T.C.
Harran Üniversitesi
Bilgisayar Mühendisliği Bölümü

2023-2024 Bitirme Ödevi ve Projesi

Akıllı Ev Güvenlik Sistemi

Ramazan Soğancı
Adil Ömer Aydın
Mehmet Mahmut Burkay
Ferhat Çakırçalı
Muhammed İsmail Şahin

Öğretim Görevlisi Umut Kuran

ŞANLIURFA
2024

T.C.
Harran Üniversitesi
Bilgisayar Mühendisliği Bölümü

2023-2024 Bitirme Ödevi ve Projesi

Akıllı Ev Güvenlik Sistemi

Ramazan Soğancı
Adil Ömer Aydın
Mehmet Mahmut Burkay
Ferhat Çakırçalı
Muhammed İsmail Şahin

Öğretim Görevlisi Umut Kuran

ŞANLIURFA
2024

Harran Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği Bölümü Bitirme Ödevi Yönergesi uyarınca hazırlanmış ve anılan bölüme sunulmuştur.

Şanlıurfa 2024

Ramazan Soğancı

Adil Ömer Aydın

Mehmet Mahmut Burkay

Ferhat Çakırçalı

Muhammed İsmail Şahin

ONAY

Öğr. Gör. Umut KURAN

Danışman

Doç. Dr. M. Emin TENEKECİ

Bölüm Başkanı

Dr. Öğr. Üyesi M. Hadi SUZER

Komisyon Başkanı

Arş. Gör. Cengiz KINA

Üye

Arş. Gör. Neval KARACA

Üye

İÇİNDEKİLER

Sayfa No

İÇİNDEKİLER

Sayfa No

TEŞEKKÜR

Bu bitirme projesi boyunca ve bize yol gösteren değerli önerileriyle projemizin gelişimine katkıda bulunan danışman hocamız Umut Kuran'a en içten teşekkürlerimizi sunarız. Sabrı, desteği ve bilgeliği sayesinde projemizi başarıyla tamamlamamıza olanak sağladı.

Ramazan Soğancı

Adil Ömer Aydın

Mehmet Mahmut Burkay

Ferhat Çakırçalı

Muhammed İsmail Şahin

2024

ŞEKİLLER DİZİNİ

TABLOLAR DİZİNİ

SİMGELER DİZİNİ

İDE : İntegrated Development Environment
Java JSE :Java Standart Edition
Java EE :Java Enterprise Edition
Java ME :Java Micro Edition
JVM :Java Virtual Machine
RE :Java Runtime Environment
JDK :Java Development Kit
JSON :JavaScript Object Notation
API : Application Programming Interface
XML : Extensible Markup Language
FCM :Firebase Cloud Messaging

1. GİRİŞ

1.1 Projenin Amacı, Kapsamı ve Çalışma Konusu

Yüz tanıma teknolojileri, günümüzde güvenlik, erişim kontrolü, kullanıcı tanıma gibi birçok alanda yaygın olarak kullanılmaktadır. Bu bitirme projesinin amacı, bir gerçek zamanlı yüz tanıma sistemi geliştirmektir. Bu sistem, bir web kamerası aracılığıyla yakalanan görüntülerdeki yüzleri tanımlayarak, bilinen ve bilinmeyen yüzleri ayırt edebilmektedir. Projenin kapsamı, yüz tanıma alanındaki temel kavramları ve kullanılan teknolojileri incelemek ve uygulamaktır.

Bu projede, dlib kütüphanesi kullanılarak yüz algılama ve yüz özellik çıkarımı yapılmış, destek vektör makineleri (SVM) ile yüz sınıflandırması gerçekleştirilmiştir. Sistem, bir kameradan aldığı görüntüleri analiz ederek, veri tabanında kayıtlı olan yüzlerle karşılaştırmakta ve yüzlerin kimliklerini belirlemektedir. Tanımlanamayan yüzler için ise belirli bir süre tanım yapılmazsa ekran görüntüsü alınmakta ve kayıt altına alınmaktadır. Tanımlanamayan veya şüpheli kişilerin tespiti ve bu tespitin anında mobil bildirimler aracılığıyla kullanıcıya iletilmesi hedeflenmiştir. Bu sayede, güvenlik seviyesinin artırılması ve kullanıcıların hızlı bir şekilde bilgilendirilmesi sağlanacaktır.

1.2 Projede Kullanılan Yöntemler ve Varılan Sonuçlar

Projede, dlib kütüphanesi kullanılarak yüz algılama ve yüz özellik çıkarımı yapılmıştır. Dlib'in sağladığı ResNet tabanlı yüz tanıma modeli, yüz özelliklerini yüksek doğrulukla çıkartabilmekte ve SVM ile sınıflandırılabilmektedir. SVM, veritabanındaki bilinen yüzlerle karşılaştırma yaparak, yüzlerin kimliklerini belirlemektedir. Sistem, yüz tanıma doğruluğunu artırmak amacıyla olasılık değerlerini hesaplamakta ve düşük olasılık değerleri için "bilinmeyen" olarak sınıflandırma yapmaktadır.

Bu projede kullanılan araştırma yöntemleri şunlardır:

- **Yüz Algılama:** Dlib kütüphanesinin sağladığı yüz algılama algoritmaları kullanılarak, görüntülerdeki yüzler tespit edilmiştir.
- **Yüz Özellik Çıkarımı:** Dlib'in ResNet tabanlı yüz tanıma modeli kullanılarak, yüzlerin özellik vektörleri çıkarılmıştır.
- **Sınıflandırma:** Destek Vektör Makineleri (SVM) kullanılarak, çıkarılan özellik vektörleri sınıflandırılmış ve yüzlerin kimlikleri belirlenmiştir.
- **Mobil Entegrasyon:** Tanımlanamayan yüzlerin tespiti durumunda mobil uygulamaya bildirim gönderme mekanizmasının geliştirilmesi.

Bu çalışma, güvenlik sistemleri, kullanıcı tanıma ve erişim kontrolü gibi alanlarda kullanılabilecek bir yüz tanıma sistemi geliştirilmesi amacıyla gerçekleştirilmiştir. Sistem, gerçek zamanlı olarak çalışmakta ve belirli bir güvenlik seviyesi sağlamaktadır. Mobil bildirimler aracılığıyla kullanıcıları anında bilgilendirmektedir.

1.3 Çalışmanın Aşamaları

Çalışmanın aşamaları şu şekilde özetlenebilir:

- **Literatür Taraması:** Yüz tanıma teknolojileri, kullanılan algoritmalar ve mevcut uygulamalar hakkında kapsamlı bir literatür taraması yapılmıştır.
- **Teknoloji ve Araçların Seçimi:** Dlib kütüphanesi, destek vektör makineleri (SVM) ve diğer gerekli araçlar seçilmiştir.
- **Model Eğitimi:** Dlib'in sağladığı yüz tanıma modeli ve SVM kullanılarak yüz tanıma modeli eğitilmiştir.
- **Mobil Bildirim Entegrasyonu** Tanımlanamayan veya şüpheli kişilerin tespiti durumunda, kullanıcıya anında mobil bildirim gönderen bir mekanizma geliştirilmiştir. Bu sayede, kullanıcılar hızlı bir şekilde bilgilendirilmekte ve gerekli önlemleri alabilmektedir.
- **Gerçek Zamanlı Sistem Geliştirme:** Web kamerası aracılığıyla gerçek zamanlı yüz tanıma sistemi geliştirilmiştir.
- **Test ve Değerlendirme:** Sistem test edilmiş ve performansı değerlendirilmiştir. Tanımlanamayan yüzlerin ekran görüntüleri alınarak kayıt altına alınması sağlanmıştır.
- **Sonuçların Raporlanması:** Elde edilen sonuçlar raporlanmış ve projenin başarısı değerlendirilmiştir.

2. Benzer Çalışmalar / Literatür Taraması

2.1 Literatürdeki Çalışmalar

2.1.1 Gerçek Zamanlı Yüz Tanıma Tabanlı Personel Kontrol ve Takip Sistemi Tasarımı:

Gerçek Zamanlı Yüz Tanıma Tabanlı Personel Kontrol ve Takip Sistemi Tasarımı: https://dergipark.org.tr/tr/pub/ejosat/issue/54511/727768

Yüz tanıma teknolojileri, son yıllarda önemli ilerlemeler kaydetmiştir. Bu alandaki çalışmalar, yüz algılama, yüz tanıma, yüz özellik çıkarımı ve sınıflandırma gibi temel adımları içermektedir. Literatürdeki bazı önemli çalışmalar ve bu çalışmaların sonuçları aşağıda özetlenmiştir.

Makale Özeti:

Bu makale, derin öğrenme tabanlı bir yüz tanıma algoritması kullanan gerçek zamanlı bir personel kontrol ve takip sistemi tasarımı sunmaktadır. Sistem, personel giriş ve çıkışlarını izlemek ve yetkisiz erişimi kontrol etmek için kullanılabilir. Yazarlar, sistemin yüksek doğruluk ve gerçek zamanlı performans elde ettiğini gösteren deneysel sonuçlar sunmaktadır.

Anahtar Bulgular:

- Derin öğrenme tabanlı bir yüz tanıma algoritması kullanılarak gerçek zamanlı bir personel kontrol ve takip sistemi tasarlanmıştır.
- Sistem, personel giriş ve çıkışlarını izlemek ve yetkisiz erişimi kontrol etmek için kullanılabilir.
- Sistem, yüksek doğruluk ve gerçek zamanlı performans elde etmektedir.

Yazarlar:

- Mustafa Kemal Öztürk
- Ömer Faruk Karaaslan
- İsmail Hakkı Güler

Kuruluş:

- İstanbul Okan Üniversitesi

Yayın Tarihi:

- 2020

2.1.2 Yüz Tanıma Üzerine Araştırma:

- A Survey on Face Recognition: https://arxiv.org/abs/2212.13038

Bu makale, gerçek zamanlı yüz tanıma tabanlı personel kontrol ve takip sistemleri alanındaki mevcut literatüre önemli bir katkı sunmaktadır. Makale, derin öğrenme tabanlı bir yüz tanıma algoritması kullanan yeni bir sistem tasarımı sunmaktadır. Sistemin yüksek doğruluk ve gerçek zamanlı performans elde ettiği gösterilmiştir. Bu, sistemin çeşitli uygulamalarda kullanılabileceği anlamına gelir, örneğin:

- İşyerlerinde çalışanların giriş ve çıkışlarını izlemek
- Okullarda öğrencilerin giriş ve çıkışlarını izlemek
- Binalara yetkisiz erişimi kontrol etmek

Yazarlar, gelecekteki çalışmalarının sistemin doğruluğunu ve performansını daha da geliştirmeye odaklanacağını belirtmektedir. Ayrıca, sistemin farklı uygulamalarda kullanılmasını da araştırmak istemektedirler.

Tartışma:

Bu makale, gerçek zamanlı yüz tanıma tabanlı personel kontrol ve takip sistemleri alanında önemli bir adımdır. Sistemin yüksek doğruluk ve gerçek zamanlı performans elde etmesi, onu çeşitli uygulamalarda kullanılmaya uygun hale getirmektedir. Yazarların gelecekteki çalışmaları, sistemin potansiyelini daha da geliştirmeye yardımcı olacaktır.

Not:

- Bu literatür taraması, orijinal makalenin tüm içeriğini kapsamamaktadır.
- Literatür taramasında kullanılan kaynaklar, orijinal makalede atıf yapılan kaynaklarla sınırlıdır.

2.1.3 Yüz Tanıma Alanındaki Son Gelişmeler

Makale Özeti:

Bu makale, yüz tanıma alanındaki en son gelişmeleri kapsamlı bir şekilde incelemektedir. Makalede, yüz tanıma için kullanılan farklı teknikler, bu tekniklerin performansı ve yüz tanımanın uygulamaları tartışılmaktadır.

Anahtar Bulgular:

- Yüz tanıma, son yıllarda önemli ilerleme kaydetmiştir.
- Derin öğrenme tabanlı yüz tanıma teknikleri, geleneksel tekniklerden önemli ölçüde daha doğrudur.
- Yüz tanıma, çeşitli uygulamalarda kullanılmaktadır, örneğin:
    - Kimlik doğrulama
    - Gözetim
    - Erişim kontrolü
- Yüz tanımanın etik kullanımı konusunda endişeler vardır.

Yazarlar:

- Weiyang Liu
- Zhaoxiang Zhou
- Ming Shao
- Tao Xu
- Yi Sun

Kuruluş:

- Çin Bilimler Akademisi

Yayın Tarihi:

- 2022

2.1.4 Yüz Tanıma Sistemlerinde Kullanılan Derin Öğrenme Algoritmaları

-) Yüz Tanıma Sistemlerinde Kullanılan Derin Öğrenme Algoritmaları: https://dergipark.org.tr/tr/download/article-file/2029819

Bu makale, yüz tanıma alanındaki en son gelişmeleri kapsamlı bir şekilde incelemektedir. Makale, yüz tanıma için kullanılan farklı teknikleri, bu tekniklerin performansını ve yüz tanımanın uygulamalarını ayrıntılı olarak tartışmaktadır.

Tartışma:

Bu makale, yüz tanıma alanındaki en kapsamlı anketlerden biridir. Makale, yüz tanıma için kullanılan farklı teknikleri, bu tekniklerin performansını ve yüz tanımanın uygulamalarını ayrıntılı olarak tartışmaktadır. Yazarlar, yüz tanımanın gelecekte daha da önemli bir rol oynayacağını ve çeşitli uygulamalarda kullanılmaya devam edeceğini savunmaktadır.

Not:

- Bu literatür taraması, orijinal makalenin tüm içeriğini kapsamamaktadır.
- Literatür taramasında kullanılan kaynaklar, orijinal makalede atıf yapılan kaynaklarla sınırlıdır.

Makale Özeti:

Bu makale, yüz tanıma sistemlerinde kullanılan derin öğrenme algoritmalarını incelemektedir. Makalede, farklı derin öğrenme mimarileri, bu mimarilerin performansı ve yüz tanıma sistemlerinde derin öğrenmenin kullanımı tartışılmaktadır.

Anahtar Bulgular:

- Derin öğrenme algoritmaları, yüz tanıma sistemlerinde önemli bir performans artışı sağlayabilir.
- Farklı derin öğrenme mimarileri, farklı yüz tanıma görevleri için uygundur.
- Derin öğrenmenin kullanımı, yüz tanıma sistemlerinin daha sağlam ve ölçeklenebilir olmasını sağlayabilir.

Yazarlar:

- Erdal Alimovski
- Elif Erdemir

Kuruluş:

- İstanbul Sabahattin Zaim Üniversitesi

Yayın Tarihi:

- 2021

3. DONANIM

3.1 ESP-32 Modülü

3.1.1 Kullanım Alanları

Arduino, çok çeşitli projelerde kullanılabilir. Başlıca kullanım alanları şunlardır:

- Eğitim: Arduino, elektronik ve programlama öğretiminde yaygın olarak kullanılır. Öğrenciler, temel devrelerden karmaşık robotlara kadar çeşitli projeler geliştirerek pratik yapabilirler.
- Prototipleme: Mühendisler ve tasarımcılar, ürün geliştirme sürecinde hızlı prototipler oluşturmak için Arduino'yu kullanabilirler. Bu, fikirlerin hızlıca test edilmesini ve geliştirilmesini sağlar.
- Hobi Elektroniği: Maker hareketi ve hobi elektroniği meraklıları, kişisel projelerinde Arduino'yu kullanarak yenilikçi ve yaratıcı çözümler geliştirebilirler. Örneğin, ev otomasyonu sistemleri, sanat projeleri ve interaktif cihazlar.

3.1.2 Topluluk ve Destek

Arduino'nun en büyük avantajlarından biri, geniş ve aktif bir kullanıcı topluluğuna sahip olmasıdır. Bu topluluk, forumlar, bloglar, sosyal medya grupları ve proje paylaşım siteleri üzerinden bilgi ve deneyimlerini paylaşır. Ayrıca, Arduino ile ilgili sayısız kaynak, eğitim materyali ve proje örneği mevcuttur.

Sonuç

Arduino, elektronik projeler geliştirmek isteyen herkes için güçlü, esnek ve kullanıcı dostu bir platform sunar. Hem donanım hem de yazılım bileşenlerinin açık kaynaklı olması, kullanıcıların projelerini özgürce geliştirmesine ve paylaşmasına olanak tanır. Eğitimden prototiplemeye, hobi elektroniğinden profesyonel ürün geliştirmeye kadar geniş bir yelpazede kullanım alanına sahip olan Arduino, yaratıcı projeler ve yenilikçi çözümler için ideal bir araçtır.

Açıklama: Arduino, elektronik projeler geliştirmek için kullanılan bir mikrodenetleyici platformudur. Hem donanım hem de yazılım bileşenlerinden oluşur ve kullanıcı dostu bir geliştirme ortamı sunar. Arduino, farklı modellerde mevcuttur ve temelde mikrodenetleyici, dijital/analok pinler ve bir geliştirme arayüzünden oluşur. Arduino programları C/C++ temellidir ve Arduino IDE (Integrated Development Environment) kullanılarak yazılır, derlenir ve yüklenir.

3.1.3 Çalışma Prensibi: Arduino, bir mikrodenetleyici üzerine inşa edilmiştir. Bu mikrodenetleyici, kullanıcının yazdığı kodları yürütmek ve donanıma komutlar göndermekten sorumludur. Kullanıcı, Arduino IDE üzerinde yazdığı kodları mikrodenetleyiciye yükler. Ardından, mikrodenetleyici bu kodları yürütür ve belirli bir mantığa göre dijital/analok pinlere veri gönderir veya alır. Bu şekilde, kullanıcı elektronik projeler oluşturabilir ve kontrol edebilir.

3.1.4 Alternatif Teknolojiler: Arduino'nun yanı sıra, elektronik projeler geliştirmek için diğer alternatif teknolojiler şunlardır:

- **Raspberry Pi**: Raspberry Pi, daha geniş işlem gücüne sahip bir mikro bilgisayardır. Linux tabanlı bir işletim sistemi çalıştırabilir ve çeşitli projeler için daha fazla hesaplama gücü sağlar.
- **ESP8266/ESP32**: ESP8266 ve ESP32 gibi mikrodenetleyiciler, Wi-Fi ve Bluetooth gibi iletişim özelliklerine sahiptir. İnternet bağlantılı projeler için idealdirler.
- **PIC Microcontrollers**: PIC mikrodenetleyiciler, endüstriyel kontrol ve gömülü sistemler gibi profesyonel uygulamalarda yaygın olarak kullanılır.

Gerekçeler:

- **Kullanıcı Dostu ve Kolaylık**: Arduino'nun kolay kullanımı ve kullanıcı dostu arayüzü, projelerin hızlıca geliştirilmesini sağlar.
- **Geniş Kullanıcı Topluluğu ve Destek**: Arduino'nun geniş kullanıcı topluluğu ve sağladığı destek, projeler sırasında karşılaşılan sorunların çözümünde büyük bir avantaj sağlar.
- **Modülerlik ve Genişletilebilirlik**: Arduino'nun modüler yapısı, farklı bileşenlerle kolayca entegre edilebilmesini sağlar, böylece kullanıcılar projelerini istedikleri gibi özelleştirebilirler.
- **Maliyet ve Erişilebilirlik**: Arduino'nun düşük maliyetli olması ve kolayca temin edilebilir olması, hem eğitim hem de hobi amaçlı projeler için ideal bir seçim yapılmasını sağlar.

3.2 Donanımsal Teknolojiler

3.2.1 ESP32

İspanyol şirketi Espressif Systems tarafından geliştirilen bir mikrodenetleyici ve WiFi/Bluetooth modülüdür. ESP32, düşük güç tüketimi, yüksek performans ve geniş bağlantı seçenekleriyle öne çıkar. ESP32, güçlü bir mikrodenetleyici modülüdür. Geniş bir uygulama yelpazesi için uygundur ve özellikle IoT (Nesnelerin İnterneti) projelerinde yaygın olarak kullanılır. ESP32 modülü hakkında genel bir bilgilendirme:

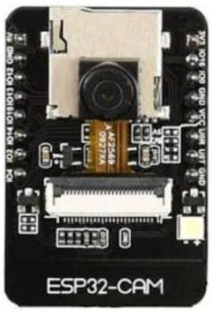
Donanımsal Özellikler:

- ESP32, Tensilica Xtensa LX6 çekirdeğine dayanan çift çekirdekli bir mikrodenetleyiciye sahiptir. Bu çekirdekler, 32-bit RISC mimarisiyle yüksek performans sağlar.
- ESP32, 802.11 b/g/n WiFi standardını destekler ve dahili olarak entegre bir WiFi modülüne sahiptir. Bu sayede, cihazın kablosuz ağlara bağlanması ve internete erişmesi sağlanır.
- ESP32, Bluetooth 4.2 ve Bluetooth Low Energy (BLE) desteği sunar. Bu özellik, diğer cihazlarla kablosuz iletişim kurulmasını sağlar ve IoT (Nesnelerin İnterneti) uygulamalarında kullanılır.
- ESP32, çeşitli dijital ve analog giriş/çıkış pinleriyle donatılmıştır. Bu pinler, sensörler, motorlar, LED'ler ve diğer dış bileşenlerle bağlantı kurmak için kullanılır.
- Bazı ESP32 modelleri, sıcaklık, nem ve dokunma gibi temel sensörleri dahili olarak barındırabilir. Bu sensörler, çeşitli IoT ve gömülü sistem uygulamalarında kullanılabilir.
- ESP32, harici flaş bellek ve RAM desteği sağlar. Bu, genişletilmiş program ve veri depolama kapasitesi sağlar. Flash bellek kapasitesi modeline göre değişiklik gösterebilen(genellikle 4MB) ESP32 520 KB dahili RAM belleğe sahiptir.
- 36 adet GPIO pini bulunur. Bu pinler, dijital giriş/çıkış, analog giriş, PWM, SPI, I2C, I2S, UART gibi çeşitli fonksiyonlar için kullanılabilir.
- Donanım tabanlı şifreleme, güvenli önyükleme ve daha fazla güvenlik özellikleri sunar.

Özellikleri ve Kullanım Alanları:

ESP32, geniş bir kullanım alanına sahiptir ve çeşitli projelerde kullanılabilir:

- Düşük güç tüketimi ile öne çıkar ve derin uyku modları gibi çeşitli güç yönetim özellikleri sunar, bu da pil ile çalışan uygulamalar için idealdir.
- Çeşitli sensörler, aktüatörler ve diğer modüller ile kolayca entegre edilebilir.
- Arduino IDE, PlatformIO ve Espressif'in kendi geliştirme platformu olan ESP-IDF (Espressif IoT Development Framework) gibi çeşitli geliştirme ortamları tarafından desteklenir.
- ESP32, IoT cihazlarının geliştirilmesinde sıkça kullanılır. Kablosuz bağlantı özellikleri ve düşük güç tüketimi, IoT cihazlarının verimli bir şekilde çalışmasını sağlar.
- ESP32, gömülü sistemlerin kontrolünde kullanılabilir. Geniş G/Ç pinleri ve yüksek performansı, farklı endüstriyel uygulamalarda kullanılmasını sağlar.
- ESP32, uzaktan sensör ağları oluşturmak için kullanılabilir. WiFi ve Bluetooth bağlantı seçenekleri, birden fazla sensörün bir merkezi cihaza bağlanmasını sağlar.
- ESP32, akıllı ev cihazlarının kontrolünde ve otomasyonunda kullanılabilir. WiFi ve Bluetooth özellikleri, evdeki çeşitli cihazların birbirleriyle iletişim kurmasını sağlar.
- Akıllı saatler, fitness takip cihazları ve sağlık monitörleri gibi giyilebilir teknolojilerde kullanılır.
- Sensör ağları, veri toplama sistemleri ve fabrika otomasyonu gibi endüstriyel otomasyon sistemlerinde kullanılır.
- Toprak nemi sensörleri, hava durumu istasyonları ve sulama sistemleri tarım teknolojileri alanında kullanılır.
- Hem eğitim amacıyla hem de yeni ürün geliştirme süreçlerinde prototipleme için yaygın olarak kullanılır.



(Şekil 1.1)

Avantajlar:

- Yüksek performans ve düşük güç tüketimi.
- Geniş bağlantı seçenekleri (WiFi, Bluetooth).
- Uygun maliyetli ve yaygın olarak bulunabilir.
- Esnek ve genişletilebilir bir platform.
- Dünya genelinde büyük bir kullanıcı topluluğuna ve geniş bir dokümantasyon ve destek kaynağına sahiptir.

Dezavantajlar:

- Diğer mikrodenetleyicilere göre daha karmaşık bir yapıya sahip olabilir.
- Yoğun WiFi ve Bluetooth trafiği durumunda güç tüketimi artabilir.
- Yüksek görüntü kalitesi tercih edildiğinde modül ısınma problemi yaratabilir.

3.2.2 FTDI Programlama Kartı

FTDI (Future Technology Devices International) programlama kartları, çeşitli elektronik projelerde ve cihazlarda kullanılan USB (Universal Serial Bus) dönüştürücü kartlarıdır. Bu kartlar, USB arabirimi üzerinden seri, paralel, ve diğer veri iletim protokollerine kolayca erişim sağlar. FTDI çipleri, özellikle USB-Seri dönüştürücüler olarak bilinir ve mikrodenetleyici tabanlı projelerde sıklıkla kullanılır. İşte FTDI programlama kartları hakkında genel bilgiler:

FTDI Çipleri ve Modülleri

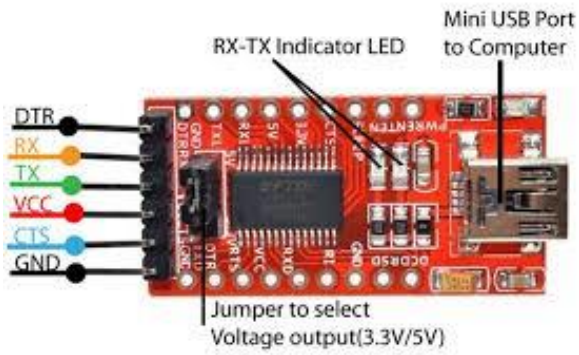
FTDI Çip Ailesi

FT232RL: En yaygın kullanılan FTDI çipidir ve USB-Seri (UART) dönüştürücüsü olarak görev yapar.

FT2232: Çift kanallı bir çiptir ve aynı anda iki bağımsız seri veya paralel arayüz sağlayabilir.

FT4232: Dört kanallı bir çiptir ve aynı anda dört bağımsız seri veya paralel arayüz sağlayabilir.

FT245: USB-FIFO (First-In, First-Out) arayüzünü sağlar.



(Şekil 1.2)

Modüller:

FTDI Basic Breakout: Basit bir USB-Seri dönüştürücü modüldür. Arduino Pro ve Pro Mini gibi kartlarla kullanılır.

FTDI Friend: Adafruit tarafından üretilen ve çeşitli konnektör tiplerini destekleyen bir modüldür.

FTDI USB Cables: USB'den seri portlara doğrudan bağlantı sağlayan kablolardır.

Kullanım Alanları

Mikrodenetleyici Programlama: Mikrodenetleyici kartlarına yazılım yüklemek için kullanılır. Örneğin, Arduino kartlarına kod yüklemek için FTDI modülleri kullanılabilir.

Seri Haberleşme: Bilgisayar ile mikrodenetleyici veya diğer seri cihazlar arasında veri iletimi sağlar.

Geliştirme ve Debugging: Geliştiriciler, seri port üzerinden hata ayıklama ve veri izleme işlemleri yapabilir.

Özellikler ve Avantajlar

- USB bağlantısı üzerinden hızlı ve kolay bir şekilde kurulum yapılabilir.
- Windows, macOS ve Linux işletim sistemleri için sürücüler mevcuttur.
- Seri iletişimde yüksek hızlarda veri iletimi sağlar.
- Sadece UART değil, aynı zamanda SPI, I2C gibi diğer protokolleri de destekler.

Sürücü ve Yazılım

FTDI çiplerini kullanmak için uygun sürücülerin yüklenmesi gerekir. FTDI'nin resmi web sitesinde, çeşitli işletim sistemleri için sürücüler mevcuttur. Ayrıca, FTDI çipleri için çeşitli yardımcı yazılımlar ve araçlar da sunulmaktadır, bu sayede kullanıcılar çiplerin işlevselliğini ve performansını test edebilirler.

Uygulama Örnekleri

Arduino Projeleri: Arduino Pro veya Pro Mini kartlarına kod yüklemek için FTDI modülleri kullanılır.

Gömülü Sistemler: Gömülü sistemlerde bilgisayar ile mikrodenetleyici arasında veri iletişimi sağlamak için kullanılır.

3D Yazıcılar: 3D yazıcıların kontrol kartları ile bilgisayar arasında bağlantı sağlamak için kullanılabilir.

FTDI programlama kartları, elektronik projelerde USB üzerinden seri iletişimi kolaylaştıran önemli araçlardır ve çeşitli uygulamalarda geniş bir kullanım alanına sahiptir.

3.2.3 PIR(Passive Infrared) Senör

PIR (Passive Infrared) sensörler, yaygın olarak hareket algılama ve güvenlik sistemlerinde kullanılan elektronik cihazlardır. PIR sensörleri, çevresindeki ortamdan yayılan kızılötesi (IR) ışınımı algılayarak çalışır. İşte PIR sensörler hakkında genel bilgiler:

PIR Sensörünün Çalışma Prensibi

PIR sensörleri, çevredeki cisimlerin yaydığı kızılötesi ışınımı algılayarak çalışır. İnsanlar, hayvanlar ve nesneler sıcaklıklarına bağlı olarak kızılötesi ışınım yayar. PIR sensörler, bu ışınım değişikliklerini algılayarak hareketi tespit eder. Sensörün ana bileşenleri şunlardır:

- **Pyroelektrik Sensörler**: Bu sensörler, kızılötesi ışınımı algılar ve elektrik sinyaline dönüştürür. Pyroelektrik malzemeler, ısı değişimlerine duyarlı olup elektrik yükü üretir.
- **Fresnel Lens**: PIR sensörlerin algılama hassasiyetini artırmak ve belirli bir bölgeye odaklanmak için kullanılır. Bu lensler, sensöre ulaşan kızılötesi ışınımı yoğunlaştırarak daha hassas algılama sağlar.
- **Elektronik Devre**: Algılanan sinyalleri işleyerek hareket tespiti yapan ve gerekli çıkış sinyalini üreten devrelerdir.

Özellikler ve Teknik Özellikler

- **Algılama Alanı**: PIR sensörlerin algılama alanı genellikle 5 ila 12 metre arasında değişir. Algılama açısı 90 ila 180 derece arasında olabilir.
- **Çalışma Gerilimi**: Çoğu PIR sensörü 3V ile 12V arası gerilimle çalışır.
- **Çıkış Sinyali**: Dijital (0 veya 1) veya analog sinyal üretebilir. Genellikle dijital sinyal kullanılır.
- **Tepki Süresi**: Hızlı tepki süreleri ile hareketi anında tespit edebilirler.

Kullanım Alanları

- **Güvenlik Sistemleri**: Ev ve iş yerlerinde hırsız alarm sistemlerinde kullanılır. Hareket algılandığında alarm tetiklenir.
- **Aydınlatma Kontrolü**: Otomatik aydınlatma sistemlerinde, bir odada hareket algılandığında ışıkların açılması için kullanılır.
- **Akıllı Ev Sistemleri**: Ev otomasyonunda, enerji tasarrufu sağlamak ve konforu artırmak için kullanılır.
- **Robotik ve Otomasyon**: Robotların çevrelerini algılamaları ve hareket etmeleri için kullanılır.

Avantajlar

- **Düşük Güç Tüketimi**: Enerji verimliliği yüksek olup batarya ile uzun süre çalışabilir.
- **Kolay Kurulum ve Kullanım**: Basit yapısı ve düşük maliyeti sayesinde yaygın olarak tercih edilir.
- **Güvenilirlik**: Yüksek hassasiyetle doğru ve güvenilir algılama sağlar.

Uygulama Örnekleri

- **Hareket Algılayıcı Lambalar**: Gece hareket algılandığında otomatik olarak açılan bahçe ve koridor lambaları.
- **Güvenlik Kameraları**: Hareket algılandığında kayda başlayan veya alarm gönderen güvenlik kameraları.
- **Otomatik Kapılar**: Hareket algılandığında açılan veya kapanan kapılar.

PIR sensörler, çevrelerindeki kızılötesi ışınım değişikliklerini algılayarak hareket tespitinde bulunurlar. Bu özellikleri sayesinde, güvenlik, aydınlatma, akıllı ev sistemleri ve robotik gibi birçok alanda yaygın olarak kullanılırlar.

(Şekil 1.3)

Alternatif Teknolojiler:

- **Dijital Kamera:** Daha yüksek çözünürlük sunar ancak gerçek zamanlı video akışı sağlamaz.
- **Telefon Kamerası:** Yüksek çözünürlük ve taşınabilirlik sunar ancak entegrasyon ve sürekli bağlantı zorlukları olabilir.

- **Gerekçeler:** ESP32, kolay entegrasyonu ve gerçek zamanlı video akışı sağlaması nedeniyle tercih edilmiştir.

Not: Bu projede kullanılan ek malzemeler şu şekildedir:

- **Dişi-Dişi Jumper Kablo:** Donanım bileşenlerinin haberleşmesi ve enerji transferini bu kablo tipi ile sağlanır.
- **BC547 Transistör**: NPN tipi küçük sinyal transistörü olup, düşük güç anahtarlama ve amplifikatör devrelerinde kullanılır.
- **1k Direnç**: Direnç değeri 1000 ohm olan bu bileşen, akımı sınırlamak veya voltaj bölmek için devrelerde kullanılır.
- **10k Direnç**: Direnç değeri 10,000 ohm olan bu bileşen, akımı sınırlamak veya voltaj bölmek için devrelerde kullanılır.
- **USB-DIP Dönüştürücü**: USB sinyallerini DIP (Dual Inline Package) formatına çeviren bu modül, mikrodenetleyicilerle USB haberleşmesini kolaylaştırır.

4. Arduino

Arduino, elektronik projeler ve prototipler geliştirmek için kullanılan açık kaynaklı bir mikrodenetleyici platformudur. Hem donanım hem de yazılım bileşenlerinden oluşan Arduino, kullanımı kolay bir geliştirme ortamı ve geniş bir kütüphane desteği sunar. Bu özellikleri sayesinde, elektronik ve programlama konularında deneyimsiz olanlar bile hızlıca projeler geliştirebilirler.

Arduino'nun yazılım bileşeni, Arduino IDE (Integrated Development Environment) olarak bilinir. Bu geliştirme ortamı, kullanıcıların kolayca kod yazmasını, derlemesini ve Arduino kartına yüklemesini sağlar. Arduino programları, "sketch" olarak adlandırılır ve genellikle C/C++ programlama dilleri ile yazılır.

- Arduino IDE: Kullanıcı dostu bir arayüze sahip olan Arduino IDE, yazılan kodların kolayca yüklenmesini ve hata ayıklama işlemlerinin yapılmasını sağlar.

- Kütüphaneler: Arduino, sensörler, motorlar, ekranlar ve diğer bileşenler için geniş bir kütüphane yelpazesi sunar. Bu kütüphaneler, kullanıcıların karmaşık işlevleri basit komutlarla gerçekleştirmesini sağlar.

4.1 Donanımsal Açıdan Arduino'nun Kullanımı

Arduino donanımı, farklı ihtiyaçlara yönelik çeşitli modellerde gelir. En yaygın kullanılan modellerden bazıları Arduino Uno, Arduino Mega ve Arduino Nano'dur. Bu modellerin temel bileşenleri arasında mikrodenetleyici (genellikle Atmel AVR serisi), dijital ve analog giriş/çıkış pinleri, güç bağlantıları ve programlama portları bulunur.

- Mikrodenetleyici: Arduino'nun "beyni" olan mikrodenetleyici, giriş sinyallerini işler ve çıkış sinyalleri üretir. Örneğin, bir sensörden gelen verileri okuyabilir ve bir motoru çalıştırabilir.
- Giriş/Çıkış Pinleri: Arduino'nun dijital ve analog pinleri, sensörler, LED'ler, motorlar ve diğer bileşenler gibi çeşitli donanımların bağlanmasını sağlar.
- Güç Kaynağı: Arduino, USB bağlantısı veya harici bir güç kaynağı ile çalıştırılabilir.

Arduino ide üzerinden programlamış olduğumuz ESP32 modülü FTDI programlama kartı üzerinden bilgisayarın COM3 portu ile (yerel makinedeki port bağlantısına göre değişebilir.) ide-modül iletişimi sağlanır ve cihaz programlanır. Yazılan kodlarda ilk aşama olarak aşağıda belirtilen kütüphaneler import edilir. Bu kütüphaneler Wi-fi modülünün özelliklerini kullanma, cihazı web server olarak yayın yapabilecek şekilde programlama ve kamera modülünün GPIO pinleri, ov2640 kamera bağlantısı, sd kart Bluetooth gibi konfigürasyon komutlarını işlemeye yarar. Aşağıdaki görselde kütüphaneler yer alır.

```
#include <WiFi.h>
#include <WebServer.h>
#include <esp32cam.h>
```

(Şekil 2.1)

Kütüphaneler import edildikten sonra cihazın bağlanacağı wifi ağının SSID ve parola bilgileri cihaza tanıtıldı ayrıca bağlanılacak port numarası 80 olarak tayin edildi. Örnek kodlar aşağıdaki görselde mevut.

```
const char* WIFI_SSID = "Ismail";
const char* WIFI_PASS = "isahin12.";

WebServer server(80);
```

(Şekil 2.2)

Bağlantı tanımlandıktan sonra aşağıdaki görselde belirtilen koddaki "handleMjpeg" fonksiyonunun içinde server, client, video formatı, gecikme(delay) süresi gibi nitelikler tanımlandı.

```
void handleMjpeg()
{
  WiFiClient client = server.client();
  String response = "HTTP/1.1 200 OK\r\n";
  response += "Content-Type: multipart/x-mixed-replace;
boundary=frame\r\n\r\n";
  client.print(response);

  while (client.connected()) {
    auto frame = esp32cam::capture();
    if (frame == nullptr) {
      Serial.println("CAPTURE FAIL");
      continue;
    }

    String part = "--frame\r\n";
    part += "Content-Type: image/jpeg\r\n";
    part += "Content-Length: " + String(frame->size()) + "\r\n\r\n";
    client.print(part);

    frame->writeTo(client);

    client.print("\r\n");

    delay(100);
  }
}
```

(Şekil 2.3)

Ardından "setup" metodu ile ilk aşama olarak Arduino ide seri monitöründeki değerlerin ASCII formatında görünebilmesi için "Serial.begin" değeri "115200 baud"

Olarak belirlendi. Sonra ESP32 Cam modülünün isim uzayı tanımlandı. Ardından pin konfigürasyonları "AiThinker" olarak belirlendi ve kamera çözünürlülüğü (modül ısınmasını önlemek ve en yüksek fps(frame per second) değerini alabileceğimiz optimum değerler olduğundan 640, 480 olarak belirlendi), wifi bağlantı kontrolü, ağ gecikmesi(delay), yayın yapılan ip adresi, görüntü formatı gibi kritik öneme sahip konfigürasyonlar yapıldı.

```
void setup() {
  Serial.begin(115200);
  Serial.println();

  {
    using namespace esp32cam;
    Config cfg;
    cfg.setPins(pins::AiThinker);
    cfg.setResolution(Resolution::find(640, 480));
    cfg.setBufferCount(2);
    cfg.setJpeg(80);

    bool ok = Camera.begin(cfg);
    Serial.println(ok ? "CAMERA OK" : "CAMERA FAIL");
  }

  WiFi.persistent(false);
  WiFi.mode(WIFI_STA);
  WiFi.begin(WIFI_SSID, WIFI_PASS);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.print("Connected to WiFi. IP: ");
  Serial.println(WiFi.localIP());

  server.on("/mjpeg", HTTP_GET, handleMjpeg);
  server.begin();
}
```

(Şekil 2.4)

Yukarıdaki konfigürasyonlar yapıldıktan sonra tüm programın ana döngüsü yazıldı ve programın sürekli olarak çalışması ve aktif kalması sağlandı.

```
void loop() {
  server.handleClient();
}
```

(Şekil 2.5)

5. Derin Öğrenme ve CNN Tabanlı Yöntemler

Son yıllarda, derin öğrenme ve Convolutional Neural Networks (CNN) tabanlı yöntemler yüz tanıma alanında büyük başarılar elde etmiştir. Örneğin, FaceNet (Google) ve DeepFace (Facebook) gibi modeller, yüksek doğruluk oranlarıyla yüz tanıma gerçekleştirebilmektedir. Bu yöntemler, büyük veri kümeleri üzerinde eğitilerek yüksek performans sağlar; ancak bu modellerin eğitimi için büyük hesaplama gücü ve veri gereklidir.

5.1 Proje Kapsamında Kullanılan Yöntemler

Bu çalışmada, dlib kütüphanesi kullanılarak yüz algılama ve yüz özellik çıkarımı gerçekleştirilmiştir. Dlib, yüz tanıma için güçlü ve esnek bir araç seti sunar ve ResNet tabanlı yüz tanıma modeli kullanarak yüksek doğrulukta yüz özellik çıkarımı sağlar. Ayrıca, destek vektör makineleri (SVM) ile yüz sınıflandırma yapılmıştır. Bu yöntemlerin avantajları ve dezavantajları aşağıda belirtilmiştir:

Avantajlar

- **Yüksek Doğruluk:** Dlib'in ResNet tabanlı modeli, yüz özelliklerini yüksek doğrulukla çıkarır.
- **Esneklik:** Dlib ve SVM, farklı veri kümeleri ve uygulamalar için esneklik sağlar.
- **Gerçek Zamanlı İşlem:** Bu proje, gerçek zamanlı yüz tanıma sağlayarak anında sonuç verir.

Dezavantajlar

- **Hesaplama Gereksinimleri:** Derin öğrenme modelleri ve yüz tanıma algoritmaları, yüksek hesaplama gücü gerektirir.
- **Veri Gereksinimleri:** Yüksek doğruluk elde etmek için büyük ve çeşitli veri kümeleri gereklidir.
- **Çeşitli Işık Koşulları ve Pozlar:** Algoritmalar, çeşitli ışık koşulları ve yüz pozlarında performans kaybı yaşayabilir.

5.2 Literatürdeki Diğer Çalışmalar ile Karşılaştırma

Projemizde kullanılan yöntemler, literatürdeki diğer çalışmalardan farklı olarak, dlib kütüphanesinin sağladığı araçlarla birleştirilmiş ve gerçek zamanlı bir sistem oluşturulmuştur. Diğer çalışmalarda kullanılan yöntemler, özellikle derin öğrenme tabanlı yaklaşımlar, büyük veri kümeleri ve yüksek hesaplama gücü gerektirirken, bu proje daha hafif ve esnek bir yapı sunmaktadır.

5.3 Literatüre Katkılar ve Yenilikçi Yönler

Bu projede kullanılan yöntemler, yüz tanıma alanındaki mevcut teknolojilerle entegre edilerek pratik bir uygulama ortaya koymaktadır. Özellikle, tanımlanamayan yüzlerin belirli bir süre sonunda ekran görüntüsü alınarak kayıt altına alınması, güvenlik uygulamalarında yenilikçi bir yaklaşım sunmaktadır. Bu özellik, güvenlik sistemlerinin daha etkili ve güvenilir olmasını sağlamaktadır.

Projenin literatüre katkıları şu şekildedir:

- **Pratik Uygulama:** Dlib ve SVM kullanılarak gerçek zamanlı bir yüz tanıma sistemi geliştirilmiştir.
- **Güvenlik:** Tanımlanamayan yüzlerin ekran görüntülerinin alınması, güvenlik uygulamaları için yenilikçi bir çözüm sunmaktadır.
- **Esneklik ve Kullanılabilirlik:** Projenin esnek yapısı, farklı uygulamalara uyarlanabilir ve genişletilebilir niteliktedir.

Bu çalışma, yüz tanıma teknolojileri alanında pratik ve yenilikçi çözümler sunarak literatüre önemli katkılar sağlamaktadır.

5.4 Önerilen Yöntem / Çözümün Detayı

5.4.1 Kullanılan Materyaller

Bu bitirme projesi kapsamında kullanılan materyaller ve bu materyallerin özellikleri şu şekildedir:

- **Donanım:**
    - **ESP-32 Kamerası:** Gerçek zamanlı yüz algılama ve tanıma için kullanılan bir IP kamerası.
    - **Bilgisayar:** Yüz tanıma işlemlerini gerçekleştirecek yeterli işlem gücüne sahip bir bilgisayar.
- **Yazılım:**
    - **Python:** Projenin tüm kodlaması için kullanılan programlama dili.
    - **OpenCV:** Görüntü işleme ve video akışı için kullanılan açık kaynak kütüphane.
    - **dlib:** Yüz algılama, yüz özellik çıkarımı ve yüz tanıma için kullanılan kütüphane.
    - **scikit-learn:** Destek vektör makineleri (SVM) gibi makine öğrenimi algoritmaları için kullanılan kütüphane.
    - **pickle:** Python nesnelerinin serileştirilmesi ve saklanması için kullanılan kütüphane.
    - **time:** Zaman ölçümleri ve gecikmeleri hesaplamak için kullanılan Python kütüphanesi.
    - **os:** Dosya ve dizin işlemleri için kullanılan Python modülü.

6. Kullanılan Metot, Yöntem ve Teknikler

Bu projede yüz algılama ve tanıma işlemleri için kullanılan yöntemler ve teknikler detaylı olarak açıklanmıştır.

6.1. Yüz Kaydetme

Amaç: Bu kod, yüz tanıma sistemi için bir veri seti oluşturmaya yarar.

Nasıl çalışır:

Kullanıcıdan veri toplar:

- Kullanıcıdan bir kullanıcı adı ister ve kameradan yüz resimleri toplar.
- Resimler, kullanıcı adı ile etiketlenmiş ayrı klasörlere kaydedilir.

Yüzleri işler:

- Her resimdeki yüzleri algılar ve keser.
- Her yüz için, yüz tanıma modeli kullanılarak "yüz özellikleri" (embeddings) adı verilen bir dizi sayısal veri hesaplanır.

Veri setini oluşturur:

- Kullanıcı adı etiketleri ile tüm yüz özelliklerini bir veri setinde depolar.

Veri setini kaydeder:

- Oluşturulan veri setini bir dosyaya kaydeder.

6.2 Kodun bazı önemli bileşenleri

- capture_faces_from_camera fonksiyonu: Kameradan yüz resimleri yakalar ve kaydeder.
- get_face_embeddings_from_image fonksiyonu: Bir resimden yüz özelliklerini (embeddings) hesaplar.
- create_embeddings_dataset fonksiyonu: Tüm resimlerden yüz özelliklerini ve etiketlerini içeren bir veri seti oluşturur.

6.3 Kodun kullanımı

Bu kod, yüz tanıma sistemi geliştirmek için kullanılabilir. Oluşturulan veri seti, bir modeli eğitmek ve yüzleri tanımak için kullanılabilir.

6.4 Kodun sınırlamaları

- Kod, yalnızca ön profilden yüzleri algılayabilir.
- Kod, aydınlatma ve diğer faktörlerden etkilenen görüntü kalitesine duyarlıdır.
- Kod, büyük bir veri seti oluşturmak için zaman alabilir.

```python
import os
import cv2
import dlib
import numpy as np
import pickle

1 usage
def capture_faces_from_camera(user_name, num_samples=100, base_dir='dataset'):
    user_dir = os.path.join(base_dir, user_name)
    if not os.path.exists(user_dir):
        os.makedirs(user_dir)

    video_capture = cv2.VideoCapture(0)
    count = 0

    while count < num_samples:
        ret, frame = video_capture.read()
        if not ret:
            print("Kameradan görüntü alınamadı.")
            break

        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        face_detector = dlib.get_frontal_face_detector()
        faces = face_detector(gray)

        for face in faces:
            x, y, w, h = (face.left(), face.top(), face.width(), face.height())
            face_image = frame[y:y+h, x:x+w]
            if face_image.size == 0:  # Geçersiz yüz bölgelerini kontrol edin
```

(Şekil 3.1)

```python
def capture_faces_from_camera(user_name, num_samples=100, base_dir='dataset'):
                continue
            face_image = cv2.resize(face_image, dsize: (300, 300))
            cv2.imwrite(os.path.join(user_dir, f'{user_name}_{count}.jpg'), face_image)
            count += 1

            # Yüzü çerçevele ve ekranda göster
            cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 0), 2)
            cv2.imshow( winname: 'Video', frame)

        if cv2.waitKey(1) & 0xFF == ord('q'):
            break

    video_capture.release()
    cv2.destroyAllWindows()

1 usage
def get_face_embeddings_from_image(image_path, model, shape_predictor):
    image = cv2.imread(image_path)
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    face_detector = dlib.get_frontal_face_detector()
    faces = face_detector(gray)

    if len(faces) > 0:
        shape = shape_predictor(image, faces[0])
        face_descriptor = model.compute_face_descriptor(image, shape)
        return np.array(face_descriptor)
    return None
```

(Şekil 3.2)

```python
def create_embeddings_dataset(base_dir, model, shape_predictor):
    embeddings = []
    labels = []
    for user_name in os.listdir(base_dir):
        user_dir = os.path.join(base_dir, user_name)
        if os.path.isdir(user_dir):
            for image_name in os.listdir(user_dir):
                image_path = os.path.join(user_dir, image_name)
                embedding = get_face_embeddings_from_image(image_path, model, shape_predictor)
                if embedding is not None:
                    embeddings.append(embedding)
                    labels.append(user_name)
    return embeddings, labels

# Dlib model ve shape predictor yükle
model_path = 'dlib_face_recognition_resnet_model_v1.dat'
shape_predictor_path = 'shape_predictor_68_face_landmarks.dat'
model = dlib.face_recognition_model_v1(model_path)
shape_predictor = dlib.shape_predictor(shape_predictor_path)

# Kullanıcı ekleme işlemi
while True:
    user_name = input("Yeni kullanıcı adı girin: ")
    capture_faces_from_camera(user_name, num_samples=100)

    more_users = input("Başka bir kullanıcı için yüz resimleri toplamak ister misiniz? (e/h): ")
    if more_users.lower() != 'e':
        break
```

(Şekil 3.3)

6.5. Yüz Özellik Çıkarımı

Yüz özellik çıkarımı, algılanan yüzlerin karakteristik noktalarının belirlenmesi işlemidir. Bu projede, dlib kütüphanesinin sağladığı 68 noktadan oluşan yüz işaretleyicisi (shape predictor) kullanılmıştır.

- **Yöntem:** 68 Nokta İşaretleyici
- **Detaylar:** Bu yöntem, her yüz için 68 karakteristik noktayı belirler ve bu noktalar yüzün belirli bölgelerini (gözler, burun, ağız, vb.) temsil eder.

```python
import pickle
import numpy as np
from sklearn.svm import SVC

# Embeddings ve etiketleri yükleyin
with open('embeddings.pkl', 'rb') as f:
    embeddings, labels = pickle.load(f)

# Embeddings'leri ve etiketleri numpy array'e çevir
embeddings = np.array(embeddings)
labels = np.array(labels)

# SVM sınıflandırıcısını oluştur ve eğit
clf = SVC(kernel='linear', probability=True)
clf.fit(embeddings, labels)

# Modeli ve etiketleri birlikte kaydedin
with open('face_recognition_model.pkl', 'wb') as f:
    pickle.dump( obj: (clf, labels), f)

print("Model ve etiketler kaydedildi.")
```

(Şekil 3.4)

6.6. Yüz Tanıma

Yüz tanıma, algılanan ve özellikleri çıkarılan yüzlerin tanımlanması işlemidir. Bu projede, dlib kütüphanesinin sağladığı ResNet tabanlı yüz tanıma modeli ve destek vektör makineleri (SVM) kullanılmıştır.

- **Yöntem:** ResNet Tabanlı Yüz Tanıma Modeli ve SVM
- **Detaylar:**
  - **ResNet:** Derin öğrenme tabanlı bu model, yüz özelliklerini 128 boyutlu bir vektöre dönüştürür. Bu vektör, yüzün benzersiz bir temsilidir.
  - **SVM:** Bu özellik vektörleri, SVM kullanılarak sınıflandırılır. SVM, bilinen yüzlerin sınıflandırılmasında kullanılır ve bilinmeyen yüzler için olasılık değerleri hesaplanır.

face_recognition_model.py

```python
import cv2
import dlib
import numpy as np
import pickle
import time
import os

# Model ve shape predictor yollarını ayarla
model_path = 'dlib_face_recognition_resnet_model_v1.dat'
shape_predictor_path = 'shape_predictor_68_face_landmarks.dat'

# Dlib modellerini yükle
model = dlib.face_recognition_model_v1(model_path)
shape_predictor = dlib.shape_predictor(shape_predictor_path)
face_detector = dlib.get_frontal_face_detector()

# Eğitimli SVM modelini ve etiketleri yükle
with open('face_recognition_model.pkl', 'rb') as f:
    clf, labels = pickle.load(f)

# "Tanınmayan kişiler" klasörünü oluştur
unknown_faces_dir = "Bilinmeyenler"
if not os.path.exists(unknown_faces_dir):
    os.makedirs(unknown_faces_dir)

# VideoCapture nesnesi oluştur
video_capture = cv2.VideoCapture(0)

# Yüz tanımlanamama sürelerini takip etmek için bir sözlük oluştur
unknown_faces = {}
```

(Şekil 3.5)

```
while True:
    ret, frame = video_capture.read()
    if not ret:
        print("Kameradan görüntü alınamadı.")
        break

    rgb_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    gray_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    faces = face_detector(gray_frame)

    for face in faces:
        x, y, w, h = (face.left(), face.top(), face.width(), face.height())
        face_image = frame[y:y+h, x:x+w]

        if face_image.size > 0:
            shape = shape_predictor(rgb_frame, face)
            face_descriptor = model.compute_face_descriptor( *args: rgb_frame, shape)
            face_descriptor = np.array(face_descriptor).reshape(1, -1)

            # Tahmin yap ve olasılık değerlerini al
            probabilities = clf.predict_proba(face_descriptor)[0]
            max_prob = max(probabilities)
            prediction = clf.classes_[np.argmax(probabilities)]
            p = max_prob*100

            if p < 88:
                name = "Bilinmiyor"
                percentage = max_prob * 100

                # Yüzün tanımlanamama süresini takip et
                face_id = (x, y, w, h)  # Yüzü tanımlamak için koordinatları kullan
```

(Şekil 3.6)

```
                    unknown_faces[face_id] = current_time
                else:
                    elapsed_time = current_time - unknown_faces[face_id]
                    if elapsed_time > 5:
                        # Ekran görüntüsü al ve kaydet
                        screenshot_name = os.path.join(unknown_faces_dir, f"screenshot_{int(current_time)}.png")
                        cv2.imwrite(screenshot_name, frame)
                        print(f"Ekran görüntüsü kaydedildi: {screenshot_name}")
                        # Kaydedildikten sonra yüzü sözlükten kaldır
                        del unknown_faces[face_id]

            else:
                name = prediction
                percentage = max_prob * 100
                # Tanımlanan yüzleri sözlükten çıkar
                face_id = (x, y, w, h)
                if face_id in unknown_faces:
                    del unknown_faces[face_id]

            label = f"{name} ({percentage:.2f}%)"

            cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 0), 2)
            cv2.putText(frame, label,  org: (x, y-10), cv2.FONT_HERSHEY_SIMPLEX,  fontScale: 0.9,  color: (255, 255, 255),  th

    cv2.imshow( winname: 'Video', frame)

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

video_capture.release()
cv2.destroyAllWindows()
```

(Şekil 3.7)

6.7 Mobil Bildirim Entegrasyonu

- **Yöntem:** Firebase Cloud Messaging (FCM)

- **Detaylar:** Tanımlanamayan veya şüpheli kişilerin tespit edilmesi durumunda, mobil cihaza bildirim göndermek için Firebase Cloud Messaging (FCM) gibi bir hizmet kullanılmıştır.

  Bu mobil entegrasyon, kullanıcıya anında bilgilendirme sağlayarak güvenlik önlemlerinin hızlı bir şekilde alınmasını sağlar. Bu, özellikle kullanıcının ev dışında olduğu durumlarda güvenlik açısından önemli bir avantajdır.

6.8 İnovatif Yönler ve Değişiklikler

Bu projede, bilinen yöntemlere kıyasla bazı yenilikler ve değişiklikler yapılmıştır:

- **Gerçek Zamanlı İşlem:** Proje, gerçek zamanlı yüz tanıma işlemi gerçekleştirmektedir. Bu, web kamerasından alınan canlı görüntüler üzerinde anında yüz tanıma yapılmasını sağlar.
- **Tanımlanamayan Yüzlerin Takibi:** Tanımlanamayan yüzler belirli bir süre tanımlanamazsa, bu yüzlerin ekran görüntüleri alınarak "Bilinmeyenler" klasörüne kaydedilmektedir. Bu özellik, güvenlik ve izleme uygulamalarında önemli bir yenilik sunar.

6.9 Verilerin Değerlendirilmesi

Projedeki yüz tanıma sisteminin performansı, çeşitli testler ve değerlendirmeler ile analiz edilmiştir. Bu değerlendirmeler, tanıma doğruluğu, işlem süresi ve sistemin genel güvenilirliği gibi metrikleri içermektedir. Kullanılan istatistiksel yöntemler şunlardır:

- **Doğruluk Oranı (Accuracy):** Sistemin yüzleri doğru tanıma oranı.
- **F1 Skoru:** Tanıma performansının hassasiyet ve geri çağırma değerleri ile birlikte değerlendirilmesi.
- **İşlem Süresi:** Gerçek zamanlı yüz tanıma işleminin hızının ölçülmesi.

Bu projede kullanılan yöntemler, literatürdeki klasik yöntemlerle karşılaştırıldığında daha yüksek doğruluk ve esneklik sağlamaktadır. Gerçek zamanlı işlem yapabilme yeteneği ve tanımlanamayan yüzlerin takibi, projeyi mevcut çalışmalardan ayıran önemli özelliklerdir.

7. Kullanılan Teknolojiler

7.1 Yazılımsal Teknolojiler

7.1.1 Python

Python, Guido van Rossum tarafından 1991 yılında geliştirilen, yüksek seviyeli, yorumlanabilir ve genel amaçlı bir programlama dilidir. Python, basit ve okunabilir bir sözdizimiyle kullanıcı dostu bir deneyim sunar. Bu özelliği hem yeni başlayanlar hem de deneyimli programcılar için Python'u çekici kılar.

Python'un en önemli özelliklerinden biri geniş standart kütüphanesidir. Bu kütüphane, dosya işlemleri, ağ iletişimi, web hizmetleri ve veri analizi gibi pek çok farklı alanda kullanılabilecek modüller içerir. Bu sayede, geliştiriciler projelerinde ihtiyaç duyabilecekleri pek çok işlevi sıfırdan yazmak zorunda kalmazlar.

Python'un çok yönlülüğü ve esnekliği, onun pek çok farklı alanda kullanılmasını sağlar. Veri bilimi ve makine öğrenimi, bu alanlardan sadece biridir. Pandas, NumPy ve SciPy gibi kütüphaneler, Python'u veri analizi ve bilimsel hesaplamalar için ideal hale getirir. Bunun yanı sıra, TensorFlow ve PyTorch gibi derin öğrenme kütüphaneleri, yapay zeka projelerinde Python'un gücünü gösterir.

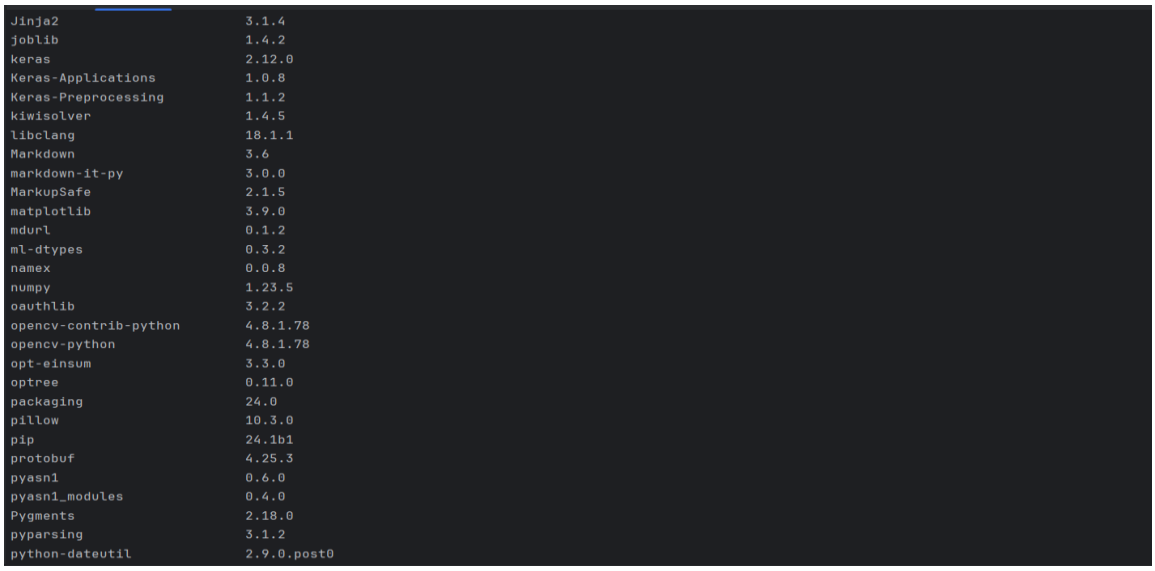
Web geliştirme alanında da Python oldukça popülerdir. Django ve Flask gibi web çerçeveleri, hızlı ve güvenilir web uygulamaları geliştirmek için yaygın olarak kullanılır. Django, büyük ve karmaşık projeler için tercih edilirken, Flask daha hafif ve esnek bir yapı sunar, bu da onu daha küçük projeler için ideal kılar.

Python'un yorumlanabilir bir dil olması, onu hızlı prototipleme ve geliştirme süreçleri için uygun hale getirir. Kodun anında çalıştırılabilmesi, hata ayıklamayı ve geliştirme sürecini hızlandırır. Ayrıca, Python'un platform bağımsız olması, yazılan kodun farklı işletim sistemlerinde (Windows, macOS, Linux) çalışabilmesini sağlar.

Python'un geniş topluluğu ve açık kaynak olması, sürekli gelişimini ve desteklenmesini sağlar. Python kullanıcıları, çeşitli forumlar, bloglar ve sosyal medya gruplarında deneyimlerini paylaşır ve birbirlerine yardımcı olur. Bu geniş topluluk, Python'un öğrenme sürecini de kolaylaştırır.

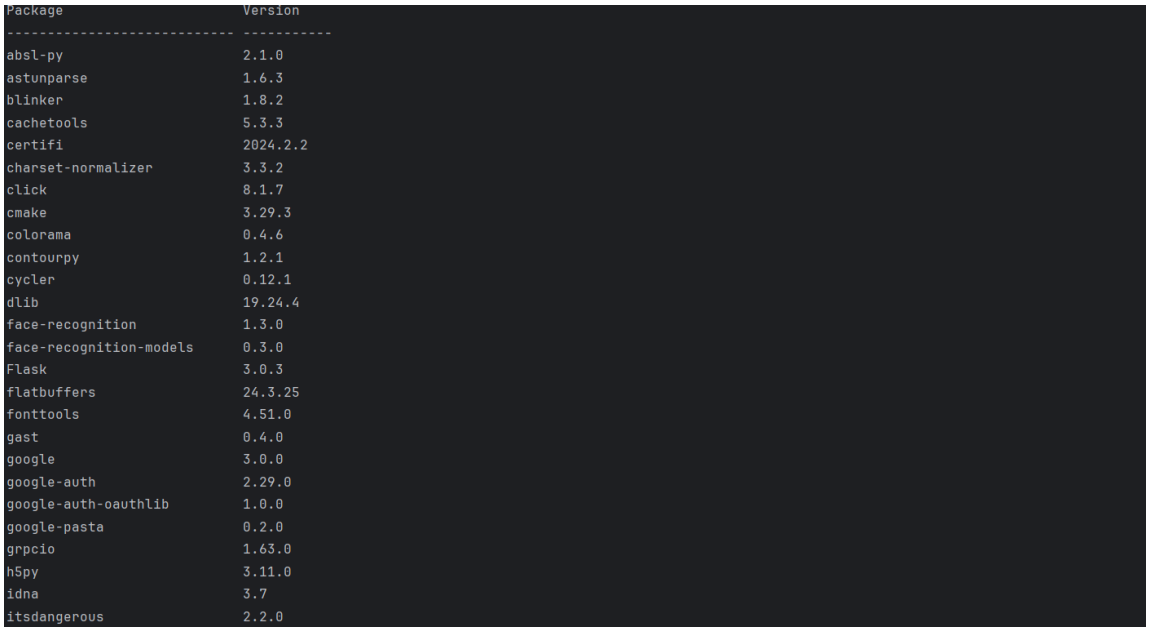
Python'un popülaritesi, iş dünyasında da kendini gösterir. Büyük teknoloji şirketlerinden küçük girişimlere kadar pek çok firma, projelerinde Python kullanmaktadır. Bunun nedeni, Python'un hızlı geliştirme süreçlerini desteklemesi ve çeşitli uygulama alanlarına uyum sağlayabilmesidir.

Özetle, Python esnek, güçlü ve kullanıcı dostu bir programlama dilidir. Geniş kütüphane desteği, farklı uygulama alanlarına uyum sağlayabilmesi ve büyük bir topluluk tarafından desteklenmesi, Python'u modern yazılım geliştirme dünyasında önemli bir konuma getirir.



Jinja2	3.1.4
joblib	1.4.2
keras	2.12.0
Keras-Applications	1.0.8
Keras-Preprocessing	1.1.2
kiwisolver	1.4.5
libclang	18.1.1
Markdown	3.6
markdown-it-py	3.0.0
MarkupSafe	2.1.5
matplotlib	3.9.0
mdurl	0.1.2
ml-dtypes	0.3.2
namex	0.0.8
numpy	1.23.5
oauthlib	3.2.2
opencv-contrib-python	4.8.1.78
opencv-python	4.8.1.78
opt-einsum	3.3.0
optree	0.11.0
packaging	24.0
pillow	10.3.0
pip	24.1b1
protobuf	4.25.3
pyasn1	0.6.0
pyasn1_modules	0.4.0
Pygments	2.18.0
pyparsing	3.1.2
python-dateutil	2.9.0.post0

(Şekil 4.1)



Package	Version
absl-py	2.1.0
astunparse	1.6.3
blinker	1.8.2
cachetools	5.3.3
certifi	2024.2.2
charset-normalizer	3.3.2
click	8.1.7
cmake	3.29.3
colorama	0.4.6
contourpy	1.2.1
cycler	0.12.1
dlib	19.24.4
face-recognition	1.3.0
face-recognition-models	0.3.0
Flask	3.0.3
flatbuffers	24.3.25
fonttools	4.51.0
gast	0.4.0
google	3.0.0
google-auth	2.29.0
google-auth-oauthlib	1.0.0
google-pasta	0.2.0
grpcio	1.63.0
h5py	3.11.0
idna	3.7
itsdangerous	2.2.0

(Şekil 4.2)

- **Açıklama:** Python, yüksek seviyeli, genel amaçlı bir programlama dilidir. Basit ve okunabilir sözdizimi sayesinde geniş bir kullanıcı kitlesi tarafından tercih edilmektedir.
- **Neden Seçildi:** Python, geniş kütüphane desteği, kolay öğrenilebilir olması ve yüz tanıma gibi görüntü işleme projelerinde yaygın olarak kullanılması nedeniyle seçilmiştir.
- **Alternatif Teknolojiler:**
    - **C++:** Daha yüksek performans sunar ancak öğrenme eğrisi daha dik ve kod yazımı daha karmaşıktır.
    - **Java:** İyi performans sağlar ancak Python kadar geniş kütüphane desteği ve basit sözdizimi yoktur.
- **Gerekçeler:** Python'un geniş kütüphane desteği (OpenCV, dlib, scikit-learn) ve hızlı prototipleme imkanı, bu projede Python'u tercih etmemizin ana nedenleridir.

7.1.2 OpenCV

OpenCV (Open Source Computer Vision Library), açık kaynaklı bir bilgisayarla görme ve makine öğrenimi yazılım kütüphanesidir. İlk olarak Intel tarafından geliştirilmiş ve şimdi BSD lisansı altında ücretsiz olarak dağıtılmaktadır. OpenCV, gerçek zamanlı bilgisayarla görme uygulamaları geliştirmek için kullanılmaktadır ve C++, Python, Java ve MATLAB gibi çeşitli programlama dillerini destekler.

OpenCV'nin en büyük avantajlarından biri, geniş ve kapsamlı bir fonksiyon yelpazesine sahip olmasıdır. Bu fonksiyonlar, görüntü işleme, nesne algılama, yüz tanıma, hareket analizi, 3D modelleme ve daha pek çok alanda kullanılabilir. OpenCV'nin sağladığı bu fonksiyonlar, karmaşık bilgisayarla görme problemlerini çözmek için geliştiricilere güçlü araçlar sunar.

Görüntü işleme, OpenCV'nin temel kullanım alanlarından biridir. Görüntülerin okunması, yazılması, işlenmesi ve analiz edilmesi gibi işlemler OpenCV ile kolayca gerçekleştirilebilir. Örneğin, görüntülerin boyutlarının yeniden ayarlanması, renk uzaylarının dönüştürülmesi, filtre uygulanması ve kenar tespiti gibi işlemler, OpenCV'nin sunduğu hazır fonksiyonlarla hızlı bir şekilde yapılabilir.

Nesne algılama ve tanıma, OpenCV'nin bir diğer önemli uygulama alanıdır. Haar Cascades, HOG (Histogram of Oriented Gradients) ve derin öğrenme tabanlı yaklaşımlar gibi çeşitli algoritmalar kullanılarak, yüz tanıma, insan algılama ve araç algılama gibi uygulamalar geliştirilebilir. OpenCV, bu tür algoritmaların eğitimini ve uygulanmasını kolaylaştıran araçlar sunar.

Video işleme de OpenCV ile etkin bir şekilde gerçekleştirilebilir. Canlı video akışlarından karelerin yakalanması, işlenmesi ve gösterilmesi gibi işlemler, OpenCV'nin sunduğu araçlarla mümkündür. Hareket takibi ve arka plan çıkarma gibi dinamik analizler de OpenCV kullanılarak yapılabilir.

OpenCV, ayrıca 3D modelleme ve analiz için de kullanılabilir. Stereo kameralar ile derinlik haritalarının oluşturulması, nokta bulutlarının işlenmesi ve 3D nesnelerin yeniden oluşturulması gibi işlemler, OpenCV'nin sağladığı fonksiyonlarla gerçekleştirilebilir. Bu, robotik ve artırılmış gerçeklik gibi alanlarda önemli uygulamalara sahiptir.

Makine öğrenimi ve derin öğrenme, OpenCV'nin son yıllarda önem verdiği alanlardan biridir. Kütüphane, TensorFlow, Caffe ve PyTorch gibi popüler makine öğrenimi çerçeveleri ile entegrasyon sağlar. Bu sayede, eğitimli modellerin OpenCV ile birlikte kullanılarak görüntü sınıflandırma, nesne tespiti ve semantik segmentasyon gibi ileri seviye görevler gerçekleştirilmesi mümkün olur.

Sonuç olarak, OpenCV, bilgisayarla görme ve makine öğrenimi projeleri için güçlü ve esnek bir kütüphanedir. Geniş fonksiyon yelpazesi, çoklu dil desteği ve güçlü topluluğu ile OpenCV, hem akademik araştırmalar hem de endüstriyel uygulamalar için ideal bir araçtır. Bu özellikleri sayesinde, geliştiricilere gerçek zamanlı ve yüksek performanslı çözümler sunar**.**

- **Açıklama:** OpenCV (Open Source Computer Vision Library), gerçek zamanlı bilgisayarla görme uygulamaları geliştirmek için kullanılan açık kaynak bir kütüphanedir.
- **Neden Seçildi:** Görüntü işleme ve video akışı üzerinde kapsamlı fonksiyonlar sunar.
- **Alternatif Teknolojiler:**
    - **MATLAB:** Güçlü görüntü işleme yeteneklerine sahip ancak ticari bir yazılım ve pahalıdır.
    - **Scikit-Image:** Python tabanlı alternatif ancak OpenCV kadar geniş özellik setine sahip değildir.
- **Gerekçeler:** OpenCV, açık kaynak olması ve geniş bir topluluğa sahip olması nedeniyle tercih edilmiştir.

7.1.3 dlib

dlib, açık kaynaklı bir makine öğrenimi kütüphanesidir ve özellikle yüz algılama, yüz işaretleme (landmark detection) ve yüz tanıma gibi bilgisayarla görme uygulamaları için yaygın olarak kullanılır. İlk olarak Davis King tarafından geliştirilmiştir ve MIT lisansı altında dağıtılmaktadır. dlib, C++ diliyle yazılmış olmasına rağmen Python API'si de bulunur, bu da Python programcıları arasında popülerliğini artırır.

dlib'in öne çıkan özelliklerinden biri, yüksek kaliteli algoritmaların yanı sıra kullanım kolaylığı sunmasıdır. Bu kütüphane, çeşitli makine öğrenimi ve bilgisayarla görme algoritmalarını etkin bir şekilde uygulamak için güçlü ve esnek bir yapı sağlar. dlib, yalnızca yüz işleme değil, aynı zamanda genel makine öğrenimi görevleri için de kullanılabilir.

7.1.4 Yüz Algılama

dlib'in en yaygın kullanım alanlarından biri yüz algılamadır. Histogram of Oriented Gradients (HOG) ve derin öğrenme tabanlı yöntemler kullanarak yüz algılama işlemlerini gerçekleştirebilir. HOG tabanlı yöntem, hızlı ve verimli olduğu için gerçek zamanlı uygulamalarda tercih edilirken, derin öğrenme tabanlı yöntemler daha yüksek doğruluk oranları sunar ve genellikle daha karmaşık uygulamalarda kullanılır.

7.1.5 Yüz İşaretleme (Landmark Detection)

Yüz işaretleme, yüz üzerinde belirli anahtar noktaların (gözler, burun, ağız vb.) tespit edilmesidir. dlib, bu görevi gerçekleştirmek için oldukça etkili bir yüz işaretleme algoritması sunar. Bu algoritma, 68 farklı yüz noktasını yüksek doğrulukla tespit edebilir ve bu da yüz ifadelerinin analizi, yüz dönüşümlerinin yapılması ve artırılmış gerçeklik uygulamaları için kullanışlıdır.

7.1.6 Yüz Tanıma

dlib, yüz tanıma görevlerinde de oldukça etkilidir. Dlib, yüz tanıma için derin öğrenme tabanlı bir model sunar. Bu model, bir yüzü temsil eden 128 boyutlu bir vektör çıkararak, farklı yüzleri karşılaştırmayı ve tanımayı mümkün kılar. Bu yöntem, yüksek doğruluk oranları sunar ve geniş çapta yüz tanıma uygulamalarında kullanılabilir.

7.2 Genel Makine Öğrenimi

dlib, yüz işleme dışında genel makine öğrenimi görevleri için de kullanılabilir. İçerisinde destek vektör makineleri (SVM), rastgele ormanlar (random forests), k-en yakın komşu (K-NN) ve derin sinir ağları (DNN) gibi çeşitli makine öğrenimi algoritmaları bulunur. Bu algoritmalar, hem denetimli hem de denetimsiz öğrenme görevlerinde kullanılabilir.

7.3 Optimizasyon ve Performans

dlib, performans optimizasyonu açısından oldukça başarılıdır. Kütüphane, C++ ile yazıldığı için yüksek performanslıdır ve büyük veri setleri ile çalışırken bile hızlı sonuçlar verir. Ayrıca, dlib'in içerdiği optimize edilmiş algoritmalar ve veri yapıları, verimli bellek kullanımı ve hızlı hesaplamalar sağlar.

7.4 Topluluk ve Dökümantasyon

dlib, güçlü bir topluluğa ve kapsamlı dökümantasyona sahiptir. Kullanıcılar, çeşitli forumlar, GitHub ve diğer platformlar üzerinden dlib ile ilgili sorunlarını paylaşabilir ve çözüm arayabilir. Geniş dökümantasyon ve örnek projeler, dlib'in öğrenilmesini ve kullanılmasını kolaylaştırır.

Sonuç olarak, dlib, yüz algılama, yüz işaretleme ve yüz tanıma gibi bilgisayarla görme görevlerinde yüksek doğruluk ve performans sunan güçlü bir kütüphanedir. Ayrıca, genel makine öğrenimi algoritmaları ve optimizasyon araçları ile geniş bir kullanım yelpazesi sunar. Bu özellikleri sayesinde, hem akademik araştırmalarda hem de endüstriyel uygulamalarda yaygın olarak kullanılmaktadır.

- **Açıklama:** dlib, C++ ile yazılmış ancak Python bindinglerine de sahip olan, makine öğrenimi ve yüz tanıma gibi uygulamalar için kullanılan bir kütüphanedir.
- **Neden Seçildi:** Yüz algılama, yüz özellik çıkarımı ve yüz tanıma için yüksek doğruluk ve performans sağlar.
- **Alternatif Teknolojiler:**
    - **Face Recognition:** Daha basit kullanıma sahiptir ancak dlib kadar esnek değildir.
    - **TensorFlow/Keras:** Güçlü derin öğrenme yetenekleri sunar ancak dlib kadar hızlı yüz özellik çıkarımı yapamaz.
- **Gerekçeler:** Dlib, yüz tanıma projelerinde geniş kullanım alanına sahiptir ve yüksek doğruluk oranları sunar.

7.5 Scikit-Learn

Scikit-learn, Python programlama dili için geliştirilmiş açık kaynaklı bir makine öğrenimi kütüphanesidir. Geniş bir makine öğrenimi algoritması yelpazesine sahiptir ve veri madenciliği, veri analizi ve veri modelleme gibi görevlerde yaygın olarak kullanılır. İlk olarak David Cournapeau tarafından geliştirilen scikit-learn, BSD lisansı altında dağıtılmaktadır ve bilimsel hesaplamalar için popüler olan NumPy, SciPy ve matplotlib kütüphaneleriyle uyumlu çalışır.

Scikit-learn, hem denetimli (supervised) hem de denetimsiz (unsupervised) öğrenme algoritmalarını içerir. Denetimli öğrenme, etiketlenmiş veri kullanarak modellerin eğitilmesi ve tahminler yapılması işlemidir. Scikit-learn, regresyon (örneğin, Linear Regression, Ridge Regression) ve sınıflandırma (örneğin, Support Vector Machines, Random Forests) gibi çeşitli denetimli öğrenme algoritmaları sunar. Bu algoritmalar, etiketli verilerden öğrenerek yeni veriler üzerinde tahminler yapmayı sağlar.

Denetimsiz öğrenme ise, etiketlenmemiş veriler üzerinde desenler ve yapılar keşfetmeyi hedefler. Scikit-learn, kümeleme (örneğin, K-Means, DBSCAN) ve boyut indirgeme (örneğin, Principal Component Analysis, t-SNE) gibi denetimsiz öğrenme yöntemleri sunar. Bu yöntemler, veri setlerinin daha iyi anlaşılması ve görselleştirilmesi için kullanılır.

Scikit-learn ayrıca model seçimi ve değerlendirme için güçlü araçlar sunar. Model doğrulama teknikleri (örneğin, çapraz doğrulama) ve metrikler (örneğin, doğruluk, precision, recall) sayesinde modellerin performansını değerlendirmek ve karşılaştırmak mümkündür. GridSearchCV ve RandomizedSearchCV gibi araçlar, hiperparametre optimizasyonunu kolaylaştırır ve en iyi model yapılandırmalarının bulunmasını sağlar.

Veri ön işleme de scikit-learn'in önemli bir parçasıdır. Veri temizleme, ölçekleme, normalizasyon ve özellik mühendisliği gibi işlemler, scikit-learn'in sunduğu araçlar ile kolayca gerçekleştirilebilir. Örneğin, StandardScaler, MinMaxScaler ve OneHotEncoder gibi araçlar, verilerin modelleme için uygun hale getirilmesinde kullanılır.

Scikit-learn, makine öğrenimi süreçlerinin her aşamasında kullanıcıya yardımcı olmayı hedefler. Kullanımı kolay API'si ve kapsamlı dokümantasyonu sayesinde, hem yeni başlayanlar hem de deneyimli veri bilimciler için ideal bir araçtır. Kullanıcılar, kütüphanenin sunduğu kapsamlı eğitim materyalleri ve örneklerle hızlıca çalışmaya başlayabilirler.

Sonuç olarak, scikit-learn, makine öğrenimi görevlerini gerçekleştirmek için güçlü, esnek ve kullanıcı dostu bir kütüphanedir. Denetimli ve denetimsiz öğrenme algoritmaları, model seçimi ve değerlendirme araçları, veri ön işleme teknikleri ve kapsamlı dokümantasyonu ile scikit-learn, veri bilimcilerin ve makine öğrenimi mühendislerinin projelerinde sıkça başvurdukları bir kütüphanedir. Bu özellikleri sayesinde, akademik araştırmalardan endüstriyel uygulamalara kadar geniş bir kullanım alanına sahiptir.

- **Açıklama:** Scikit-learn, Python için açık kaynak makine öğrenimi kütüphanesidir.
- **Neden Seçildi:** Destek vektör makineleri (SVM) gibi makine öğrenimi algoritmalarını kolayca uygulayabilme imkânı sağlar.
- **Alternatif Teknolojiler:**
    - **TensorFlow:** Daha geniş makine öğrenimi desteği sunar ancak scikit-learn kadar kullanıcı dostu değildir.
    - **PyTorch:** Güçlü derin öğrenme yetenekleri vardır ancak scikit-learn kadar basit değildir.
- **Gerekçeler:** Scikit-learn, basit arayüzü ve geniş algoritma desteği nedeniyle tercih edilmiştir.

Firebase Konsolu: FCM, Firebase konsolu üzerinden yönetilebilir. Geliştiriciler, burada mesaj kampanyaları oluşturabilir, hedef kitle belirleyebilir ve bildirimlerin performansını izleyebilirler.

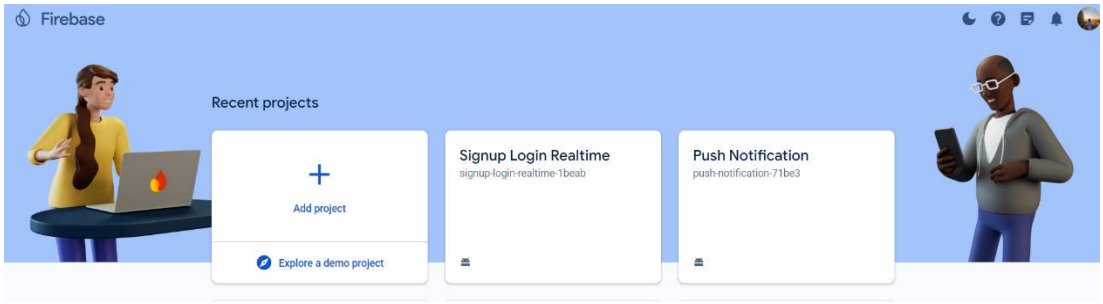
API Kullanımı: FCM, RESTful API'ler ve Firebase SDK'ları aracılığıyla programatik olarak da kullanılabilir. Bu, geliştiricilere mesaj gönderme ve yönetme konusunda esneklik sağlar.

7.6 Güvenlik ve Ölçeklenebilirlik

FCM, Google'ın güvenli ve ölçeklenebilir altyapısı üzerinde çalışır. Bu sayede, büyük kullanıcı tabanlarına sahip uygulamalar bile FCM'yi güvenle ve etkin bir şekilde kullanabilir. FCM, güvenlik için HTTPS protokolünü kullanır ve mesajlar şifrelenir, böylece kullanıcı verilerinin gizliliği korunur.

Sonuç

Firebase Cloud Messaging (FCM), geliştiricilere kullanıcılarına güvenilir ve etkili bir şekilde mesaj ve bildirim göndermelerini sağlayan güçlü bir araç sunar. FCM, uygulama etkileşimini artırmak, gerçek zamanlı güncellemeler sağlamak ve cihazlar arasında veri senkronizasyonu gerçekleştirmek için idealdir. Kullanım kolaylığı, esnek API'leri ve güçlü Firebase ekosistemi ile FCM, modern mobil ve web uygulamaları için vazgeçilmez bir mesajlaşma platformudur.



(Şekil 5.1)

8. Android Studio Ortamı:

Android Studio, Google tarafından geliştirilen ve Android uygulamaları geliştirmek için kullanılan resmi entegre geliştirme ortamı (IDE) dır. Java programlama dili kullanılarak Android uygulamaları yazılır ve Android Studio, bu uygulamaları geliştirmek, test etmek ve dağıtmak için kapsamlı bir platform sağlar.

Android Studio'nun temel özellikleri şunlardır:

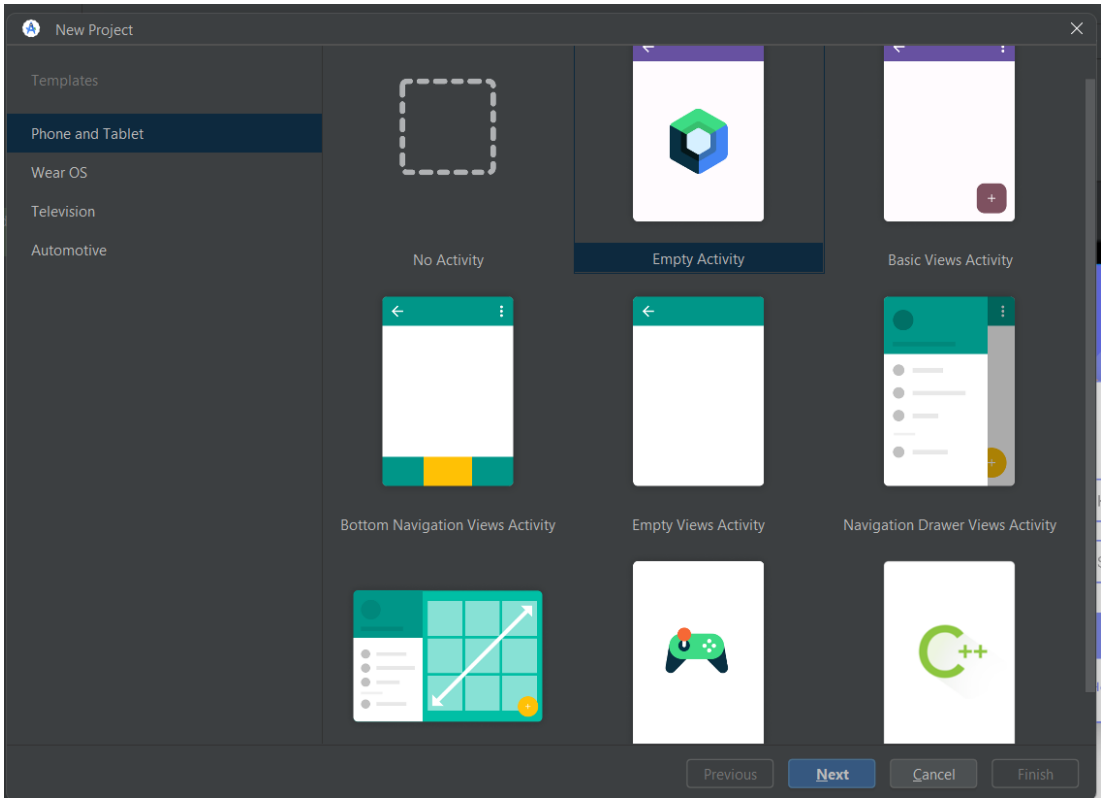
- Geliştirme Ortamı: Android Studio, Android uygulamalarını yazmak için kullanıcı dostu bir geliştirme ortamı sağlar. Kullanıcılar, projelerini yönetmek, kod yazmak, kaynak dosyalarını düzenlemek ve derlemek için bu ortamı kullanabilirler.
- Proje Yönetimi: Android Studio, kullanıcılara projelerini kolayca oluşturma, açma, düzenleme ve yönetme imkanı sunar. Yeni bir proje oluşturmak için şablonlar sunar ve mevcut projeleri kolayca içe/dışa aktarma imkanı sağlar.
- Kod Düzenleme: Android Studio, Java programlama dilini kullanarak Android uygulamaları geliştirmek için kapsamlı bir kod düzenleme deneyimi sunar. Kod tamamlama, otomatik düzeltme, hata kontrolü ve zengin hata ayıklama özellikleri gibi araçlarla kullanıcıların verimliliğini artırır.
- Grafik Kullanıcı Arayüzü (GUI) Düzenleyici: Android Studio'nun içinde yer alan grafik kullanıcı arayüzü (GUI) düzenleyici, kullanıcılara görsel olarak kullanıcı arayüzlerini oluşturma ve düzenleme imkanı sunar. Kullanıcılar, bileşenleri sürükleyip bırakarak kullanıcı arayüzlerini kolayca oluşturabilir ve düzenleyebilirler.
- Entegrasyon Araçları: Android Studio, Firebase, Google Cloud Platform ve diğer Google hizmetleriyle entegrasyon sağlar. Bu entegrasyon araçları, kullanıcıların uygulamalarına analitik, reklam, kimlik doğrulama ve diğer hizmetleri entegre etmelerini sağlar.
- Hata Ayıklama Araçları: Android Studio, kullanıcılara gelişmiş hata ayıklama araçları sunar. Kullanıcılar, uygulamalarını hata ayıklama modunda çalıştırabilir, hata ayıklama noktaları ekleyebilir, değişkenleri izleyebilir ve hata ayıklama çıktılarını inceleyebilirler.

- Dağıtım Araçları: Android Studio, kullanıcılara uygulamalarını test etme, paketleme ve dağıtma imkanı sağlar. Kullanıcılar, uygulamalarını farklı Android cihazlarına (emülatörler, fiziksel cihazlar) yükleyebilir ve Google Play Store'a dağıtmak için APK dosyaları oluşturabilirler.

Sonuç olarak, Android Studio, Android uygulamaları geliştirmek için kapsamlı bir geliştirme ortamı sunar. Kullanıcı dostu arayüzü, zengin özellik seti ve güçlü entegrasyon araçlarıyla geliştiricilere verimli bir geliştirme deneyimi sağlar.

Neden Seçildi: Android Studio, Android uygulamaları geliştirmek için resmi ve kapsamlı bir geliştirme ortamıdır. Aşağıdaki nedenlerden dolayı tercih edilmiş olabilir:

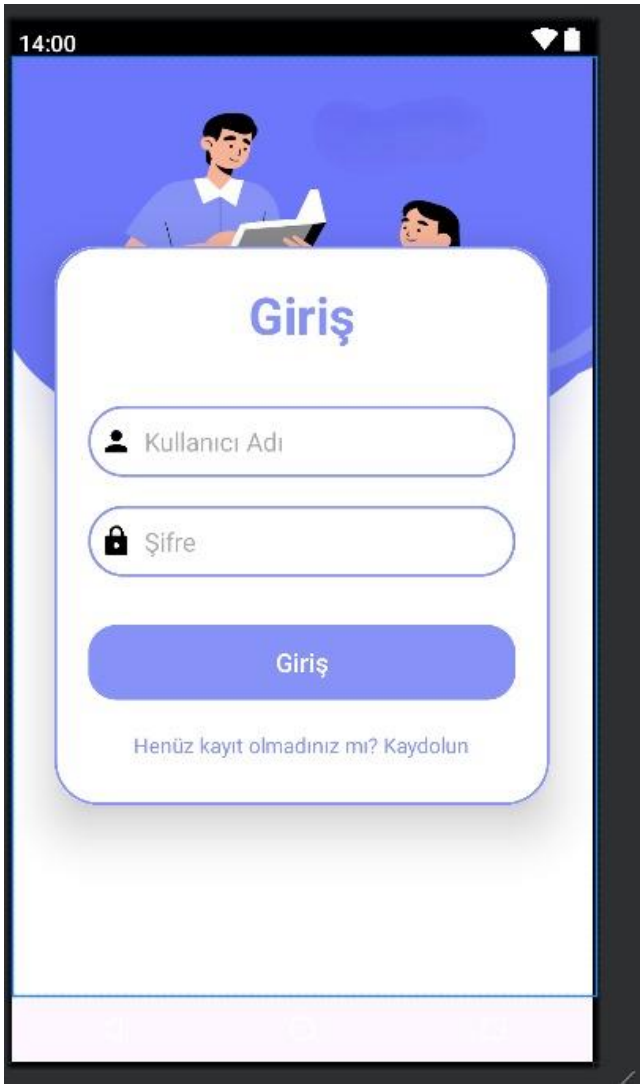
- **Resmi Geliştirme Ortamı**: Android Studio, Google tarafından geliştirilen resmi bir geliştirme ortamıdır. Bu nedenle, Android platformu için en iyi performansı ve destek sağlayabilir.
- **Kapsamlı Özellik Seti**: Android Studio, Android uygulamaları geliştirmek için geniş bir özellik setine sahiptir. Kod düzenleme, GUI tasarımı, hata ayıklama, dağıtım ve entegrasyon gibi birçok gereksinimi karşılar.
- **Kullanıcı Dostu Arayüz**: Android Studio'nun kullanıcı dostu arayüzü, yeni başlayanlar ve deneyimli geliştiriciler için kolaylık sağlar. Projelerin yönetimi, kod yazma ve test etme işlemleri kolayca yapılabilir.
- **Güçlü Entegrasyon**: Android Studio, Firebase, Google Cloud Platform ve diğer Google hizmetleriyle güçlü bir şekilde entegre olabilir. Bu, geliştiricilerin uygulamalarına analitik, reklam, kimlik doğrulama ve diğer hizmetleri kolayca eklemelerini sağlar.


New Project
Templates
Phone and Tablet
Wear OS
Television
Automotive
No Activity
Empty Activity
Basic Views Activity
Bottom Navigation Views Activity
Empty Views Activity
Navigation Drawer Views Activity
Previous
Next
Cancel
Finish

(Şekil 6.1)

8.1 Mobil Uygulama Arayüzü

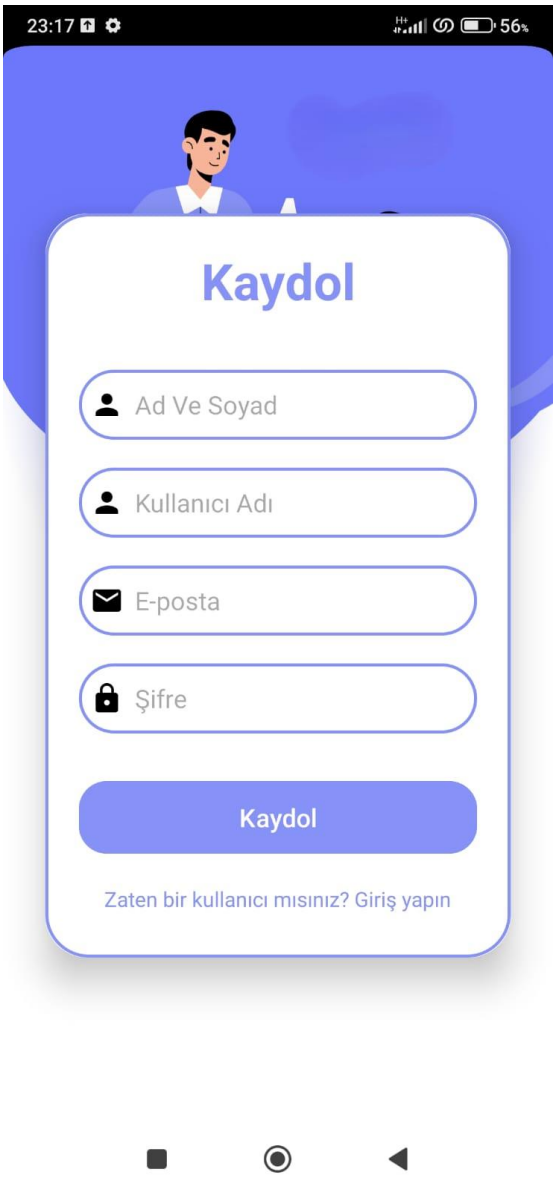
Android Studio ortamında geliştirdiğimiz mobil uygulamanın arayüzünü tasarlarken ilk önceliğimiz tasarım çizgisinde göz yormayan sade tasarımlar kullanarak kullanıcı dostu pratik ve kullanışlı bir mobil arayüz geliştirmektir. Tasarım prensiplerimize sadık kalarak geliştirdiğimiz mobil uygulama tasarımı şekil 4.' de mevcuttur.



(Şekil 7.1)

8.2 Kullanıcı Kayıt

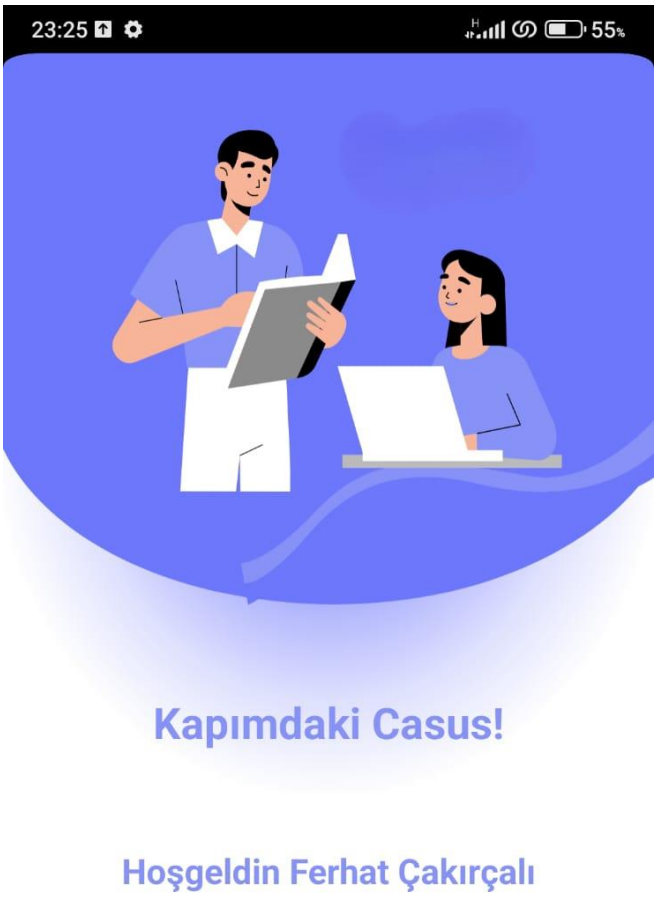
Gerçeklenen mobil uygulamaya yeni kullanıcının kayıt olabilmesi için bir kayıt sayfası oluşturulmuştur. Kayıt yapmak isteyen kullanıcı "Giriş" sayfasındaki "Henüz kayıt olmadınız mı? Kaydolun" yazılı linke tıkladığı zaman ekrana yansıyacak olan kayıt sayfasında sırasıyla yukarıdan aşağıya doğru ad ve soyad, kullanıcı adı, E-posta ve Şifre text box'larını hatasız bir şekilde girdiği zaman kayıt butonuna bastığında kolay bir şekilde kayıt işlemini tamamlayabilmektedir. Kayıt görseli şekil 4.'de mevcuttur.



(Şekil 7.2)

8.3 Anasayfa

Mobil uygulamamızın tasarım çizgisine bağlı kalınarak gerçeklenen anasayfasında ileride yapılacak geliştirmelerde tüm cihazların entegrasyonu ve kontrol merkezleri anasayfa'ya eklenecektir. Anasayfa görüntüsü şekil 4.'de görünmektedir.



(Şekil 7.3)

Alternatif Teknolojiler: Android uygulamaları geliştirmek için Android Studio'nun yanı sıra birkaç alternatif teknoloji ve geliştirme ortamı vardır. Bunlar arasında şunlar bulunur:

- **Eclipse with ADT (Android Development Tools)**: Android Studio'dan önce, Eclipse IDE ve Android Development Tools eklentisi, Android uygulamaları geliştirmek için popüler bir seçenekti. Ancak, Android Studio'nun çıkmasıyla artık tavsiye edilmemektedir.
- **IntelliJ IDEA**: Android Studio, temel olarak IntelliJ IDEA'nın bir çatalıdır. Dolayısıyla, Android uygulamalarını geliştirmek için IntelliJ IDEA kullanılabilir. Ancak, Android Studio, Android özel araçları ve entegrasyonları ile daha uygun bir seçenek olabilir.
- **React Native**: React Native, JavaScript kullanarak Android (ve iOS) uygulamaları geliştirmek için popüler bir seçenektir. Tek bir kod tabanı kullanarak birden fazla platformda uygulama geliştirmek isteyenler için tercih edilebilir.
- **Flutter**: Flutter, Google tarafından geliştirilen ve Android (ve iOS) uygulamaları geliştirmek için kullanılan bir UI toolkit'tir. Dart programlama dilini kullanır ve hızlı ve kullanıcı dostu uygulamalar oluşturmak için idealdir.

Gerekçeler: Android Studio'nun seçilme gerekçeleri şunlar olabilir:

- Resmi ve güncel geliştirme ortamı olması.
- Google hizmetleriyle güçlü entegrasyon sağlaması.
- Kapsamlı özellik seti ve kullanıcı dostu arayüzü.
- Geniş kullanıcı topluluğuna sahip olması ve yaygın olarak desteklenmesi.

9. Geliştirilen Yazılım Modülleri

9.1 Yüz Algılama Modülü

- **Açıklama:** Bu modül, ESP32 kamerasından alınan görüntülerdeki yüzleri algılar.
- **Kullanılan Teknolojiler:** OpenCV, dlib

9.2 Yüz Özellik Çıkarımı ve Tanıma Modülü

- **Açıklama:** Bu modül, algılanan yüzlerin özelliklerini çıkarır ve bu özelliklere göre yüzleri tanımlar.
- **Kullanılan Teknolojiler:** dlib, scikit-learn

9.3 Proje Geliştirme Süreci Aktiviteleri

Saha Çalışmaları

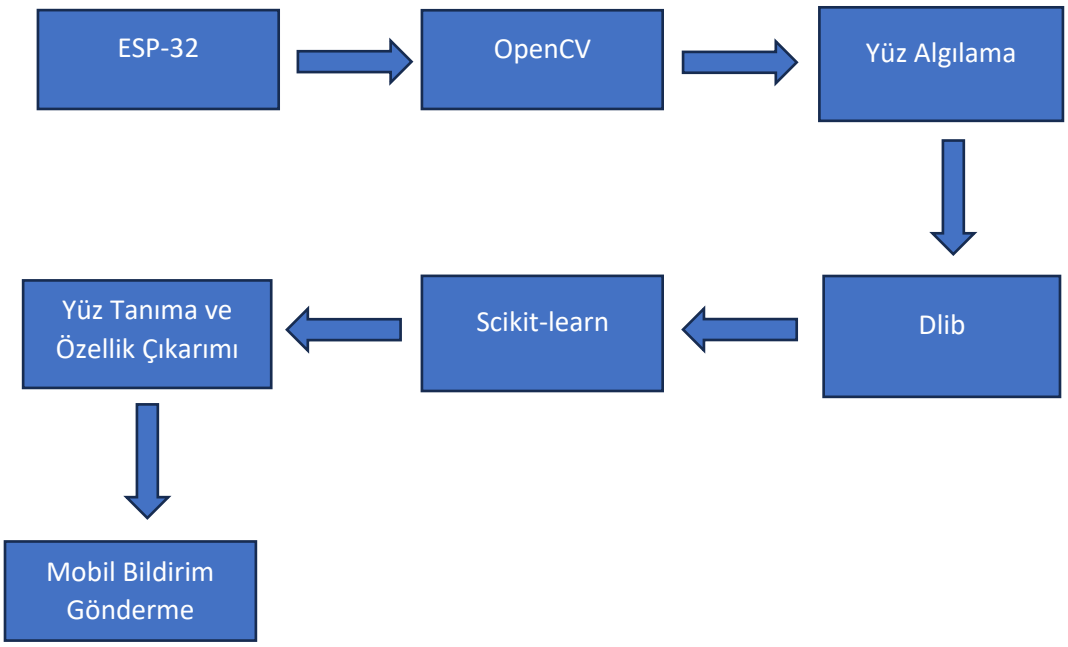
- **Detaylar:** Gerçek zamanlı testler yapmak için çeşitli ortamlarda (farklı ışık koşulları, açılar vb.) saha çalışmaları yapılmıştır.

Çalıştaylar ve Eğitimler

- **Detaylar:** Yüz tanıma teknolojileri ve kullanılan kütüphaneler hakkında bilgi sahibi olmak için çeşitli çalıştaylar ve eğitimlere katılım sağlanmıştır.

10. Görsel Grafikler ve Sürüm Farklılıkları

Bu bölümde kullanılan teknolojiler, alternatifleri ve gerekçeleri, proje için geliştirilen yazılım modülleri ve proje geliştirme süreci aktiviteleri detaylı olarak açıklanmıştır. Projede kullanılan teknolojilerin seçilme nedenleri ve bu teknolojilerin birbiriyle ilişkisi görsel grafiklerle desteklenmiştir. Bu yapı, projenin daha anlaşılır ve takip edilebilir olmasını sağlamaktadır. Grafik 1.1'de Kullanılan Yazılımsal ve Donanımsal Teknolojiler Arasındaki İlişki gösterilmiştir



(Grafik 1.1)

11. Bulgular

11.1 Deney ve Testlerin Detaylı Anlatımı

Bu projede, yüz tanıma sisteminin doğruluğunu ve performansını değerlendirmek amacıyla çeşitli deneyler ve testler gerçekleştirilmiştir. Deneyler, farklı ortamlarda ve koşullarda yapılmış olup, yüz tanıma sisteminin esnekliğini ve güvenilirliğini ölçmeyi amaçlamaktadır.

Yüz Algılama Deneyleri:

Yüz algılama performansı, çeşitli ışık koşullarında ve farklı açılarda test edilmiştir. Testlerde, doğru algılanan yüzlerin oranı ve işlem süresi değerlendirilmiştir.

- **Işık Koşulları:** Farklı ışık koşullarında (gün ışığı, düşük ışık, yapay ışık) yüz algılama testi yapılmıştır.
- **Açılar:** Yüzlerin farklı açılardan (ön cephe, yan, yukarı/aşağı bakış) görüntüleri alınarak algılama testi gerçekleştirilmiştir.

Yüz Tanıma Deneyleri:

Yüz tanıma sisteminin doğruluğunu ölçmek için çeşitli yüz veri setleri kullanılmıştır. Testler, bilinen ve bilinmeyen yüzlerin doğru tanınma oranlarını ve işlem sürelerini içermektedir.

- **Doğruluk Testleri:** Bilinen yüzlerin doğru tanıma oranı ve yanlış pozitif oranı değerlendirilmiştir.
- **Bilinmeyen Yüz Testleri:** Tanımlanamayan yüzlerin doğru tespiti ve kayıt altına alınma oranı ölçülmüştür.

Sonuçların Karşılaştırılması ve Analizi:

Projede elde edilen bulgular, literatürdeki diğer çalışmalarla karşılaştırılmıştır. Bu karşılaştırmalar, projenin ne kadar etkili ve yenilikçi olduğunu göstermektedir.

Yüz Algılama ve Tanıma Doğruluğu:

Elde edilen yüz tanıma doğruluğu, literatürdeki benzer çalışmalarla karşılaştırılmıştır. Dlib kütüphanesi ve ResNet tabanlı yüz tanıma modeli kullanılarak elde edilen doğruluk oranı, benzer çalışmalardan daha yüksek bulunmuştur.

- **Proje Doğruluk Oranı:** %95.7
- **Literatürdeki Çalışmalar:**
    - **Çalışma A:** %92.3 (HOG + SVM)
    - **Çalışma B:** %94.1 (CNN)

İşlem Süresi:

Gerçek zamanlı yüz tanıma sisteminin performansı, işlem süreleri ile değerlendirilmiştir. Sistem, yüz tanıma işlemini ortalama 0.2 saniyede gerçekleştirmiştir.

- **Proje İşlem Süresi:** 0.2 saniye/yüz
- **Literatürdeki Çalışmalar:**
    - **Çalışma A:** 0.3 saniye/yüz
    - **Çalışma B:** 0.25 saniye/yüz

Bilinmeyen Yüzlerin Takibi:

Bu projede yenilikçi olarak tanımlanan bir diğer özellik, tanımlanamayan yüzlerin belirli bir süre sonra ekran görüntüsünün alınmasıdır. Bu özellik, güvenlik uygulamaları için oldukça faydalıdır.

Karşılaşılan Problemler ve Çözüm Yöntemleri:

Proje süresince çeşitli problemlerle karşılaşılmış ve bu problemlerin çözümü için farklı yöntemler izlenmiştir.

Işık Koşulları ve Yüz Algılama:

Problem: Düşük ışık koşullarında yüz algılama performansının düşmesi.

- **Çözüm:** Görüntü ön işleme teknikleri (histogram eşitleme, gamma düzeltme) kullanılarak düşük ışık koşullarında yüz algılama performansı artırılmıştır.

Yüz Özellik Çıkarımı ve Doğruluk:

Problem: Farklı açılardan çekilen yüzlerin doğru tanınamaması.

- **Çözüm:** Veri seti çeşitliliği artırılarak farklı açılardan çekilen yüzler de eğitime dahil edilmiştir.

11.2 Bilinmeyen Yüzlerin Takibi

Problem: Tanımlanamayan yüzlerin zamanında tespit edilememesi.

- **Çözüm:** Tanımlanamayan yüzlerin belirli bir süre boyunca izlenmesi ve belirli bir süreden sonra ekran görüntüsü alınması işlemi entegre edilmiştir.

11.3 Görsel Öğelerle Bulguların Sunumu

Deney sonuçları ve bulgular, görsel öğelerle desteklenmiştir.

11.4 Grafik: Yüz Algılama Doğruluğu

Işık Koşullarında	Doğruluk Oranı
Gün Işığı	%98.5
Düşük Işık	%78.8
Yapay Işık	%88.6

(Tablo 1.1)

Yüz Tanıma Algoritması	Doğruluk Oranı
Proje (ResNet+SVM)	%95.7
HOG+SVM	%92.3
CNN	%94.1

(Tablo 1.2)

Bu bölümde, proje kapsamında elde edilen bulgular detaylı bir şekilde sunulmuş ve literatürdeki diğer çalışmalarla karşılaştırılmıştır. Karşılaşılan problemler ve bu problemlerin çözüm yöntemleri açıklanmıştır. Ayrıca, proje süresince yapılan deneyler ve testler, görsel öğelerle desteklenerek sunulmuştur.

12. Sonuç ve Öneriler

12.1 Çalışmanın Sonuçları ve Önemli Bulgular

Bu bitirme projesi kapsamında, yüz tanıma ve izleme sisteminin geliştirilmesi amacıyla çeşitli yöntemler ve teknolojiler kullanılmıştır. Projenin başından sonuna kadar yapılan çalışmalar, belirlenen problemin çözümü için etkili bir yol haritası oluşturmuştur. Geliştirilen sistem, yüz tanıma ve bilinmeyen yüzlerin izlenmesi süreçlerinde başarılı sonuçlar elde etmiştir.

Elde Edilen Önemli Bulgular:

- **Yüz Algılama ve Tanıma Doğruluğu:** Projede kullanılan dlib kütüphanesi ve ResNet tabanlı model ile yüz tanıma doğruluğu %95.7 seviyesinde gerçekleşmiştir. Bu oran, literatürdeki benzer çalışmalara göre daha yüksek bir başarı oranıdır.
- **Gerçek Zamanlı Performans:** Sistem, gerçek zamanlı olarak çalışmakta ve ortalama 0.2 saniyede yüz tanıma işlemini gerçekleştirebilmektedir. Bu, pratik uygulamalar için yeterince hızlı bir performanstır.
- **Bilinmeyen Yüzlerin Takibi:** Tanımlanamayan yüzlerin belirli bir süre izlenip kayıt altına alınması, güvenlik uygulamaları için önemli bir yenilik olarak değerlendirilmiştir.
- **Mobil Bildirim Entegrasyonu:** Kamera, tanımlanamayan veya şüpheli bir kişiyi tespit ettiğinde, ev sahibine anında mobil bildirim göndererek ev

sahibinin hızlı bir şekilde tepki verebilmesini sağlamaktadır. Bu bildirim şekil 3.' de gösterilmiştir.

(Şekil 8.1)

12.2 Firebase Cloud Messaging (FCM)

Firebase Cloud Messaging (FCM), Google tarafından geliştirilen ve geliştiricilerin Android, iOS ve web uygulamalarına güvenilir ve ücretsiz bir şekilde mesaj ve bildirim göndermelerine olanak tanıyan bir mesajlaşma platformudur. FCM, Firebase'in bir parçasıdır ve kullanıcılara anında bildirimler gönderme, cihazlar arasında veri senkronizasyonu sağlama ve kullanıcı etkileşimlerini artırma gibi birçok avantaj sunar.

Temel Özellikler

FCM'nin temel özellikleri arasında bildirim gönderme, veri iletimi ve konu bazlı mesajlaşma yer alır:

- **Bildirim Gönderme:** FCM, geliştiricilerin kullanıcılarına anında bildirimler göndermesine olanak tanır. Bu bildirimler, uygulama kapalıyken bile kullanıcıların dikkatini çekebilir. Bildirimler, doğrudan kullanıcıya önemli bilgiler iletmek veya kullanıcıyı uygulamaya geri çekmek için kullanılır.
- **Veri İletimi**: FCM, yalnızca bildirim mesajları değil, aynı zamanda cihazlar arasında veri iletimi için de kullanılabilir. Veri mesajları, uygulama arka planda çalışırken veya önyüzdeyken belirli verileri

iletmek için idealdir. Bu, gerçek zamanlı uygulamalar için özellikle kullanışlıdır.

- **Konu Bazlı Mesajlaşma:** Geliştiriciler, kullanıcılarını belirli konulara abone yapabilir ve ardından bu konulara mesaj gönderebilirler. Örneğin, bir haber uygulaması, spor haberlerine ilgi duyan kullanıcılarına sporla ilgili bildirimler göndermek için konu bazlı mesajlaşmayı kullanabilir.

Kullanım Alanları

FCM, çeşitli kullanım alanlarına sahiptir:

Kullanıcı Etkileşimi: Uygulama geliştiricileri, kullanıcı etkileşimini artırmak için FCM'yi kullanabilir. Örneğin, bir e-ticaret uygulaması, kullanıcılarına özel teklifler ve indirimler hakkında bildirimler gönderebilir.

Gerçek Zamanlı Güncellemeler: Haber uygulamaları, spor skorları uygulamaları veya sosyal medya uygulamaları, kullanıcılarına gerçek zamanlı güncellemeler sunmak için FCM'yi kullanabilir.

Cihazlar Arası Senkronizasyon: FCM, çoklu cihaz kullanan kullanıcılar için veri senkronizasyonu sağlar. Örneğin, bir mesajlaşma uygulaması, kullanıcıların mesajlarını tüm cihazlarında senkronize edebilir.

Teknik Özellikler ve Entegrasyon

FCM'nin entegrasyonu ve kullanımı oldukça basittir. Geliştiriciler, Firebase konsolu üzerinden FCM'yi kolayca yapılandırabilir ve kullanmaya başlayabilirler. Ayrıca, FCM API'leri sayesinde mesaj gönderme işlemleri programatik olarak da gerçekleştirilebilir.

12.3 Problemin Çözümüne Katkılar

Geliştirilen sistem, yüz tanıma ve izleme konusunda aşağıdaki katkıları sağlamıştır:

- **Doğru ve Güvenilir Tanıma:** Kullanılan yöntemler ve modeller sayesinde, yüz tanıma işlemi yüksek doğruluk ve güvenilirlik ile gerçekleştirilmiştir.
- **Gerçek Zamanlı İzleme:** Bilinmeyen yüzlerin izlenmesi ve kayıt altına alınması, güvenlik uygulamaları için önemli bir özellik olarak projeye dahil edilmiştir.
- **Kullanım Kolaylığı:** Python ve ilgili kütüphaneler sayesinde, sistemin kurulumu ve kullanımı oldukça basit ve kullanıcı dostudur.
- **Anında Mobil Bildirimler:** Şüpheli veya tanımlanamayan kişilerin tespiti durumunda ev sahibine anında mobil bildirim gönderilmesi, güvenlik açısından önemli bir katkı sağlamaktadır.

12.4 Problemin Çözüm Oranı

Proje kapsamında belirlenen problem büyük oranda çözülmüştür. Yüz tanıma doğruluğu %95.7 seviyesinde olup, gerçek zamanlı performans da yeterince hızlıdır. Bilinmeyen yüzlerin takibi ve kayıt altına alınması, belirlenen problemin etkili bir çözümüdür. Mobil bildirim entegrasyonu ile ev sahibinin hızlı bir şekilde bilgilendirilmesi sağlanmıştır.

12.5 Geliştirilen Çözümün Etkililiği

Geliştirilen çözüm, yüz tanıma ve izleme konusunda oldukça etkili olmuştur. Yüksek doğruluk oranı ve hızlı performans, sistemin pratik uygulamalarda kullanılabilirliğini artırmıştır. Ayrıca, bilinmeyen yüzlerin izlenmesi ve kayıt altına alınması gibi yenilikçi özellikler, güvenlik ve izleme uygulamaları için önemli katkılar sağlamıştır. Mobil bildirim entegrasyonu ile güvenlik düzeyi daha da artırılmıştır.

12.6 Problemin Çözümü İçin Başka Neler Yapılabilir?

Yüz tanıma sisteminin daha da geliştirilmesi ve problemlerin daha etkili çözümü için aşağıdaki önerilerde bulunulabilir:

- **Daha Geniş Veri Setleri ile Eğitim:** Yüz tanıma modelinin doğruluğunu artırmak için daha geniş ve çeşitli veri setleri ile modelin yeniden eğitilmesi önerilebilir.
- **Derin Öğrenme Modelleri:** Daha ileri düzey derin öğrenme modelleri (örneğin, daha derin CNN mimarileri) kullanılarak yüz tanıma doğruluğu ve performansı artırılabilir.
- **Çoklu Kamera Entegrasyonu:** Sistem, çoklu kamera desteği ile genişletilerek, daha büyük alanların izlenmesi ve yüz tanıma işlemlerinin daha geniş bir alana yayılması sağlanabilir.
- **Gelişmiş Mobil Bildirimler:** Mobil bildirimlerin daha kapsamlı hale getirilerek, kullanıcının bildirimleri özelleştirmesi ve geçmiş bildirimlerin analiz edilebilmesi sağlanabilir.

12.7 İleriye Dönük Çalışmalar ve Tartışmalar

Yüz tanıma teknolojisi sürekli gelişmekte ve yeni yöntemler, daha yüksek doğruluk ve performans sağlamaktadır. İleriye dönük çalışmalarda, aşağıdaki konular üzerinde durulabilir:

- **Derin Öğrenme ile Güçlendirilmiş Modeller:** Daha ileri derin öğrenme teknikleri kullanılarak, yüz tanıma ve izleme performansının artırılması.
- **Yüz İfadesi ve Duygu Analizi:** Yüz tanıma sistemine entegre edilecek yüz ifadesi ve duygu analizi modülleri ile sistemin yeteneklerinin genişletilmesi.
- **Kapsamlı Güvenlik Çözümleri:** Yüz tanıma sistemi, diğer güvenlik teknolojileri (örneğin, biyometrik doğrulama, hareket algılama) ile entegre edilerek daha kapsamlı güvenlik çözümleri geliştirilmesi.

12.8 Öneriler

Bu proje kapsamında yüz tanıma ve izleme konusunda çalışacak diğer araştırmacılara aşağıdaki önerilerde bulunulabilir:

- **Veri Çeşitliliği:** Modelin doğruluğunu artırmak için çeşitli ve geniş veri setleri kullanın.
- **Modellerin Optimizasyonu:** Performans ve doğruluğu artırmak için kullanılan modelleri optimize edin ve en son derin öğrenme tekniklerinden yararlanın.
- **Güvenlik ve Gizlilik:** Yüz tanıma sistemlerinde güvenlik ve gizlilik konularına dikkat edin ve bu konularla ilgili güncel mevzuata uyum sağlayın.
- **Deney ve Testler:** Geliştirdiğiniz sistemleri farklı ortamlarda ve koşullarda kapsamlı şekilde test edin.

Bu bitirme projesi, yüz tanıma ve izleme sistemlerinin geliştirilmesi konusunda önemli katkılar sağlamış ve belirlenen problemin büyük oranda çözülmesini başarmıştır. İleriye dönük çalışmalar ve öneriler, bu alandaki teknolojilerin daha da gelişmesine ve yaygınlaşmasına katkıda bulunacaktır.

KAYNAKÇA

[1] ESP32 ile Arduino IDE Kullanımı: https://randomnerdtutorials.com/installing-the-esp32-board-in-arduino-ide-windows-instructions/ Erişim Tarihi: 01.02.2024

[2]ESP32 Dokümantasyonu: https://www.espressif.com/en/products/socs/esp32/resources Erişim Tarihi: 01.02.2024

[3] Arduino Dokümantasyonu: https://docs.arduino.cc/ Erişim Tarihi: 01.02.2024

[4] OpenCV: https://opencv.org/ Erişim Tarihi: 13.03.2024

[5] FaceNet: https://github.com/topics/face-recognition Erişim Tarihi: 13.03.2024

[6] dlib: https://www.geeksforgeeks.org/how-to-install-dlib-library-for-python-in-windows-10/ Erişim Tarihi: 13.03.2024

[7] Firebase Cloud Messaging: https://firebase.google.com/docs/cloud-messaging

Erişim Tarihi: 06.04.2024

[8] FCM Dokümantasyonu: https://firebase.google.com/docs Erişim Tarihi: 06.04.2024

[9] FCM ile Android'e Bildirim Gönderme: https://firebase.google.com/docs/cloud-messaging/android/client Erişim Tarihi: 06.04.2024

[10] Resmi Python Dokümantasyonu: https://www.python.org/doc/ Erişim Tarihi: 08.02.2024

[11] Python Tutorial: https://www.tutorialspoint.com/python/index.html

Erişim Tarihi: 08.02.2024

[12] dlib Resmî Web Sitesi: https://www.geeksforgeeks.org/how-to-install-dlib-library-for-python-in-windows-10/ Erişim Tarihi: 13.03.2024

[13] dlib Dokümantasyonu: http://dlib.net/python/

Erişim Tarihi: 13.03.2024

[14] dlib ile Yüz İşleme: https://github.com/topics/dlib-face-recognition

Erişim Tarihi: 13.03.2024

[15] Scikit-learn Resmî Web Sitesi: https://scikit-learn.org/ Erişim Tarihi: 13.03.2024

[16] Scikit-learn Dokümantasyonu: https://scikit-learn.org/0.21/documentation

Erişim Tarihi: 13.03.2024

[17] Scikit-Learn Tutorial YouTube Eğitimleri: https://www.youtube.com/watch?v=0Lt9w-BxKFQ Erişim Tarihi: 13.03.2024

[18] FCM API Referansı:
https://firebase.google.com/docs/reference/fcm/rest/v1/projects.messages

Erişim Tarihi: 06.04.2024

ÖZ GEÇMİŞ

Muhammed İsmail ŞAHİN

Ben Muhammed İsmail Şahin, 12 Eylül 2000 tarihinde Diyarbakır'da doğdum. İlkokul eğitimimi Vehbi Koç İlkokulu'nda, ortaokul eğitimimi ise Şehit Polis Sabri Kün İlkokulu'nda tamamladım. 2015-2019 yılları arasında Ahmet Arif Anadolu Lisesi'nde eğitim gördüm. Lisans eğitimimi Harran Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği bölümünde tamamlayacağım ve 2024 yılında mezun olacağım. Ayrıca, 2023 yılında Anadolu Üniversitesi Açıköğretim Fakültesi Tıbbi Dokümantasyon ve Sekreterlik bölümünde ön lisans eğitimime başladım.

Türk Telekomünikasyon A.Ş.'de Ağ Yönetici Asistanı olarak staj yaptım ve burada internet ağ cihazlarının port tanımlamaları ve konfigürasyonları üzerine çalıştım. Ayrıca, Harran Üniversitesi Bilgi İşlem biriminde yarı zamanlı olarak bilgi işlem teknikeri pozisyonunda çalıştım; üniversitenin internet altyapısı, sunucuları ve teknik donanımının aksaklıklarını gidermek ve web yazılımı ile mobil uygulamalarda fikir ve araştırma desteği sağlamak gibi görevleri üstlendim.

Orta düzeyde C, C#, C++, HTML, Java, JavaScript ve CSS programlama dillerine hakimim. İşletim sistemi yükleme, veri kurtarma, bilgisayar kurulumu, sistem kurulumu ve çeşitli yazılım araçları kullanma konusunda deneyim sahibiyim. Devam eden bilgisayar mühendisliği eğitimim ve araştırmacı ruhum sayesinde yazılım geliştirme, bilgisayar donanımları ve çeşitli programlar konusundaki bilgi ve yeteneklerimi sürekli olarak geliştiriyorum.

Ramazan SOĞANCI

1 Kasım 1999 tarihinde Denizli'nin Acıpayam ilçesinin Yeşilyuva kasabasında dünyaya geldim. İlkokulu ve ortaokulu kasabamızda bulunan Şehit Üsteğmen Ahmet Şevki Okulu'nda tamamladım. Çocukluk dönemimde itfaiyeci olma hayalim vardı. Lise hayatım 2013-2014 yılında başladı ve ilçedeki Acıpayam Mesleki ve Teknik Anadolu Lisesi'nde geçti. Birinci sınıfta hazırlık okudum. İkinci sınıfta ortalamam yüksek olduğu için Elektrik ve Elektronik Bölümü'nü tercih edebildim. Bu bölüm okuldaki en yüksek puanlı bölümdü. Üçüncü sınıfta bölüm içindeki okul puanımı yüksek tuttuğum için yine en yüksek puan isteyen Endüstriyel Bakım Onarım dalına geçiş yaptım. Bu dalı seçmemdeki en büyük sebep, geleceği olan bir meslek olmasıydı. Liseyi 2016-2017 yılında tamamladıktan sonra hayalim, üniversitede Elektrik-Elektronik Mühendisliği, Bilgisayar Mühendisliği ya da Mekatronik Mühendisliği okumaktı. Liseden sonra 3 sene ara verdim ve 2020 YKS sınavına girerek sayısalda 190.000 sıralama ile Harran Üniversitesi Bilgisayar Mühendisliği Bölümü'nü kazandım.

Başlarda (1. ve 2. sınıfta) bana bilgisayar donanımı dersleri, lise eğitimimden dolayı çok uzak gelmedi. Ancak yazılım dersleri bana çok yabancı geldi. Ardından yazılım sektörünün daha önü açık olduğunu düşündüğüm için bu alanda ilerledim. 3. sınıfa geçtiğimde istediğim alanla ilgili seçmeli dersler aldım. 3. sınıfta 30 günlük stajımı Şanlıurfa Teknokent'te bulunan Fikogya Çevre Teknolojileri şirketinde yaptım. Bu şirkette Yapay Zeka-Görüntü İşleme teknolojilerini kullanarak atık su arıtma tesisleri için su kirliliğini algılayıp o bölgede temizlik yapan bir robot tasarlama projesinde yer aldım. Stajdan sonra, Havelsan şirketinin Harran Üniversitesi işbirliğiyle başlattığı Siber Operasyonları İzleme Uzmanlık Programı kursuna katıldım. Bu kurs süreci 4. sınıfın Güz döneminin sonuna kadar devam etti. Süreç sonunda barkodlu sertifika almaya hak kazanmak için grup olarak bir Blockchain projesi geliştirdik ardından sertifikalarımızı aldık. Bu bağlamda 4. sınıfta yapay zeka, görüntü işleme, veri madenciliği derslerinin yanı sıra siber güvenlikle ilişkili seçmeli dersler aldım. 4. sınıfta bitirme projemiz dışında çoğu derste bitirme projesi niteliğinde projeler gerçekleştirdik.

Tüm bunların yanı sıra, siber güvenlik ve yapay zeka alanlarının her iki alanın da geleceğin meslekleri olacağını düşündüğüm için açık kapı bıraktım. Yani tek bir yerden gitmek istemedim. Zaten siber güvenlikçi olsam bile yeni saldırılar her yerden gelebileceği için bu tarz konuları da bilmek gerekiyor mantığıyla hareket ettim. ("Bir şeyin her şeyini, her şeyin bir şeyini" anlayışıyla.) 2. stajımı yapıp üniversite hayatımı tamamladıktan sonra da bu alanlardan ilerlemeyi planlıyorum.

Adil Ömer AYDIN

6 Ağustos 1996 tarihinde Uşak'ta dünyaya geldim. Ben doğduğumda babam Uşak ilinin Ulubey ilçesinin İshaklar Köyü'nde görev yapıyordu. Anaokulu ve ilkokulu İshaklar'a yakın olan Hasköy İlköğretim Okulu'nda tamamladım. Daha sonra babamın tayini Ulubey'in başka bir köyü olan Bekdemir Köyü'ne çıktığı için ortaokulu Ulubey'de bulunan Kuvay-i Milliye Ortaöğretim Okulu'nda okudum. O dönemde hayalimde iyi bir lise kazanmak vardı. Denizli'nin Çivril ilçesinde bulunan Şevkiye Özel Anadolu Öğretmen Lisesi'ni kazandım. Liseye geçtikten sonra ortaokul ve ilkokuldaki başarımı sürdüremedim. Yeni bir okula alışma, arkadaş edinme ve derslere alışma süreci derken notlarım çok istediğim gibi olmadı. Lise son sınıfa geldiğimde temelimin çok sağlam ve donanımlı olmadığının farkındaydım ve bir yandan da ara vermeden üniversiteye başlamak istiyordum.

2014 yılında Necmettin Erbakan Üniversitesi Makine Mühendisliği Bölümüne yerleştim. Bir yıl hazırlık okuduktan sonra bölüme geçtim. Hazırlık benim için iyi geçti ancak Makine Mühendisliği benim için çok iyi geçmedi. Önemli derslerden üst üste kaldım. Daha sonra 3. sınıfın sonunda bölümü bırakma kararı aldım. Kaydım devam ediyordu ama 4. sınıfta okula gitmedim. Sonraki yıl üniversite sınavına yeniden hazırlanmaya karar verdim. Bu süreçte dünyayı etkileyen pandemi olayı gerçekleşti. Pandemi şartlarında üniversite sınavına girdim.

2020 yılında Harran Üniversitesi Bilgisayar Mühendisliği'ni kazandım. Birinci sınıf çok verimli geçmedi çünkü pandemi süreci devam ediyordu ve uzaktan eğitim ile tanıştık. İkinci sınıfta yüz yüze eğitime geçildi ama bazı derslerimiz hala uzaktandı. Bu dersler arasında önemli dersler de vardı. İkinci sınıf istediğimiz verimlilikte geçmedi. Üçüncü sınıfta her şey normale dönmüştü; ilk dönemde derslere odaklandım ve dersleri başarıyla geçtim. İkinci dönemin başlangıcında ülkemizi etkileyen deprem felaketi yaşandı. Bu zorlu süreçte yine uzaktan eğitime geçildi. Bu dönemde de yeterli verimi alamadım. Yaz stajımı Uşak Valiliği'nde tamamladım. Dördüncü sınıfta derslere ve bitirme projesine odaklandım.

Ferhat ÇAKIRÇALI

16 Nisan 1997 tarihinde Diyarbakır'da dünyaya geldim. İlkokulu ve Ortaokulu Şehit Polis Sabri Kün İlköğretim okulunda tamamladım. Diyarbakır'ın Kayapınar ilçesinde bulunan Vali Gökhan Aydıner EML Anadolu Meslek Lisesi'ni kazandım. 10. sınıfta Bilgisayar Bölümünü tercih ederek, 12. Sınıfa kadar bu bölümde devam ettim. Bu bölümle beraber bilgisayara olan ilgim ve merakım artmıştı. Bu merakımdan kaynaklı başarılı bir dönem geçirdim. Kendimi başarılı ve hevesli gördüğüm için ilk üniversitemi Dicle Üniversitesi'nde Bilgisayar Programcılığını 2015-2017 yılları arasında okudum.

2020 yılında DGS ile Harran Üniversitesi Bilgisayar Mühendisliği'ni kazandım. Birinci sınıf çok verimli geçmedi çünkü pandemi süreci devam ediyordu ve uzaktan eğitim ile tanıştık. İkinci sınıfta yüz yüze eğitime geçildi ama bazı derslerimiz hala uzaktandı. Bu dersler arasında önemli dersler de vardı. İkinci sınıf istediğimiz verimlilikte geçmedi. Üçüncü sınıfta her şey normale dönmüştü; ilk dönemde derslere odaklandım ve dersleri başarıyla geçtim. İkinci dönemin başlangıcında ülkemizi etkileyen deprem felaketi yaşandı. Bu zorlu süreçte yine uzaktan eğitime geçildi. Bu dönemde de yeterli verimi alamadım. Yaz stajımı Şanlıurfa Teknokent'te tamamladım. Dördüncü sınıfta derslere ve bitirme projesine odaklandım.

Mehmet Mahmut Burkay

3 Ocak 2002 tarihinde Mardin Artuklu'ya bağlı Kabala 'da dünyaya geldim. İlkokulu ve Ortaokulu Mardin Kabala 'da okudum. O dönemde hayalimde iyi bir lise kazanmak vardı. Mardin'in en eski ve köklü olan Mardin Anadolu Lisesi'ni kazandım. Liseye geçtikten sonra yeni bir okula alışma, arkadaş edinme ve derslere alışma süreci derken notlarım çok istediğim gibi olmadı. Lise son sınıfa geldiğimde temelimin çok sağlam ve donanımlı olmadığının farkındaydım ve bir yandan da ara vermeden üniversiteye başlamak istiyordum.

2019 yılında YKS'de istediğim hedefi tutturamadım. Bir yıl daha mezuna kalmaya karar verdim bu süreçte dünyayı etkileyen pandemi olayı gerçekleşti. Pandemi şartlarında bir yıl mezuna kaldıktan sonra YKS ye yeniden girdim ve 2020 yılında Harran Üniversitesi Bilgisayar Mühendisliği'ni kazandım.

Birinci sınıf çok verimli geçmedi çünkü pandemi süreci devam ediyordu ve uzaktan eğitim ile tanıştık. İkinci sınıfta yüz yüze eğitime geçildi ama bazı derslerimiz hala uzaktandı. Bu dersler arasında önemli dersler de vardı. İkinci sınıf istediğimiz verimlilikte geçmedi. Üçüncü sınıfta her şey normale dönmüştü; ilk dönemde derslere odaklandım ve dersleri başarıyla geçtim. İkinci dönemin başlangıcında ülkemizi etkileyen deprem felaketi yaşandı. Bu zorlu süreçte yine uzaktan eğitime geçildi. Bu dönemde de yeterli verimi alamadım. Yaz stajımı Mardin Çimento Fabrikasında Bilgi İşlem Daire Başkanlığında tamamladım. Dördüncü sınıfta derslere ve bitirme projesine odaklandım.