BURSA TEKNİK ÜNİVERSİTESİ
MÜHENDİSLİK VE DOĞA BİLİMLERİ FAKÜLTESİ
Bilgisayar Mühendisliği Bölümü

ERA

DİJİTAL DEPREM ÇÖZÜMLERİ

HACKATHON RAPORU

Hazırlayanlar

Ahmet YUMUTKAN

Alihan GÜNDOĞDU

Koray GARİP

Kerem ÖZCAN

Özet

Bu çalışma, Türkiye gibi deprem riski yüksek bölgelerde arama-kurtarma operasyonlarına katkı sağlamayı hedeflemektedir. Örneğin, 2023 yılında yaşanan ve “asrın felaketi” olarak nitelendirilen Kahramanmaraş merkezli depremler, Türkiye’nin güneydoğusunda geniş bir coğrafyayı etkileyerek büyük bir insani krize yol açmıştır. Bu felaket yalnızca fiziksel yıkıma değil; aynı zamanda bilgi kirliliği, iletişim aksaklıkları ve organizasyon zorluklarına da neden olmuştur. Afet sırasında sosyal medyada dolaşan doğrulanmamış ve yanlış bilgiler, yardım ekiplerinin yönlendirilmesinde ciddi sorunlar yaratmıştır.

ERA (Earthquake Relief Assistant) adlı bu mobil uygulama, sesli yardım çağrılarını yapay zeka aracılığıyla analiz ederek, konum bilgisi ile birleştirip merkezi bir harita üzerinden görselleştirmektedir. Kullanıcı girişi olmadan yalnızca acil yardım butonuyla yardım talebi oluşturulabilirken, giriş yapan kullanıcılar harita üzerinden yıkılmış binaları, yardım çadırlarını ve gıda noktalarını işaretleyebilmektedir.

ERA, deprem sonrası süreçleri de göz önünde bulundurarak barınma ve temel ihtiyaç lokasyonlarının harita üzerinde renk kodlarıyla gösterilmesini sağlar. Bu sayede ihtiyaç sahipleri yardıma erişebilirken, gönüllüler ve yardım kuruluşları da daha organize hareket edebilir. React Native (Expo) ile geliştirilen uygulama, Amazon Web Services (AWS) altyapısında sunucusuz (serverless) bir mimari ile çalışmaktadır. Backend katmanında AWS Lambda, API Gateway, DynamoDB, Amazon Transcribe ve Cognito servisleri yer almaktadır.

1. Giriş

Depremler yalnızca fiziksel yıkıma yol açan doğa olayları değil, aynı zamanda bilgi sistemlerinin dayanıklılığını ve toplumların kriz anındaki dijital reflekslerini test eden olaylardır. Deprem ülkesi olan Türkiye'de yüzbinlerce binanın yıkılmasına ve binlerce can kaybına neden olurken, aynı zamanda bilgi kirliliği ve koordinasyon sorunlarını da beraberinde getirmiştir.

Sosyal medyada yayılan doğrulanmamış yardım çağrıları ve yanlış konum bildirimi, kurtarma ekiplerinin zamanını boşa harcamasına yol açmıştır. Bu durum, doğru bilgiyi hızlı ve güvenilir biçimde iletebilecek bir sistem ihtiyacını ortaya koymuştur. ERA, tam da bu ihtiyaca yanıt olarak geliştirilmiştir.

Uygulamanın sunduğu özellikler arasında, tek tuşla sesli yardım çağrısı gönderme, konum bilgisinin otomatik iletimi, yapay zeka destekli analiz ve kategorilendirme, harita üzerinden interaktif işaretlemeler gibi işlevler bulunmaktadır. Kullanıcı girişi gerektirmeyen acil yardım butonu, afetzedelerin hızlıca yardım talep etmesini sağlarken; giriş yapan kullanıcılar sahadaki durumu güncel olarak işaretleyebilir.

ERA ayrıca deprem sonrası süreci de kapsayarak geçici barınma, gıda dağıtımı, sağlık hizmetleri gibi lokasyonların harita üzerinde gösterilmesini sağlar. Bu bilgiler farklı renkler ve ikonlarla sunularak ihtiyaç sahiplerinin bu alanlara yönelmesini kolaylaştırır. Böylece yalnızca enkaz altında kalanlar değil, yardım etmeye çalışan gönüllüler de bu sistem aracılığıyla organize olabilir.

2. Sistem Mimarisi ve Veri Akışları

ERA sistemi, temel olarak bir mobil uygulama (frontend) ve bulut tabanlı bir backend altyapısından oluşmaktadır. Bu iki bileşen, afet durumlarında kritik bilgilerin toplanması, işlenmesi ve sunulması için entegre bir şekilde çalışır.

2.1. Kullanılan Servisler ve Teknolojiler

Sistemin geliştirilmesinde kullanılan temel teknolojiler ve servisler aşağıdaki tabloda özetlenmiştir:

Katman	Teknoloji / Servis	Açıklama
Mobil Uygulama (Frontend)	React Native (Expo)	Platformlar arası (iOS & Android) mobil uygulama geliştirme çatısı. Hızlı geliştirme ve kolay dağıtım için Expo yönetimli iş akışı.
	react-native-maps	Harita görüntüleme, Marker ekleme ve kullanıcı etkileşimleri için temel harita bileşeni. OpenStreetMap ile entegrasyon.
	expo-av (Audio)	Cihaz mikrofonundan ses kaydı alma ve yönetme.
	expo-location	Kullanıcının mevcut coğrafi konumunu (enlem, boylam) elde etme.
	expo-file-system	Kaydedilen ses dosyalarını geçici olarak saklama ve base64 formatına çevirme.
	react-native-pager-view	Ana uygulama içinde farklı özellikler (Harita, Konum Bilgisi, Ses Kayıt) arasında sekmeli geçiş sağlama.
	aws-amplify	AWS Cognito ile kimlik doğrulama, AWS API Gateway üzerinden güvenli API çağrıları.

Kimlik Doğrulama	AWS Cognito	Kullanıcı kaydı, girişi, oturum yönetimi ve API Gateway yetkilendirmesi için OAuth 2.0 tabanlı yönetilen servis.
Backend Servisleri (AWS)	AWS Lambda	Python ile yazılmış, olay tabanlı, sunucusuz işlem fonksiyonları.
	API Gateway	Mobil uygulamadan gelen HTTP isteklerini Lambda fonksiyonlarına yönlendiren, API ile endpoint'leri oluşturma ve yönetme servisi. Cognito ile yetkilendirme.
	Amazon S3	Kaydedilen ses dosyalarının (.wav), konum verilerinin (location.json) ve transkript sonuçlarının (transcript.json) depolanması için obje depolama servisi.
	Amazon Transcribe	S3'e yüklenen ses dosyalarını otomatik olarak metne dönüştürme (Speech-to-Text). Türkçe dil desteği ile.
	Amazon Bedrock	Transkript metinlerini analiz ederek yapılandırılmış JSON formatında yardım özeti üreten üretken yapay zeka servisi.
Veritabanı (AWS)	Amazon DynamoDB	POINT verileri ve potansiyel olarak diğer uygulama verileri için NoSQL, anahtar-değer ve doküman veritabanı.

Yardımcı Araçlar	Python (boto3)	AWS SDK for Python; Lambda fonksiyonları içinde AWS servisleriyle etkileşim için.
	Geohash Algoritması	Coğrafi koordinatları, yakınlık tabanlı sorgulamalara olanak tanıyan kısa alfanümerik stringlere dönüştürme.

2.2. Temel Veri Akışları

Sistemin iş mantığı iki ana akış üzerinden yürütülür: Sesli yardım çağrısı ve harita üzerinden nokta ekleme/görüntüleme.

Sesli Yardım Çağrısı Akışı:
Kullanıcı mobil uygulamadaki "Acil Yardım" butonuyla ses kaydı başlattığında, `expo-av` ile ses alınır ve `expo-location` ile GPS koordinatları elde edilir. Kayıt tamamlandığında, ses dosyası `expo-file-system` ile base64 formatına çevrilir ve konum bilgisiyle birlikte JSON payload olarak AWS API Gateway üzerinden `/acil-ses` endpoint'ine gönderilir. Bu istek, AWS Cognito ile yetkilendirilir. API Gateway, isteği bir AWS Lambda fonksiyonuna yönlendirir. Bu Lambda fonksiyonu, base64 ses verisini decode ederek `.wav` formatında ve konum bilgisini `location.json` olarak benzersiz bir UUID ile Amazon S3'teki `deprem-data/audios/{UUID}/` klasörüne kaydeder.
S3'e yeni bir `.wav` dosyası yüklendiğinde, S3 Event Notification otomatik olarak bir başka Lambda fonksiyonunu tetikler. Bu fonksiyon, Amazon Transcribe servisine Türkçe dil desteğiyle bir transkripsiyon görevi başlatır. Transkripsiyon tamamlandığında sonuç `transcripts/` klasörüne `.json` olarak kaydedilir ve bu olay yeni bir S3 Event Notification ile üçüncü bir Lambda fonksiyonunu tetikler. Bu son Lambda, transkript metnini okur ve Amazon Bedrock servisine (Anthropic Claude 3 Sonnet modeli) özel bir prompt ile göndererek yapılandırılmış bir JSON formatında (açıklama, sağlık durumu, konum açıklaması, kat vb.) yardım özeti üretir. Orijinal konum bilgisi S3'ten çekilir, bu konum için bir Geohash hesaplanır

ve tüm bu toplanan bilgiler (yapılandırılmış yardım özeti, konum, Geohash) Amazon DynamoDB'deki `POINT` tablosuna kaydedilir.

Harita Üzerinden Nokta Ekleme/Görüntüleme Akışı:
Giriş yapmış kullanıcılar, harita üzerinde uzun basarak veya bir buton aracılığıyla yeni bir nokta (örn: gıda dağıtım yeri, yıkılmış bina) ekleyebilir. Seçilen noktanın koordinatları, kullanıcının girdiği açıklama ve seçtiği nokta tipi (GIDA, BARINMA, TIBBI_YARDIM, YIKINTI, DIGER) mobil uygulama tarafından alınır ve AWS API Gateway üzerinden `/point` endpoint'ine Cognito ile yetkilendirilmiş bir POST isteği ile gönderilir. API Gateway isteği bir Lambda fonksiyonuna iletir. Bu Lambda, Cognito'dan kullanıcı kimliğini alır, gelen koordinatlar için bir Geohash hesaplar ve veriyi (kullanıcı ID, koordinatlar, açıklama, tip, Geohash) DynamoDB'ye kaydeder.
Noktaları görüntülemek için, kullanıcı haritada gezindiğinde veya zoom yaptığında, mobil uygulama mevcut harita merkezi koordinatlarını ve bir yarıçap bilgisini API Gateway üzerinden `/point` endpoint'ine GET isteği ile gönderir. İlgili Lambda fonksiyonu, bu merkez koordinatları ve yarıçapa göre bir Geohash prefix'i hesaplar. DynamoDB'deki `POINT` tablosunda, bu Geohash prefix'i ile başlayan (`begins_with` sorgusu) tüm noktalar sorgulanır ve sonuçlar JSON formatında mobil uygulamaya döndürülür. Bu sayede coğrafi olarak yakın noktalar verimli bir şekilde listelenir.

2.3. Veritabanı Tasarımı ve Geohash Kullanımı

Veritabanı olarak Amazon DynamoDB (NoSQL) tercih edilmiştir. Coğrafi yakınlık tabanlı sorguların verimli bir şekilde yapılabilmesi için Geohash algoritması kullanılmaktadır. Geohash'ler, coğrafi koordinatları kısa alfanümerik stringlere dönüştürerek, birbirine yakın konumların benzer ön eklere (prefix) sahip olmasını sağlar. DynamoDB'nin `begins_with` sorgu operatörü ile belirli bir Geohash prefix'ine sahip tüm noktalar, tüm veritabanını taramak yerine sorguyu belirli bir alt kümeye indirgeyerek hızlıca bulunabilir. Bu, okuma performansını artırır ve maliyetleri düşürür. Geohash'in karakter uzunluğu (precision), sorgulanan coğrafi alanın büyüklüğünü belirler; daha uzun Geohash, daha küçük ve kesin bir alanı temsil eder. Sistem, istenen yarıçapa göre sorgulanacak Geohash prefix'inin uzunluğunu dinamik olarak ayarlar.

3. Mobil Uygulama Arayüzü ve Fonksiyonları

ERA mobil uygulaması, kullanıcı dostu bir arayüz ve kritik anlarda kolay erişilebilir fonksiyonlar sunmak üzere React Native ve Expo kullanılarak geliştirilmiştir.

Kullanıcı kimlik doğrulama ve yetkilendirme işlemleri AWS Cognito servisi aracılığıyla yönetilir. Kullanıcılar, e-posta adresleri, kullanıcı adı ve şifre ile kaydolabilir ve e-posta yoluyla gönderilen doğrulama koduyla hesaplarını aktif hale getirebilirler. Başarılı giriş sonrası, aws-amplify SDK'sı oturum token'larını yönetir ve API Gateway üzerinden yapılan yetkilendirilmiş API çağrılarında bu token'lar kullanılır. Kullanıcılar, "Çıkış Yap" butonu ile oturumlarını güvenli bir şekilde sonlandırabilirler.

Uygulamanın ana ekranı, `react-native-maps` bileşeni ile oluşturulmuş interaktif bir harita sunar ve OpenStreetMap altlık haritasını kullanır. Kullanıcının mevcut konumu veya varsayılan bir bölge merkezlenir. Harita üzerinde gezinildiğinde, o anki görünümün merkez koordinatları ve uygun bir yarıçap kullanılarak `fetchPointsFromApi` fonksiyonu çağrılır. Bu fonksiyon, backend Lambda fonksiyonu aracılığıyla DynamoDB'den Geohash tabanlı sorgu ile ilgili bölgedeki yardım noktalarını ve çağrılarını çeker. Çekilen noktalar, tipine göre farklı renklerde (`getPinColor` fonksiyonu ile belirlenir) `Marker` bileşenleri olarak haritada gösterilir. Bir Marker'a tıklandığında, o noktaya ait detay bilgileri (açıklama, tip vb. `getPrettyTypeName` ile formatlanır) bir modal pencerede sunulur. Harita bölgesi her değiştiğinde API'ye sık istek gitmesini önlemek için `fetchPoints` çağrısı `useDebounce` hook'u ile sarmalanarak performans optimizasyonu sağlanmıştır. Giriş yapmış kullanıcılar, haritaya uzun basarak yeni bir yardım noktası ekleyebilir; seçilen koordinatlar ve kullanıcı tarafından girilen açıklama/tip bilgileri `createPointApi` fonksiyonu ile backend'e gönderilerek kaydedilir ve haritada anında görüntülenir.

Sesli acil yardım çağrısı özelliği için, ekran yüklendiğinde `Audio.requestPermissionsAsync()` ile mikrofon izni istenir. "Kayıt Başlat" butonuyla `Audio.Recording.createAsync` kullanılarak `.wav` formatında ses kaydı başlatılır ve "Kaydı Durdur" ile sonlandırılır. Kaydedilen ses dosyası URI'si üzerinden `FileSystem.readAsStringAsync` ile base64 formatına dönüştürülerek, `expo-location` ile alınan mevcut konum bilgisiyle birlikte backend'e gönderilir. Bu veri, Bölüm 2.2'de açıklanan ses işleme akışını tetikler.

Ayrıca, "Konumum" sekmesinde `Location.requestForegroundPermissionsAsync()` ile konum izni alınır ve `Location.getCurrentPositionAsync({})` ile elde edilen enlem/boylam değerleri kullanıcıya gösterilir. Giriş yapıldıktan sonra ulaşılan ana uygulama arayüzü, `react-native-pager-view` kullanılarak "Harita", "Konumum" ve "Ses Kayıt" arasında geçiş yapılabilen sekmeli bir yapı sunar.

4. Performans, Ölçeklenebilirlik ve Test Stratejileri

ERA sisteminin tasarımı, yüksek performans, ölçeklenebilirlik ve güvenilirlik hedeflenerek yapılmıştır.

Performans açısından, mobil uygulamada `react-native-maps` bileşeninin çok sayıda Marker gösteriminde optimize çalışması kritik olup, harita üzerindeki API çağrıları `useDebounce` hook'u ile gereksiz istekleri azaltacak şekilde düzenlenmiştir. Backend'de ise Python ile yazılmış AWS Lambda fonksiyonları ve DynamoDB'nin Geohash tabanlı anahtar şeması ile yapılan `begins_with` sorguları, coğrafi yakınlık tabanlı aramalar için yüksek performans sunar. Amazon Transcribe ve Bedrock gibi yönetilen servisler de kendi içlerinde performans optimizasyonlarına sahiptir.

Ölçeklenebilirlik, AWS Lambda, API Gateway, S3, DynamoDB, Transcribe ve Bedrock gibi sunucusuz (serverless) servislerin kullanımıyla doğal olarak sağlanır. Bu servisler, gelen yük arttıkça AWS tarafından otomatik olarak ölçeklenir, bu da manuel sunucu yönetimi ihtiyacını ortadan kaldırır. DynamoDB, doğru anahtar tasarımıyla yüksek okuma/yazma kapasitelerine ölçeklenebilir. API Gateway de gelen istekleri karşılamak için otomatik ölçeklenir.

Testler, konusunda kapsamlı bir strateji benimsenmiştir. Backend Lambda fonksiyonlarının temel iş mantığı, Python'un `unittest` veya `pytest` gibi kütüphaneleriyle, AWS servis etkileşimleri ise `moto` veya `unittest.mock` ile mock'lanarak birim testlerine tabi tutulmuştur. Veritabanı erişim metodları için de birim testleri yazılmıştır. S3'e dosya yüklenmesiyle başlayan tüm Lambda akışının entegrasyon testleri ve API Gateway endpoint'lerinin doğrulaması

yapılmıştır. Frontend tarafında, Jest ve React Native Testing Library kullanılarak UI bileşenlerinin ve yardımcı fonksiyonların birim/komponent testleri gerçekleştirilmiştir.

5. Sonuç ve Gelecek Çalışmalar

ERA projesi, afet anlarında yalnızca fiziksel yardımı değil, dijital yardımı da öncelikli hale getiren, teknolojinin insan hayatını kurtarabilecek düzeyde etkili olduğunu gösteren bir çalışmadır. Sunucusuz mimarisiyle düşük maliyetli, ölçeklenebilir ve hızlı bir altyapı sunan sistem, hem enkaz altındaki bireylerin yardım çağrılarının yapay zeka destekli analiziyle doğru şekilde yönlendirilmesini hem de sahada bulunan gönüllülerin ve resmi ekiplerin harita tabanlı güncel bilgilerle daha organize şekilde çalışmasını sağlamaktadır. Geohash tabanlı veritabanı sorgulamaları ve modüler Lambda fonksiyonları, sistemin performansını ve bakımını kolaylaştırmaktadır.

6. Gelecek Geliştirmeler

- Uygulama olan kişilerin yakınında olan depremde bilgi verilmesi ve birbiriyle haberleşmesi.
- Yetkili arama-kurtarma ekipleri ve resmi kurumlar için özel bir web tabanlı yönetim paneli ve harita arayüzü geliştirilmesi.
- Push notification sistemi entegre edilerek, kullanıcılara yakınlarındaki önemli olaylar veya yardım noktaları hakkında anlık bildirimler gönderilmesi.
- Toplanan veriler üzerinde daha kapsamlı analizler yaparak afet sonrası ihtiyaçların ve riskli bölgelerin belirlenmesine yönelik raporlama araçları geliştirilmesi.
- Diğer afet türleri (sel, yangın vb.) için de uyarlanabilir bir altyapı oluşturulması.