CMS Publish Webhook & ISR/CRM/Slack Otomasyon Planlama Şablonu — Yazılım / Webhooks (v1.0)

Asset Amaç: Bu şablon; İçerik Yönetim Sistemindeki (CMS) yayınlama (publish), güncelleme (update) ve silme (delete/unpublish) olaylarını (events) güvenli webhook entegrasyonları ile dinleyerek Next.js önbellek yenileme (Incremental Static Regeneration - ISR), Slack bilgilendirme hatları ve CRM tetikleyicilerini uçtan uca planlamayı amaçlar. Olay seçimi, veri paketi (payload) sözleşmesi, uç nokta (endpoint) güvenliği, loglama ve otomatik alarm katmanını tek bir teknik dokümanda toplayarak manuel içerik senkronizasyonu işlerini ve sistemsel hata risklerini sıfıra indirir.

Kim Kullanır?: Teknik Liderler (Tech Lead), Ajans Yazılım Takımları ve İçerik Operasyon Yöneticileri.

Nasıl Kullanılır?

1. CMS altyapınız üzerinde hangi olay türlerinin (publish/delete) hangi içerik modellerini (oda, kampanya, blog) tetikleyeceğini netleştirerek işe başlayın.
2. Sistemler arası veri alışverişi için veri paketi şablonunu (payload) ve uç nokta güvenliğini (gizli anahtar, rate limit, IP kısıtlaması) sözleşmeye bağlayın.
3. Tetikleme sonrasında Next.js tarafında önbellekten düşecek sayfa yollarını (revalidation paths), Slack/CRM aksiyonlarını ve hata anı operasyon kurallarını doldurarak şablonu uygulamaya alın.

TEMPLATE — CMS Publish Webhook & ISR/CRM/Slack Otomasyon Planlama Şablonu

A) Boş Şablon Alanları (Webhook Otomasyon Matrisi)

1) Olay (Event) ve İçerik Modeli Envanteri

CMS panelinde içerik üreticisi işlem yaptığında arka planda fırlatılacak webhook olaylarının seçimi:

- **Aktif Edilecek Webhook Olay Türleri:** `[ ] publish` / `[ ] update (Sadece canlı yayındaki içerik güncellendiğinde)` / `[ ] delete` / `[ ] unpublish`
- **Webhook'a Bağlanacak İçerik Tipleri (Content Types):** `[ ] room (Oda)` / `[ ] campaign (Kampanya/Teklif)` / `[ ] blog` / `[ ] document (Rehber/PDF)` / `[ ] service (Otel İçi Hizmetler)`

2) Teknik Payload (Veri Paketi) Sözleşmesi

Hedef sunucunun gelen isteği pürüzsüz işleyebilmesi için gönderilmesi zorunlu olan minimum JSON veri yapısı:

JSON

```
{
  "event_type": "_____",   // Örn: content.publish
  "content_type": "_____",   // Örn: room
  "entity_id": "_____",   // Örn: cms_room_9921
  "slug_path": "_____",   // Örn: /odalar/deluxe-suit
  "locale": "_____",   // Örn: tr, en
  "timestamp": "_____",   // Örn: 1719659100
  "signature": "_____"    // Şifrelenmiş doğrulama hash'i
}
```

3) Hedef Altyapı Aksiyonları (Destinations)

Webhook verisi endpoint'e ulaştığında eş zamanlı tetiklenecek hedef sistem senaryoları:

- **Ön Yüz Performans Katmanı (Next.js):** `[ ] revalidate path (Sadece ilgili sayfayı yenile)` / `[ ] full rebuild (İstisnai büyük sistem güncellemeleri)`
- **Kurumsal İletişim Kanalları (Slack Entegrasyonu):** `[ ] #content (İçerik bildirim odası)` / `[ ] #ops (Yazılım operasyon odası)` / `[ ] #seo (Arama motoru takip odası)`
- **Müşteri İlişkileri ve Pazarlama (CRM):** `[ ] Yeni kampanya tetiklemesi` / `[ ] Otomatik nurturing e-posta akışı başlatma` / `[ ] Satış ekibine anlık görev/bildirim fırlatma`
- **Arama Motoru Dizini (Search Index):** `[X] Otomatik Google/Bing sitemap ping gönderimi` *(Varsayılan Yetkili Sistem: Otomatik SEO Kancası)*

4) Önbellek Yenileme (Revalidation) ve Dağıtım Planı

Sitenin hem çok hızlı çalışması (ISR) hem de her zaman güncel kalması için dağıtım kuralları:

- **Revalidate Paths Üretim Kuralı (Örn: İçerik dili ve tipine göre dinamik URL inşası):** _____
- **İçeriğin CMS'ten Yayınlanması ile Sitede Canlıya Geçmesi Arasındaki Hedef Süre:** ____ Dakika / Saniye.
- **Hatalı İstek Durumunda Yeniden Deneme Algoritması (Retry / Exponential Backoff):** `[X] Var` / `[ ] Yok` *(Varsayılan Durum: Üstel gecikmeli yeniden deneme mekanizması aktiftir).*

5) Endpoint Güvenlik Katmanı Kontrol Listesi

Dışarıdan gelebilecek sahte webhook isteklerini ve siber saldırıları (DDoS, Brute-Force) engelleme bariyerleri:

- **[ ] Webhook Secret Doğrulama:** Gelen isteklerin CMS'ten geldiğini doğrulamak için header alanındaki imzanın (X-Hub-Signature) private key ile çözülmesi.
- **[ ] IP Allowlist Sınırlandırması:** Webhook dinleyici endpoint'inin sadece CMS sunucusunun sabit IP adreslerinden gelen isteklere açık olması.
- **[ ] Rate Limiting (İstek Sınırlandırma):** Saniyede veya dakikada gelebilecek maksimum webhook istek adetlerinin kilitlenmesi.
- **[ ] Idempotency Kontrolü:** Ağ gecikmesi nedeniyle aynı webhook isteği mükerrer (2 kez) geldiğinde sistemin hata vermeden tek bir işlem yapması.
- **[ ] Entegre Log & Alert Mekanizması:** Güvenlik bariyerine takılan veya başarısız olan tüm isteklerin anlık olarak izlenmesi.

6) Günlükleme (Loglama) ve Hata Alarm Koşulları

Operasyonel sürdürülebilirliği sağlamak adına sistemin izleyeceği metrikler ve kırmızı alarm çizgileri:

- **Kalıcı Log Veri Alanları:** `event_id`, `event_type`, `slug_path`, `http_status_code`, `duration_ms`
- **Kritik Alarm Tetiklenme Koşulu:** `[ ] Üst üste 3 kez 5xx Sunucu Hatası fırladığında` / `[ ] 401 Yetkisiz Erişim algılandığında` / `[ ] Next.js revalidate fail durumuna düştüğünde` otomatik olarak yazılım nöbetçisine (SRE/DevOps) bildirim gönderilir.

B) Nasıl Doldurulur? (5 Kritik Kural)

1. **İmza Doğrulamasını (Signature Verification) Asla Atlamayın:** Webhook endpoint'leri dış dünyaya açık URL yapılardır. CMS tarafından gönderilen `signature` hash değerini backend tarafında kendi gizli anahtarınızla doğrulamadan gelen istekleri kesinlikle işlemeyin.
2. **Next.js'i Tamamen Yeniden Derlemekten (Full Rebuild) Kaçının:** Sitede tek bir otel odası veya blog içeriği güncellendiğinde tüm web sitesini ayağa kaldırmaya (full build) çalışmayın. Sadece değişen sayfanın URL yolunu önbellekten düşüren `res.revalidate(path)` mimarisini (ISR) kullanın.
3. **Webhook İşlemlerini Kuyruk Yapısıyla (Queue) Asenkron Yönetin:** Webhook isteği geldiğinde Next.js yenilemesini, Slack bildirimini ve CRM tetiklerini aynı HTTP isteği içinde yapmaya çalışıp bağlantıyı bekletmeyin (timeout riski). İsteği alır almaz "200 OK" dönün ve arka planda iş kuyruğuna (Redis/BullMQ, AWS SQS vb.) atarak asenkron işleyin.
4. **Mükerrer İsteklere Karşı Idempotency Anahtarı Kullanın:** İnternet kesintilerinde CMS sistemleri aynı webhook olayını tekrar gönderebilir. Sistemde veri kirliliği veya çift kayıt oluşmaması için `event_id` veya `timestamp` kontrolü yaparak mükerrer istekleri eleyin.
5. **İçerik Tiplerine Göre İlişkili Sayfa Yollarını Temizleyin:** Bir oda içeriği güncellendiğinde sadece `/odalar/deluxe-suit` sayfasını önbellekten düşürmek

yetmez. O odanın listelendiği ana sayfayı (`/odalar`) veya ana sayfadaki kampanya bandını da (`/`) tetikleyerek önbellekten silmeyi unutmayın.

C) Teknik Kontrol Listesi (Checklist)

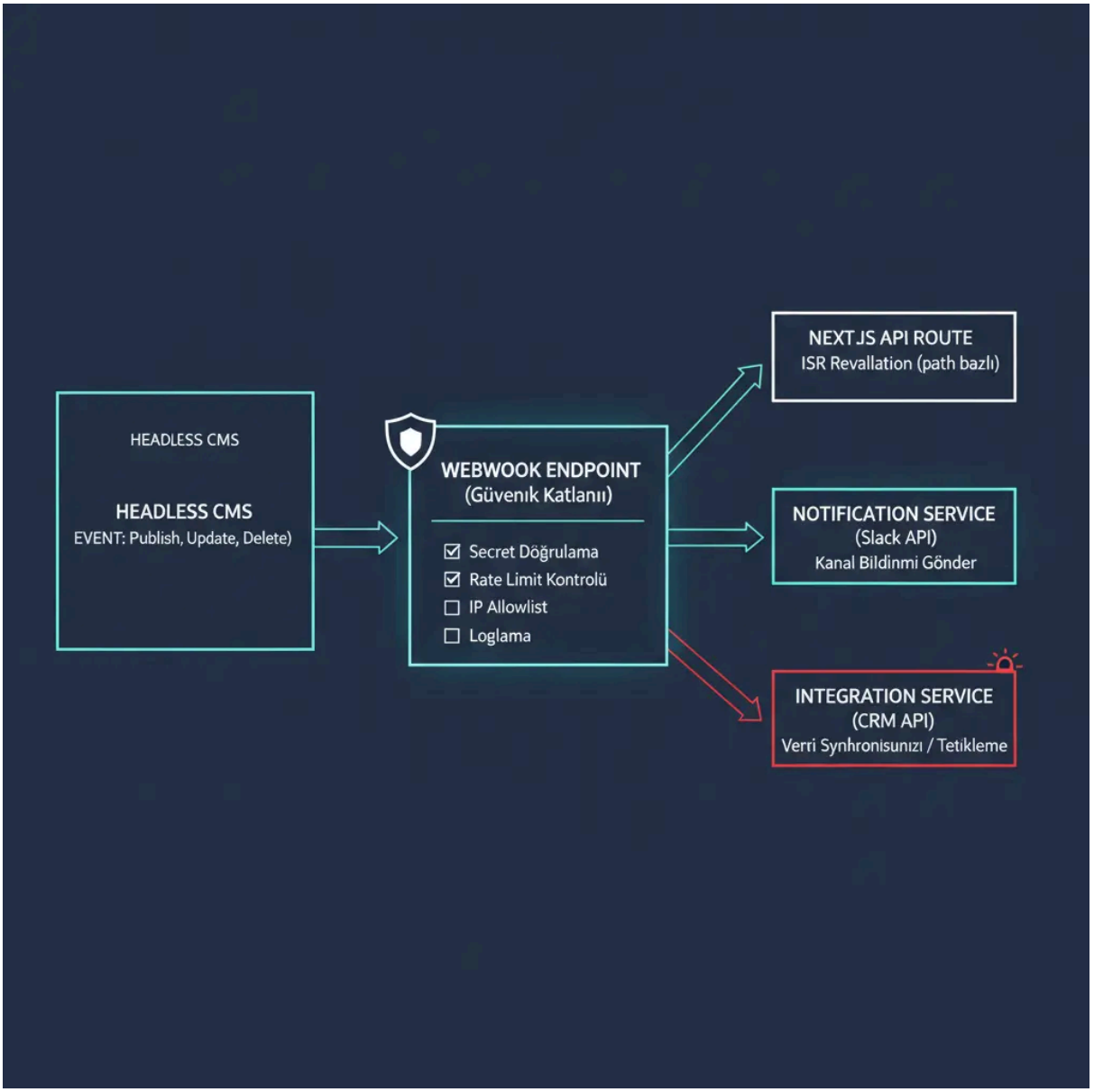
- [ ] **Olaylar Tanımlandı:** CMS üzerinde hangi içerik modellerinin webhook mekanizmasını harekete geçireceği listelendi.
- [ ] **Payload Standardı Sağlandı:** Sistemler arasında konuşulacak asgari JSON veri sözleşmesi taslağı kilitlendi.
- [ ] **Hedef Sistemler Bağlandı:** Tetikleme anında çalışacak Next.js ISR, Slack bildirim kanalları ve CRM otomasyon entegrasyonları haritalandırıldı.
- [ ] **Güvenlik Çemberi Hazır:** Webhook imza doğrulama, IP kısıtlama, rate limit ve mükerrer istek engelleme (idempotency) kuralları kuruldu.
- [ ] **Kuyruk ve Yeniden Deneme Aktif:** Başarısız istekleri arka planda üstel gecikmeyle (exponential backoff) tekrar deneyecek hata yönetim kurgusu tamamlandı.
- [ ] **Operasyonel Alarm Kuruldu:** 3 kez üst üste hata alınması veya revalidate başarısızlığı durumunda DevOps ekiplerine uyarı fırlatacak alarm koşulları tanımlandı.

Deliverables (Teslim Edilecek Çıktılar)

- Yazılım Ekiplerine İletilecek 1 Adet Standart JSON Webhook Event ve Payload Sözleşmesi Dokümanı
- Entegrasyon Akışını Çizen 1 Adet Dinamik Revalidation, Slack ve CRM Bildirim Akış Planı
- Endpoint Sızmalarını Engelleyen 1 Adet Güvenlik, Loglama ve Operasyonel Kontrol Listesi Kılavuzu

[Checklist / Framework Card]

Yazılım mimarları ve sistem entegratörleri için tasarlanmış, 1200x1200px kare formatında yüksek çözünürlüklü bir kurumsal webhook güvenlik ve otomasyon checklist kartı. Gece siyahı bir IDE kod arayüzü teması zemininde, kurumsal neon moru ve mavi hatlarla ayrılmış panel yapısı bulunuyor. Kartın üzerinde; "İmza Doğrulama (Signature Check)", "IP Filtresi", "Hız Sınırı (Rate Limiting)" ve "Mükerrer İstek Engeli (Idempotency)" gibi siber güvenlik ve veri bütünlüğü adımları, yanlarında parlayan yeşil onay ikonları bulunan scannable bir kontrol listesi tasarımıyla sergileniyor.



[Diagram / Flow]

Bir içerik yayınlama olayının sistemler arasında tetiklediği asenkron otomasyon zincirini açıklayan, 1920x1080px Full HD çözünürlükte tasarlanmış teknik bir veri akış ve mimari entegrasyon diyagramı. Sol taraftaki kurumsal CMS arayüzünde basılan bir "Yayınla (Publish)" butonundan çıkan parlak mor renkli bir webhook veri paketi (payload) oku, ortadaki güvenli API ağ geçidine (Gateway) ulaşıyor. Bu ağ geçidinden çıkan asenkron akış okları; Next.js sunucusuna önbellek temizleme (ISR) emri gönderirken, eş zamanlı olarak Slack kanalına bildirim kartı fırlatıyor ve CRM sisteminde pazarlama otomasyon süreçlerini tetikliyor. Tüm bu entegrasyon zinciri net teknik etiketlerle ve "clarity at a glance" felsefesiyle şemalaştırılıyor.