

İÇİNDEKİLER

1. GİRİŞ 1.1. Projenin Amacı ve Kapsamı
1.2. Projenin Hedef Kitleleri
1.3. Proje Kısıtları ve Sınırlandırmalar
2. KULLANILAN TEKNOLOJİLER VE YAZILIM MİMARİSİ
2.1. Arka Plan (Backend) Mimarisi (.NET Core, Repository & Unit of Work Pattern)
2.2. Veritabanı ve Veri Yönetimi (Entity Framework Core & Dapper)
2.3. Ön Yüz (Frontend) Mimarisi (React, Vite, Component Yapısı)
2.4. Güvenlik Katmanı (JWT Authentication & Authorization)
3. PROJENİN ÖNE ÇIKAN SİSTEM ÖZELLİKLERİ
3.1. Akıllı Önbellekleme (MemoryCache) Sistemi
3.2. Merkezi ve Dosya Tabanlı Loglama (Serilog)
3.3. Google Gemini Yapay Zeka (AI) Entegrasyonu
3.4. Dinamik Yönetim (Admin) Paneli (CRUD İşlemleri)
4. SİSTEM KURULUMU
4.1. Geliştirme Ortamı Gereksinimleri
4.2. Adım Adım Kurulum ve Çalıştırma
5. SONUÇ VE GELECEK ÇALIŞMALAR
6. KAYNAKÇA

1. GİRİŞ

1.1. Projenin Amacı ve Kapsamı

Günümüzde dijital içerik tüketiminin hızla artması, kullanıcıların film, dizi ve sinema salonu seçimi yaparken karar verme süreçlerini zorlaştırmaktadır. "CineCompare" projesinin temel amacı; kullanıcıların vizyondaki veya popüler filmleri inceleyebildiği, sinema salonlarındaki seans ve fiyat bilgilerini tek bir ekranda karşılaştırabildiği, yapay zeka desteğiyle kişiselleştirilmiş öneriler alabildiği modern bir web platformu geliştirmektir. Proje, uçtan uca (Full-Stack) olarak geliştirilmiş olup, arka planda güvenli ve hızlı bir API hizmeti, ön yüzde ise etkileşimli bir kullanıcı deneyimi sunmaktadır.

1.2. Projenin Hedef Kitlesi

Projenin hedef kitlesi; sinema sektörünü yakından takip eden, hangi filme gideceğine karar vermekte zorlanan ve bütçesine en uygun sinema seansını arayan tüm internet kullanıcılarıdır. Ayrıca, platformun içeriğini yönetebilmek adına yetkilendirilmiş sistem yöneticileri (Adminler) de projenin doğrudan kullanıcıları arasındadır.

1.3. Proje Kısıtları ve Sınırlandırmalar

- **Yapay Zeka Erişimi:** Sistemdeki akıllı öneri motoru, Google Gemini AI API'sine bağımlıdır. API'nin ücretsiz sürümü kullanıldığı için saatlik sorgu kotaları mevcuttur.
- **Canlı Veri Sınırlaması:** Sistemdeki sinema salonu fiyatları ve bilet bilgileri, eğitim amaçlı geliştirme ortamında simüle edilmiş (mock) verilerle desteklenmektedir; gerçek zamanlı bilet satış sistemleriyle (Biletix vb.) entegrasyon projenin ilerleyen aşamalarına bırakılmıştır.
- **Dosya Yönetimi:** Filmlere ait afiş görselleri sunucuya yüklenmek yerine doğrudan URL bağlantıları aracılığıyla çekilmekte olup, sunucu depolama maliyeti sıfıra indirilmiştir.

2. KULLANILAN TEKNOLOJİLER VE YAZILIM MİMARİSİ

Proje, yazılım mühendisliği prensiplerine uygun olarak Katmanlı Mimari (N-Tier Architecture) üzerine inşa edilmiştir.

2.1. Arka Plan (Backend) Mimarisi (.NET Core, Repository & Unit of Work Pattern)

Arka plan servisleri C# programlama dili ve .NET 10.0 altyapısı kullanılarak "Web API" standartlarında geliştirilmiştir. Projede kod tekrarını önlemek, test edilebilirliği artırmak ve veritabanı işlemlerini soyutlamak amacıyla **Generic Repository Design Pattern** uygulanmıştır. Ayrıca, birden fazla veritabanı işleminin (örneğin film eklerken log atılması) tek bir işlem bloğunda güvenle (Transaction) tamamlanması veya hata anında geri alınması (Rollback) için **Unit of Work Pattern** kullanılmıştır.

2.2. Veritabanı ve Veri Yönetimi (Entity Framework Core & Dapper)

Projede veri saklama ortamı olarak Microsoft SQL Server kullanılmıştır. ORM (Object-Relational Mapping) aracı olarak **Entity Framework Core** ile Code-First yaklaşımı benimsenmiştir. Bununla birlikte, veritabanından yüksek performanslı okuma (Select) işlemi gerektiren çoklu tablo birleştirme (Join) operasyonlarında, EF Core'a kıyasla daha hafif ve hızlı olan mikro-ORM **Dapper** kullanılmış; projenin ihtiyacına göre iki teknoloji hibrit bir yapıda harmanlanmıştır.

2.3. Ön Yüz (Frontend) Mimarisi (React, Vite, Component Yapısı)

Uygulamanın arayüzü, Facebook tarafından geliştirilen **React.js** kütüphanesi kullanılarak oluşturulmuştur. Modern bir derleme aracı olan **Vite** sayesinde geliştirme süreci hızlandırılmıştır. Stil süreçlerinde **TailwindCSS** framework'ü tercih edilerek; karanlık tema (Dark Mode), cam efekti (Glassmorphism) ve mobil cihazlarla tam uyumlu duyarlı (Responsive) bir tasarım inşa edilmiştir. React bileşenleri (Components) tekrar kullanılabilir şekilde tasarlanmış, durum yönetimi (State Management) React Hooks ile sağlanmıştır.

2.4. Güvenlik Katmanı (JWT Authentication & Authorization)

Sistemdeki yetkilendirme işlemleri **JSON Web Token (JWT)** mekanizması ile sağlanmıştır. Kullanıcıların şifreleri veritabanına açık metin (Cleartext) olarak kaydedilmemiş, Microsoft Identity kütüphanesi ile şifrelenerek (Hash) güvence altına alınmıştır. Sisteme giriş yapan (Login) bir kullanıcıya sunucu tarafından imzalanmış bir JWT iletilir ve kullanıcı yapmak istediği API isteklerinde bu token'ı doğrulama aracı olarak kullanır.

3. PROJENİN ÖNE ÇIKAN SİSTEM ÖZELLİKLERİ

3.1. Akıllı Önbellekleme (MemoryCache) Sistemi

Yüksek trafikli durumlarda veritabanı sunucusunun (SQL Server) yorulmasını engellemek amacıyla **IMemoryCache** entegrasyonu sağlanmıştır. Örneğin, kullanıcıların sıklıkla görüntülediği "Vizyondaki Filmler" listesi veritabanından çekildikten sonra 10 dakika boyunca sunucu RAM'inde tutulmakta, aynı listeyi talep eden diğer kullanıcılara veritabanına gidilmeden milisaniyeler içerisinde cevap dönülmektedir.

3.2. Merkezi ve Dosya Tabanlı Loglama (Serilog)

Sistemde oluşan API isteklerinin başarı durumları, çalışma süreleri ve olası hata mesajlarının izlenebilirliğini sağlamak adına **Serilog** kütüphanesi kurulmuştur. Standart konsol loglamasının yanı sıra, tüm kayıtlar logs/cinecompare-log-{Tarih}.txt formatıyla günlük olarak metin dosyalarına basılmaktadır. Bu yapı, sistemin bakımını ve hata tespit (Debugging) sürecini büyük oranda kolaylaştırmaktadır.

3.3. Google Gemini Yapay Zeka (AI) Entegrasyonu

Platformun en yenilikçi özelliklerinden biri yapay zeka tabanlı sohbet ve öneri robotudur. Güvenlik standartları gereği yapay zeka (Google Gemini) çağrıları doğrudan tarayıcı üzerinden değil, .NET Backend içindeki AiController üzerinden geçecek şekilde tasarlanmıştır. Bu yapı sayesinde sistemdeki API şifreleri korunmuş ve kullanıcıların metin bazlı film tavsiyesi istekleri hızlıca karşılanmıştır.

3.4. Dinamik Yönetim (Admin) Paneli (CRUD İşlemleri)

Projenin tamamen yönetilebilir olması amacıyla gelişmiş bir Yönetim Paneli geliştirilmiştir. MovieController içerisinde yazılan HTTP POST ve HTTP DELETE metodları sayesinde; sisteme yetkili olarak giriş yapan bir kullanıcı, React ön yüzündeki arayüzü kullanarak yeni filmler ekleyebilmekte ve süresi dolan filmleri sistemden saniyeler içerisinde tamamen silebilmektedir.

4. SİSTEM KURULUMU

4.1. Geliştirme Ortamı Gereksinimleri

- Backend API için: **.NET 10.0 SDK**
- Frontend Arayüzü için: **Node.js (v18.0 ve üzeri)**
- Veritabanı için: **Microsoft SQL Server LocalDB**

4.2. Adım Adım Kurulum ve Çalıştırma

Backend API Çalıştırılması:

1. Proje dizininde yer alan CineCompare.Web klasörüne gidilir.
2. appsettings.json dosyası içerisinde DefaultConnection bağlantı cümlesi ve JWT güvenlik anahtarları doğrulanır.
3. Terminalde dotnet run komutu çalıştırılır. API sunucusu ayağa kalkarak istekleri beklemeye başlar.

Frontend React Çalıştırılması:

1. Proje dizininde yer alan CineCompare.UI klasörüne gidilir.
2. Projeye ilk defa giriliyorsa terminalde npm install komutu çalıştırılarak bağımlı kütüphaneler indirilir.
3. Sonrasında npm run dev komutu çalıştırılır ve arayüz ayağa kalkar.
4. Herhangi bir web tarayıcısından <http://localhost:3000> adresine girildiğinde, sistem arka plan (API) ile tam senkronize bir şekilde çalışmaya başlayacaktır.

5. SONUÇ VE GELECEK ÇALIŞMALAR

"CineCompare" projesi, güncel web programlama teknolojilerinin uyum içerisinde nasıl kullanılabileceğini kanıtlayan, modüler ve güvenli bir bitirme projesidir. Proje sürecinde; veritabanı tasarımı, katmanlı mimari, güvenli kimlik doğrulama (JWT), önbellekleme (Caching), profesyonel loglama ve yapay zeka (AI) gibi günümüz yazılım endüstrisinin aktif olarak kullandığı konseptler başarıyla uygulanmıştır.

Gelecek Çalışmalar: Projenin bir sonraki aşamasında, gerçek sinema firmalarının API'leri (API Entegrasyonu) platforma dahil edilerek canlı bilet satış işlemleri gerçekleştirilebilir. Ayrıca, Identity altyapısı genişletilerek yetkilendirme (Role-based Authorization) özellikleri detaylandırılabilir.

6. KAYNAKÇA

- [1] Microsoft .NET Architecture Guides, "N-Tier and Repository Patterns", <https://learn.microsoft.com/en-us/dotnet/architecture/>
- [2] React Documentation, "Building Interactive UIs", <https://react.dev/>
- [3] TailwindCSS, "Rapidly build modern websites without ever leaving your HTML", <https://tailwindcss.com/>
- [4] Entity Framework Core & Dapper, "Micro-ORMs vs Full ORM", <https://learn.microsoft.com/en-us/ef/core/>
- [5] Google Gemini AI, "Generative AI API Documentation", <https://ai.google.dev/>
- [6] Serilog, "Structured Logging for .NET", <https://serilog.net/>