1. Giriş

Bu rapor, Görüntü Analiz ve Araştırma Aracı projesinin teknik detaylarını özetlemektedir. Aşağıda proje kapsamında uygulanan mimari, kod bileşenleri, metodolojiler, veri kaynakları, ön/ardıl işlemler ve modellerin açıklamaları yer almaktadır.

Proje önce terminalde çalışacak şekilde yapılandırılmış sonra django ile arayüzü yapılmıştır .Raporda ana bileşenlere odaklanılmıştır arayüz yapım aşamasına yer verilmemiştir.

Projeyi çalıştırmak için: python manage.py runserver yapılmalıdır

2. Ana Bileşenler ve Kod Detayları

Proje dört ana bileşenden ve bu bileşenlere ait kod modüllerinden oluşmaktadır. Her bir bileşenin sorumlulukları, kullanılan kütüphaneler ve hata yönetimi şu şekildedir:

2.1 ImageLoader

- **Amaç**: Kullanıcının verdiği yerel dosya yolu veya URL üzerinden görüntüyü güvenli biçimde yüklemek.
- **Kütüphaneler**: PIL (Image, UnidentifiedImageError), requests, io, os.
- **Temel Metodlar**:
    - load_from_source(source: str): Kaynağı tespit ederek _load_from_url veya _load_from_file işlemlerine yönlendirir.
    - _load_from_url(url: str): HTTP isteği (User-Agent, timeout ile), content-type kontrolü, byte verisini PIL.Image.open ve verify() ile doğrulama.
    - _load_from_file(file_path: str): Dosya varlık kontrolü, PIL.Image.open ve verify() ile bütünlük kontrolü.
- **Hata Yönetimi**:
    - requests.exceptions.Timeout, RequestException → ağ hatası/timeout mesajı.
    - IOError, UnidentifiedImageError, SyntaxError → bozuk veya desteklenmeyen format bildirimi.
    - FileNotFoundError → yerel dosya bulunamadı uyarısı.

2.2 GeminiMultiModalAnalyzer

- **Amaç**: Google Gemini API'si ile hem görüntü analizi (nesne çıkarımı ve anahtar kelimeler) hem de metin özetleme yapmak.

- **Kütüphaneler**: google.generativeai as genai, PIL, re.
- **Yapı ve Metodlar**:
    - __init__(api_key: str): genai.configure, vision_model ve text_model nesnelerini ön tanımlı modellerle başlatır.
    - _configure_gemini(): API anahtarını doğrular; başarısızlıkta ConnectionError fırlatır.
    - analyze_image(image: Image): Görüntüyü prompt ile gönderir, model yanıtını alır ve _parse_analysis_response ile ayrıştırır.
    - _parse_analysis_response(text_response: str): Regex temelli Nesne: ve Anahtar Kelimeler: formatlarını ayıklar.
    - summarize_text(context: str, object_name: str): Web içeriğinden derlenen metni prompt ile özetler.
- **Hata Yönetimi**:
    - API çağrılarında genel Exception yakalama, hata mesajı basımı.
    - Ayrıştırma hatalarında ham model yanıtını _print_failed_response ile yazdırma.

2.3 WebScraper

- **Amaç**: DuckDuckGo arama sonuçlarından gelen sayfaları indirip, HTML temizliği ve metin çıkarımı yapmak.
- **Kütüphaneler**: duckduckgo_search.DDGS, requests, bs4.BeautifulSoup, re, time.
- **Metodlar**:
    - search_and_extract(query: str, num_results: int): DDGS ile arama, bulunan URL'leri filtreleyip _fetch_and_parse_url ile içerik çıkarımı.
    - _fetch_and_parse_url(url: str): HTTP isteği, content-type kontrolü, BeautifulSoup ile gereksiz etiketlerin dekompozisyonu, get_text ile metin elde etme, maksimum karakter uzunluğu sınırlaması.
- **Hata Yönetimi**:
    - requests.exceptions.Timeout, RequestException → bağlantı/indirme hatası.
    - Genel Exception → parse ve ağ hataları.
    - Sunucuları yormamak için isteklere time.sleep(0.5) eklenmiştir.

2.4 ResultStorage

- **Amaç**: Analiz sonuçlarını JSON formatında dosyaya kaydetmek ve geçmiş kayıtları yüklemek.
- **Kütüphaneler**: json, os.
- **Metodlar**:
    - load_history(): JSON dosyasını varlık kontrolü ve json.load, format kontrolü ile listeleri döndürür.
    - save_result(result_data: dict): load_history ile mevcut listeyi alır, yeni sonucu ekler, json.dump ile dosyaya yazar (ensure_ascii=False, indent=4).
- **Hata Yönetimi**:
    - json.JSONDecodeError, IOError → dosya okuma/yazma hatalarında kullanıcı bilgilendirmesi.

3. Kullanılan Yapay Zeka Metodolojileri ve Algoritmalar

Ön Eğitilmiş Model (Google Gemini Multimodal API):

Tercih Sebebi:

1. Çokmodlu Analiz Desteği: Görüntüden nesne ve anahtar kelime çıkarımı ile metin özetlemeyi tek bir API içerisinde sağlaması, ayrı ayrı modelleri entegre etme yükünü ortadan kaldırır.
2. Türkçe Dil Yetkinliği: Türkçe prompt ve çıktı sağlayabilmesi, modelin yerel dildeki sonuçların kalitesini artırır.
3. Yüksek Doğruluk ve Güncel Performans: Transformer tabanlı mimarisi sayesinde hem görsel hem metinsel görevlerde öne çıkan doğruluk değerleri sunar.
4. Hızlı Prototipleme ve Kolay Entegrasyon: API tabanlı kullanım, altyapı kurulum ve bakım maliyetini düşürerek geliştirme sürecini hızlandırır.
5. Ölçeklenebilirlik ve Güvenilirlik: Google Cloud altyapısı üzerinde çalıştığı için yüksek istek hacimleri altında bile stabil hizmet sağlar.

- Nesne ve Anahtar Kelime Tespiti: Transformer tabanlı Gemini multimodal modeli; image-to-text yetenekleri.
- Metin Özetleme: Transformer mimarili özetleme; uzun metinler için timeout artırımı.

4. Veri Kaynakları

- **Girdi Verisi**: Kullanıcıdan dosya veya URL yoluyla alınan gerçek zamanlı görüntüler.
- **Web İçerikleri**: Anahtar kelime odaklı dinamik DuckDuckGo aramaları.