

# WeatherAppV2 – TASARIM RAPORU (DESIGN REPORT)

## 1. GİRİŞ

Bu doküman, Expo ve React Native kullanılarak geliştirilen **WeatherAppV2** (Rota Bazlı Hava Durumu Analiz Sistemi) uygulaması için hazırlanmış Tasarım Raporu'dur. Rapor, projenin analiz aşamasında belirlenen gereksinimlerin; modern yazılım mimarisi prensipleri, servis odaklı yaklaşım ve kullanıcı deneyimi (UX) kriterleri doğrultusunda nasıl yapılandırıldığını detaylandırmaktadır. Tasarımın temel odağı; harita servisleri, rota optimizasyonu ve anlık hava durumu verilerini modüler bir mimaride birleştirmektir.

## 2. TASARIM HEDEFLERİ

Uygulamanın tasarım sürecinde aşağıdaki temel mühendislik hedefleri gözetilmiştir:

- 2.1 Servis ve Arayüz Ayrımı (SoC):** Harici API çağrıları (Mapbox, OpenWeatherMap) ile kullanıcı arayüzü mantığının birbirinden ayrılması hedeflenmiştir. Bu sayede API değişiklikleri UI katmanını etkilemez.
- 2.2 Genişletilebilirlik:** Sistemin gelecekte farklı hava durumu servisleri (örn. AccuWeather) veya ek taşıma modları (toplu taşıma, bisiklet) eklenebilecek esneklikte olması planlanmıştır.
- 2.3 Hata Toleransı ve Dayanıklılık:** Ağ kesintileri veya API limit aşımı gibi durumlarda uygulamanın çökmesini önleyen, kullanıcıya anlamlı geri bildirimler veren bir yapı kurulmuştur.
- 2.4 Performans ve Veri Yönetimi:** Özellikle uzun rotalarda artan API isteklerini optimize etmek için veri örnekleme ve gereksiz render işlemlerini önleme teknikleri uygulanmıştır.

## 3. SİSTEM MİMARİSİ

### 3.1 Mimari Yaklaşım

WeatherAppV2, bileşen tabanlı (Component-Based) ve kavramsal olarak katmanlı bir mimari üzerine inşa edilmiştir:

1. **Sunum Katmanı (Presentation):** React Native bileşenleri, MapView görselleştirmeleri ve kullanıcı etkileşim panelleri.
2. **Uygulama Servisleri Katmanı (Service):** Geocoding, Directions ve Weather API yönetim mantığı.
3. **Veri/Model Katmanı (State):** Uygulama durumu (Region, RouteCoords) ve harici servislerden dönen ham verilerin işlenmiş hali.

### 3.2 Teknoloji Yığını

- **Çatı (Framework):** Expo + React Native (Hızlı prototipleme ve çapraz platform desteği için).
- **Harita ve Rota:** react-native-maps ve Mapbox Directions API.
- **Veri İletişimi:** Axios (HTTP istemcisi).
- **Hava Durumu:** OpenWeatherMap API<sup>16</sup>.

## 4. ALT SİSTEMLER

### 4.1 Harita ve Görselleştirme Alt Sistemi

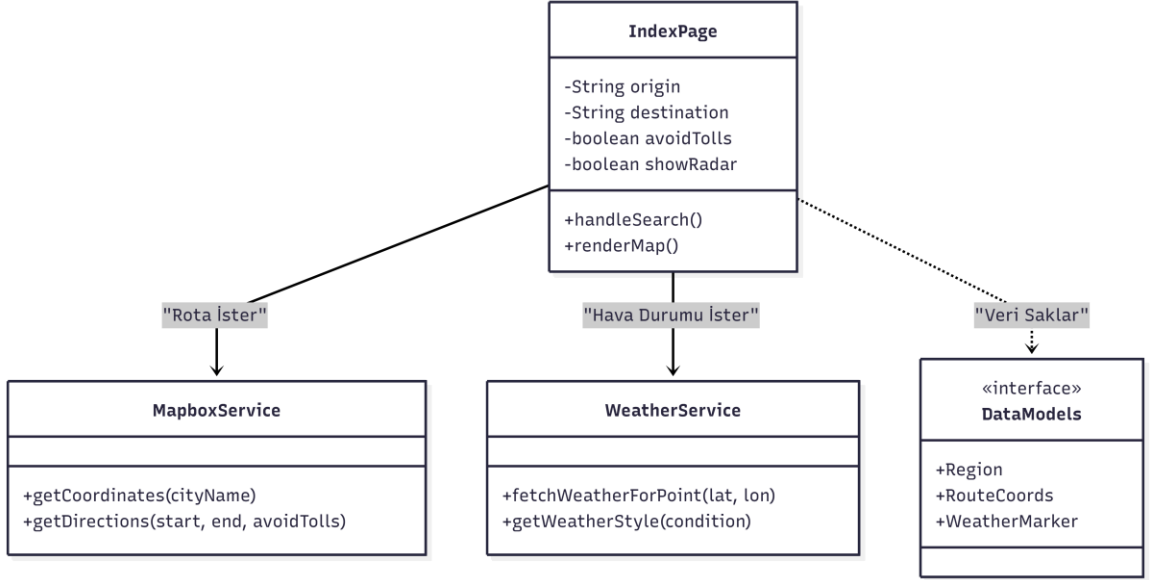
- **Sorumluluk:** Rota çizimi (Polyline), başlangıç/bitiş noktalarının işaretlenmesi ve yağış radarı katmanının yönetilmesi.
- **Özellik:** Ur1Tile kullanılarak OpenWeatherMap'ten gelen anlık yağış verileri harita üzerinde canlı katman olarak gösterilir.

### 4.2 Rota Hesaplama ve Coğrafi Kodlama Alt Sistemi

- **Sorumluluk:** Şehir isimlerini koordinata dönüştürme (Geocoding) ve iki nokta arasındaki en uygun sürüş rotasını belirleme.
- **Kısıt Yönetimi:** Kullanıcının "Paralı Yollardan Kaçın" (Avoid Tolls) seçeneğine göre API isteği dinamik olarak güncellenir.

### 4.3 Hava Durumu Analiz Alt Sistemi

- **Sorumluluk:** Rota üzerindeki stratejik noktalardan hava durumu verisi toplama ve bu verileri görsel bir "stil" (ikon, renk) ile eşleştirme.
- **Veri İçeriği:** Sıcaklık, rüzgar hızı ve rüzgar yönü (ok işareti ile) kullanıcıya sunulur.



## 5. MODÜL VE FONKSİYON TASARIMI

### 5.1 Ana Kontrol Mekanizması (handleSearch)

Bu fonksiyon, uygulamanın iş akışını yöneten merkezi mantıktır:

1. **Validasyon:** Girdilerin boş olup olmadığını kontrol eder.
2. **Koordinat Dönüşümü:** getCoordinates ile şehir isimlerini Mapbox üzerinden lat/lon değerlerine çevirir.
3. **Rota Çizimi:** İki nokta arasındaki yolu hesaplar ve routeCoords state'ini günceller.
4. **Hava Durumu Örneklemesi:** Rota üzerindeki noktalardan fetchWeatherForPoint ile veri çeker.

### 5.2 UI Bileşen Tasarımı

- **Giriş Paneli:** İki adet TextInput ve filtreleme seçenekleri (Switch) içerir.
- **Hava Durumu Kartları:** Harita üzerinde kompakt "kule" tasarımlı, alt üçgen pointer içeren marker'lar kullanılır.

## 6. TASARIM KARARLARI VE TRADE-OFF'LAR

Karar	Gerekçe	Avantaj / Dezavantaj
API Tabanlı Mimari	Veriyi cihazda saklamak yerine anlık olarak çekmek güncelliği sağlar.	(+) Her zaman güncel veri. (-) İnternet bağımlılığı ve API limitleri.
Expo Framework	Hızlı geliştirme süreci ve kolay kütüphane yönetimi.	(+) Cross-platform uyumu. (-) Native modüllere erişimde kısıtlar.
Radar Katmanı (Switch)	Kullanıcıya görsel yoğunluğu kontrol etme imkanı sunar.	(+) UX esnekliği. (-) Ekstra network yükü (opsiyonel olsa da).
Tek Ekran (Single Page) Akışı	Kullanıcının rotadan kopmadan tüm verilere erişmesini sağlar.	(+) Düşük bilişsel yük. (-) Karmaşık durumlarda UI kalabalığı riski.

## 7. SONUÇ

WeatherAppV2 tasarımı, analiz raporundaki rota ve hava durumu entegrasyonu hedeflerini başarıyla karşılamaktadır. Seçilen katmanlı mimari ve modüler fonksiyon yapısı sayesinde, uygulama sürdürülebilir bir temele oturtulmuştur. API güvenliği (Environment Variables) ve performans optimizasyonları (Caching) gibi önerilen iyileştirmelerle, sistemin endüstriyel standartlara tam uyumu hedeflenmektedir.