İÇİNDEKİLER

[İÇİNDEKİLER 0](#_Toc165397670)

[ŞEKİL LİSTESİ 1](#_Toc165397671)

[TABLO LİSTESİ 2](#_Toc165397672)

[1. Takım Şeması 3](#_Toc165397673)

[2. Proje Mevcut Durum Değerlendirmesi 4](#_Toc165397674)

[3. Algoritmalar ve Sistem Mimarisi 5](#_Toc165397675)

[3.1. Veri Setleri 5](#_Toc165397676)

[3.2. Veri Önişleme 6](#_Toc165397677)

[3.3. Algoritmalar 9](#_Toc165397678)

[3.4. Mobil Uygulama 11](#_Toc165397679)

[3.5. Akış Şeması 21](#_Toc165397680)

[4. Projede kullanılacak yöntem, donanımlar ve yazılımlar ile ilgili bilgiler 22](#_Toc165397681)

[4.1. Donanımlar 22](#_Toc165397682)

[4.2. Yazılımlar 22](#_Toc165397683)

[5. Özgünlük 24](#_Toc165397684)

[5.1.1. Veri Setinin Homojen Bir Şekilde Eğitim Veri Seti ve Test Veri Seti Olarak Ayrılması 24](#_Toc165397685)

[5.1.2. Veri Setinin Yapay Zekadan Yardım Alınarak Etiketlenmesi 24](#_Toc165397686)

[5.1.2.1. LabelImg 25](#_Toc165397687)

[6. PROJE TAKVİMİ 26](#_Toc165397688)

[7. Sonuçlar ve İnceleme 28](#_Toc165397689)

[8. Kaynakça 29](#_Toc165397690)

ŞEKİL LİSTESİ

[Şekil 1. Takım Şeması 3](#_Toc168579846)

[Şekil 2. Veri Setine Ait Örnek Görseller 5](#_Toc168579847)

[Şekil 3. BSR GAN Uygulanmamış Görseller 7](#_Toc168579848)

[Şekil 4. BSR GAN Uygulanmış Görseller 7](#_Toc168579849)

[Şekil 5. Öteleme Veri Arttırma Tekniği Kullanılmamış Görseller 9](#_Toc168579850)

[Şekil 6. Öteleme Veri Arttırma Tekniği Kullanılarak Elde Edilen Görseller 9](#_Toc168579851)

[Şekil 8. Mobil Uygulama Arayüz Tasarımı 12](#_Toc168579852)

[Şekil 9. Uygulama Kod -1 13](#_Toc168579853)

[Şekil 10. Uygulama Kod -2 14](#_Toc168579854)

[Şekil 11. Uygulama Kod -3 14](#_Toc168579855)

[Şekil 12. Uygulama Kod -4 14](#_Toc168579856)

[Şekil 13. Uygulama Kod -5 15](#_Toc168579857)

[Şekil 14. Uygulama Kod -6 16](#_Toc168579858)

[Şekil 15. Uygulama Kod -7 17](#_Toc168579859)

[Şekil 16. Uygulama Kod -8 18](#_Toc168579860)

[Şekil 17. Uygulama Kod -9 19](#_Toc168579861)

[Şekil 18. Uygulama Kod -10 20](#_Toc168579862)

[Şekil 19. Akış Şeması 21](#_Toc168579863)

[Şekil 20. LabelImg Etiketleme Programı 25](#_Toc168579864)

[Şekil 21. YOLOv5 Model Eğitimi Test Sonuçları 28](#_Toc168579865)

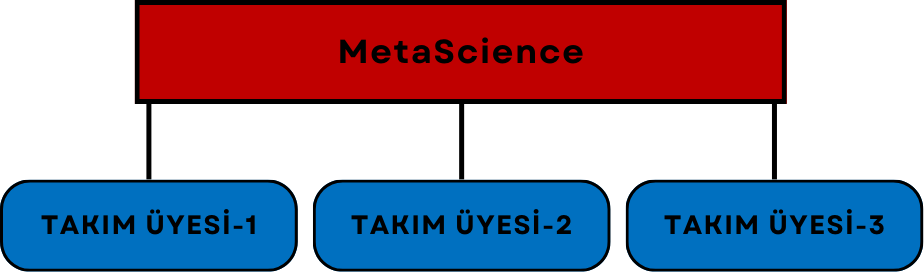
# TABLO LİSTESİ

[Tablo 1. YOLO Algoritmaları Karşılaştırma Tablosu 10](#_Toc165397385)

[Tablo 2. Proje Takvimi 26](#_Toc165397386)

# Takım Şeması

MetaScience Yapay Zekaya Giriş takımı 2023 yılında Necmettin Erbakan Üniversitesi Bilgisayar Mühendisliği bölümünde öğrenim gören 3 öğrenci tarafından kurulmuştur. Takım şeması **Şekil 1**‘de belirtilmiştir.



Şekil 1. Takım Şeması

**Takım üyesi-1:** Nesne tespiti ile ilgili tez, makale, rapor araştırılması ve incelenmesi, kullanılabilecek veri setlerinin araştırılması, veri setlerinin indirilmesi ve kullanılacak veri setlerinden proje kategorisine uygun olan görüntülerin seçilmesi, data augmentation teknikleri kullanarak veri çoğaltılması, veri setlerinin eğitim veri seti ve test veri seti olarak ayrılmasından sorumludur.

**Takım üyesi-2:** Proje hedeflerine en uygun algoritmanın belirlenmesi, belirlenen nesne tespit algoritması ile model eğitimi gerçekleştirilmesi, mobil uygulama geliştirme, arayüz tasarımı ve API bağlantısından sorumludur.

**Takım üyesi-3:** Veri setinin etiketlenmesi, oluşturulan modelinin test veri setleri ile kontrolünün gerçekleştirilmesi, modeldeki eksikliklerin belirlenmesi ve optimizasyonundan sorumludur.

# Proje Mevcut Durum Değerlendirmesi

Teknolojinin gelişimiyle birlikte nesne tespiti, yapay zeka alanında önemli bir araştırma ve uygulama alanı haline geldi. Bu bağlamda, kol saatlerinin markalarını tespit ederek kullanıcılarına daha iyi bir deneyim sunmayı hedefleyen bir projeyi hayata geçirilmektedir. Proje, kol saati görsellerinden oluşan geniş bir veri seti üzerine odaklanıyor ve bu veri setiyle birlikte YOLOv5 nesne tespit algoritması kullanılarak bir yapay zeka modelinin eğitilmesi hedeflenmektedir.

Projenin ilk aşamasında, 10 farklı kol saati firmasına ait görsellerden oluşan bir veri seti toplandı ve etiketlendi. Bu veri seti, çeşitli ortamlarda farklı açılardan çekilmiş saat görsellerini içeriyor. Eğitim sürecinde veri çeşitliliğini artırmak amacıyla öteleme tekniklerinden faydalanıldı, görüntülerdeki saati çeşitli yönlere kaydırarak veri seti zenginleştirme işlemleri gerçekleştirildi. Ayrıca, görsel kalitesi artırmak için BSR GAN algoritması kullanılarak çözünürlük iyileştirildi.

Eğitim aşaması tamamladıktan sonra, YOLOv5 öğrenme algoritmasıyla geliştirilen model, saatin markasını belirlemek için girdi olarak verilen görselleri analiz etmektedir. Modelin doğruluğunu test etmek için ayrılmış olan doğrulama veri setiyle modelin performansı değerlendirilecektir. Model doğruluk oranı yeterli düzeye ulaştığında, geliştirilen model, kullanıcıya rahat kulanım kolaylığı sağlaması açısından mobil uygulamaya entegre edilecektir.

Mobil uygulama, Android Studio platformu aracılığıyla Flutter yazılım geliştirme kiti kullanılarak geliştirilecektir. Kullanıcılar uygulama üzerinden galeriyi kullanarak sisteme bir kol saati fotoğrafını yükleyeceklerdir. YOLOv5 ile geliştirilen nesne tespit algoritması, girdi olarak gönderilen görseldeki saat markasını tespit edecektir. Sonrasında, ilgili tespit edilen marka bilgisi ekranda gösterilecektir.

Geliştirilmekte olan proje, yapay zeka, mobil uygulama geliştirme ve veri analitiği gibi farklı alanları bir araya getirerek kullanıcıya değerli bir deneyim sunmayı hedeflemektedir. Şu anda proje planlama ve hazırlık aşamasında bulunmaktadır, yakın gelecekte kullanıcılarla buluşması planlanmaktadır.

# Algoritmalar ve Sistem Mimarisi

## Veri Setleri

Analiz ve çalışmalarda kullanılan veri seti Kaggle platformundan temin edilmiştir. Kaggle, veri bilimi ve makine öğrenimi alanında dünyanın önde gelen topluluk tabanlı platformlarından biridir. Kullanıcılar, gerçek dünya veri setleri üzerinde yarışmalar düzenleyebilir, projeler geliştirebilir ve eğitim materyallerine erişebilirler. Ayrıca, Kaggle, veri bilimcileri ve makine öğrenimi uzmanlarının bir araya gelerek fikir alışverişi yapabileceği ve iş birliği yapabileceği bir ortam sunar. Bu platform, veri odaklı inovasyonu teşvik ederken, topluluk etkileşimini artırarak öğrenme ve gelişmeyi destekler.

Projede kullanılması planlanan veri seti, toplamda 10 farklı saat markasına ait görsellerden oluşmaktadır [1]. Veri setinde, Audemars Piguet, Breguet, Breitling, Maurice Lacroix, Omega, Patek Philippe, Rado, Rolex, TISSOT ve Vacheron Constantin saat markalarına ait görseller bulunmaktadır. Her bir marka saat için 200 adet görüntü bulunmaktadır.

Veri seti, toplamda 2000 görüntüden oluşmaktadır. Her bir görsel, farklı açılardan ve farklı ortamlarda çekilmiş saatlerin resimlerini içermektedir. Görüntüler değişken çözünürlük değerlerine sahiptir ve çeşitli ışık koşullarında çekilmiş olmaları nedeniyle renk ve parlaklık farklılıkları gösterebilmektedir. Veri setindeki görseller, PNG formatında ve ortalama boyutları 474x474 piksel olarak tespit edilmiştir. Kaggle platformundan temin edilen verilere ek olarak, Google görseller üzerinden saat markalarına ait görüntüler toplanarak mevcut veri seti genişletilecektir.

saat, kol saati, Analog saat, askı, kayış içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 2. Veri Setine Ait Örnek Görseller

## Veri Önişleme

Projede kullanılması planlanan veri seti, ortalama olarak 474x474 piksel boyutlarında ve düşük kalitede görüntülerden oluşmaktadır. Bu durum, geliştirilecek olan nesne tespit modeli için olumsuz senaryolar oluşturmaktadır. Bu nedenle, olumsuz senaryoların etkilerini azaltmak amacıyla çeşitli veri ön işleme tekniklerinin araştırılması ve uygulanması kararlaştırılmıştır. Geliştirilmesi planlanan yapay zeka modelinin etkili bir şekilde tespit yapabilmesi için veri setindeki fotoğrafların kalitesinin artırılması gerekmektedir. Bu amaç doğrultusunda, literatür taraması yapılmıştır, aşağıdaki çıkarımlar elde edilmiştir.

Super-Resolution, düşük çözünürlüklü (LR) bir görüntünün çözünürlüğünü yapay olarak artırarak yüksek çözünürlüklü bir görüntü (HR) elde etme işlemidir. Bu işlem, görüntünün detaylarını ve keskinliğini artırmayı amaçlamaktadır. Literatürde çeşitli Super-Resolution algoritmaları mevcuttur. Bu algoritmalar arasında öne çıkanlardan biri BSRGAN'dır. BSRGAN, gerçekçi ve kesin görüntüler üretme yeteneği ile dikkat çekmektedir.

Bu işlemi yapmak için yapay zeka kullanmaktadır ve şimdiye kadar geliştirilmiş en güçlü Super-Resolution tekniklerinden biri olarak tanımlanmaktadır.

BSRGAN, genel amaçlı blind image super-resolution (kör görüntü süper çözünürlüğü) ve bulanık görüntü giderme için güçlü bir araçtır. Kör görüntü süper çözünürlüğü, bir görüntünün düşük çözünürlüklü bir versiyonunu elde edip, bu düşük çözünürlüklü girdiye dayanarak yüksek çözünürlüklü bir versiyonunu tahmin etmeye çalışan bir super-resolution yöntemidir. BSRGAN'ın amacı, genel amaçlı kör görüntü süper çözünürlüğü sorununu çözmektir. Ayrıca, bir görüntünün Laplace varyansına bağlı olarak bulanık görüntülerini de gidermesini sağlar [2].

Kaggle platformundan temin edinilen veri seti üzerinde, BSRGAN algoritmasıyla üretilen çıktılar aşağıda gösterilmektedir.

saat, Analog saat, kol saati, ok içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 3. BSR GAN Uygulanmamış Görseller

saat, Analog saat, moda aksesuar, kuvars saat içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 4. BSR GAN Uygulanmış Görseller

Son zamanlarda, radyoloji ve radyoterapi alanlarında, yapay zeka araştırmaları, derin öğrenmeye dayalı algoritmaların kullanımına giderek daha fazla bağımlı hale gelmiştir. Bu algoritmalar, geleneksel makine öğrenimi yöntemlerine göre önemli ölçüde daha iyi performans göstermekle birlikte, daha büyük eğitim veri kümelerine dayanmaktadır. Ancak, tıbbi görüntüler gibi genellikle sınırlı veriye sahip alanlarda, bu büyük veri kümeleri bulunmayabilir. Bu sorunu çözmek için popüler bir yöntem olan veri arttırma teknikleri, eğitim veri kümesinin boyutunu artırmaya yöneliktir. Bu teknikler, derin öğrenme algoritmalarında, eğitim için mevcut veri miktarını artırarak daha iyi sonuçlar elde edilmesini sağlamaktadır [3].

Literatürde, temel büyütme teknikleri, geometrik dönüşümler, filtreleme gibi veri arttırma yöntemleri detaylı bir şekilde araştırılmıştır.

Temel büyütme teknikleri, artırılmış bir görüntü oluşturmak için görüntünün noktalarını farklı bir konuma eşleyen veya görüntü yoğunluk değerlerini değiştiren bir görüntüye dönüşüm uygulayan teknikleri içermektedir [3].

Geometrik dönüşümler kullanılan en yaygın temel büyütmedir. Bu dönüşümler ölçekleme, öteleme, döndürme, yansıtma ve kesme gibi afin veya bazen çarpıklık içeren perspektif dönüşümlerdir [3].

**Ölçekleme (Scaling):** bir görüntünün boyutunu değiştirme işlemidir. Ölçekleme, görüntüyü büyütme veya küçültme olarak gerçekleşebilir. Bu işlem, her bir boyut için bir ölçek faktörü kullanılarak gerçekleştirilir [3].

**Öteleme (Translation):** Bir görüntüyü belirli bir yönde hareket ettirme işlemidir. Öteleme, görüntüyü yatay ve dikey olarak belirli bir mesafe kadar kaydırarak gerçekleştirilir [3].

**Döndürme (Rotation):** Bir görüntüyü belirli bir nokta etrafında döndürme işlemidir. Döndürme, bir merkez noktası ve bir açı derecesi kullanılarak gerçekleştirilir [3].

**Yansıtma (Reflection):** Bir görüntüyü aynalayarak simetrik bir şekilde değiştirme işlemidir. Yansıtma, görüntünün belirli bir eksen boyunca yansıtılmasıyla gerçekleştirilir [3].

**Kesme (Shearing):** Bir görüntünün belirli bir yönde belirli bir açıyla kaydırılmasıdır. Bu işlem, görüntünün her bir pikselinin, belirli bir eğimle belirli bir yönde hareket ettirilmesiyle gerçekleştirilir [3].

Bir görüntünün filtrelenmesi evrişim kullanılarak yapılmaktadır. Bunu başarmak için evrişimsel bir çekirdek, çevredeki piksellerin değerlerine dayalı olarak her pikseldeki yoğunluğu değiştirmek üzere görüntü boyunca hareket ettirilir. Bu yöntemi kullanarak bir görüntü keskinleştirilebilir, bulanıklaştırılabilir veya yumuşatılabilir ve böylece artırılmış bir görüntü elde edilebilir [3].

Yapılan literatür taramaları sonucunda, geliştirilmesi planlanan yapay zeka modelinde veri arttırma teknikleri olarak geometrik dönüşümlerin tercih edilmesinin daha uygun olduğu sonucuna ulaşılmıştır. Bu tercih, temel büyütme tekniklerinin verinin boyutunu ve yoğunluğunu değiştireceği ve dolayısıyla farklı bir görüntü elde edileceği gerçeğinden kaynaklanmaktadır.

Diğer yandan, filtreleme işlemleri, örneğin gürültü azaltma veya keskinleştirme gibi, bu yapay zeka problemi için uygun bir veri arttırma yöntemi olarak görülmemektedir. Bu nedenle, sadece öteleme yöntemi geometrik dönüşümler içinde veri arttırma amacıyla kullanılacaktır. Öteleme yöntemi ile var olan görüntülerin koordinatları değiştirilerek yeni görüntüler elde edilecektir.

saat, Analog saat, moda aksesuar, askı, kayış içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 5. Öteleme Veri Arttırma Tekniği Kullanılmamış Görseller

saat, Analog saat, moda aksesuar, askı, kayış içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 6. Öteleme Veri Arttırma Tekniği Kullanılarak Elde Edilen Görseller

## Algoritmalar

Derin öğrenme nesne tespiti dedektörleri iki kategoriye ayrılmaktadır. Bunlardan birincisi aday alanı çıkarıp derin öğrenme gerçekleştirerek tespit ve sınıflandırmayı uygulayan iki aşamalı nesne dedektörleridir. Bu dedektör tipik olarak diğer yöntemlerden daha yüksek bir doğruluk oranına sahiptir ancak tespit sürecinde birden fazla aşamanın yer alması nedeniyle çıkarım hızları daha yavaştır [4]. Ren ve ark. tarafından önerilen iki aşamalı nesne dedektörü olan Faster R-CNN, bir görüntüyü girdi olarak alıp o görüntünün özellik haritasını döndüren ConvNet'e (CNN) iletir. Özellik haritaları üzerine Region Proposal Network (RPN) uygular. RPN, nesne adaylarını, aday puanları (objectness scores) ile birlikte döndürür. Bölgeler sabit bir boyutta yeniden boyutlandırılmak üzere ROI pooling katmanına gönderilir. Son olarak, bölgeleri fully connected katmanda sınıflandırılmak üzere besler [5,6]. İkinci dedektör sınıfı ise tek aşamalı nesne detektörleridir. Bu dedektörler bölge önerme adımını kaldırılarak hem nesne lokalizasyonunu hem de sınıflandırmayı aynı adımda yapmaktadır.

YOLO, nesne tespiti için evrişim sinir ağını kullanan tek aşamalı bir derin öğrenme algoritmasıdır. Hızı ve doğruluğu nedeniyle popülerdir. Çeşitli derin öğrenme algoritmaları vardır ancak bunlar bir nesneyi tek seferde tespit edemezler. YOLO ise algılamayı bir sinir ağı üzerinden tek bir ileri yayılımda yapar. Bu da onu gerçek zamanlı uygulamalara uygun hale getirir. Bu özellik YOLO algoritmasını diğer derin öğrenme algoritmaları arasında popüler hale getirmiştir [7].

YOLOv1, bir görüntüyü S × S boyutundaki ızgara hücrelerine ayırarak işlemektedir. Her bir ızgara hücresi, bir nesnenin merkezinin o hücreye düşmesi durumunda bu nesneyi tespit etmekten sorumludur. Her bir hücre, belirli bir güven puanıyla ilişkilendirilmiş B adet sınırlayıcı kutuyu tahmin edebilir. Bu sınırlayıcı kutular x, y, w, h değerlerinden ve bir güven puanından oluşur. Burada x, y, w ve h sırasıyla sınırlayıcı kutunun merkezi, genişliği ve yüksekliğini ifade etmektedir. YOLO, bir nesneyi algılamak için önce sınırlayıcı kutular tahmin eder. Daha sonra, her bir ızgara hücresi için tahmin edilen sınırlayıcı kutular arasında, bir nesnenin gerçek konumunu en iyi temsil eden kutuyu seçmek için Birleşim Üzerinden Kesişme (IOU) kriterini kullanır. Ancak, bu süreçte birden fazla sınırlayıcı kutu olabilir. Bu fazlalığı ortadan kaldırmak için YOLO maksimum olmayan bastırmayı (Non-Maximum Suppression) kullanır. Bu, yüksek IOU'ya sahip olan ve güven düzeyi düşük olan sınırlayıcı kutuları kaldırarak nesne algılama doğruluğunu artırır. Özetle, IOU değeri 0.5'in üzerinde olan sınırlayıcı kutuları bir araya getirirken düşük güven puanına sahip fazla sınırlayıcı kutuları elimine eder. YOLOv1, kaybı hesaplamak için hata kareler toplamını kullanır. YOLOv2'de doğruluğu artırmak ve aşırı uyum problemini azaltmak için evrişim katmanlarıyla birlikte toplu normalizasyon eklendi [8]. YOLOv3'te, küçük nesneleri tespit etmekte zorlanan Darknet19'un özellik çıkarma omurgası, bu sorunu çözmek için Darknet53 olarak değiştirildi [9]. YOLOv4'te özellik çıkarıcıların omurgası ise CSPDarknet53 olarak değiştirildi. Bu durum algoritmanın hızını ve doğruluğunu önemli ölçüde arttırdı. YOLOv5, önceki YOLO algoritmalarının en yeni ve hafif versiyonudur. Darknet çerçevesi yerine PyTorch çerçevesini kullanır [7].

Yapılan literatür taraması sonucunda geliştirilecek nesne tespit modeli için YOLOv3, YOLOv4 ve YOLOv5 algoritmaları incelenip aşağıdaki çıkarımlar elde edinilmiştir [10, 11, 12].

Tablo 1. YOLO Algoritmaları Karşılaştırma Tablosu

metin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu

Yapılan literatür tarama sonuçları göz önünde bulundurularak YOLOv5 nesne tespit algoritmasının diğer alternatif öğrenme algoritmalarına kıyasla daha hızlı çalıştığı ve daha yüksek doğruluk oranına sahip olduğu gözlemlenmiştir. Bu nedenle, geliştirilmesi planlanan yapay zeka modelinde YOLOv5 algoritmasının kullanılması planlanmaktadır. İlerleyen süreçlerde farklı öğrenme algoritmalarıyla yapılan model eğitim ve test işlemleri sonucunda, nihai olarak kullanılacak olan en uygun algoritmanın belirlenmesi amaçlanmaktadır.

## Mobil Uygulama

Kullanıcıya arayüz sağlamak için flutter yazılım geliştirici kiti ile mobil uygulama geliştirilecektir.

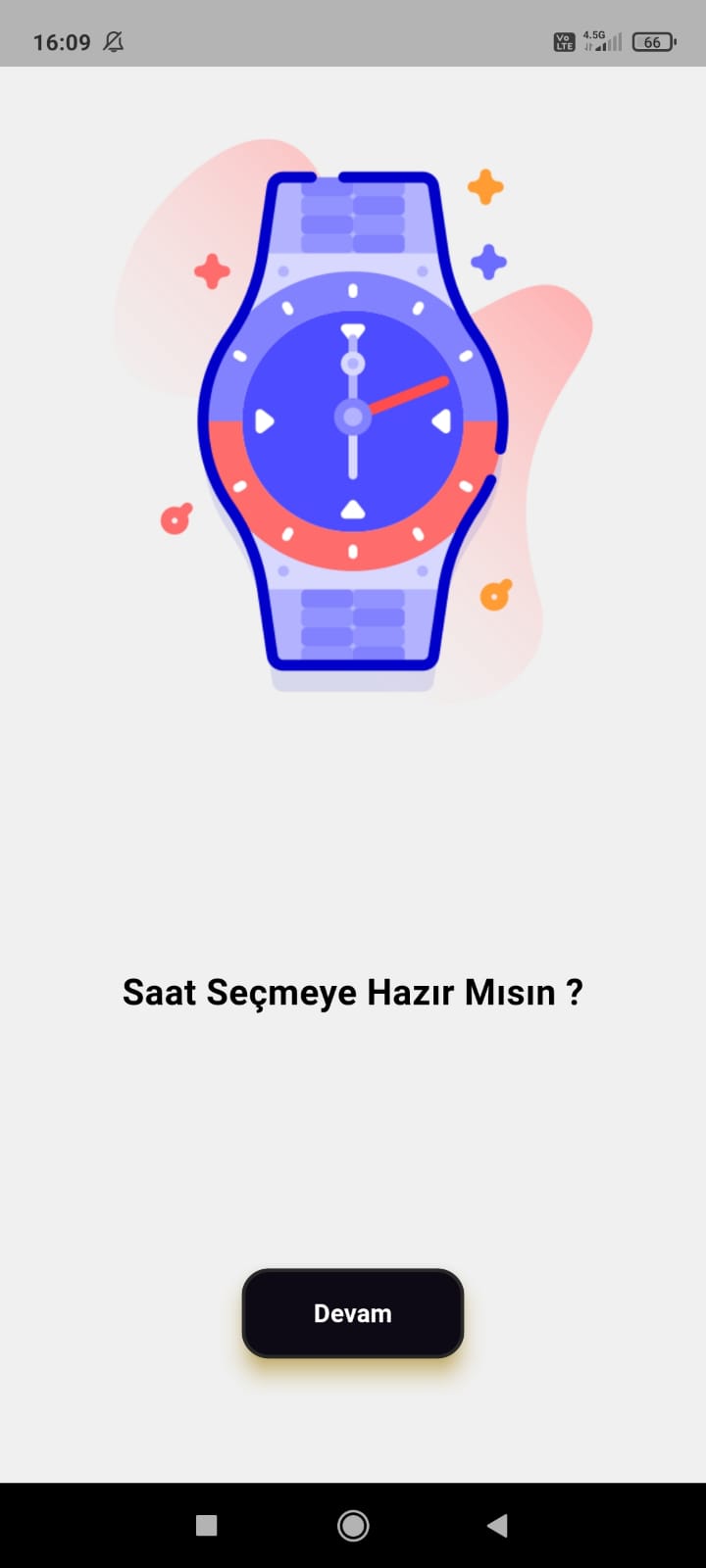
Google tarafından geliştirilen Flutter, açık kaynaklı bir mobil uygulama geliştirme çerçevesidir ve Dart programlama dilini kullanır. Bu çerçeve, Android ve iOS platformlarında çalışan, benzer bir görünüme sahip, yüksek performanslı, estetik ve kullanıcı dostu mobil uygulamaların geliştirilmesine imkan sağlamaktadır [13].

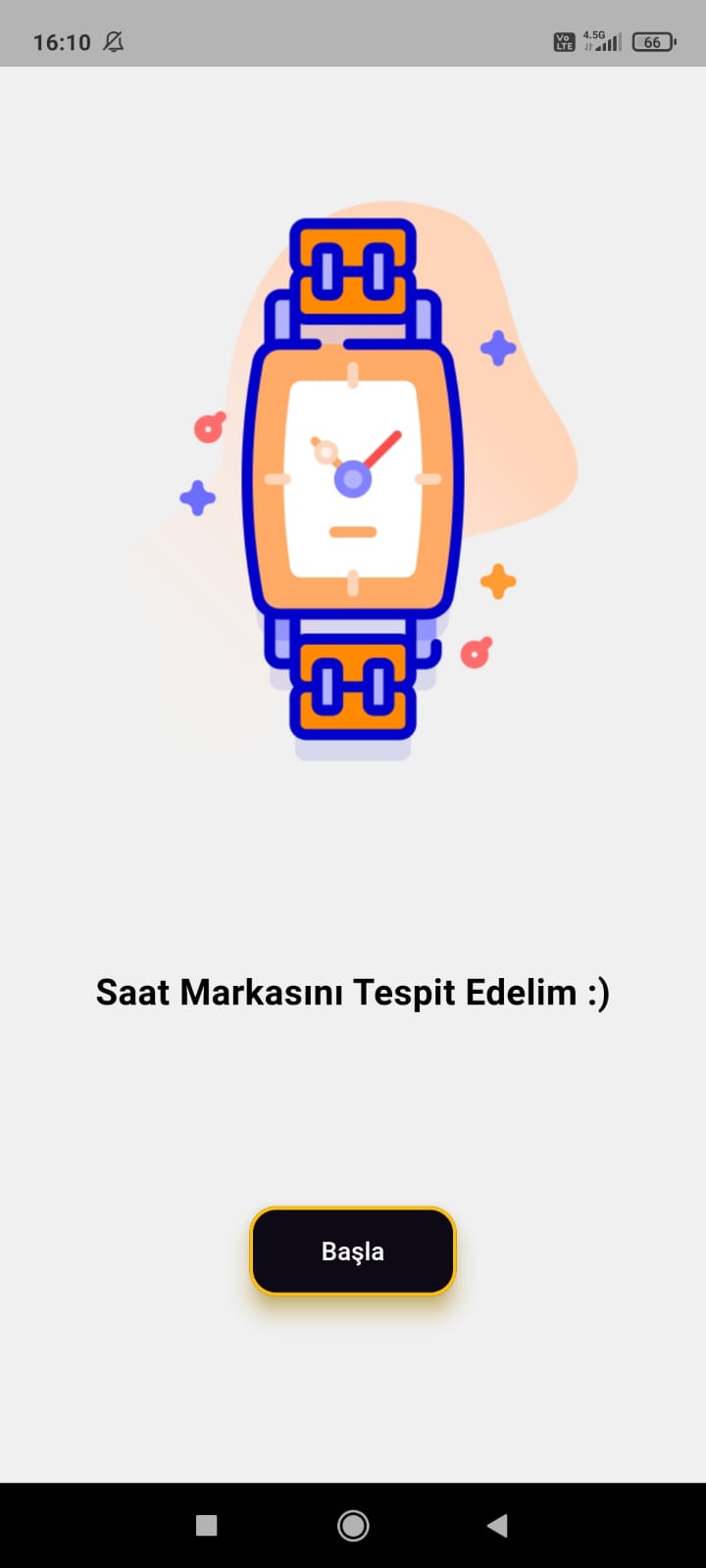
Flutter, Dart adlı bir yazılım dili üzerine inşa edilmiştir. Bu dil, açık kaynaklı bir yapıya sahiptir, C ve Java gibi diğer programlama dillerine benzer bir sözdizimine sahiptir. Flutter, Dart dilini kullanarak uygulama geliştiricilerinin hızlı ve verimli bir şekilde mobil uygulamalar oluşturmalarına olanak sağlamaktadır [13].

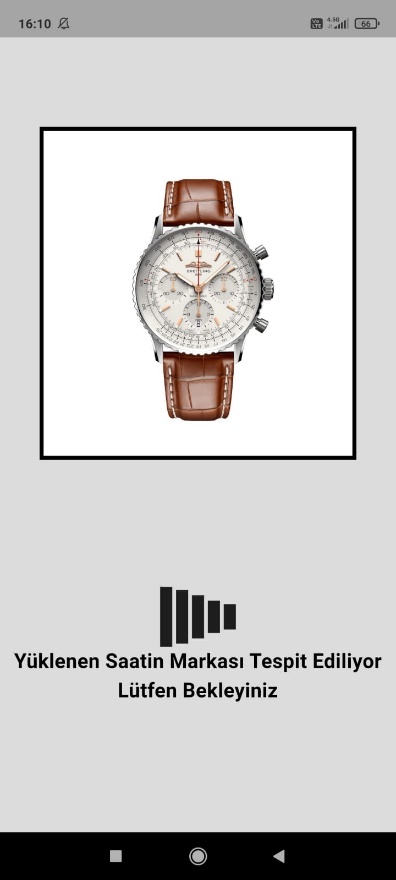
Projede kullanılacak olan mobil uygulama, kullanıcıların uygulamaya saat resimleri yüklemelerine ve ardından ilgili resmin markasını tespitine olanak sağlamaktadır. Uygulamanın geliştirilmesi için Android Studio IDE ortamı tercih edilmiştir.

Android Studio, Android uygulama geliştirme işlemlerini kolaylaştırmak için Google tarafından sunulan bir entegre geliştirme ortamıdır.

Bu yazılım, Android uygulamaları ve oyunları oluşturmak, test etmek ve hata ayıklamak için gerekli olan araçları ve kaynakları sağlamaktadır.

metin, ekran görüntüsü, yazılım, işletim sistemi içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, saat, ekran görüntüsü, moda aksesuar içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 7. Mobil Uygulama Arayüz Tasarımı

Kullanıcı uygulamayı açtığında "Saat Seçmeye Hazır Mısın?" ekranı ile karşılaşacaktır. "Başla" butonuna tıklandığında, "Sisteme Fotoğraf Yüklemek İçin Son Bir Adım Kaldı😊" ekranına geçiş yapılacaktır. Ardından "Başla" butonuna tıklanarak "Saat Markasını Tespit Edelim 😊" ekranına geçilecektir. Son olarak, "Başla" butonuna tıklayarak galeriden fotoğraf seçme işlemi gerçekleştirilecektir. Fotoğraf sisteme yüklendiğinde, sunucuda bulunan YOLOv5 modeli ile saat fotoğrafının marka tespiti gerçekleştirilecektir. Marka tespit sonucu doğruluk oranı ile birlikte uygulama ekranında gösterilecektir. Kullanıcı isterse "Yeni Tespit Yapalım" butonuna tıklayarak yeni bir tespit işlemi gerçekleştirebilecektir.

**Uygulamada Kullanılan Kütüphaneler:**

* **import 'dart:io'** 🡪 Giriş-çıkış işlemleri gerçekleştirmek için kullanılmaktadır.
* **import 'package:flutter/material.dart'** 🡪 Flutter ugulamalarında kullanılan yaygın widgetlerin kullanılması için gereklidir.
* **import 'package:flutter\_riverpod/flutter\_riverpod.dart'** 🡪 Durum yönetimi için kullanılmaktadır. Navigator gibi yapı elemanlarından daha esnek bir yapı sunmaktadır.
* **import 'package:image\_picker/image\_picker.dart'** 🡪 Galeriden fotoğraf çekmek için kullanılmaktadır.
* **import 'package:http/http.dart' as http** 🡪 http istekleri yapmak için kullanılmaktadır.
* **import 'package:flutter\_spinkit/flutter\_spinkit.dart'** 🡪 Veri yüklenirken veya arka planda belirli bir işlem gerçekleştiğinde animasyon özellikleri sağlamak için kullanılmaktadır.

**Uygulama Kod Bölümü:**

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 8. Uygulama Kod -1

* **StateProvider** ile durum değişikliği sağlanmıştır.
* **PageController** ise sayfalar arası geçiş yapmak için kullanılmaktadır.
* **initialPage** başlangıç sayfasını ifade etmektedir.
* **keepPage** ise sayfalar arası geçiş yaparken mevcut sayfanın hatırlanmasını sağlar ve mevcut sayfayı uygulama yeniden başlatıldığında korumaktadır.

metin, yazı tipi, ekran görüntüsü, çizgi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 9. Uygulama Kod -2

* **imageProvider**, StateProvider yapısı ile birlikte ref parametresi kullanılarak geriye bir XFile değişkeni döndürülmüştür. Bu yapı galeriden resim seçme işlemlerinde kullanılmıştır.
* **markaProvider**, StateProvider yapısı ile birlikte dynamic yani api işlemleri gerçekleştiğinde karşıdan gelen verinin farklı türde bir değer ifade edeceğini belirtmek için kullanılmıştır. Yani farklı türden değer alabileceğini ifade etmektedir.

metin, yazı tipi, ekran görüntüsü, grafik içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 10. Uygulama Kod -3

* Ana fonksiyonumuzu belirtmektedir. **runApp** uygulamamızı başlatmayı sağlar ve **ProviderScope** ile child parametresinie **MyApp** class ismi verilerek uygulamamızın başlatılması sağlanılmıştır.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 11. Uygulama Kod -4

* **StatelessWidget**’ı kullanmamızın sebebi durumsuz olmasından kaynaklanmaktadır. Çünkü bu sınıf miras alınarak bir kez oluşturulan ve içindeki veri veya durumu değişmeyen bir sınıf tanımlanmıştır. Kısacası basit bir arayüz gerçekleştirmek için kullanılmıştır. **StatefullWidget** daha dinamik ve kullanıcı etkileşimine daha duyarlıdır.
* **StatelessWidget**’ı, sınıfı kullanıldığı için sadece **build** metodu override edilmiştir.
* **Context** ise şuanki sayfamızı belirtmektedir. MaterialApp widgeti ile ugulamamızın temel yapısı ve teması ayarlanmıştır.
* **debugShowCheckedModeBanner** uygulama sayfasında sağ üstte yer alan debug yazısını ayarlamaktadır.
* **routers** ile de uygulama sayfa aralarında geçişler gerçekleştirilmiştir.
* **InfoPage** anasayfayı belirtmektedir.
* **/select** ise **SelectFileWidget** sayfasına yönlendirmektedir.

metin, ekran görüntüsü, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 12. Uygulama Kod -5

* **ConsumerWidget** widgetinden miras alınarak sayfalar arası **ref** durum değişkenin kullanılmasını sağlamaktadır. Yani durum değişkenine bağlı olarak arayüzün güncellenmesini sağlayacaktır.
* **Scaffold** uygulamamızın iskeletini oluşturmaktadır.
* Kaydırılabilir sayfa tanımlaması gerçekleştirmek için **PageView** widgeti kullanılmıştır.
* **physics** parametresine **NeverScrollableScrollPhysics** özelliği verilerek sayfaların manuel olarak kaydırılması engellenmiştir.
* Sayfalar arası geçişi **ref** değişkeni kullanılarak controller parametresinin içersine **controllerProvider** değişkeni atanarak bulunan sayfa içersinde build metodunun tekrardan güncellenmesi ile farklı sayfalar arsında geçişlerinin gerçekleştirilmesi sağlanılmıştır.
* **PageView** içersinde bulunan **child** parametresi içerisinde bulunan değişkenler ise geçiş yapıldığında geçilecek olan farklı sayfaları belirtmektedir.



Şekil 13. Uygulama Kod -6

* Widget boyutunun dinamik olarak ayarlanabilmesi için **MediaQuery** sınıfında bulunan size ile ekranın genişlik ve yükseklik değerleri çekilmiştir.
* Arka plana bir widget tanımlamak için **Container** yapısını kullandım ve oluşturmuş olduğum tüm widgetları bu widget üzerine atanmıştır.
* Widgetların alt alta durabilmesi için child parametresine **Column** widgetını eklenmiştir.
* Widgetların Column widgetında hizalı durabilmesi için **MainAxisAlignment.spaceAround** değeri kullanılmıştır.
* Belirli yükseklik ve boyutlarda bir widget oluşturabilmek için **SizedBox** widgetı kullanılmıştır. **Child** parametresi olarak **assets** klasörü içerisinde bulunan “**1.png**” resmi eklenmiştir.
* **Text** widgetı kullanılarak ekrana “**Saat Seçmeye Hazır mısın?**” metni yazılmıştır. Daha sonrasında rengi ve kalınlığı ayarlanmıştır.
* **ElevatedButonn** widgetı eklenilmiştir. Butona basıldığında 1 parametresi ile **PageView** widgeti içersinde bulunan sayfaya Animasyon geçiş süresi verilerek ve yavaş bir animasyon eğrisi sağlanılarak 1 indeksli sayfaya geçiş yapılır.
* **RoundedRectangleBorder** özelliği kullanılarak butonun kenarlarının yuvarlatılması sağlanılmıştır.
* **Elevation** parametresi ile düğmenin buton gölgelendirilmesi sağlanılmıştır.
* Daha sonrasında **child** parametresi ile buton üzerine bilgi mesajı yazdırılmıştır.

metin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 14. Uygulama Kod -7

* Yukarıdaki kodlar sayfa 1 ile benzer kodlara sahiptir. Ek olarak uygulama ekranında bulunan resim ve bilgi mesajı değiştirilmiştir.

metin, ekran görüntüsü, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, ekran görüntüsü, yazılım, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, ekran görüntüsü, yazılım, işletim sistemi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 15. Uygulama Kod -8

* Yukarıdaki kodlar sayfa 1 ile benzer kodlara sahiptir. Ek olarak uygulama ekranında bulunan resim ve bilgi mesajı değiştirilmiştir. Sayfa 1 ve 2’den farklı olarak galeriden seçim işlemi gerçekleştirilmiştir. Seçilen resim **file2** değişkenine atanmıştır.
* Galeriden seçilen resmin **null** olup olmama durumu belirlenerek şuanki durum bilgi kontrolü gerçekleştirilmiştir. Şuanki durum bilgisi güncellenerek **SelectFileWidget** sayfasına geçiş işlemi gerçekleştirilmiştir.
* File nesnesi oluşturularak **http.MultipartRequest** ile **'https://api.furkanakyol.tech/api/Data/Image'** yolu belirtilen sunucuya veri gönderim işlemi gerçekleştirilmiştir.
* http’de dosya yüklemeleri için kullanılan ‘**http.MultipartFile**’ nesnesi oluşturulmuştur. Bu nesneye galeriden seçilen dosyanın yolu verilmiştir.
* **‘file’** parametresi ise sunucundan sunucudan hangi parametre adı altında alınacağını belirtmek için kullanılmıştır.
* **await** parametresi kullanılarak asenkron bir şekilde işlemlerin gerçekleştirilmesi sağlanılmıştır.
* İstek sunucuya gönderilir ve sunucudan dönen istek başarılı ise **200** durum kodu verilir.
* Daha sonrasında sunucudan dönen cevap bilgisi uygulama ekranında gösterilir.
* **markaProvider** durum bilgisi ise sunucudan dönen cevap bilgisine atanır. Bu değişken hatırlarsak **dynamic** veri tipindeydi.

metin, ekran görüntüsü, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 16. Uygulama Kod -9

metin, ekran görüntüsü, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, ekran görüntüsü, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

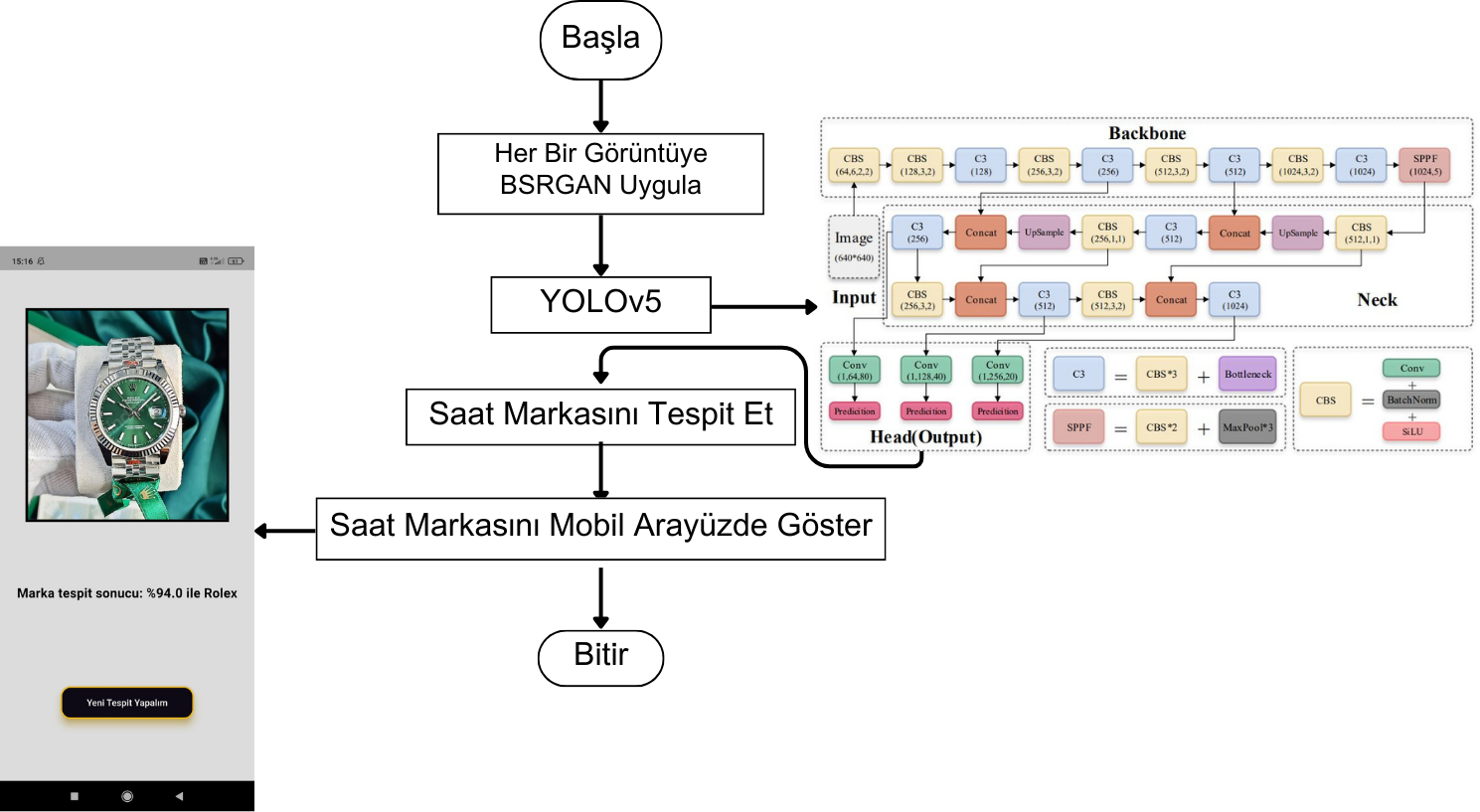
metin, ekran görüntüsü, yazı tipi, tasarım içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 17. Uygulama Kod -10

* Yukarıdaki kodlar galeriden dosya seçme bölümüne aittir.
* İlk olarak bir Container widgetı oluşturularak galeriden seçilen dosyanın yolu durum değişkeninin yoluna verilmiştir ve uygulama ekranında bir çerçeve yapısı içerisinde gösterilmiştir.
* Eğer **markaProvider** durum değişkenin içeriği boş ise **SpinKitWave** animasyon widgetından yararlanılarak bir animasyon efekti ekranda gösterilmiştir.
* Serverdan ilgili görsele ait cevap mesajı geldiğinde marka tespit sonucu ekranda gösterilmiştir.
* Yeniden tespit yapılabilmesi için ElevatedButton widgetının **onPressed** metodu içersine request ve response mesajlarına yer verilmiştir. Bu yöntem, markaProvider durum belirtecinin değerini boş bir dize ("") olarak ayarlayarak, sunucudan gelen eski yanıt bilgisini almamayı sağlamaktadır.

## Akış Şeması



Şekil 18. Akış Şeması

# Projede kullanılacak yöntem, donanımlar ve yazılımlar ile ilgili bilgiler

## Donanımlar

Model eğitiminde kullanılması planlanan donanım özellikleri aşağıdaki gibidir.

**Marka:** Lenovo Legion 5I gen 7

**Ekran kartı:** NVIDIA GeForce RTX 3070 8 GB GDDR6, Boost clock 1620MHz, tgp 140W.

**RAM:** 2X 8 GB SO-DIMM DDR5-4800

**SSD:** 512 GB SSD M.2 2280 PCI-e 4.0x4 NVMe.

**İşlemci:** Intel Core i7-12700H, 14C (6p + 8e) / 20t, p-Core 2.3 / 4.7GHz, e-Core 1.7 / 3.5GHz, 24MB.

## Yazılımlar

**Python:** Python genel amaçlı, yüksek seviyeli bir dildir. Sözdizimi basit ve okunabilir olacak şekilde tasarlanmıştır. Kod blokları girinti tabanlıdır. Python, nesne yönelimli programlama konseptlerini destekler. Sınıflar ve nesneler kullanarak programlarınızı modüller ve sürdürülebilir bir şekilde organize etmenizi sağlar. Geniş bir standart kütüphane koleksiyonuna sahiptir. Bu kütüphaneler, birçok işlevi yerine getirmek için hazır fonksiyonlar ve modüller içerir. Makine öğrenimi, görüntü işleme vb. yapay zeka alanlarında sıklıkla kullanılan programlama dilidir.

**OpenCV:** (Open Source Computer Vision), bilgisayar görüşü ve makine görüşü uygulamaları geliştirmek için kullanılan açık kaynaklı bir bilgisayar görüntü işleme kütüphanesidir. OpenCV'nin temel özellikleri arasında Görüntü İşleme, Nesne Tanıma, Kamera Takibi, Derin Öğrenme ve Kullanım Kolaylığı bulunmaktadır. Projede kullanılması planlanan diğer kütüphaneler ise PyTorch, Pandas, Requests, IPython, Pillow, Psutil, Torchvision, Tqdm, Matplotlib, Seaborn'dur.

**Anaconda:** Ücretsiz ve açık kaynaklı, Python ve R programlama dillerinin bilimsel hesaplama kullanımında paket yönetimini kolaylaştırmayı amaçlayan özgür ve açık kaynaklı dağıtımdır. Paket sürümleri conda paket yönetim sistemi ile yönetilir. Anaconda dağıtımı Windows, Linux ve MacOS işletim sistemlerinde kullanılabilen veri bilimi paketleri içerir. Aynı zamanda JupyterLab, Jupyter Notebook, QtConsole, Spyder, Glue (yazılım), Orange, RStudio, Visual Studio Code, PyCharm gibi editörlerine erişim desteği sunmaktadır [14]

* **PyCharm:**
  + Linux, Windows, macOS işletim sistemlerinde kullanılabilir [15].
  + Full-fledged Professional ve Free Community sürümlerinden oluşmaktadır [15].
  + JetBrains tarafından geliştirilmiştir. PyCharm, hızlı düzeltmelere olanak tanıyan üretkenlik araçlarıyla ve gelişmiş özellikleriyle öne çıkmaktadır.
* **Jupyter Notebook:**
  + Linux, Windows, macOS, bulut tabanlı sistemlerde kullanılabilir [15].
  + Ücretsiz bir platformdur [15].
  + Tarayıcı tabanlıdır [15].
  + Jupyter ismi, veri bilimi için açık kaynaklı diller olan Julia, Python ve R programlama dillerinin kısaltmasından oluşur. Veri bilimi (data science) konusuna odaklanmış bir geliştirme ortamıdır ve bu konuda oldukça kullanışlı araçlar sunar [15].
  + Jupyter Notebook, gerçek zamanlı kodlama ile denklemler, görselleştirmeler ve anlatı metni içeren belgeler oluşturmaya ve paylaşmaya izin verir. Veri temizleme, dönüştürme ve görselleştirme, sayısal simülasyon, istatistiksel modelleme, makine öğrenimi ve daha birçok veri bilimi alanında pratik dahili araçlara sahiptir [15].
  + Jupyter Notebook yalnızca bir IDE değildir, aynı zamanda bir eğitim aracı olarak, sunumlar için ve hatta blog yazmak için yaygın olarak kullanılmaktadır.
  + Sürüm kontrolü, linterler ve otomatik testler en büyük eksikliklerindendir. Gelişmiş IDE'lerdeki kadar ileri düzey otomatik kod tamamlaması yoktur [15].

# Özgünlük

### Veri Setinin Homojen Bir Şekilde Eğitim Veri Seti ve Test Veri Seti Olarak Ayrılması

Eğitilen modelin aşırı öğrenme (overfitting) durumuna sahip olup olmadığını gözlemlemek amacıyla model eğitimi gerçekleştirildikten sonra, model test işlemlerinin gerçekleştirilmesi gerekmektedir. Yapılan literatür taraması sonucunda eğitim veri setinin %80, test veri setinin %20 olarak ayrılması kararlaştırılmıştır. Kaggle ve Google görseller platformlarından temin edilen veriler ile genel bir veri seti oluşturulmuştur. Oluşturulan veri seti farklı kategorilerdeki nesneleri içermektedir. Bununla birlikte, veri setinin sınıf dağılımındaki düzensizlik, eğitim ve test veri kümesini homojen bir şekilde ayırırken makine öğrenmesi yöntemlerinden scikit-learn kütüphanesinde bulunan train\_test\_split() fonksiyonu kullanımını zorunlu hale getirmektedir. Veri kümesinin eğitim ve test seti olarak homojen bir şekilde bölünmesinin temel nedeni, modelin eğitim sırasında bir sınıf değerini daha iyi öğrenmesini engellemektir. Veri kümesini homojen olmayan şekillerde bölmek, modelin yanlı öğrenmesine sebep olmaktadır. Bu durum modelin nesne tespit etme işlemi sırasında genelleme yeteneğinin zayıflamasına yol açmaktadır.

Oluşturulan genel veri setinden, homojen bir test veri seti oluşturabilmek için farklı temalara ait görüntüler yeniden gruplandırılmıştır. Yeniden gruplandırma işlemi, içerdiği nesne sınıflarına göre (Audemars Piguet, Breguet, Breitling, Maurice Lacroix, Omega, Patek Philippe, Rado, Rolex, TISSOT ve Vacheron Constantin) yeniden gruplandırılmıştır. Oluşturulan her bir grupta ayrı ayrı scikit-learn kütüphanesinde bulunan train\_test\_split() fonksiyonu kullanılarak %20 test veri seti elde edilir. Son olarak her bir grup için ayrı ayrı oluşturulmuş olan test veri seti birleştirilerek genel bir test veri seti oluşturulmuştur.

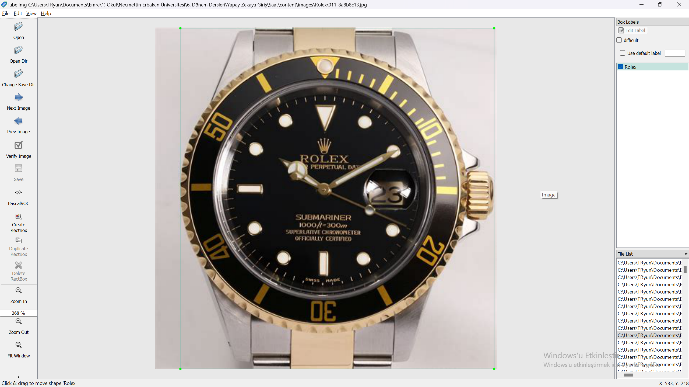
### Veri Setinin Yapay Zekadan Yardım Alınarak Etiketlenmesi

YOLO nesne tespit algoritması ile eğitim gerçekleştirebilmek için, görüntüler ve bu görüntülere ait ".txt" uzantılı dosyalar kullanılmaktadır. Veri seti etiketleme işlemlerinde, etiketleme işlemini hızlandırmak amacıyla 1000 adet görüntü etiketlenerek bir model eğitilmiştir. Eğitilen model kullanılarak, veri setinde henüz etiketlenmemiş görüntüler üzerinde nesne tespit etme işlemi yapılmıştır. Yapılan nesne tespit etme işlemleri sonucunda, tespit edilen nesnelerin bilgileri, YOLO formatına uygun bir biçimde .txt dosyasına aktarılmaktadır. Bu işlemlerin devamında, geliştirilen model aracılığıyla etiketlenen verilerin etiket dosyaları (.txt), “LabelImg” etiketleme programına konularak nesne koordinatları üzerinde düzenlemeler yapılmıştır. Geliştirilen model aracılığıyla etiketlenen veriler, model eğitiminde kullanılmak üzere hazır hale getirilmiştir. Veri seti etiketleme işlemi için "LabelImg" programının tercih edilme nedeni, uygun formatta gelen etiket dosyaları ile veri seti arasında görsel bir ilişki kurabilmesi ve koordinatlar üzerinde dönüşüm yapabilmeye imkan tanımasından kaynaklanmaktadır.

#### LabelImg

Python programlama diliyle geliştirilmiş ve açık kaynaklı olarak sunulan bir yazılım, verileri PASCAL VOC ve YOLO formatlarında dışa aktarabilme özelliğine sahiptir. Bu yazılım, varsayılan olarak sınırlayıcı kutuları yalnızca dörtgen biçiminde sunar. Ancak kullanıcılar, GitHub sayfasında bulunan ilave kod aracılığıyla farklı yöntemler ile etiketleme işlemleri gerçekleştirebilirler [16].

Aşağıda **Şekil 9**’da LabelImg etiketleme programında, eğitim veri seti üzerinde yapılan etiketleme işlemine ait görsellere yer verilmiştir.

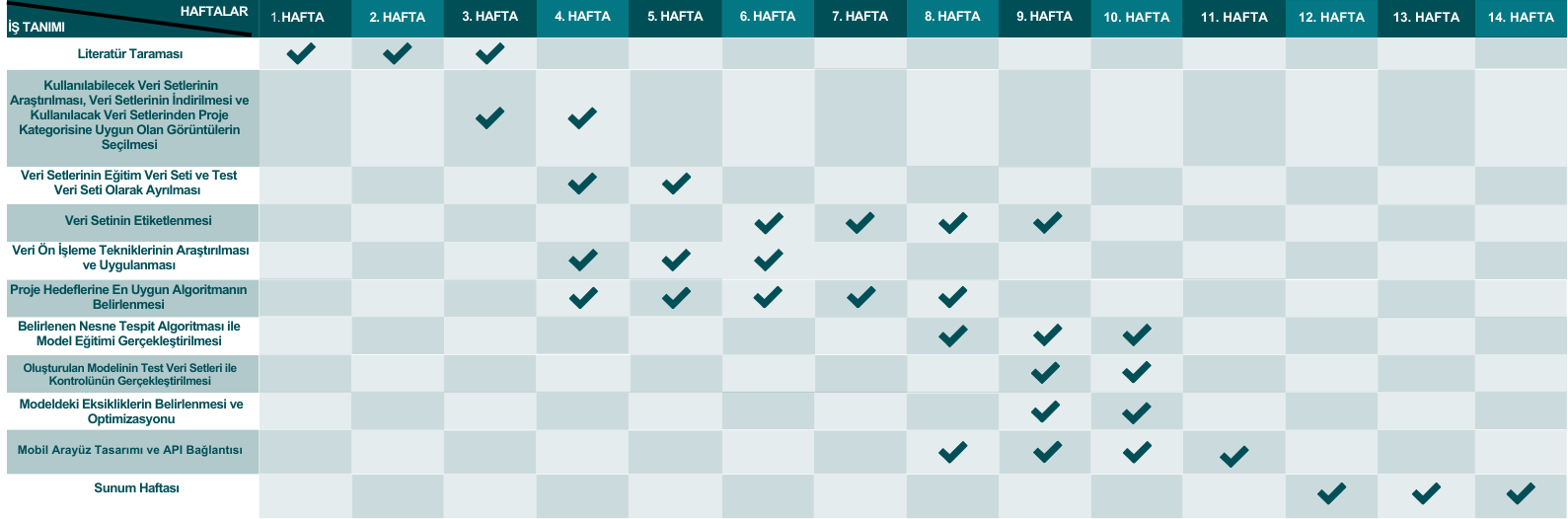
metin, kol saati, saat, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 19. LabelImg Etiketleme Programı

# PROJE TAKVİMİ

Tablo 2. Proje Takvimi



**Literatür Taraması:** Projemizin temelini oluşturmak amacıyla nesne tespitiyle ilgili literatür taraması yapılacak ve konuyla ilgili mevcut bilimsel makaleler, raporlar ve benzeri kaynaklar incelenerek güncel bilgilere erişilecektir. Belirlenen nesne tespit modeline dayalı çalışmalar başlatılacaktır.

**Kullanılabilecek Veri Setlerinin Araştırılması, Veri Setlerinin İndirilmesi ve Kullanılacak Veri Setlerinden Proje Kategorisine Uygun Olan Görüntülerin Seçilmesi:** Bu iş paketi, projemizin temel veri gereksinimlerini karşılamak adına gerçekleştirilecek adımları içermektedir. İlk olarak, kullanılabilecek veri setleri titizlikle araştırılacaktır. Bu araştırma sürecinde, projemizin hedeflerine uygun, güvenilir ve geniş kapsamlı veri setleri belirlenerek indirilecektir. İndirilen veri setleri, proje isterleri göz önünde bulundurularak yeniden düzenlenecektir.

**Veri Setlerinin Eğitim Veri Seti ve Test Veri Seti Olarak Ayrılması:** Modelin eğitim ve test verisi olarak ayrılmasının temel nedeni, modelin gerçek dünya verilerine genelleme yapabilmesini sağlamaktır. Eğitim verisi, modelin öğrenme sürecinde kullanılırken test verisi modelin genel performansını değerlendirmek için ayrılmaktadır. Bu, modelin eğitim verilerine aşırı uyum yapmasını önlemektedir ve gerçek dünya verilerine daha iyi uyarlanmasını sağlamaktadır. Bu durumda, geliştirilen nesne tespit modelinin gerçek performansı ölçülmüş olacaktır.

**Veri Setinin Etiketlenmesi:** Belirlenen nesne tespit algoritmasına uygun etiketleme formatı belirlenecektir. Sonrasında etiketleme programı ile veri setinin etiketleme işlemi yapılacaktır.

**Veri Ön İşleme Tekniklerinin Araştırılması ve Uygulanması:** Veri ön işleme, makine öğrenimi ve derin öğrenme gibi yapay zeka modellerinin başarılı bir şekilde çalışabilmesi için temel adımlardan biridir. Bu adım, ham veri setinin temizlenmesi, düzenlenmesi ve hazırlanmasını içerir. Bu nedenle, bu iş paketinde projenin başarısı için doğru veri ön işleme tekniklerinin araştırılması ve uygulanması sağlanacaktır.

**Proje Hedeflerine En Uygun Algoritmanın Belirlenmesi:** Yapılan literatür taramaları sonucunda elde edilen bilgiler doğrultusunda genel doğruluk oranı, küçük nesnelerde doğruluk oranı, çalışma hızı, eğitim süresi vb. spesifik kriterler belirlenecek ve bu kriterlere dayalı olarak algoritmalar arasında bir karşılaştırma yapılacaktır. Elde edilen sonuçlar, projenin gereksinimleri ve hedefleri doğrultusunda en etkili ve uygun algoritmanın belirlenmesine yardımcı olacaktır.

**Belirlenen Nesne Tespit Algoritması ile Model Eğitimi Gerçekleştirilmesi:** Belirlenen nesne tespit algoritmasına uygun hiper parametre (batch size, epoch, image size, loss value vb.) değerleri belirlenecektir. Belirlenen bu kriterler doğrultusunda modelin eğitim süreci gerçekleştirilecektir. “Proje konusu” kısmında belirtilen ikinci görev için gerekli çalışmalar yapılacaktır.

**Oluşturulan Modelinin Test Veri Setleri ile Kontrolünün Gerçekleştirilmesi:** Geliştirdiğimiz modelin performansını değerlendirmek ve güvenilirliğini sağlamak amacıyla, modelin test verileri üzerinde model başarı kontrolü gerçekleştirilecektir.

**Modeldeki Eksikliklerin Belirlenmesi ve Optimizasyonu:** Geliştirilen modelin test sonuçlarının ilgili değerlendirme kriterleri (doğruluk, hassasiyet, duyarlılık ve F1-skoru) ile değerlendirilmesi sonucunda gerekli görülen optimizasyon işlemleri gerçekleştirilecektir.

**Mobil Arayüz Tasarımı ve API Bağlantısı:** Mobil uygulama geliştirme sürecinde, kullanıcıların etkileşimde bulunacakları arayüzün doğru şekilde tasarlanması ve kullanıcı deneyiminin optimize edilmesi son derece önemlidir. Ayrıca, uygulamanın veri kaynaklarıyla etkileşim halinde olabilmesi için uygun API'ların belirlenmesi ve entegrasyonunun sağlanması gerekmektedir. Bu iş paketinde mobil arayüz tasarımı ve uygun API bağlantısı sağlanacaktık.

# Sonuçlar ve İnceleme

Yapılan literatür taramaları sonucunda YOLOv3-v4-v5 nesne tespit algoritmaları üzerinde karşılaştırmalar yapılmıştır. Geliştirilmesi planlanan model isterleri göz önünde bulundurulduğunda, YOLOv5 algoritmasının diğer 2 nesne tespit algoritmasına kıyasla model isterlerini daha iyi karşıladığı gözlemlenmiştir. Teknolojinin ve yazılımların ilerlemesiyle birlikte YOLO sürümlerinde de gelişmeler yaşanmıştır. Bu bağlamda YOLOv7 ve YOLOv8 sürümleri üzerinde de literatür taramaları yapılmıştır. Ancak proje isterlerini karşılayan herhangi bir literatüre rastlanmamıştır.

Bunun sonucunda, YOLOv5-v7-v8 nesne tespit algoritmaları ile eğitimlerin gerçekleştirilmesi kararlaştırılmıştır. Gerçekleştirilen eğitimler sonucunda, Accuracy, F1-score, Precision ve Recall gibi performans metrikleri göz önünde bulundurularak proje için en uygun algoritma seçilecektir ve seçilen algoritma üzerinde iyileştirmeler yapılacaktır.

Kaggle sitesi kullanılarak başlangıçta 2000 adet, 474x474 boyutlarından oluşan bir veri seti elde edildi ve YOLOv5 nesne tespit algoritmasıyla eğitim aşamasına konuldu [1]. Eğitilen modelin img-size değeri **640**, batch-size değeri **8**, epochs değeri **200**, ve yolo sürümü **yolov5s** olarak belirlenmiştir. Eğitimler Tesla T4 16 GB GPU ile Google Colab platformu üzerinden gerçekleştirildi. Veri setinin %80’i eğitim ve %20’si test çalışmalarında kullanıldı. Modelin eğitim aşaması yaklaşık 100 dakika sürdü. 400 adet görüntüyle modelin test aşaması gerçekleştirildi. Geliştirilen modelin test aşamasında elde edilen sonuçlar aşağıda **Şekil 10**’da gösterilmektedir.

saat, kol saati, Analog saat içeren bir resim

Açıklama otomatik olarak oluşturuldusaat, kol saati, Analog saat, altın içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 20. YOLOv5 Model Eğitimi Test Sonuçları

Eğitimi gerçekleştirilen modelin performansı şu şekildedir: Accuracy: 0.995, F1-score: 0,998, Precision: 0.996, ve Recall: 1

# Kaynakça

[1] <https://www.kaggle.com/datasets/ahedjneed/fancy-watche-images>

[2] K. Zhang, J. Liang, L. Van Gool and R. Timofte, "Designing a Practical Degradation Model for Deep Blind Image Super-Resolution," 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, 2021, pp. 4771-4780, doi: 10.1109/ICCV48922.2021.00475.

[3] Chlap, P., Min, H., Vandenberg, N., Dowling, J., Holloway, L., & Haworth, A. (2021). A review of medical image data augmentation techniques for deep learning applications. Journal of Medical Imaging and Radiation Oncology, 65(5), 545-563.

[4] Tan Li, Lv Xinyue, Lian Xiaofeng and Wang Ge, "YOLOv4\_Drone: UAV image target detection based on an improved YOLOv4 algorithm[J]", Computers and Electrical Engineering, pp. 93, 2021. https://doi.org/10.1016/j.compeleceng.2021.107261

[5] Kaur R. Singh S. A comprehensive review of object detection with deep learning. Digital Signal Processing, 132, 2022, 103812. https://doi.org/10.1016/j.dsp.2022.103812

[6] Ren, S. He, K. Girshick, R. Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39, 2016 ,1137– 1149. https://doi. org/10.1109/TPAMI.2016.2577031

[7] Upesh Nepal and Hossein Eslamiat, Comparing Yolo3, Yolo4, and Yolo5 for Autonomous Landing Spot Detection in Faulty UAVs. Sensors, 22(2),464. https://pubmed.ncbi.nlm.nih.gov/35062425/

[8] Redmon, J.; Farhadi, A. YOLO9000: Better, faster, stronger. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 7263–7271

[9] Redmon, J. Darknet: Open Source Neural Networks in C; 2013–2016. Available online: https://pjreddie.com/darknet/ (accessed on 20 October 2021)

[10] Kıvrak, O. Gürbüz, M. Z. Performance Comparison of YOLOv3, YOLOv4 and YOLOv5 Algorithms: A Case Study for Poultry Recognition. Avrupa Bilim ve Teknoloji Dergisi, 38, 2022, 392-397. https://doi.org/10.31590/ejosat.1111288

[11] Murthy, J.S. Siddesh, G.M. Lai, W.-C. Parameshachari, B.D. Patil, S.N. Hemalatha, K.L. ObjectDetect: A Real-Time Object Detection Framework for Advanced Driver Assistant Systems Using YOLOv5. Wirel.Commun.Mob.Comput. 2022, 9444360. <https://doi.org/10.1155/2022/9444360>

[12] Shu-Jun, Ji. Qing-Hua, Lin. Fei, Han. An improved algorithm for small object detection based on YOLOv4 and multi-scale contextual information. Computers and Electtrical Engineering, 105, 2022, 108490. https://doi.org/10.1016/j.compeleceng.2022.108490

[13]<https://medium.com/@muti.coding/flutter-nedir> 2d031b73c27b#:~:text=Flutter%2C%20Google%20taraf%C4%B1ndan%20geli%C5%9Ftirilen%2C%20Dart,mobil%20uygulamalar%20geli%C5%9Ftirmeye%20olanak%20tan%C4%B1r

[14] <https://tr.wikipedia.org/wiki/Anaconda_(Python_da%C4%9F%C4%B1t%C4%B1m%C4%B1)>

[15] <https://alikasimoglu.com/blog/python-icin-en-iyi-ide-gelistirme-ortami-editor-hangisi/#3>

[16] <https://github.com/HumanSignal/labelImg>