🌦️ AuraHealth - Hava Durumu & AI Asistan Projesi

📋 Proje Özeti

AuraHealth, kullanıcılara kapsamlı hava durumu bilgileri, deprem verileri ve yapay zeka destekli sohbet deneyimi sunan modern bir web uygulamasıdır. Proje, Python Flask backend'i ve vanilla JavaScript frontend'i ile geliştirilmiştir.

🎯 Proje Hedefleri

Ana Hedefler:

- ✅ Gerçek zamanlı hava durumu bilgisi sağlama

- ✅ Machine Learning ile hava durumu tahmini

- ✅ Kandilli Rasathanesi deprem verilerini sunma

- ✅ Hava kirliliği ölçümlerini gösterme

- ✅ AI destekli chatbot ile kullanıcı etkileşimi

- ✅ Responsive ve kullanıcı dostu arayüz

Hedef Kitle:

- Hava durumu takipçileri

- Deprem bilgisi arayan kullanıcılar

- Çevre kirliliği konusunda bilinçli bireyler

- Teknoloji meraklıları

🛠️ Teknoloji Stack'i

Backend (Python):

- Flask: Web framework

- Joblib: ML model yükleme

- Pandas: Veri manipülasyonu

- Google Generative AI: Chatbot için Gemini API

- BeautifulSoup: Web scraping

- Requests: HTTP istekleri

- Flask-CORS: CORS desteği

Frontend:

- HTML5: Yapısal temel

- CSS3: Styling ve responsive tasarım

- Vanilla JavaScript: İnteraktivite

- Boxicons: İkon kütüphanesi

- Google Fonts: Typography

External APIs:

- OpenWeatherMap API: Hava durumu ve hava kirliliği

- Google Gemini API: AI sohbet

- Kandilli Rasathanesi: Deprem verileri

⚡ Temel Özellikler

1. 🌤️ Hava Durumu Modülü

- Anlık Hava Durumu: Sıcaklık, nem, hissedilen sıcaklık

- Konum Tabanlı: GPS koordinatları ile otomatik şehir tespiti

- Şehir Arama: Manuel şehir seçimi

- Görsel Gösterim: Hava durumuna uygun SVG ikonları

2. 🤖 Machine Learning Tahmini

- Gelişmiş Tahmin: Scikit-learn tabanlı ML modeli

- Tarih Seçimi: İleri tarihler için tahmin

- Çoklu Parametre: Sıcaklık, yağış miktarı tahmini

- Komut Tabanlı: "Tahmin et [Şehir] [DD/MM/YYYY]" formatı

3. 🏔️ Deprem Bilgi Sistemi

- Gerçek Zamanlı Veri: Kandilli Rasathanesi entegrasyonu

- Mesafe Hesaplama: Haversine formülü ile en yakın depremler

- Detaylı Bilgi: Büyüklük, derinlik, zaman, konum

- Görsel Sınıflandırma: Büyüklüğe göre renk kodlaması

4. 🌫️ Hava Kirliliği Monitörü

- AQI Değerleri: Air Quality Index hesaplaması

- Pollutant Detayları: PM2.5, PM10, O3, NO2 seviyeleri

- Renk Kodlaması: Kirlilik seviyesine göre görsel gösterim

- Sağlık Tavsiyeleri: Kirlilik seviyesine uygun öneriler

5. 💬 AI Chatbot (E.V.A)

- Google Gemini Entegrasyonu: Gelişmiş doğal dil işleme

- Komut Sistemi: Özel komutlarla işlevsellik

- Session Yönetimi: Sohbet geçmişi saklama

- Çok Dilli Destek: Türkçe optimizasyonu

📊 API Endpoint'leri

| Endpoint | Method | Açıklama |

|----------|--------|----------|

| /get\_weather\_info | POST | Hava durumu bilgisi |

| /chat | POST | AI chatbot etkileşimi |

| /get\_air\_pollution | GET | Hava kirliliği verileri |

| /normalize\_location | POST | Koordinat → şehir çevirimi |

| /get\_city\_from\_coords | POST | Koordinatlardan şehir adı |

| /get\_earthquake\_data | GET | Ham deprem verileri |

| /chat\_status | GET | Chat session durumu |

🎨 Kullanıcı Arayüzü

Tasarım Prensipleri:

- Responsive Design: Mobil ve desktop uyumlu

- Clean Interface: Minimal ve modern tasarım

- Intuitive Navigation: Alt navigasyon menüsü

- Visual Feedback: Loading states ve error handling

Panel Sistemi:

1. Hava Durumu Paneli: Ana hava durumu bilgileri

2. Hava Kirliliği Paneli: AQI ve pollutant verileri

3. Deprem Paneli: En yakın deprem listesi

4. Chat Paneli: AI asistan ile sohbet

Renk Paleti:

- Primary: #43affc (Mavi)

- Success: #00e400 (Yeşil)

- Warning: #ffff00 (Sarı)

- Danger: #ff0000 (Kırmızı)

- Background: #fff (Beyaz)

🚀 Kurulum ve Çalıştırma

Backend Kurulumu:

pip install flask flask-cors joblib pandas beautifulsoup4 google-generativeai requests

python app.py

Frontend Kurulumu:

python -m http.server 5500

Gerekli Dosyalar:

- hava\_durumu\_model.pkl: ML modeli

- encoder.pkl: Şehir encoder'ı

- API anahtarları (OpenWeatherMap, Gemini)

🔧 Geliştirme Süreci

Aşama 1: Backend Geliştirme

- ✅ Flask API yapısı kurulumu

- ✅ External API entegrasyonları

- ✅ ML model entegrasyonu

- ✅ Web scraping implementasyonu

Aşama 2: Frontend Geliştirme

- ✅ Responsive UI tasarımı

- ✅ JavaScript API çağrıları

- ✅ Panel navigasyon sistemi

- ✅ Error handling ve loading states

Aşama 3: Entegrasyon

- ✅ Frontend-Backend bağlantısı

- ✅ CORS konfigürasyonu

- ✅ Session yönetimi

- ✅ Geolocation API entegrasyonu

Aşama 4: Optimizasyon

- ✅ Kod temizleme

- ✅ Performance iyileştirmeleri

- ✅ Error handling güçlendirme

- ✅ UI/UX iyileştirmeleri

📈 Teknik Detaylar

Machine Learning Modeli:

- Algoritma: XGBoost regression

- Özellikler: Gün, ay, hafta günü, şehir

- Çıktılar: Max sıcaklık, min sıcaklık, yağış

- Encoding: One-hot encoding şehirler için

Veri Kaynakları:

- OpenWeatherMap: Hava durumu ve kirlilik

- Kandilli Rasathanesi: Deprem verileri

- Google Gemini: AI sohbet

- Navigator Geolocation: Kullanıcı konumu

Security:

- Flask session yönetimi

- API key güvenliği

- CORS policy

- Input validation

🔮 Gelecek Planları

Kısa Vadeli:

- 📱 Mobile app geliştirme

- 🔔 Push notification sistemi

- 📊 Veri visualizasyon iyileştirme

- 🌍 Çoklu dil desteği

Uzun Vadeli:

- 🤖 Daha gelişmiş AI özellikleri

- 📈 Kullanıcı analytics

- ☁️ Cloud deployment

- 🔗 Sosyal medya entegrasyonu

Özellik Geliştirmeleri:

- Hava durumu alarm sistemi

- Deprem erken uyarı

- Kişiselleştirilmiş öneriler

- Kullanıcı profil sistemi

📊 Proje İstatistikleri

Kod Metrics:

- Backend: ~1,000 satır Python kodu

- Frontend: ~800 satır JavaScript kodu

- CSS: ~600 satır styling kodu

- API Endpoints: 7 aktif endpoint

Özellik Sayısı:

- 🌦️ 4 Ana panel

- 🔧 7 API endpoint'i

- 🤖 1 AI chatbot

- 📱 Tam responsive tasarım

🎯 Sonuç

AuraHealth projesi, modern web teknolojileri ve yapay zeka entegrasyonu ile kapsamlı bir hava durumu ve bilgi platformu sunmaktadır. Kullanıcı dostu arayüzü, güvenilir veri kaynakları ve gelişmiş özellikler ile hedef kitlesine değer katmayı amaçlamaktadır.

Proje, sürekli geliştirme ve iyileştirme odaklı bir yaklaşımla tasarlanmış olup, gelecekte daha da gelişmiş özellikler ve platformlar için genişletilebilir bir yapıya sahiptir.

---

Geliştirici: Ephesta

Geliştirme Tarihi: 2025

Teknoloji Stack: Python Flask + Vanilla JavaScript

Lisans: Özel Proje