Teknik Mimari (MVP → Scale)

1) Yüksek Seviye Bileşenler

Client’lar

Web: Next.js (SSR/ISR ile hızlı + SEO)

Mobil: Flutter veya React Native (tek kod tabanı)

API Gateway

REST (veya GraphQL) → auth, rate limit, caching

Core Backend (App API)

Python FastAPI (veya Node.js NestJS)

İş mantığı: arama, karşılaştırma, kargo & taksit hesaplama, favori/alarmlar

Fiyat/Kargo Veri Katmanı

Connector Service’ler (Trendyol, Hepsiburada, Amazon TR, n11)

Varsa Resmi API; yoksa “nazik scraping” (robots.txt’ye saygı, düşük frekans, cache)

Scheduler + Queue: Celery/RQ + Redis / RabbitMQ (periyodik veri çekim)

Ürün Eşleştirme (Matching)

Normalizasyon + fuzzy match (GTIN/EAN, SKU, başlık/marka/model)

Dedup & “master product” oluşturma

Depo & Arama

OLTP: PostgreSQL (ürün, satıcı, fiyat, kargo, kampanya)

Cache: Redis (sıcak aramalar)

Full-text/aranabilirlik: OpenSearch/Elasticsearch (opsiyonel)

Bildirim

Fiyat alarmı, kampanya push: Firebase Cloud Messaging + e-posta

Gözlemleme

Log: Loki/ELK, Metrics: Prometheus + Grafana, Tracing: OpenTelemetry

DevOps

Docker + Kubernetes (başta ECS/Cloud Run da olabilir)

CI/CD: GitHub Actions

Obj. Storage: S3/GCS (görseller & raporlar)

2) Veri Akışı (Arama → Karşılaştırma → Alarm)

Kullanıcı arama yapar (örn. “AirPods Pro 2”).

App API cache’e bakar (Redis). Yoksa:

Search Service, MasterProduct’ı bulur (eşleştirme).

Price Aggregator, ilgili satıcıların en güncel fiyat + kargo + teslimat verilerini çeker (önbellek + periyodik güncelleme).

Toplam Tutar Hesabı: ürün fiyatı + kargo + (varsa) komisyon/kapıda ödeme + teslimat süresi.

Sonuç tablosu döner; kullanıcı favoriye ekleyip alarm kurabilir.

Scheduler periyodik günceller; eşik altına inince push/mail gönderir.

3) Kargo & Taksit Hesaplama Motoru

Kargo:

Satıcının kural seti: “₺500 üstü ücretsiz”, “standart kargo 49₺”, “aynı gün 79₺” vb.

Bölge/mağaza özel farklar (ileride posta kodu bazlı).

Promolar: “Sepette kargo ücretsiz” (kupon motorundan tetik).

Taksit:

Banka/pos entegrasyonu olmadan, siteye göre ilan edilen planları toplar (ör. “Axess 6 taksit”).

Aylık taksit tutarı ve toplam maliyet (faiz/komisyon varsa) hesaplanır.

Çıktı:

toplam\_fiyat (ürün + kargo)

teslimat\_tahmini (gün)

taksit\_tablosu[] (ay, aylık, toplam)

uygulanabilir\_kuponlar[]

4) Ürün Eşleştirme (Matching) Stratejisi

Öncelik sırası:

GTIN/EAN/Barkod (net eşleşme)

Model No / MPN + Marka

Başlık temizleme (stop-word, ölçü/renk varyantı) + fuzzy ratio

Özellik vektörleri (embedding) ile benzerlik (scale aşaması)

Eşleşme güven skoru (0–1). Düşük skorlu eşleşmeleri manuel queue’ye at (review panel).

5) API Taslağı (REST)

GET /search?q= → master ürün + en iyi eşleşmeler

GET /products/{id} → ürün detay + fiyat/kargo/taksit/kupon

GET /products/{id}/prices → satıcı kırılımı (site, ürün fiyatı, kargo, toplam, teslimat)

POST /favorites {user\_id, product\_id}

POST /alerts {product\_id, threshold\_price, channels:[push,email]}

GET /coupons?site=trendyol&product\_id=...

Admin:

POST /ingest/{connector}/run

GET /matching/review?status=pending

6) Örnek Şema (PostgreSQL)

products(id, brand, model, name, gtin, mpn, category\_id, created\_at)

product\_attrs(product\_id, key, value)

sellers(id, name, site, affiliate\_tag)

listings(id, product\_id, seller\_id, seller\_sku, url, last\_seen\_at)

prices(id, listing\_id, price, currency, fetched\_at)

shipping\_rules(id, seller\_id, rule\_json, updated\_at) -- koşullar

shipping\_quotes(id, listing\_id, method, fee, region, fetched\_at)

installments(id, seller\_id, bank, months, monthly, total, updated\_at)

coupons(id, seller\_id, code, desc, min\_spend, expires\_at, source)

users(id, email, hashed\_pw, created\_at)

favorites(user\_id, product\_id, created\_at)

alerts(id, user\_id, product\_id, threshold, channel, created\_at)

7) Connectors (Veri Toplama) – MVP → Scale

MVP (2 site): Trendyol (API + RSS benzeri feed’ler) / Hepsiburada (nazik scraping)

Teknik:

Rotating proxy + backoff + etiketli user-agent

Saatlik/dönemsel refresh; popüler ürünlerde daha sık

Yoğun talep ürünlerde “on-demand fetch” (kullanıcı isteyince taze çek)

Scale: Amazon TR, n11, Teknosa, MediaMarkt, Çiçeksepeti, Migros Online…

Not (Uyumluluk): Mümkün olan yerde resmi ortak/Affiliate API’ler; scraping’de robots.txt’ye saygı, makul istek hızı, yasal kullanım ve telif haklarına özen.

8) Performans & Maliyet Optimizasyonu

Önbellek: Popüler aramalar 10–30 dk Redis cache

Precompute: Gecelik/ saatlik “toplam\_fiyat snapshot” tablosu

Resim/CDN: Görselleri hot-link yerine imaj proxy + CDN

Sorgu optimizasyonu: listings/prices üzerinde composite index

Mikroservisleşme: Trafik artınca connectors ve aggregator ayrı ölçeklenir

9) Güvenlik & Gizlilik

OAuth 2.1 / JWT

HTTPS zorunlu, HSTS

Oran sınırlama (rate limit), WAF

PII minimizasyonu; KVKK uyumu; açık gizlilik politikası

Secrets: Vault/Secret Manager

10) MVP Geliştirme Sprint Planı (12 Hafta)

S1–S2: Proje setup, DB şeması, temel modeller, Auth

S3–S4: Trendyol connector + Matching v1

S5–S6: Hepsiburada connector + Price Aggregation + Toplam fiyat

S7: Web (Next.js) Arama & Sonuç Tablosu

S8: Mobil temel akış (Arama → Sonuç)

S9: Favori & Alarm + Push (Firebase)

S10: Kupon veri modeli + basit kupon listesi

S11: İzleme/Loglama, caching, basit admin panel

S12: Beta, yük testi, bugfix, küçük UX cilası

11) Farkı Netleştiren Özellik Seti (V1)

Kargo dahil toplam fiyat sıralaması (varsayılan)

Teslimat süresi etiketi (1 gün / 2–3 gün)

Taksit özet rozeti (örn. “6 taksit: 12×₺X toplam ₺Y”)

Uygulanabilir kupon etiketi (var/yok + tahmini indirim)

12) Ölçekleme Yol Haritası (V2+)

Gelişmiş eşleştirme (embedding, ML)

Bölgesel kargo (posta kodu bazlı)

Dinamik fiyat grafikleri (30/90 gün)

Tarayıcı eklentisi (checkout’ta kupon otomatik uygula)

B2B raporlama paneli (anonim trendler)

Sonraki adım önerim

İlk sprint’e girebilmemiz için:

Stack seçimi (FastAPI+Postgres+Redis+Next.js+Flutter öneriyorum)

DB şema ve endpoint’leri finalize

Trendyol connector POC (bir ürün için uçtan uca akış)