

# KOCAELİ ÜNİVERSİTESİ BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

## YAZILIM LAB. I — II.Proje

### WEB TABANLI SOSYAL KÜTÜPHANE PLATFORMU

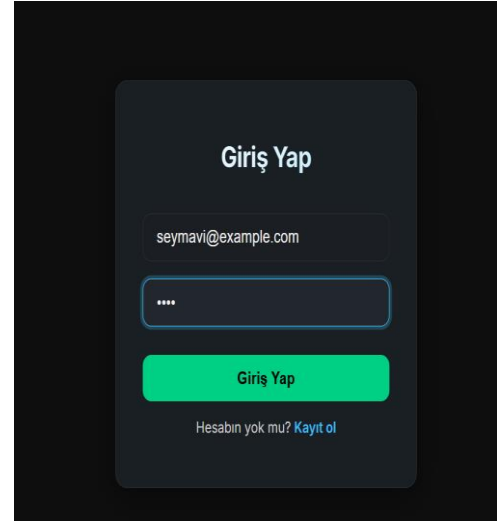
Meryem Şeyma Gençer 240201118

Ecesu Yıldırım 240201121

## ÖZET

Bu çalışmada, kullanıcıların kitap ve film içeriklerini keşfedebildiği, puanlayabildiği, yorumlayabildiği ve kişisel listeler oluşturarak sosyal bir akış üzerinden diğer kullanıcılarla etkileşim kurabildiği bir Web Tabanlı Sosyal Kütüphane Platformu geliştirilmiştir. Sistem; FastAPI tabanlı RESTful bir backend, React tabanlı modern bir frontend ve PostgreSQL veri tabanı üzerinde çalışmaktadır. İçerik verileri TMDb ve Google Books API'lerinden dinamik olarak elde edilmektedir. Proje kapsamında kullanıcı yönetimi, sosyal akış, etkileşim özellikleri (beğeni/yorum), listeleme sistemi ve içerik detay yönetimi gibi fonksiyonlar geliştirilmiştir. Çalışma; mimari tasarım, veri tabanı modellemesi, algoritmik süreçler ve deneysel performans çıktılarıyla birlikte sunulmaktadır.

Bu hedef doğrultusunda sistem; frontend, backend ve veri tabanı katmanlarından oluşan çok katmanlı bir mimari ile geliştirilmiştir.



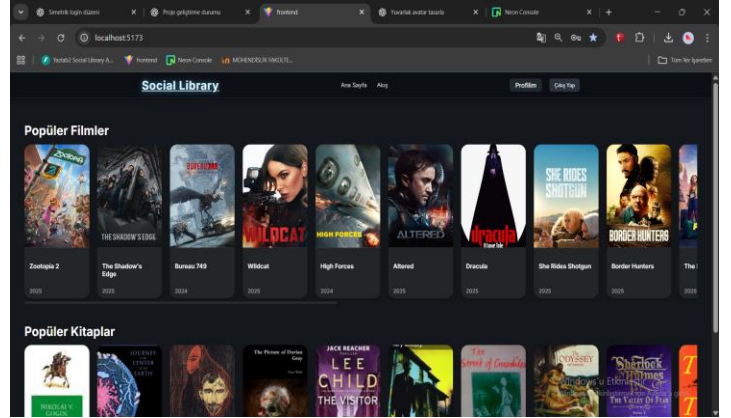
Görsel-1: Web tabanlı platformun giriş ekranı.

## GİRİŞ

Dijital içerik tüketiminin artmasıyla kullanıcıların kitap ve film gibi kültürel öğeleri takip edebileceği, listeleyebileceği ve sosyal etkileşim sağlayabileceği çevrimiçi platformlara olan ihtiyaç yükselmiştir. Bu proje, Kocaeli Üniversitesi Yazılım Laboratuvarı-I kapsamında verilen “Web Tabanlı Sosyal Kütüphane Platformu” gereksinimlerini karşılamak amacıyla geliştirilmiştir.

Projenin temel amaçları şunlardır:

- Kullanıcıların kişisel kütüphaneler oluşturabilmesini sağlamak,
- Beğendikleri içerikleri puanlayabilmesini ve yorumlayabilmesini sağlamak,
- Takip sistemiyle sosyal bir akış oluşturmak,
- Harici API'ler üzerinden dinamik içerik toplayabilmek,
- Modern, ölçeklenebilir ve güvenli bir web platformu sunmak.



Görsel -2: Popüler film ve kitapların gösterildiği anasayfa.

## YÖNTEM

### Sistem Mimarisi

Sistem üç ana bileşenden oluşmaktadır:

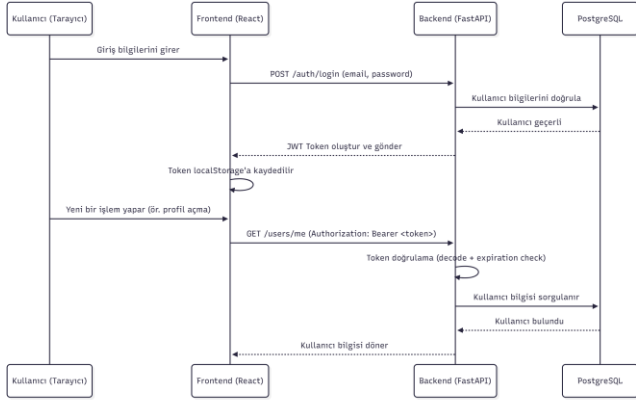
#### a) Frontend (React)

Sistemin istemci tarafı, yüksek performanslı ve komponent tabanlı yapısıyla modern web geliştirme için standart hâline gelen React.js kullanılarak oluşturulmuştur. Arayüz; yeniden

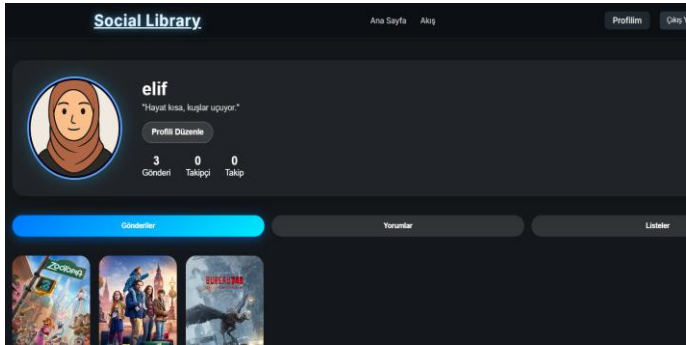
kullanılabilir komponentlerden, sayfa yönlendirme yapısından ve API servislerine erişimi soyutlayan bir istemci katmanından oluşmaktadır.

Frontend katmanında kullanıcı arabirimi (UI) tasarlanırken hem kullanılabilirlik hem de görsel tutarlılık dikkate alınmış, bileşen-temelli modüler bir yapı tercih edilmiştir. Kullanıcı ile sunucu arasındaki tüm veri alışverişi Axios üzerinden yapılmakta ve istekler ortak bir servis yapısı üzerinden yönetilmektedir. Bu sayede API adresleri merkezi olarak kontrol edilebilmekte, kimlik doğrulama token'ı otomatik olarak tüm isteklere eklenebilmekte, hata yönetimi tek bir noktadan düzenlenebilmektedir.

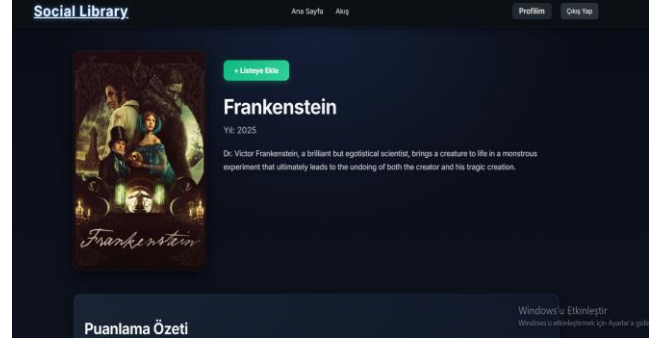
Kullanıcı kimlik doğrulaması, web sistemlerinde yaygın olarak kullanılan JWT (JSON Web Token) tabanlı Bearer Token mekanizması ile sağlanmıştır. Kullanıcı giriş yaptığında backend tarafından üretilen token, tarayıcıda saklanmakta ve tüm yetkili isteklerde HTTP Authorization başlığına otomatik olarak eklenmektedir.



Şekil-1: JWT tabanlı kimlik doğrulama sürecinin akış diyagramı.



Görsel - 3: Kullanıcının Profilini gösteren UserProfile sayfası.



Görsel - 4: Film ve Kitaplar hakkındaki bilgileri, puanları ve kullanıcıların yaptığı yorumları gösteren ItemDetail sayfası.

## b) Backend (FastAPI)

Sistemin sunucu tarafı, modern Python ekosisteminde yüksek performans ve asenkron işlem desteğiyle öne çıkan FastAPI framework'ü kullanılarak geliştirilmiştir. Backend katmanı; kullanıcı işlemleri, liste yönetimi, içerik etkileşimleri, harici API bağlantıları ve sosyal akış hesaplamaları gibi tüm iş mantığını kapsayan modüler bir yapıdadır.

FastAPI'nin sağladığı otomatik şema üretimi, Swagger UI dokümantasyonu, asenkron çalışma modeli ve yüksek performanslı request işleme kapasitesi, sistemin genişlemeye açık bir mimari ile geliştirilmesine olanak tanımıştır.

Backend, fonksiyonel açıdan bağımsızlaştırılmış router yapılarıyla organize edilmiştir. Her bir işlev grubu farklı bir endpoint dosyası olarak tasarlanmıştır:

/auth → Kullanıcı kayıt, giriş, şifre sıfırlama

/items → İçerik detaylarının yönetimi ve harici API entegrasyonları

/lists → Liste oluşturma, listeye içerik ekleme, varsayılan listelerin yönetimi

/rating → Puanlama işlemleri

/review → Yorum işlemleri

/activity → Sosyal akış, beğeni ve yorum aktivitelerinin kaydı

Bu modüler yapı sayesinde hem okunabilirlik hem de sürdürülebilirlik artırılmıştır.

Backend içinde tüm kullanıcı doğrulama süreci JWT token çözümü (decode) mekanizması üzerinden kontrol edilmektedir. Her istek geldiğinde token doğrulanır, geçersiz veya süresi dolmuş token'lar reddedilir. Bu yaklaşım, sistem güvenliği açısından önem taşımaktadır.

### c) Veri Tabanı (PostgreSQL)

Sistemin veri tabanı, ilişkisel bütünlüğü ve ölçeklenebilirliği ile bilinen **PostgreSQL** üzerinde inşa edilmiştir. Veri modelleme için **SQLAlchemy ORM** tercih edilmiş ve tablolar arasında güçlü ilişkiler tanımlanmıştır.

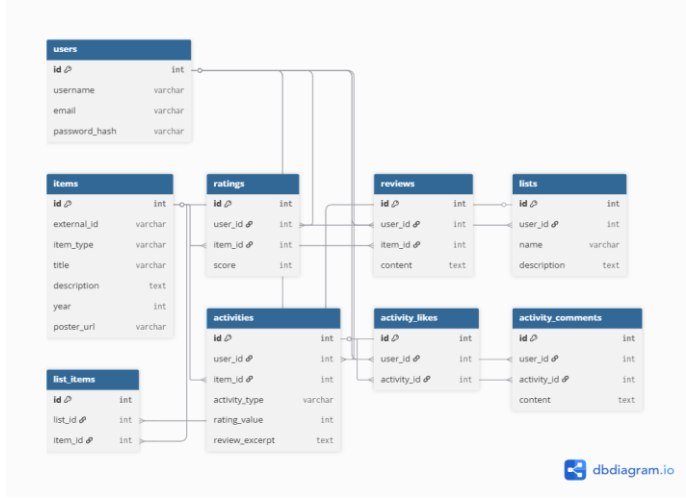
Modelleme sürecinde kullanıcı-içerik etkileşimleri göz önünde bulundurularak aşağıdaki temel tablolar oluşturulmuştur:

- **Users:** Kullanıcı profilleri ve kimlik doğrulama bilgileri
- **Items:** Kitap ve film içerik bilgileri
- **Lists:** Kullanıcının içerikleri gruplayabildiği listeler
- **ListItems:** Bir listeye eklenen içeriklerin eşleştirme tablosu
- **Ratings:** Kullanıcıların içeriklere verdiği puanlar
- **Reviews:** Kullanıcı yorumları
- **Activities:** Sistem genelindeki sosyal akış verileri
- **ActivityLikes / ActivityComments:** Aktivitelere verilen etkileşimler

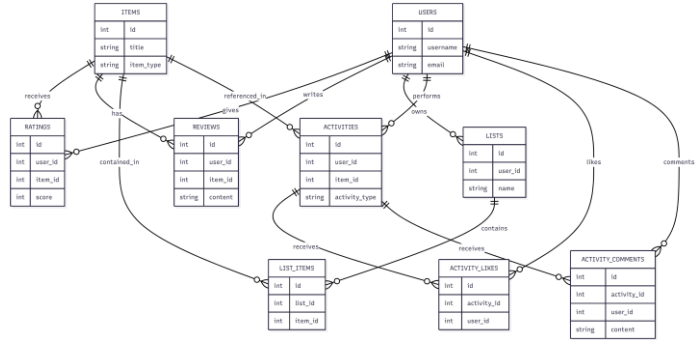
Bu veri yapısı, sosyal medya benzeri bir içerik akışının, kullanıcı listelerinin ve kişisel kütüphane yönetiminin bütüncül şekilde yürütülmesini sağlamaktadır.

Model ilişkileri esas alınarak oluşturulan ER ve UML Diyagramları Şekil- 2 ve Şekil-3'te sunulmuştur.

### Akış Diyagramları



Şekil-2: Projede kullanılan veri tabanı tablolarının ER diyagramı.

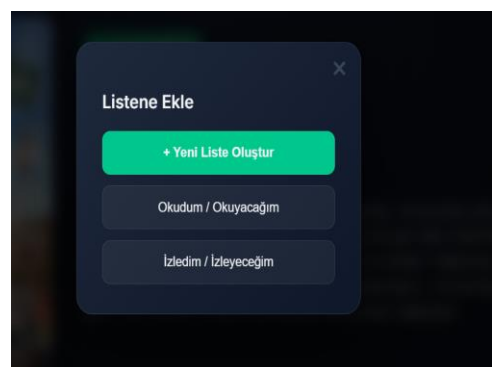


Şekil-3: Projede kullanılan veri tabanı tablolarının UML diyagramı.

### Pseudocode Örneği

#### Kullanıcı bir içeriği listeye eklediğinde çalışan süreç:

```
onClick(AddToListButton):
    token = loadJWT()
    send GET /lists/me
    if defaultList not exists:
        create system lists
    send POST /lists/{listId}/items/{itemId}
    if success:
        showToast("Başarıyla eklendi")
```



Görsel-5: Listeye ekle butonuna tıklandığında karşılaşılan ekran.

akışı sağlanmış, kullanıcılar arasında sosyal bir paylaşım deneyimi sunulmuştur.

Gelecek çalışmalar olarak:

- Öneri motoru (machine learning tabanlı içerik önerisi),
- Gerçek zamanlı bildirim sistemi,
- Mobil uygulama entegrasyonu

gibi geliştirmeler planlanabilir.

## SONUÇ

Bu proje ile; modern web mimarisine uygun, ölçeklenebilir ve sosyal etkileşim temelli bir kütüphane platformu başarıyla gerçekleştirilmiştir. Harici API entegrasyonları sayesinde zengin içerik