

ANKARA ÜNİVERSİTESİ  
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

BLM4531-AĞ TABANLI TEKNOLOJİLER VE  
UYGULAMALARI



Metehan ULUOCAK - 21290433

GitHub: <https://github.com/Metehan-Uluocak/SLIP-IOS-WEB/tree/master/slip-web>

## **Security Leak Intelligence Platform**

### **Kurumsal Güvenlik Zafiyet Takip ve Analiz Sistemi**

#### **Özet**

Bu belge, slip-web istemcisinin amaçlarını, mimarisini, veri akışını, güvenlik gereksinimlerini ve geliştirme/teşhir planını ayrıntılı ve teknik bir biçimde açıklar. Hedef; öğretim görevlisinin beklentilerini karşılayacak seviyede netlik ve teknik derinlik sağlamaktır.

#### **Amaç**

- SlipAPI tarafından sunulan sızıntı verilerini güvenli, erişilebilir ve rol-bazlı bir kullanıcı arayüzünde sunmak.
- Analist ve Admin iş akışlarını destekleyen CRUD işlemleri (Sources, Platforms) ile hızlı keşif ve inceleme imkânı sağlamak.
- Mobil ve masaüstünde tutarlı, erişilebilir ve performanslı bir deneyim sunmak.

#### **Mevcut Kod Tabanı (kısa)**

- Teknoloji: React + TypeScript, Vite/CRA benzeri yapı (package.json mevcut).
- Önemli dosyalar:
  - src/App.tsx — Route yapılandırması
  - src/components/MainLayout.tsx — Ana çatı/layout
  - src/contexts/AuthContext.tsx, ThemeContext.tsx — global auth ve tema yönetimi
  - src/pages/\* — Dashboard, Login, Platforms, Sources, Users
  - src/services/api.ts — API client (HTTP katmanı)
  - src/models/\* — Tip tanımları (Leak, Platform, Source, User)

## 1. Veri Modeli

- \* Leak: Id, PlatformName, Title, Summary, PublishDate, SourceUrl, SourceName
- \* Source: Id, Name, Url, Description
- \* Platform: Id, Name, Description
- \* User: Id, Username, Email, Role, password

## 2. API & Backend Mimarisi

\* Teknoloji: .NET 8 üzerinde çalışan, RESTful prensiplerine uygun bir API'dir. Veri erişimi için Entity Framework Core ve mysql veritabanı kullanılmıştır.

\* Controllers: Controllers altında, her bir veri modeli için standart CRUD operasyonlarını ve iş mantığını yöneten denetleyiciler bulunur:

- \* LeaksController: Sızıntı verilerini listeler, filtreler.
- \* PlatformsController: Platform yönetimi.
- \* SourcesController: Kaynak yönetimi.
- \* UsersController: Kullanıcı kimlik doğrulama (login) ve yönetimi.

## 3. Örnek Endpoint'ler:

\* `GET /api/leaks`: Tüm sızıntıları getirir. Filtreleme için `?platform=...`, `?q=...` gibi query parametreleri destekler.

- \* `POST /api/leaks`: Yeni bir sızıntı kaydı oluşturur (Admin/Analist yetkisi).
- \* `GET /api/sources`: Tüm kaynakları listeler.
- \* `POST /api/sources`: Yeni bir kaynak ekler (Admin/Analist yetkisi).
- \* `GET /api/platforms`: Tüm platformları listeler.
- \* `POST /api/platforms`: Yeni bir platform ekler (Admin/Analist yetkisi).

#### 4. Frontend Mimarisi

Teknoloji: React (TypeScript ile) kullanılarak geliştirilmiş modern bir uygulamadır.

State Yönetimi: Global state (kullanıcı oturum bilgisi, tema) için React'in `Context API`'ı (`AuthContext`, `ThemeContext`) kullanılmıştır.

Routing: `React Router` kütüphanesi ile sayfa yönlendirmeleri yönetilir. `ProtectedRoute` bileşeni, sadece oturum açmış ve yetkili kullanıcıların belirli sayfalara erişimini sağlar.

API İletişimi: api.ts dosyasında yapılandırılmış olan `axios` istemcisi, backend ile iletişimi yönetir. `Authorization` başlığına JWT token'ı otomatik ekler.

#### 5. Roller & Yetkilendirme

API ve Frontend katmanlarında rol bazlı erişim kontrolü (RBAC) uygulanmıştır.

Admin: Tüm CRUD operasyonlarına ve kullanıcı yönetimine tam yetkilidir.

Analist: `Source` ve `Platform` ekleyip düzenleyebilir, tüm sızıntıları görebilir.

User: Sisteme sadece okuma amaçlı (read-only) erişebilir; sızıntıları görüntüleyebilir ancak değişiklik yapamaz.

#### Proje Planı

Hafta 1: Proje analizi, mimari tasarım ve teknoloji seçimi (React, .NET).

Hafta 2: Veritabanı şeması (EF Core modelleri) ve API projesinin temel altyapısının oluşturulması.

Hafta 3: API için `SeedData` mekanizmasının geliştirilmesi ve temel API endpoint'lerinin test edilmesi.

Hafta 4: React tarafında `Dashboard` ve `Login` arayüzlerinin tasarlanması, API entegrasyonu.

Hafta 5: `Sources` yönetimi modülünün (CRUD) frontend ve backend tarafında geliştirilmesi.

Hafta 6: `Platforms` yönetimi modülünün geliştirilmesi.

Hafta 7: Rol yönetimi ve yetkilendirme (`AuthContext`, `ProtectedRoute`) mekanizmalarının entegrasyonu.

Hafta 8: Genel testler, hata düzeltmeleri ve projenin son haline getirilmesi.

## Notlar

**Backend Kaynak Kodu:** SlipAPI klasöründe yer almaktadır. `Program.cs` ana başlangıç dosyasıdır.

**Veri Erişimi:** Veritabanı bağlantısı ve sorguları AppDbContext.cs üzerinden yönetilmektedir.

**Frontend Kaynak Kodu:** slip-web klasöründe yer almaktadır. `src/App.tsx` ana yönlendirme ve layout dosyasını içerir.