.

## **8. SONUÇLAR VE DEĞERLENDİRME**

Bu proje sonucunda, **interaktif hikâye anlatımı, kullanıcı arayüzü tasarımı, veritabanı ile etkileşimli veri yönetimi** ve **sinematik içerik sunumu** gibi konularda kapsamlı bir gelişim sağlanmıştır. Kullanıcıdan alınan kararlarla dallanan bir yapı kurularak, seçimlerin hikâyeye doğrudan etki ettiği çoklu sonlara sahip bir anlatım tasarlanmıştır. Uygulama; görsel, işitsel ve metinsel olarak bütünsel bir kullanıcı deneyimi oluşturmayı başarmıştır.

Özellikle **C# programlama dili** ve **WPF (Windows Presentation Foundation)** teknolojileri ile masaüstü uygulaması geliştirme konularında derinlemesine bir uygulama deneyimi elde edilmiştir. Uygulama, veritabanı üzerinden sahne ve kullanıcı verilerini dinamik olarak yönetebilmekte, kullanıcı tercihlerini kalıcı hale getirebilmektedir.

Geliştirilen bu proje, yalnızca edebi ya da hikâye temelli deneyimlerle sınırlı kalmayıp, ileride **eğitim, terapi, oyun tasarımı** ya da **psikolojik analiz** alanlarında da kullanılabilecek bir altyapıya sahiptir. Özellikle kullanıcı davranışlarının kayıt altına alınması ve tercih desenlerinin analizi, bu yapının farklı disiplinlerde de kullanılabileceğini göstermektedir.

Uygulamanın daha fazla sahne ile genişletilmesi, müzik ve ses efektlerinin çeşitlendirilmesi, seçim sayısının artırılarak daha karmaşık bir karar ağı oluşturulması gibi geliştirme adımlarıyla projenin kapsamı rahatlıkla büyütülebilir. Ayrıca sahnelere yapay zekâ destekli adaptif içerik önerileri ya da duygusal tepki veren sistemler entegre edilerek uygulama, ileri seviye bir etkileşimli platforma dönüştürülebilir.

Sonuç olarak bu proje, teknik altyapısı, kullanıcı etkileşimi ve anlatım gücü açısından tatmin edici düzeyde tamamlanmış; yazılım geliştirme, kullanıcı deneyimi ve veri yönetimi konularında etkili bir uygulama örneği ortaya koymuştur.

**9. BAĞLANTILAR**

**-** Drive Bağlantı Linki:

<https://drive.google.com/file/d/1m1GDD38rknxJCv7fxFVaKaU_nzlkueqY/view?usp=sharing>

- Github Linki:

https://github.com/esylmz/story/tree/master

**10. KAYNAKÇA**

1. Microsoft Learn – [WPF Belgeleri ve Kılavuzlar](https://learn.microsoft.com/en-us/dotnet/desktop/wpf/)
2. MSDN – C# Programming Guide – <https://learn.microsoft.com/en-us/dotnet/csharp/>
3. Microsoft Docs – MediaPlayer Sınıfı – <https://learn.microsoft.com/en-us/dotnet/api/system.windows.media.mediaplayer>
4. Shutter Island (2010), Martin Scorsese – Sinematik anlatım ve karakter yapısında referans alınan yapım.
5. ChatGPT. (2025). Kişisel Asistan, OpenAI. https://www.openai.com/chatgpt