



T.C. PAMUKKALE ÜNİVERSİTESİ



# WEB TABANLI TEKNOLOJİLER

## Dönem Sonu Projesi

Kütüphane Bilgi Sistemi

Sinem Ertural

21253026

# Proje Gelişim Süreci

Projemin geliştirme sürecinde belirlediğim sıralama şu şekildedir:

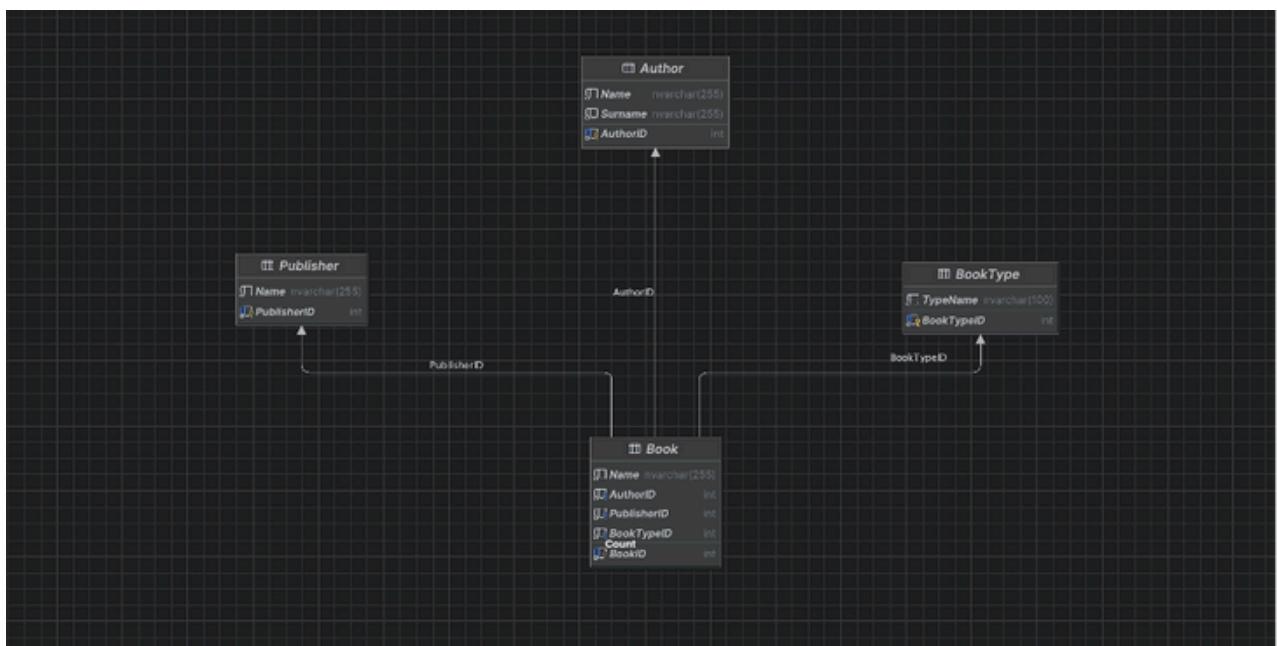
- 1) Veri Tabanı Tasarımı :** Bu görsel sayesinde elimde kaç tablo var , tablolar arasında hangi ilişkiler var ve tablodaki alanların ne olduğunu görebilirim.
- 2) Backend Geliştirme :** Projenin backend tarafını Dotnet Framework kullanarak oluşturdum.
- 3) Frontend Geliştirme :** Projenin kullanıcı arayüzü, React Framework kullanılarak oluşturdum.

Veri Tabanı Tablom şu şekilde ;

Book – Author arasında bire çok ilişki vardır.

Book – Publisher arasında bire çok ilişki vardır.

Book – BookType arasında çoka çok ilişki vardır.



Bu alanlar ve ilişkileri göz önünde bulundurarak projemin backendini oluşturdum. Önce appsetting.json dosyam içinde veri tabanımın bilgilerini yazdım. Veri tabanı olarak docker da kurduğum SQL container'ını çekiyorum. Aynı zamanda veri tabanımı bağlamak için Program.cs dosyamda değişiklikler yaptım. Daha sonra temiz kod ilkesine dayanarak projemin okunurluğunu arttırmak adına şu klasörleri oluşturdum;

- Services
- Models
- Dtос
- Controllers

Program.cs dosyamda bulunan bazı özel yaoılar şunlardır;

- **CORS**

Bu bağlantıyı sağlamamın sebebi http isteklerini kabul etmek.

```
//useCors bağlandı

builder.Services.AddCors(options =>
{
    options.AddDefaultPolicy(policy =>
    {
        policy.WithOrigins("http://localhost:5173", "https://localhost:5173").
            AllowAnyMethod().AllowAnyHeader().AllowCredentials();
    });
});
```

- **JWT Authentication**

Bu bağlantıyı sağlamamın sebebi kimlik doğrulama ve yetkilendirme yapabilmek.

Services klasörü içerikleri ;

- **AppDbContext.cs** : DbContext, veritabanı ile CRUD (Create, Read, Update, Delete) işlemlerini yapabilmek için kullanılır.

- **PasswordHasher.cs** : Bu sınıf, kullanıcı şifrelerini güvenli bir şekilde saklamak için hashleme işlemi yapar.

Models klasörü içerikleri ;

- Modelste veri tabanı alanları ve ilişkilerim bulunur.

- Bire çok (1:N kısmında N ilişkisine sahip yapıda) yada çoka çok bir ilişkim varsa listeleme işlemi kullanılır.

0 references

```
public virtual List<Book> Books { get; set; } = [];
```

Bire çok yada bire bir ilişkilerde aşağıdaki yapı kullanılır burada örneğin book.cs modelimin içinde bu yapıyı kullanıyorsam kastetmek istediğim bir kitap sadece bir yayinevine ait olabilir.

4 references

```
public required virtual Publisher Publisher { get; set; }
```

Dtos klasörü içerikleri ;

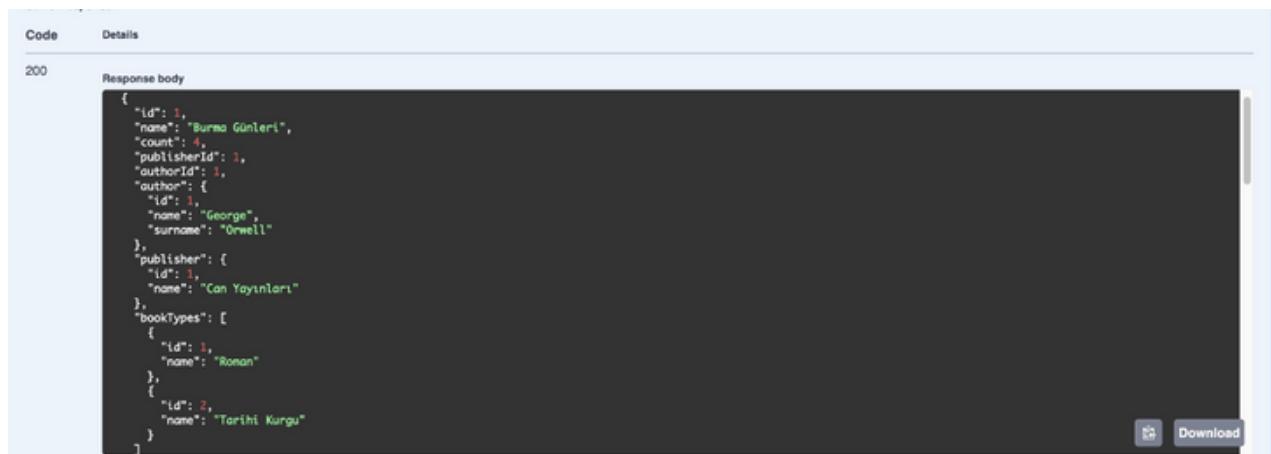
- Dto'lar sadece taşınmak istenen verilerin alındığı kısımlardır. Örneğin her alanın Id'si Dto'larda verilmez.

- Bir başka önemli nokta ise çoka çok ilişkim hangi iki alan arasındaysa Dtos kısmında ara bir tablo oluşturmalıyım. Veri tabanım arkada bu ilişkiye bana sağlayacaktır. Book ile BookType arasında bir çoka çok ilişkim olduğu için BookBookTypeDto.cs adında bir dosya oluşturdum.

Controllers klasörü içerikleri ;

- CORS işlemlerimi yaptığım kısım bu alanda bulunur. Her tablom için bir controllers oluşturduğum.
- Aynı zamanda [Authorize] işlemi ile sadece adminin yapabileceği kısımları yetkilendirdim.

Tüm bu kısımlar bittikten sonra swaggerda veri girişi işlemimi yaptım.



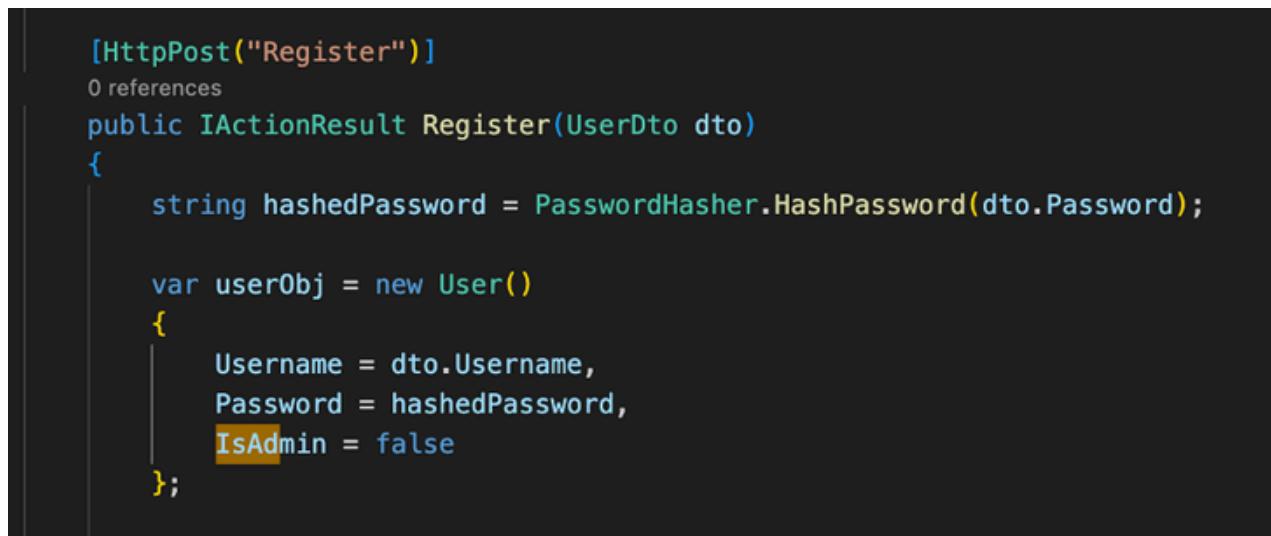
Code Details

200 Response body

```
{
  "id": 1,
  "name": "Burma G\u0111nleri",
  "count": 4,
  "publisherId": 1,
  "authorId": 1,
  "author": {
    "id": 1,
    "name": "George",
    "surname": "Orwell"
  },
  "publisher": {
    "id": 1,
    "name": "Can Yayınlari"
  },
  "bookTypes": [
    {
      "id": 1,
      "name": "Roman"
    },
    {
      "id": 2,
      "name": "Tarihi Kurgu"
    }
  ]
}
```

Download

Authentication kısmında login , logout , register kısımlarım vardır registerda oluşturduğum her kişi kullanıcı rolünde oluyor çünkü ;



```
[HttpPost("Register")]
0 references
public IActionResult Register(UserDto dto)
{
    string hashedPassword = PasswordHasher.HashPassword(dto.Password);

    var userObj = new User()
    {
        Username = dto.Username,
        Password = hashedPassword,
        IsAdmin = false
    };
}
```

Bu kısımda isAdmin = true yapıp bir admin oluşturduğum ve tekrar kodu eski haline getirdim.

Backendimden sonra yeni bir react projesi başlatıp projemin react kısmına başladım.

React projemde bulunan src dosyamın içinde components adlı bir klasör oluşturduğum ve tüm componentlerimi bu kısma yazdım. Aynı zamanda sitemde kullanacağım resimleri de public dizinin altında images klasörü oluşturduğum ve resimleri orada sakladım .

App.tsx içinde sayfa rotalarımı belirledim . Bunu path ve element ile yaptım.

```
        },
        {
          path: "/Books",
          element: <BookList />,
        },
      ],
    
```

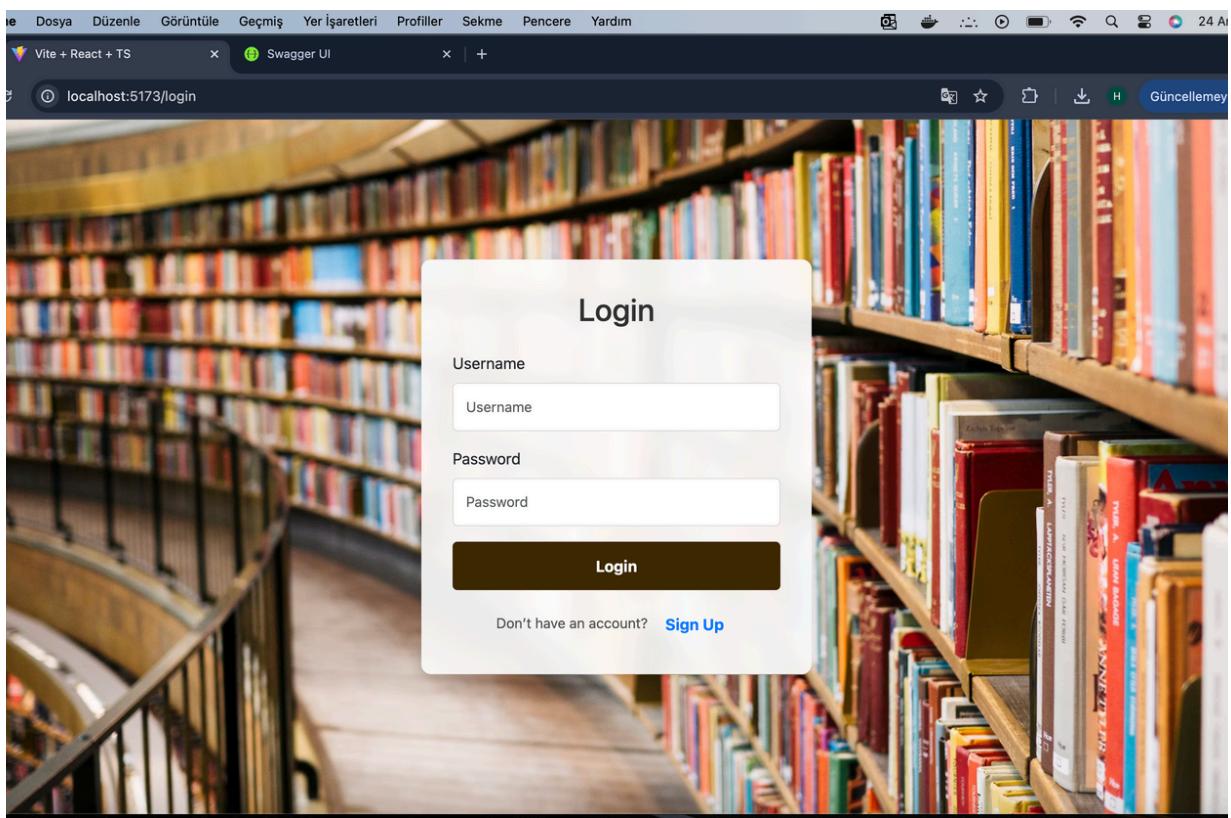
Swaggerdan çekmek istediğim her veri için bir fonksiyon yazdım fetch ile okudum hangi methodu yaptırmak istediysem bunu post, delete gibi ifade ettim.

```
function handleLogin() {
  fetch("http://localhost:5196/api/authentication/login", {
    method: "post",
    credentials: "include",
    headers: { "content-type": "application/json" },
    body: JSON.stringify({
      username: username,
      password: password,
    })
  })
}
```

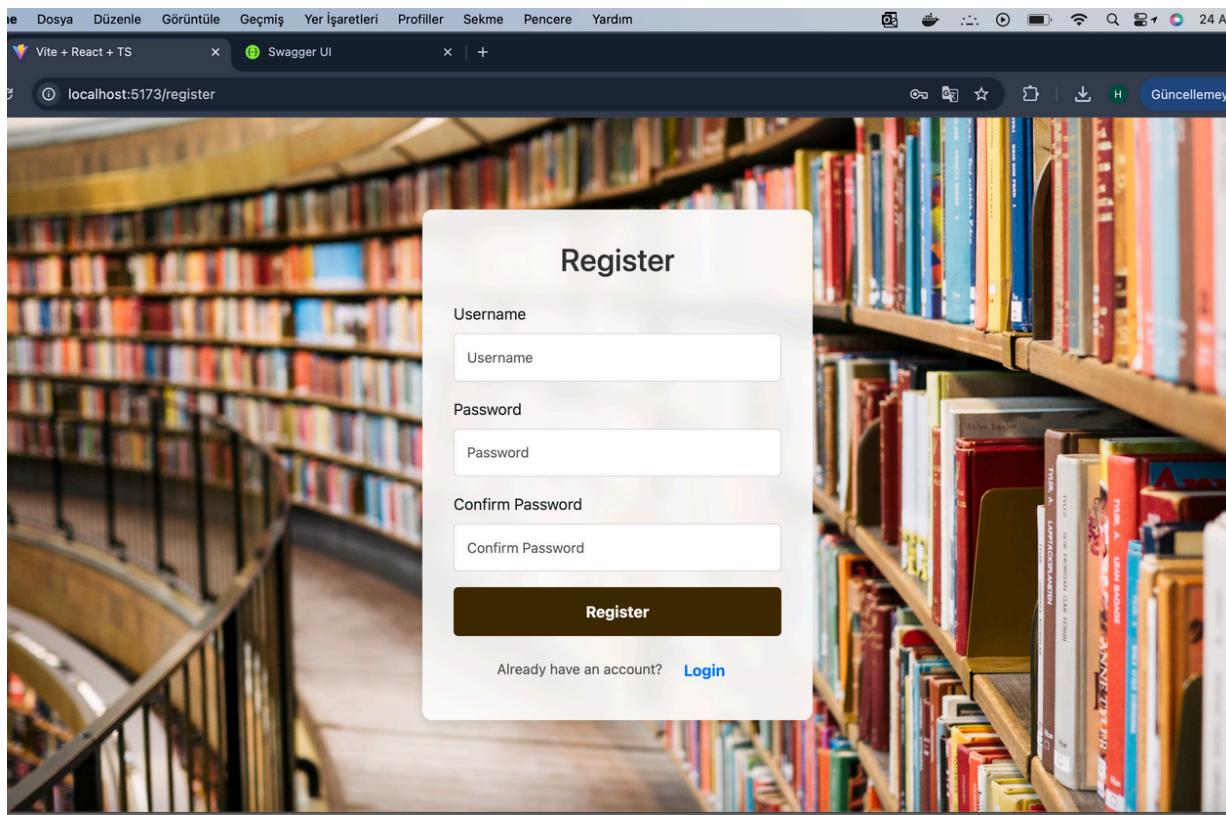
React projemde kullandığım hook ve fonksiyonlar;  
Bu Hook'lar ve fonksiyonlar, React uygulamamın etkileşimli ve dinamik bir şekilde çalışmasını sağladı ;

- Form işlemleri (handleSubmit) kullanıcı girdilerini yakalayıp backend'e göndermeyi sağlar.
- API işlemleri (handleFetch) uygulamanın verilerini güncel tutar.
- Kullanıcı deneyimi (handleClose) etkileşimli bileşenler sunar.
- State ve Effect yönetimi (useState ve useEffect) React bileşenlerinin veri akışını ve davranışını düzenler.

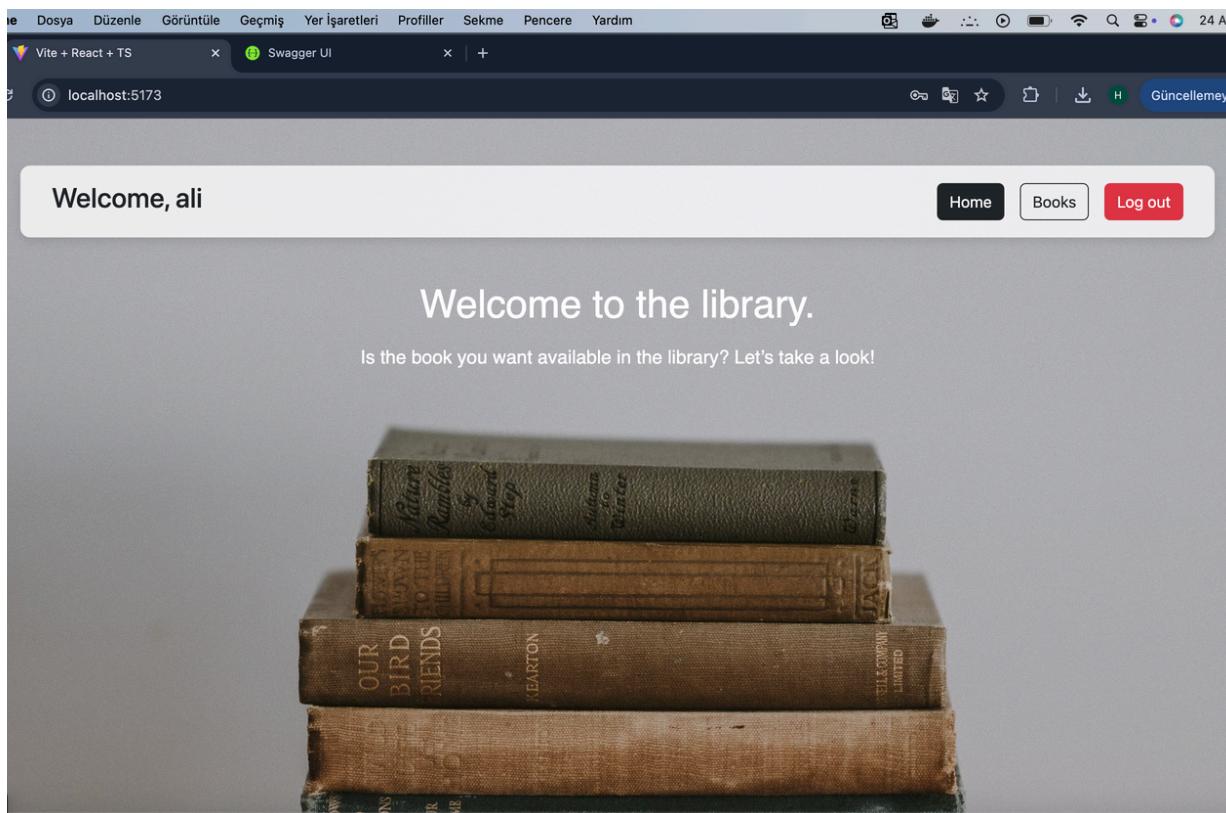
## Projemin Arayüz Tasarımı



bu kısımda kullanıcılar ve admin varsa hesapları giriş yapıyorlar eğer yoksa sign in yazısına tıklayıp kayıt ol sayfasında yeni hesap oluşturuyorlar.



Hesabı olmayan kullanıcılar için hesap oluşturma işlemi bu sayfada gerçekleşmektedir.



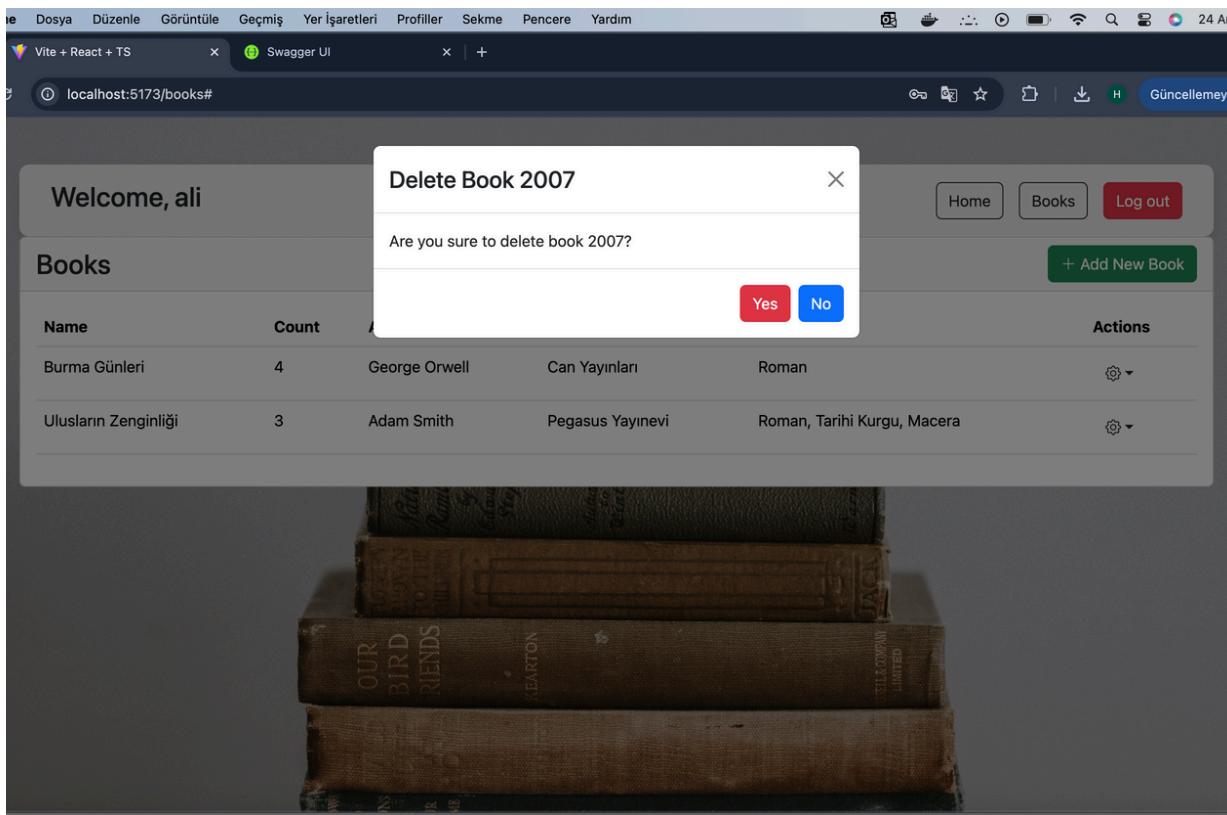
Giriş yapan kullanıcı ve admin anasayfaya yönlendirilir.

The screenshot shows a web application interface. At the top, there is a navigation bar with links like 'Dosya', 'Düzenle', 'Görüntüle', 'Geçmiş', 'Yer işaretleri', 'Profil', 'Sekme', 'Pencere', and 'Yardım'. Below the navigation bar, the address bar shows 'localhost:5173/books'. The main content area has a header 'Welcome, ali' and a 'Books' section. The 'Books' section contains a table with columns: Name, Count, Author, Publisher, Categories, and Actions. Two book entries are listed: 'Burma Günleri' (Count 4, Author George Orwell, Publisher Can Yayınları, Category Roman) and 'Uluslararası Zenginliği' (Count 3, Author Adam Smith, Publisher Pegasus Yayınevi, Categories Roman, Tarihi Kurgu, Macera). A green button '+ Add New Book' is located at the top right of the table. In the background, there is a large, semi-transparent image of several old, worn books stacked vertically.

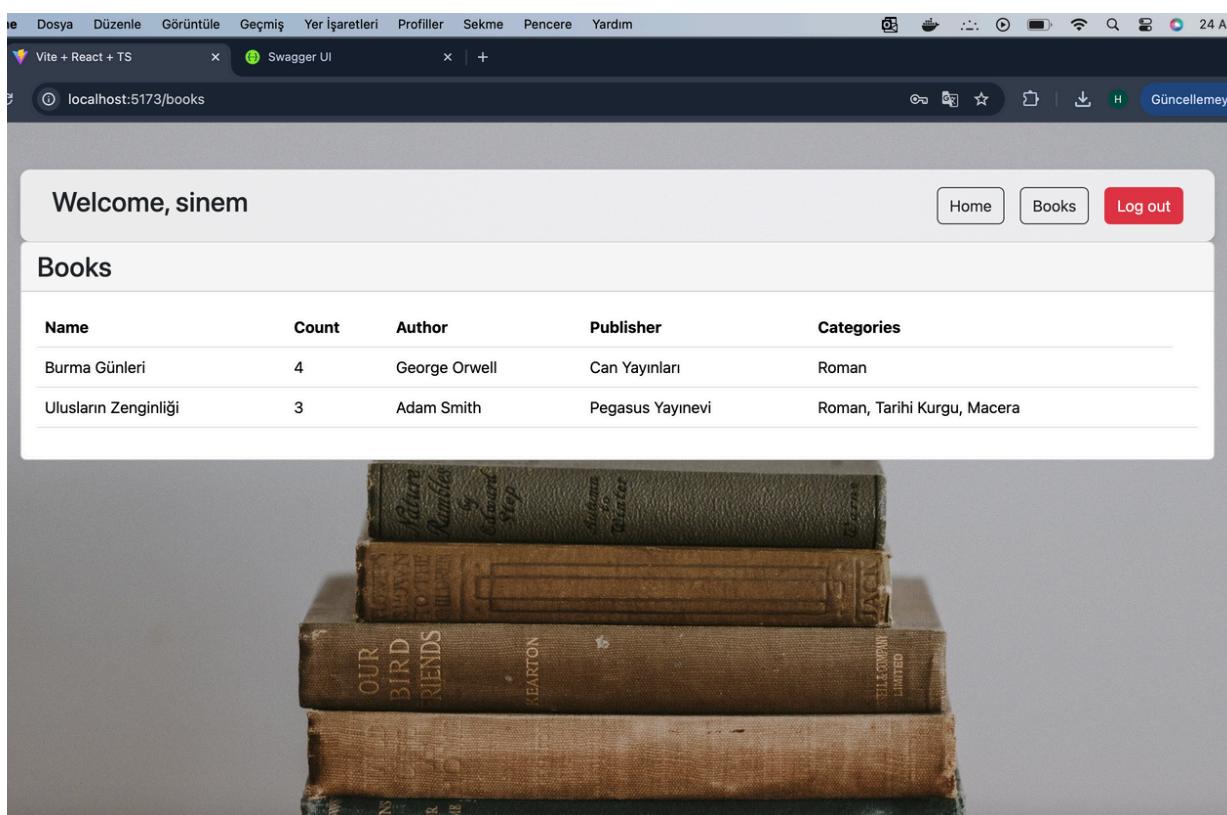
Eğer admin giriş yaptıysa kullanıcılarından farklı olarak yeni kitabı ekle ve aksiyon kısımlarında kitabı değiştir veya sil butonlarına tıklayarak işlem yapabilir.

The screenshot shows the same web application interface, but with a modal dialog box overlaid on it. The modal is titled 'Add New Book' and contains fields for 'Kitap Adı' (Book Name), 'Kitap Sayısı' (Book Count), 'Yazar' (Author), 'Yayınçı' (Publisher), and 'Kitap Kategorileri' (Book Categories). Under 'Kitap Kategorileri', there are three checkboxes: 'Roman', 'Tarihi Kurgu', and 'Macera'. A green 'Kaydet' (Save) button is at the bottom right of the modal. The background of the application shows the same book list and background image as the first screenshot.

Adminin değiştirmek istediği veya eklemek istediği kitaplar için doldurması gereken alanlar bu kısımda gözükmür.



Kitap eğer admin tarafından silinmek isterse bu şekilde bir uyarı mesajı alır.



Herhangi bir kullanıcı ise kitaplar sayfasında bu şekilde görüntüler herhangi bir ekleme , silme veya güncelleme işlemi yapamaz . Sadece kitap listesini görüntüler.

**Proje video linkim aşağıda bulunmaktadır ;**

[https://drive.google.com/file/d/1HFehFFrxQnpYxLYXgBrPMYcnJgG1crR0  
/view?usp=sharing](https://drive.google.com/file/d/1HFehFFrxQnpYxLYXgBrPMYcnJgG1crR0/view?usp=sharing)