# DİJİTAL DEPREM ÇÖZÜMLERİ HACKATHONU

# ÖN TASARIM RAPORU

## 1. Proje Başlığı

**ERA(Earthquake Relief Assistant)**

## 2. Problem Tanımı

Depremler, sadece fiziksel yapıları değil; aynı zamanda iletişim altyapılarını, ulaşım ağlarını ve insan hayatını doğrudan etkileyen büyük afetlerdir. Böyle kriz anlarında, hayatta kalanların en temel ihtiyaçlarını (yiyecek, içecek, barınma, tıbbi yardım vb.) hızlıca bildirebilecekleri, kurtarma ekiplerinin ve gönüllülerin sahadaki durumu anlık görebileceği etkili bir dijital koordinasyon sistemine ihtiyaç duyulmaktadır.

Mevcut durumda, sosyal medya ve mesajlaşma uygulamaları üzerinden yapılan bireysel yardım çağrıları; konumsuz, dağınık ve doğrulanmamış bilgiler içerdiğinden dolayı, müdahale ekipleri için organize edilmesi güç ve zaman alıcıdır. Aynı zamanda, hangi bölgede ne tür yardıma ne kadar ihtiyaç olduğu net olarak belirlenemediği için, mevcut kaynaklar (gıda, ekipman, insan gücü) rastgele dağılmakta ve kritik alanlara zamanında ulaştırılamamaktadır.

Kurumlar ve gönüllüler için sahadaki ihtiyaçları anlık, konum bazlı ve türüne göre sınıflandırılmış şekilde görebilecekleri merkezi bir bilgi sisteminin eksikliği, müdahale ve yardım süreçlerini yavaşlatmakta, afet sonrası koordinasyonu ciddi şekilde zorlaştırmaktadır.

Bu bilgi eksikliği, karar vericilerin stratejik hareket etmesini engellerken, sahadaki vatandaşlar için de hayati yardımın gecikmesine neden olmaktadır.

## 3. Çözümünüz / Proje Fikri

ERA, deprem gibi kitlesel afetler sonrasında, bireylerin ihtiyaçlarını ve yardım imkanlarını konum tabanlı olarak harita üzerinden bildirebilecekleri mobil ve web tabanlı bir platformdur. Kullanıcılar, birkaç basit adımla bulundukları konuma göre 5 farklı kategoride (gıda, barınak, tıbbi yardım, kurtarma ihtiyacı, destek) nokta bırakabilirler ve ihtiyacı olanlar bu noktalara giderek ihtiyaçlarını karşılayabilirler.

Bu veriler, AWS altyapısı üzerinde çalışan sunucusuz bir backend tarafından toplanır, Geohash algoritması ile konumsal olarak gruplanır ve DynamoDB üzerinde ölçeklenebilir biçimde saklanır. Böylece belirli bir bölgedeki yardım çağrıları veya destek noktaları hızlıca analiz edilebilir.

Yetkili kurumlar (AFAD, belediyeler, STK’lar) ve gönüllü ekipler, bu platform sayesinde sahadaki ihtiyaçları anlık olarak harita üzerinden görebilir, yoğunluk haritaları ile öncelikli müdahale bölgelerini tespit edebilir ve yardımları bu doğrultuda yönlendirebilir. Depremden etkilenen insanlar kendisine en yakın bulunan destek noktasına giderek en hızlı şekilde yardım alabilirler.

## 4. Teknolojik Yaklaşım

* Frontend (Mobil): React Native (Expo)- Hızlı geliştirme ve cross-platform uyumluluğu için.
* Backend: AWS Lambda (Python)- Sunucusuz, ölçeklenebilir ve olay tabanlı işlem gücü için.
* API Gateway: AWS API Gateway- Lambda fonksiyonlarını RESTful API olarak sunmak ve yönetmek için.
* Veritabanı: AWS DynamoDB- NoSQL, yüksek performanslı, ölçeklenebilir ve esnek şemalı veri depolama için.
* Coğrafi İndeksleme ve Sorgulama: Python ile geliştirilmiş Geohash algoritması ve DynamoDB'nin sorgu yeteneği.
* Harita Servisi: OpenStreetMap

## 5. Hedef Kullanıcı / Fayda

* Afetzedeler: Acil ihtiyaçlarını (yiyecek, su, barınak, tıbbi yardım, kurtarma talebi) konumlarla birlikte kolayca giderebilecektir, göçük altında olanlar konumlarını hızlıca belirtebilir.
* Yardım Gönüllüleri ve Bireysel Yardımseverler: Harita üzerinden yardım noktalarının, yıkılan binalarını konumlarını belirtebilirler.
* Sivil Toplum Kuruluşları ve Yardım Organizasyonları: Sahadaki durumu anlık takip ederek kaynaklarını (insan gücü, malzeme) etkin bir şekilde planlayıp yönlendirebilecekler.
* Yetkili Kurumlar (AFAD, Kızılay, Belediyeler vb.): Genel afet tablosunu, ihtiyaç yoğunluklarını ve türlerini görerek stratejik kararlar alabilecek, büyük ölçekli koordinasyonu sağlayabilecekler.
* Fayda: Hızlı ve doğru bilgi akışı, kaynakların etkin kullanımı, müdahale sürelerinin kısalması, koordinasyonun artırılması, hayat kurtarma potansiyelinin yükseltilmesi.

## 6. Uygulama Planı (MVP için 24 saat)

| **Zaman Aralığı** | **Hedef** |  |
| --- | --- | --- |
| 0–4 saat | Temel React Native arayüzü (harita görünümü, nokta ekleme butonları), react-native-maps entegrasyonu. |  |
| 4–10 saat | AWS Lambda (Python) fonksiyonlarının DynamoDB tablo şeması ve Geohash entegrasyonunun tamamlanması. API Gateway kurulumu. |  |
| 10–18 saat | Frontend (React Native) ile Backend (API Gateway üzerinden Lambda) entegrasyonu. Temel nokta ekleme ve haritada görüntüleme işlevselliğinin uçtan uca testi. |  |
| 18–24 saat | Hata ayıklama, kullanıcı arayüzünde son rötuşlar, sunum ve demo senaryosunun hazırlanması. Raporun son haline getirilmesi. |  |

## 7. Takım Üyeleri ve Rolleri

| **Ad Soyad** | **Sorumluluk** |
| --- | --- |
| Koray Garip | Backend Api yönetimi ve araştırma |
| Alihan Gündoğdu | Backend |
| Kerem Özcan | Frontend |
| Ahmet Yumutkan | Frontend |

## 8. Riskler & Sınırlılıklar (1–2 madde yeterli)

* Süre Kısıtı: 24 saatlik süre zarfında, gelişmiş filtreleme seçenekleri, kullanıcı kimlik doğrulama (Cognito ile), anlık push bildirimleri gibi özellikler prototip aşamasında kalabilir veya tamamen MVP kapsamı dışında bırakılabilir. Temel "nokta ekleme" ve "yakındaki noktaları listeleme" işlevlerine odaklanılacaktır.
* Veri Simülasyonu: Gerçek zamanlı, büyük ölçekli ve çeşitli afetzede verisi elde etmek mümkün olmayacağından, sistemin işlevselliğini göstermek ve test etmek için veriler manuel olarak veya basit scriptlerle simüle edilecektir. API'ye gönderilen userId sabit bir değer olacaktır.