# NPC Engineering - Güvenlik İyileştirmeleri ve UI Düzeltme Planı

Bu belge, NPC Engineering projesinde tespit edilen güvenlik açıklarını kapatmak ve eksik UI bileşenlerini (Giriş/Kayıt butonları) entegre etmek için oluşturulmuş görev listesidir.

## 🚨 AŞAMA 1: Kritik Güvenlik Altyapısı (Hemen Uygulanmalı)

### 1.1. Environment Variable Validation (Çevre Değişkeni Doğrulama)

**Hedef:** Uygulama başlarken kritik değişkenlerin varlığını kontrol etmek.

* [ ] zod kütüphanesini kullanarak lib/env.mjs (veya .ts) dosyası oluştur.
* [ ] NEXT\_PUBLIC\_SUPABASE\_URL, NEXT\_PUBLIC\_SUPABASE\_ANON\_KEY, SHOPIER\_API\_KEY, SHOPIER\_API\_SECRET değişkenlerini zorunlu kıl.
* [ ] next.config.mjs içinde bu dosyayı import ederek build/start anında kontrol sağla.

### 1.2. Middleware & Route Protection (Sunucu Taraflı Koruma)

**Hedef:** /admin ve /dashboard rotalarını sunucu tarafında korumak.

* [ ] Kök dizinde middleware.ts dosyasını oluştur.
* [ ] @supabase/ssr kullanarak session kontrolü sağla.
* [ ] /admin/\* rotasına erişen kullanıcının role bilgisini kontrol et (Metadata veya veritabanı sorgusu ile). Admin değilse 403 veya redirect.
* [ ] /dashboard/\* rotasına erişen kullanıcının oturum açtığını doğrula. Açık değilse /login'e yönlendir.

### 1.3. Güvenlik Header'ları (Security Headers)

**Hedef:** XSS, Clickjacking ve diğer saldırıları engellemek.

* [ ] next.config.mjs dosyasını güncelle.
* [ ] Aşağıdaki headerları headers() fonksiyonuna ekle:
  + X-Frame-Options: DENY
  + X-Content-Type-Options: nosniff
  + Strict-Transport-Security
  + Referrer-Policy: strict-origin-when-cross-origin
  + Content-Security-Policy (Başlangıçta report-only veya script-src 'self' 'unsafe-inline' ile dikkatli yapılandır).

## 🔴 AŞAMA 2: API ve Veri Güvenliği

### 2.1. Shopier Callback Güvenliği

**Hedef:** app/api/callback/route.ts dosyasını sertleştirmek.

* [ ] **Rate Limiting:** Basit bir in-memory veya veritabanı tabanlı (IP tabanlı) rate limit mekanizması ekle.
* [ ] **Signature Validation:** Shopier'den gelen signature doğrulamasını try-catch blokları ile sarmala ve başarısız denemeleri detaylı logla (console.error yerine yapılandırılmış log).
* [ ] **IP Kontrolü (Opsiyonel):** Mümkünse Shopier IP aralıklarını kontrol et.

### 2.2. Supabase RLS (Row Level Security) Politikaları

**Hedef:** Veritabanına doğrudan erişimi kısıtlamak.

* [ ] SQL Editör veya Migration dosyası ile profiles, orders tabloları için RLS'i etkinleştir (ALTER TABLE ... ENABLE ROW LEVEL SECURITY).
* [ ] **Policy Örnekleri:**
  + Kullanıcılar sadece kendi profillerini görebilir/düzenleyebilir.
  + Adminler her şeyi görebilir.
  + Anonim kullanıcılar sadece products tablosunu (public) okuyabilir.

### 2.3. XSS ve Input Sanitization

**Hedef:** Kullanıcı girdilerini temizlemek.

* [ ] app/admin/page.tsx ve diğer form alanlarında kullanıcıdan alınan verileri (isim, email vb.) render ederken React'in varsayılan korumasına güven, ancak dangerouslySetInnerHTML kullanımından kaçın.
* [ ] Form validasyonu için zod şemaları oluştur ve hem client hem server tarafında uygula.

## 🟡 AŞAMA 3: UI/UX Düzeltmeleri (Kullanıcı Paneli)

### 3.1. Header Component Güncellemesi

**Sorun:** Ana sayfada "Giriş Yap" / "Kayıt Ol" butonları veya kullanıcı menüsü görünmüyor.

**Dosya:** components/header.tsx

**Görevler:**

* [ ] Header bileşenini "Client Component" olarak işaretle (use client) veya Auth durumunu kontrol eden bir wrapper kullan.
* [ ] Supabase auth.getUser() veya onAuthStateChange ile kullanıcının giriş durumunu dinle.
* [ ] **Durum: Giriş Yapılmamışsa:**
  + Sağ üst köşeye "Giriş Yap" (variant="ghost") ve "Kayıt Ol" (variant="default") butonlarını ekle.
* [ ] **Durum: Giriş Yapılmışsa:**
  + Mevcut UserMenu bileşenini render et.
* [ ] Mobilde (Hamburger menü içinde) de bu linklerin göründüğünden emin ol.

### 3.2. Dashboard ve Auth Sayfaları

* [ ] app/login/page.tsx ve app/register/page.tsx sayfalarında şifre karmaşıklığı kontrolü ekle (Min 8 karakter, sembol zorunluluğu uyarısı).
* [ ] Başarılı giriş/kayıt sonrası yönlendirmelerin (router.push('/dashboard')) doğru çalıştığını test et.

## 🟢 AŞAMA 4: Bakım ve İzleme

### 4.1. Logging

* [ ] Kritik hatalar (Ödeme hataları, Auth hataları) için bir loglama servisi (Sentry vb.) veya veritabanında bir error\_logs tablosu kur.

### 4.2. Dependency Security

* [ ] npm audit çalıştır ve yüksek riskli paketleri güncelle.
* [ ] Kullanılmayan paketleri temizle.

## Uygulama Sırası Önerisi

1. Önce **UI Düzeltmesi (3.1)** yapılmalı ki sistem test edilebilir olsun.
2. Ardından **Middleware (1.2)** ve **Env Validation (1.1)** eklenmeli.
3. Son olarak **API Güvenliği (2.1)** ve **RLS (2.2)** yapılandırılmalı.