

Codenight Case: Turkcell Smart Allocation – Dinamik Kaynak ve Öncelik Yönetim Platformu

Amaç

Ekipler, Turkcell servislerinde oluşan talepleri ve sınırlı kaynakları dikkate alarak, **önceliklendirme ve atama yapan** bir yazılım sistemi geliştirir.

Sistem; farklı kaynaklardan gelen talepleri analiz eder, mevcut kaynak durumunu değerlendirir ve **en uygun atamayı** otomatik olarak gerçekleştirir.

Temel Özellikler

1. Talep (Request) Yönetimi

- Sistem, farklı servislerden gelen talepleri **request** olarak işler.
- Her talep aşağıdaki alanları içermelidir:
 - talep eden kullanıcı
 - servis
 - talep türü
 - aciliyet seviyesi
 - oluşturulma zamanı

Örnek talep:

```
{  
  "request_id": "REQ-2041",  
  "user_id": "U8",  
  "service": "Superonline",  
  "request_type": "CONNECTION_ISSUE",  
  "urgency": "HIGH",  
  "created_at": "2026-03-12T14:25:00Z"  
}
```

2. Kaynak (Resource) Yönetimi

- Sistem, kullanılabilir kaynakları takip etmelidir.
- Kaynaklar şunları içerebilir:
 - teknik ekip
 - destek personeli
 - sanal kapasite (ör. eş zamanlı destek limiti)

Örnek kaynak:

```
{  
  "resource_id": "RES-11",  
  "resource_type": "TECH_TEAM",  
  "capacity": 3,  
  "city": "Ankara",  
  "status": "AVAILABLE"  
}
```

3. Durum ve Öncelik Hesaplama

- Her talep için sistem bir **öncelik skoru** hesaplamalıdır.
- Öncelik hesabında şu faktörler kullanılabilir:
 - aciliyet
 - talep türü
 - kullanıcının geçmiş durumu
 - bekleme süresi

Örnek:

$\text{priority_score} = \text{urgency_weight} + \text{waiting_time_weight} + \text{user_risk_weight}$

4. Atama (Allocation) Motoru

- Sistem, talepleri uygun kaynaklarla eşleştirmelidir.

- Atama yapılırken:
 - kaynak kapasitesi
 - şehir bilgisi
 - mevcut aktif atamalar dikkate alınmalıdır.

Bir kaynak kapasitesi dolduğunda yeni atama yapılmamalıdır.

5. Atama Çakışmaları

- Aynı anda birden fazla yüksek öncelikli talep oluşabilir.
 - Sistem:
 - en yüksek öncelik skoruna sahip talebi seçmeli
 - diğer talepleri beklemeye almalı
 - bu durumu kayda geçmelidir.
-

6. Bildirim (BiP Mock)

- Atama sonucu ilgili kullanıcıya bildirim gönderilmelidir (mock).

Örnek:

```
{  
  "user_id": "U8",  
  "message": "Talebiniz öncelikli olarak işleme alındı. Teknik ekip yönlendirildi."  
}
```

7. Kayıt ve İzlenebilirlik

- Sistem tüm atamaları ve kararları kaydetmelidir.

Örnek kayıt:

```
{
```

```
"allocation_id": "AL-501",  
"request_id": "REQ-2041",  
"resource_id": "RES-11",  
"priority_score": 92,  
"status": "ASSIGNED",  
"timestamp": "2026-03-12T14:30:00Z"  
}
```

8. Yönetim ve İzleme Ekranı (Dashboard)

Dashboard'da aşağıdaki bilgiler yer almalıdır:

- Bekleyen talepler
- Aktif atamalar
- Kaynak doluluk oranları
- Öncelik skorlarına göre talep sıralaması

Dashboard web tabanlı olabilir; mobil zorunlu değildir.

Bonus Özellik: Allocation Kural Yönetim Ekranı (Opsiyonel)

Sistem içinde öncelik ve atama kurallarının kullanıcı arayüzü üzerinden yönetilebildiği bir ekran geliştirilmesi beklenir.

Beklenen Fonksiyonlar

- Öncelik hesaplama kurallarını görüntüleme
- Ağırlık değerlerini değiştirme
- Belirli talep türleri için özel öncelik tanımlama
- Kuralları aktif / pasif yapabilme

Yapılan değişiklikler kaydedilmeli ve sistem bir sonraki atamada güncel kuralları kullanmalıdır.

Veri Modeli (CSV / JSON)

users

user_id, name, city

requests

request_id, user_id, service, request_type, urgency, created_at

resources

resource_id, resource_type, capacity, city, status

allocations

allocation_id, request_id, resource_id, priority_score, status, timestamp

allocation_rules

rule_id, condition, weight, is_active

API Önerisi

POST /requests

GET /requests

GET /resources

POST /allocate

GET /allocations

GET /dashboard/summary

Puanlama Kriterleri (Genel – 100 Puan)

Kategori	Puan
Temel İşlevsellik ve Doğru Çalışma	30
Veri Modeli ve Sistem Tasarımı	20
Öncelik ve Atama Mekanizması	20
Kod Kalitesi ve Yapı	15

Kategori	Puan
Görsellik ve Anlatılabilirlik	10
Bonus Özellikler	5