



**TRAKYA ÜNİVERSİTESİ  
MÜHENDİSLİK FAKÜLTESİ  
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**

**MOBİL UYGULAMA GELİŞTİRME I  
PROJE RAPORU**

**PROJE ADI:**

Veri Listesi Yönetimi

**PROJE EKİBİ:**

1221602040-TAHA DEMİR

**PROJE DANIŞMANI:**

CEM TAŞKIN

**Edirne – 2025**

**İÇİNDEKİLER**

1. PROJE ÖZETİ VE AMACI .....	1
2. KULLANILAN TEKNOLOJİLER VE MİMARİLER.....	1
3. UYGULAMA ÖZELLİKLERİ .....	3
4. KULLANICI ARAYÜZÜ VE TASARIM.....	4
5. SONUÇ.....	7
6. KAYNAKÇA.....	8

## 1. PROJE ÖZETİ VE AMACI

Uygulama "Film Veri Yönetim Sistemi" olarak tanımlanabilir. Jetpack Compose kullanılarak geliştirilen bu proje, "Veri Listesi Yönetimi" konusunu sadece listeleme bazında değil, verinin eklenmesi, filtrelenmesi, kategorize edilmesi ve detaylandırılması gibi gereksinimleri kapsayacak şekilde ele alıyor.

## 2. KULLANILAN TEKNOLOJİLER VE MİMARİLER

Film veri yönetim sistemi, performans ve sürdürülebilirlik açısından en güncel Android Jetpack kütüphaneleri kullanılarak, "**Veri Listesi Yönetimi**" prensipleri çerçevesinde geliştirilmiştir:

- **Kotlin:** Android'in resmi dili olan Kotlin'in modern söz dizimi ve **Null Safety** özellikleri kullanılarak, bellek sızıntıları ve uygulama çökmeleri minimize edilmiş, güvenli bir kod tabanı oluşturulmuştur.
- **Jetpack Compose:** Kullanıcı arayüzü tamamen deklaratif yöntemle geliştirilmiştir. Bu sayede karmaşık liste yapılarında bile yüksek performanslı, akıcı ve modern bir tasarım dili elde edilmiştir.
- **VVM Mimarisi (Model-View-ViewModel):** Uygulamanın iş mantığı (Logic) ile kullanıcı arayüzü (UI) tamamen birbirinden ayrılmıştır. Bu mimari, verinin tek bir kaynaktan yönetilmesini ve kodun modüler yapıda kalmasını sağlamıştır.
- **Room Database & Kotlin Flow:** Yerel veri depolama için SQLite tabanlı Room tercih edilmiştir. Veritabanındaki tüm değişimler **Flow** aracılığıyla asenkron olarak dinlenir; böylece bir veri silindiğinde veya eklendiğinde arayüz manuel işleme gerek kalmadan "tepkisel" (reactive) olarak güncellenir.
- **Coil & HTTP Header Entegrasyonu:** Afiş görsellerinin dinamik yüklenmesi için Coil kütüphanesi kullanılmıştır. Dış kaynaklı sunucuların güvenlik protokollerini aşmak adına **User-Agent (Chrome)** başlıklarını eklenderek, verinin sürekliliği ve başarılı şekilde çekilmesi garanti altına alınmıştır.
- **Kotlin Coroutines:** Arama işlemleri, filtreleme ve veritabanı kayıtları gibi ağır işlemler, ana arayüzü (Main Thread) dondurmadan arka planda asenkron olarak yürütülür.

- **Gelişmiş Arama ve Filtreleme Algoritmaları:** Uygulama içerisinde film adı, oyuncu kadrosu ve tür bazlı eşzamanlı filtreleme yapılmaktadır. Bu yapı, veri listesi yönetiminde kullanıcının aradığı veriye en kısa yoldan ulaşmasını sağlayan akıllı bir arama motoru ile desteklenmiştir.
- **Dinamik Kategorizasyon:** Ham veri listesi, tür bilgilerine göre çalışma anında (runtime) otomatik olarak grupperlərlərək kullanıcıya organize bir kütüphane deneyimi sunacak şekilde tasarlanmıştır.

### 3. UYGULAMA ÖZELLİKLERİ

**Dinamik Veri Kategorizasyonu:** Uygulama, veritabanındaki ham film listesini otomatik olarak analiz eder ve tür (Aksiyon, Dram, vb.) bilgilerine göre grupperler. Bu sayede kullanıcıya karmaşık bir liste yerine, düzenli ve yatay kaydırılabilir (LazyRow) kategorilerden oluşan profesyonel bir kütüphane arayüzü sunulur.

**Gelişmiş Arama ve Filtreleme Sistemi:** Veri Listesi Yönetimi'nin en kritik parçası olan arama mekanizması şu özellikler içerir:

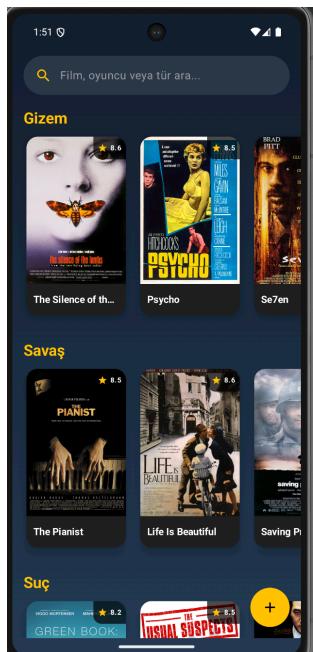
- **Çok Kanallı Arama:** Sadece film ismine göre değil, oyuncu kadrosuna ve film türüne göre de eş zamanlı filtreleme yapar.
- **Arama Geçmişi (Search History):** Kullanıcının önceki aramalarını yerel hafızada (SharedPreferences/DataStore) saklayarak hızlı erişim imkanı tanır.
- **Anlık Geri Bildirim:** Kullanıcı yazdığı anda (Real-time) liste güncellenir ve sonuç bulunmadığında kullanıcıya özel uyarı ekranı gösterilir.

**Akıllı Veri Girişi ve Düzenleme (CRUD):** Yeni veri ekleme süreci, hatalı girişleri minimize edecek şekilde tasarlanmıştır:

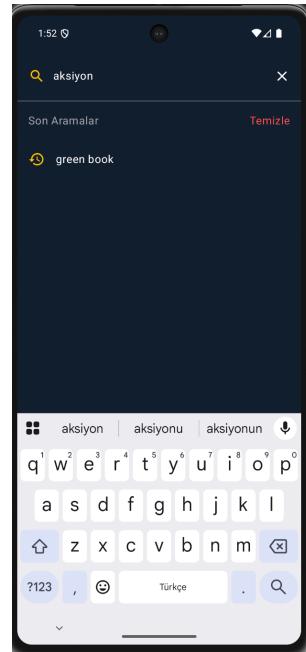
- **Dinamik Chip Yapısı:** Oyuncu isimleri ve film türleri, modern "Chip" bileşenleri kullanılarak listeye eklenir veya listeden kolayca silinebilir.
- **Görsel Önizleme:** Film afişi için bir URL girildiğinde, kaydedilmeden önce anlık olarak önizleme penceresinde görüntülenir.

**Tepkisel Silme ve Onay Mekanizmaları:** Veri kaybını önlemek amacıyla, bir öğe silinmek istendiğinde (uzun basma ile tetiklenir) kullanıcıya modern bir onay diyalogu gösterilir. Silme işlemi onaylandığında, veritabanındaki değişim **Kotlin Flow** üzerinden tüm ekranlara anında yansıtılır.

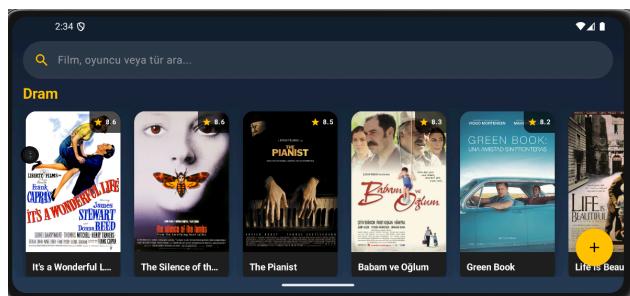
## 4. KULLANICI ARAYÜZÜ VE TASARIM



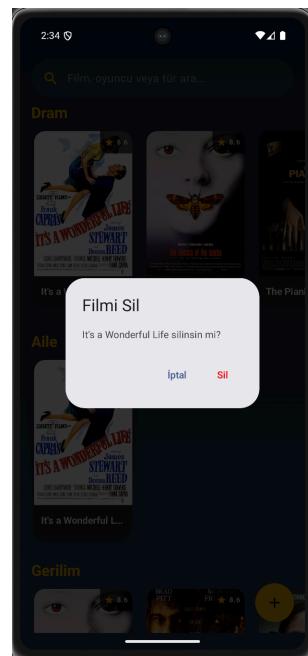
Şekil 4.1.Anasayfa



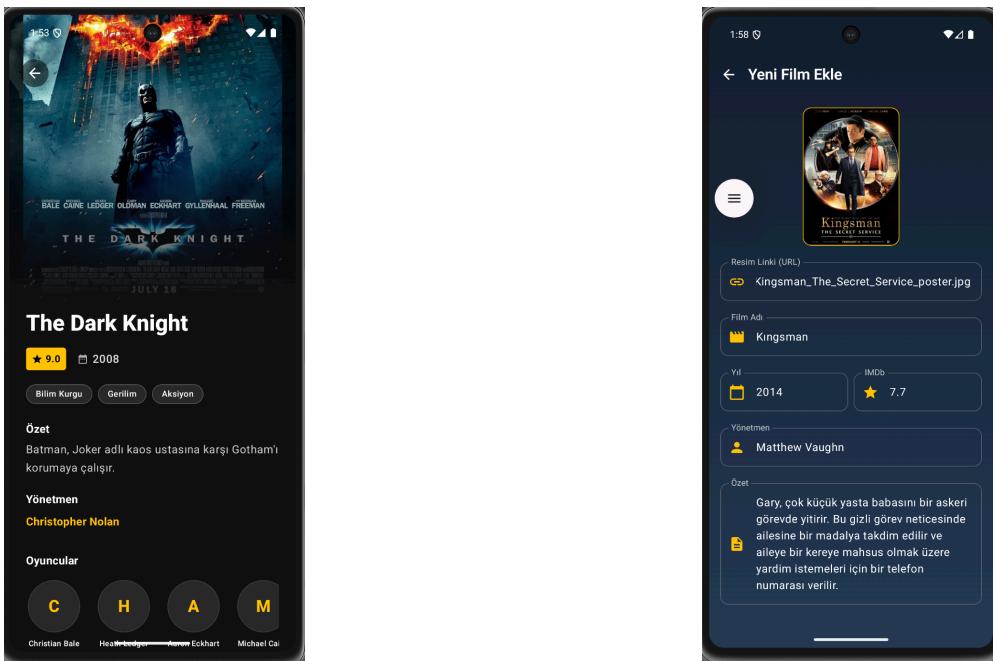
Şekil 4.2.Arama sayfası



Şekil 4.3.Anasayfa yan çevrilmiş

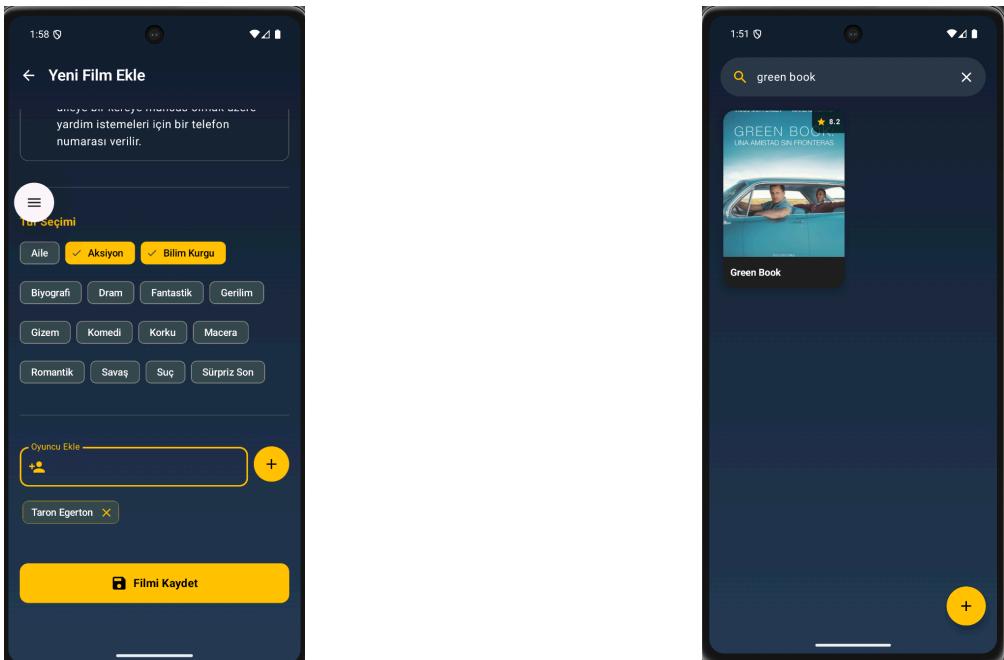


Şekil 4.4.Film silme pop-up



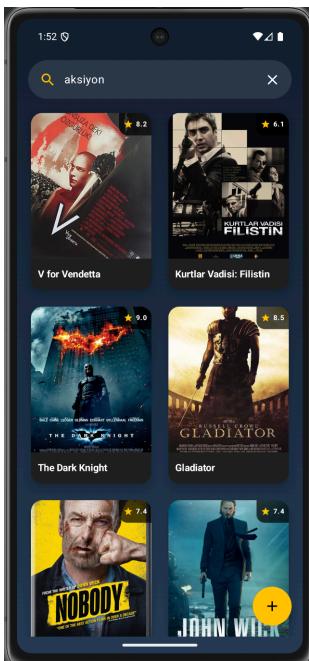
**Şekil 4.5.**Film detay sayfası

**Şekil 4.6.**Film ekleme sayfası



**Şekil 4.7.**Film ekleme sayfası devamı

**Şekil 4.8.**Filmleri isme göre arama örneği



**Şekil 4.8.**Filmleri kategorilere göre arama örenğı

## **5. SONUÇ**

Uygulama, "Veri Listesi Yönetimi" ana başlığı altında; verinin yerel veritabanında saklanması ,asenkron olarak işlenmesi ve kullanıcıya kategorize edilmiş bir hiyerarşi ile sunulması süreçlerini uçtan uca kapsamaktadır.Proje kapsamında geliştirilen akıllı arama algoritması ve dinamik tür eşleştirme sistemi, büyük veri setlerinin kullanıcı tarafından kolayca yönetilebileceğini kanıtlamıştır.Projenin ilerleyen aşamalarında, bir API entegrasyonu yapılarak verilerin otomatik olarak bulut üzerinden çekilmesi ve kullanıcılara izleme alışkanlıklarına göre yapay zeka destekli film önerileri sunan bir Öneri Motoru eklenmesi hedeflenmektedir.

## 6. KAYNAKÇA

- **Google Developers.** (2025). *Jetpack Compose: Android's modern toolkit for building native UI*. Erişim adresi: <https://developer.android.com/jetpack/compose>
- **Kotlin Foundation.** (2025). *Kotlin Programming Language Documentation*. Erişim adresi: <https://kotlinlang.org/docs/home.html>
- **Google Developers.** (2025). *Guide to App Architecture (MVVM)*. Erişim adresi: <https://developer.android.com/topic/architecture>
- **Google Developers.** (2025). *Android SpeechRecognizer & TextToSpeech API*. Erişim adresi: <https://developer.android.com/reference/android/speech/package-summary>
- **Coil Library.** "Image Loading for Android with Kotlin Coroutines." [coil-kt.github.io](https://coil-kt.github.io)