



ZONGULDAK BÜLENT ECEVİT ÜNİVERSİTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

**BLM497 – BİLGİSAYAR MÜHENDİSLİĞİ PROJE TASARIMI DERSİ
GEREKSİNİMLER ŞARTNAMESİ VE ÖN ANALİZ BELGESİ (GŞÖA)**

Öğrenciler		Adı Soyadı	Öğrenci No	İmza
	1.	Şevval Nur Çapur	220106109072	
	2.	Merve Sünbüle	220106109067	
	3.	Rabia Ceylan	220106109023	
	4.	Kamile Burcu İpek	220106109066	
Tarih:	30.10.2025			
Proje Danışmanı:	Dr. Öğretim Üyesi Erkan Çetiner			

Projenin Başlığı: Görüntü İşleme ile Zehirli Mantar Tespiti

1. Giriş

Bu belgenin amacı, görüntü işleme ve derin öğrenme yöntemlerini kullanarak zehirli ve yenilebilir mantarların tespitini gerçekleştiren bir mobil uygulamanın gereksinimlerini ve ön analizini açıklamaktır. Proje, kullanıcıların doğada buldukları mantarların fotoğraflarını çekip sisteme yükleyerek zehirli olup olmadığını öğrenmelerine olanak tanır. Böylece hem gıda güvenliği hem de insan sağlığı açısından risklerin önüne geçilmesi hedeflenmektedir.

2. Tanımlar ve Kısaltmalar

Kısaltma	Açıklama
CNN	Convolutional Neural Network – Evrişimli Sinir Ağı
TensorFlow	Derin öğrenme modelleri için açık kaynak framework
OpenCV	Görüntü işleme kütüphanesi
TFLite	TensorFlow Lite – Mobil cihazlarda model çalıştırma
Dataset	Modelin eğitimi için kullanılan veri kümesi
UI / UX	Kullanıcı Arayüzü / Deneyimi
API	Uygulama Programlama Arayüzü

3. Ürün

3.1. Ürün Kapsamı

Uygulama, kullanıcıdan alınan mantar fotoğrafını derin öğrenme tabanlı modelle analiz ederek 'Zehirli' veya 'Yenilebilir' sonucunu verir. Sistem TensorFlow ortamında eğitilmiş bir CNN modelini kullanır ve mobilde çalışabilmesi için TensorFlow Lite formatına dönüştürülür. Eğitim verisi Kaggle üzerindeki açık kaynak mantar görüntü veri setlerinden sağlanacaktır.

3.2. Ürün Değeri

Proje, doğa yürüyüşçüleri ve biyolojiyle ilgilenen kullanıcılar için pratik bir araç sunar. Yanlış mantar tüketiminden kaynaklı zehirlenmeleri önleyerek insan sağlığı açısından değer yaratır.

3.3. Hedef Kitle

Doğa yürüyüşü ve kamp severler, mantar toplayıcıları, botanik/biyoloji öğrencileri, gıda güvenliği çalışanları.

3.4. Planlanan Kullanım

Kullanıcı mantar fotoğrafını çeker veya galeriden yükler. Görüntü işlenir, model tahmin yapar ve sonuç gösterilir.

3.5. Genel Tanım

Sistem dört ana modülden oluşur: Veri alma, ön işleme, model tahmini ve sonuç gösterimi. Transfer learning (ör. MobileNetV2/EfficientNet) tercih edilecek; veri artırma (döndürme, parlaklık, yakınlaştırma) ve sınıf ağırlıklandırma kullanılarak dengesiz sınıf problemleri azaltılacaktır.

4. İşlevsel (Fonksiyonel) Gereksinimler

ID	Gereksinim	Açıklama
FR-1	Fotoğraf Çekme	Kullanıcı uygulama içinden kamera ile fotoğraf çekebilmelidir.
FR-2	Görüntü Yükleme	Kullanıcı galeriden fotoğraf seçebilmelidir.
FR-3	Görüntü İşleme	Görüntü modele uygun hale getirilmelidir (ör. 224x224, RGB, normalizasyon).
FR-4	Model Çalıştırma	TensorFlow Lite modeli çalıştırılarak sonuç üretilmelidir.
FR-5	Sonuç Gösterimi	Zehirli/Yenilebilir sonucu ve güven oranı kullanıcıya sunulmalıdır.
FR-6	Eşik ve Uyarılar	Belirsizlik durumunda 'Emin Değilim' çıktısı ve uyarı mesajı gösterilmelidir.
FR-7	Sonuç Kaydı	Kullanıcı, geçmiş analizlerini isteğe bağlı olarak görüntüleyebilmelidir.
FR-8	Hata Yönetimi	Başarısız durumlarda kullanıcıya uygun uyarı ve yönlendirme sağlanmalıdır.

5. Dış Arayüz Gereksinimleri

5.1. Kullanıcı Arayüzü Gereksinimleri

Uygulama arayüzü **ReactNative** tabanlı olarak geliştirilecektir.

Tasarım, sade, anlaşılır ve mobil ekranlara tam uyumlu bir yapıdadır.

Kullanıcı ana ekranda “Fotoğraf Çek” ve “Galeriden Seç” seçenekleriyle mantar görüntüsünü sisteme aktarır.

Sonuç ekranında mantarın “Zehirli” veya “Yenilebilir” durumu, güven oranı ile birlikte gösterilir. Renk kodlamasıyla sonuçlar görsel olarak da belirtilir (örnek: kırmızı – zehirli, yeşil – yenilebilir).

5.2. Donanım Arayüzü Gereksinimleri

Uygulama, **Android ve iOS** işletim sistemli akıllı telefonlarda çalışacaktır.

Minimum gereksinimler:

- **RAM:** En az 4 GB
- **Kamera:** Fotoğraf çekim özelliği
- **Depolama:** Görüntü işleme ve model dosyası için yeterli yer (yaklaşık 100 MB)
Ağ bağlantısı yalnızca model güncellemeleri veya istatistiksel veri aktarımı için kullanılabilir.
Tahmin işlemleri çevrimdışı yapılabilir.

5.3. Yazılım Arayüzü Gereksinimleri

Uygulama, **TensorFlow Lite, OpenCV, NumPy** ve **Expo** kütüphaneleri kullanılarak geliştirilecektir. Kod geliştirme ortamı olarak **Visual Studio Code** kullanılacaktır.

Model dosyası cihaz içinde çalışacak, sunucu bağlantısı gerektirmeyecektir.

Arayüz (frontend) ve model işlem katmanı (backend) React Native içinde entegre biçimde çalışacaktır.

5.4. İletişim Arayüzü Gereksinimleri

Uygulama temelde **çevrimdışı** çalışacak şekilde tasarlanmıştır.

Fotoğraf işleme ve model tahmini işlemleri cihaz üzerinde yapılır.

İnternet bağlantısı yalnızca **model güncellemeleri** veya **performans istatistiklerinin** gönderimi gibi isteğe bağlı işlemler için gereklidir.

Herhangi bir dış e-posta veya sunucu tabanlı iletişim modülü bulunmamaktadır.

6. İşlevsel Olmayan Gereksinimler

6.1. Güvenlik

- Fotoğraflar cihazda işlenmeli, buluta yüklenmemelidir.
- Uygulama yalnızca gerekli izinleri (kamera, depolama) istemelidir.
- Veri aktarımı (varsa) TLS üzerinden şifrelenmelidir.
- Kullanıcıdan açık rıza alınmalı, KVKK/GDPR ilkelerine uyulmalıdır.

6.2. Kapasite

- Model boyutu hedefi: 30–60 MB (TFLite, kuantizasyonlu).
- Uygulama toplam boyutu hedefi: ≤ 100 MB.
- Geçmiş kayıtları (opsiyonel) için cihaz başına ~50–100 kayıt saklanacaktır.

6.3. Uyumluluk

- Minimum işletim sistemi: Android 10+, iOS 14+.
- Minimum donanım: 64-bit CPU, 4 GB RAM, kamera erişimi.
- TFLite NNAPI/Metal hızlandırma mevcutsa kullanılacaktır.

6.4. Güvenilirlik

- Hedef doğruluk (toplam): ≥ 80 (öğrenci projesi için gerçekçi hedef); esneklik aralığı: 80–85.
- Zehirli sınıfı için duyarlılık (recall): ≥ 85 (güvenlik önceliği).

- Zehirli sınıfı için kesinlik (precision): $\geq \%80$.
- Belirsiz örneklerde 'Emin Değilim' çıktısı verilecek ve kullanıcı uyarılacaktır (eşik tabanlı).

6.5. Ölçeklenebilirlik

Yeni mantar türleri ve ek sınıflar kolayca eklenebilecek şekilde veri ve model boru hattı tasarlanacaktır.

6.6. Sürdürülebilirlik

Proje, uzun vadeli sürdürülebilirlik hedefiyle tasarlanmıştır. Görüntü işleme tabanlı yapay zekâ modeli, **transfer learning (aktarım öğrenmesi)** yöntemiyle geliştirildiği için, zaman içinde yeni veri kümeleriyle yeniden eğitilerek güncel kalabilir.

Bu sayede sistemin doğruluk oranı çevresel değişiklikler, yeni mantar türleri veya görüntü kalitesi farklılıklarından etkilenmeden **en az 5–10 yıl boyunca** sürdürülebilir bir şekilde çalışması öngörülmektedir. Modelin ömrü, kullanılan **TensorFlow Lite** altyapısı ve düzenli versiyon güncellemeleriyle uzatılabilir. Uygulamanın bakım planı, hem bulut tabanlı hem de cihaz üzerinde model güncelleme desteğini içerecek şekilde kurgulanmıştır.

Ayrıca açık kaynak kütüphaneler ve modüler yapı sayesinde, yeni teknolojilere geçiş süreci kolaylaştırılmıştır.

6.7. Kullanılabilirlik

Mantar Tanıma Sistemi, her yaştan ve teknik bilgiden kullanıcının rahatlıkla kullanabileceği şekilde tasarlanmıştır. Arayüz sade, yönlendirici ve adım adım ilerleyen bir yapıya sahiptir. Kullanıcı, yalnızca **3 basit adımda** sonuca ulaşabilir:

1. Fotoğraf çekmek veya galeriden bir görüntü seçmek,
2. Sistemin tahmin sürecini başlatmak,
3. Sonuç ekranında tahmin edilen “**Zehirli / Yenilebilir**” bilgisini ve güven oranını görüntülemek.

Arayüzdeki butonlar, ikonlar ve renkler kullanıcıyı yönlendirecek biçimde tasarlanmıştır. Sistem, mobil cihazlar (Android/iOS) ve masaüstü ortamlarda çalışabilir; bu da erişilebilirliği artırır. Ayrıca kullanıcıdan özel bir teknik bilgi beklenmez — sadece temel akıllı telefon kullanım becerisi yeterlidir.

7. Ön Analiz

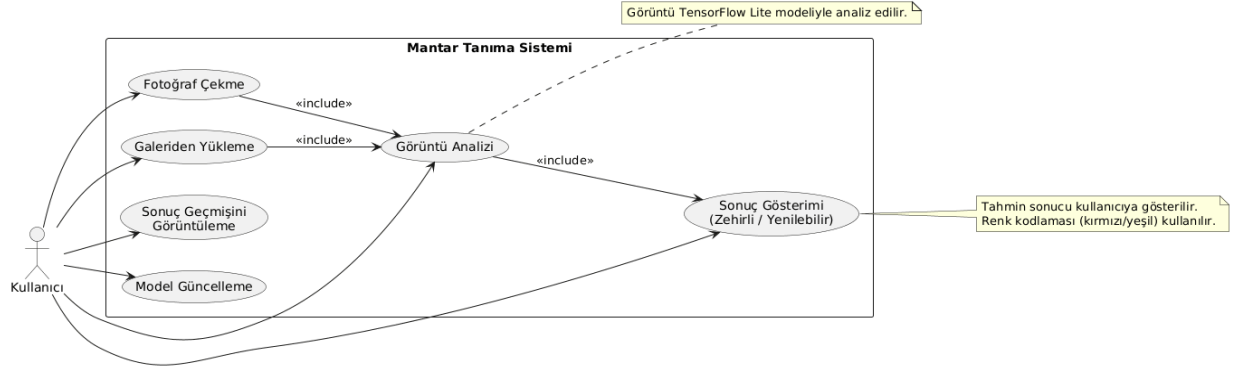
Problem, nesne-yönelimli bir yaklaşımla ele alınarak sisteme ilişkin aşağıdaki UML (Unified Modeling Language = Birleşik Modelleme Dili) grafikleri verilmelidir. Bu bölümde sözel anlatımlar da sunularak grafik gösterimler desteklenebilir.

7.1. Kullanım Durumu (Use Case) Grafikleri

Use Case Diyagramı: Fotoğraf çekme, işleme ve tahmin akışı.

Bu diyagramda, kullanıcı ve sistem arasındaki temel etkileşimler gösterilmiştir.

- **Aktör:** Kullanıcı
- **Senaryolar:**
 - Fotoğraf çekme veya galeriden yükleme
 - Görüntünün analiz edilmesi
 - Sonucun gösterilmesi (Zehirli / Yenilebilir)
 - (Opsiyonel) Sonuç geçmişinin görüntülenmesi
 - (Opsiyonel) Model güncelleme



7.2. Sınıf (Class) Grafikleri

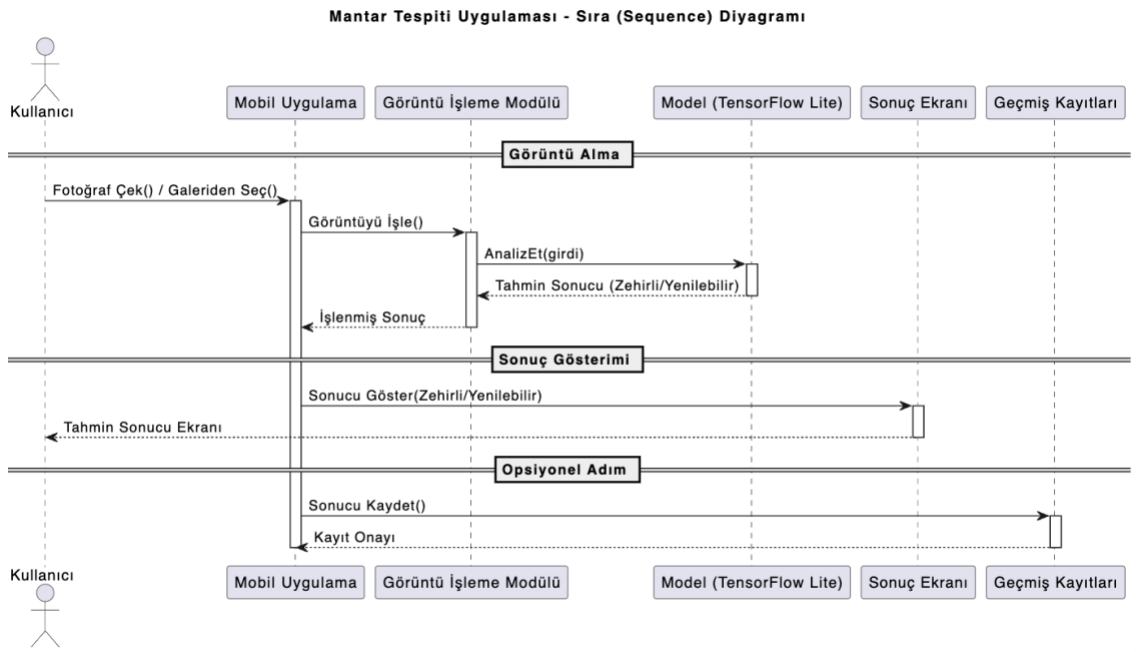
Class Diyagramı: Görüntü işleme ile model eğitileceği için class yapısı bulunmadığı için bu grafik çizilmemiştir.

7.3. Sıra (Sequence) Grafikleri

Sequence Diyagramı: Fotoğraf çek → İşleme → Tahmin → Sonuç Gösterimi.

Bu diyagramda, işlemlerin zaman sırasına göre akışı gösterilmiştir:

1. Kullanıcı fotoğraf çeker veya galeriden seçer.
2. Uygulama görüntüyü işler (yeniden boyutlandırma, normalizasyon).
3. Model tahmini gerçekleştirilir.
4. Sonuç ekranda gösterilir.
5. (Opsiyonel) Tahmin sonucu geçmişe kaydedilir.



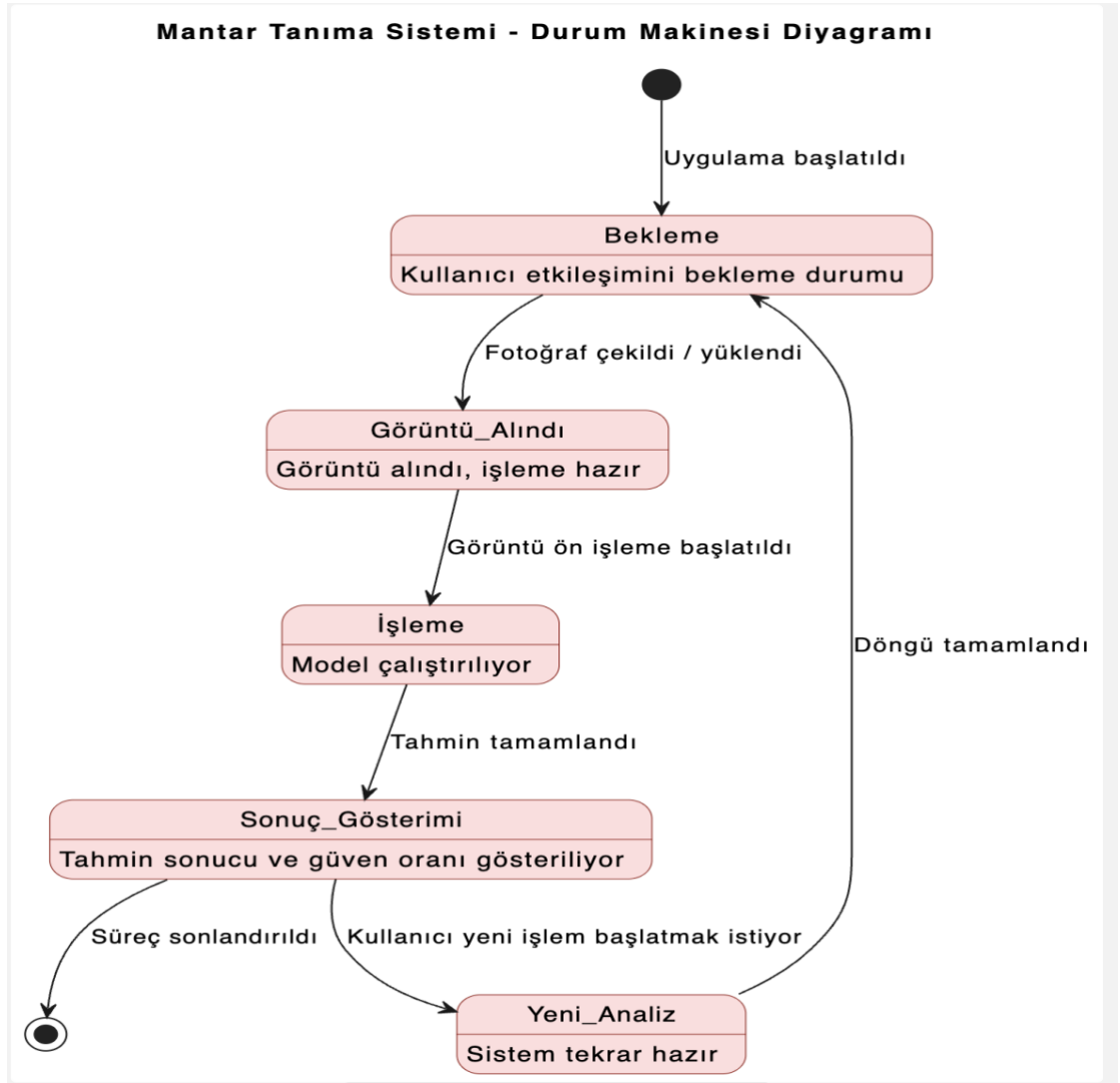
7.4. Durum Makinesi (State Machine) Grafikleri

State Machine Diyagramı: Bekleme → İşleme → Sonuç Gösterimi → YeniAnaliz.

Bu diyagram, sistemin durum değişikliklerini ve geçişlerini açıklar:

Bekleme: Kullanıcı etkileşimini bekleme durumu

- Görüntü Alındı: Fotoğraf çekme veya yükleme tamamlandı
- İşleme: Görüntü ön işleme ve model tahmini aşaması
- Sonuç Gösterimi: Tahmin sonucu ve güven oranı kullanıcıya sunulur
- Yeni Analiz: Kullanıcı yeni bir işlem başlatabilir



7.5. Etkinlik (Activity) Grafikleri

Activity Diyagramı: Kullanıcının fotoğraf yükleyip sonucu alması akışı.

Bu diyagram, işlemlerin sıralı adımlarını ve karar noktalarını göstermektedir:

- Başlat
- Görüntü seç / çek
- Görüntü ön işleme
- Model tahmini çalıştırma
- **[Karar: Güven oranı düşük mü?]**
 - Evet → “Emin Değilim” uyarısı → Sonuç ekranı
 - Hayır → Sonuç (Zehirli / Yenilebilir) gösterimi
- (Opsiyonel) Sonucun geçmişe kaydedilmesi
- Bitir

Mantar Tanıma Sistemi - Etkinlik (Activity) Diyagramı

