**T.C.**

**BANDIRMA ONYEDİ EYLÜL ÜNİVERSİTESİ**

**MÜHENDİSLİK VE DOĞA BİLİMLERİ FAKÜLTESİ**



**LİSANS BİTİRME PROJESİ**

**Python ve Flet (Flutter) Tabanlı Derinleştirilmiş Yapay Zekâ Uygulaması**

**Huzeyfe AKTAŞ**

**Zubair Najim DOST**

**Elektrik Elektronik Mühendisliği Bölümü**

**HAZİRAN 2024**

**T.C.**

**BANDIRMA ONYEDİ EYLÜL ÜNİVERSİTESİ**

**MÜHENDİSLİK VE DOĞA BİLİMLERİ FAKÜLTESİ**



**LİSANS BİTİRME PROJESİ**

**Python ve Flet (Flutter) Tabanlı Derinleştirilmiş Yapay Zekâ Uygulaması**

**Huzeyfe AKTAŞ**

**Zubair Najim DOST**

**DANIŞMAN**

**Doktor Öğretim Üyesi Kadir İLERİ**

**Elektrik Elektronik Mühendisliği Bölümü**

**HAZİRAN 2024**

**ONAY**

Huzeyfe AKTAŞ ve Zubair Najim DOST tarafından hazırlanan “**Python ve Flet (Flutter) Tabanlı Derinleştirilmiş Yapay Zeka Uygulaması**”adlı proje çalışması …/…/… tarihinde yapılan sınavla aşağıdaki jüri tarafından oybirliği/oyçokluğu ile Bandırma Onyedi Eylül Üniversitesi, Mühendislik ve Doğa Bilimleri Fakültesi, Elektrik- Elektronik Mühendisliği Bölümünde LİSANS BİTİRME PROJESİ olarak kabul edilmiştir.

**Jüri** **Üyeleri** **İmza**

Dr. Öğr. Üyesi KADİR İLERİ

(Danışman)

Dr. Öğr. Üyesi ONURSAL ÇETİN

(Üye)

Doç. KUTLUCAN GÖRÜR

(Üye)

Doç. ABDULLAH GÖKYILDIRIM

(Üye)

Prof. Dr. Muhammet Nuri SEYMAN

Bölüm Başkanı

**ÖZET**

Hızla ilerleyen bu teknolojik çağda Python ve Flutter'ın birleşimi güçlü ve akıllı uygulamalar yaratmanın yolunu açtı. Bu uygulamalar, Gemini API ve Flet gibi kütüphaneler aracılığıyla AI ve ML'nin yeteneklerinden yararlanır. Bu uygulamanın temel amacı, büyük dil modellerini, görselleri ve akademik araştırma makalelerini analiz etmek ve yorumlamaktır. Uygulamamız, yapay zekâ tabanlı modelleme tekniklerinin potansiyelinden yararlanarak, çeşitli uygulama alanlarında insan yaşamını iyileştirmeye katkıda bulunan akıllı mobil uygulamalara kapsamlı bir genel bakış sunmayı amaçlamaktadır. Derin öğrenme çözümlerini kullanarak uygulama, gerçek zamanlı performans gerçekleştirebilir. Akıllı telefonlarda derin öğrenme çıkarım ağlarının konuşlandırılması. Bu, akıllı telefonların yalnızca daha güvenli olmasını değil, aynı zamanda potansiyel tehditleri tahmin edebilmesini ve önleyebilmesini de sağlar. Mezuniyet projemizi özellikle Python tabanlı ön uç ve arka uç yazılım mühendisliğini hedefleyerek yapay zekâ ve onun yardımcı programlarına dayalı yapmayı hedefledik. Flutter genellikle bir ön uç UI/UX kolaylaştırıcısı olarak kullanıldığından, bu yeni kütüphane Flet ile Flutter'ı yapay zekâ odaklı arka uca da entegre edebildik. Python'un kullanımıyla, bir dizi yapay zekâ tekniği makine öğrenimi, algılayıcı, karar ağacı grafikleri, kural tabanlı sistemler, sınıflandırma ve derin öğrenme gibi uygulamalar sorunsuz bir şekilde uygulanabiliyor ve uygulamaya entegre edilebiliyor. Uygulama, çeşitli kaynaklardan veri topluyor ve bunları eğitim veri tabanını kullanarak analiz ediyor. Bu uygulamanın özel olarak GPT'ye benzemesi veya en azından pahalı kaynaklar tarafından gerçekleştirilen görevlerin çoğunu yapabilen ve sınava çalışmak gibi günlük akademik performansları için öğrenciler ve öğretmenler için erişilebilir hale getirebilen, kendi kendine organize edilen AI sohbetimiz olması hedeflenmektedir, bir araştırma makalesi yazmak veya hatta veri analizi yazmak. OpenCV kullanan profesörler için sınav kâğıdı kontrol sürecini bile kolaylaştırmak amacıyla görüntü işleme yükümlülüklerini dahil ettik. Uygulama aynı zamanda Python işlevlerini web uygulamalarına dönüştürmeye odaklanıyor ve kullanıcıların uygulamaya elektronik ders kitapları aracılığıyla erişmesine olanak tanıyor.

**TEŞEKKÜR**

Çalışmayı yöneten danışmanımıza ve hep yanımızda olan değerli hocalarımıza teşekkürlerimi iletiyorum. Hocalarımızın bugüne kadar bize canı gönülden destek çıktıkları için minnettarız. Bir gün insanlara ve daha doğrusu gezegenimize hizmet vermek pozisyonda bulunduğumuz zaman sizin fedakarlıklarınızı unutmayacağız. Pandemi döneminde zorluklarla beraber bizi buraya yönlendiren hevesimize ve bilime karşı sevgimiz hep baki kalacaktır.

**Huzeyfe AKTAŞ**

**Zubair Najim DOST**

**İÇİNDEKİLER**

[1. GİRİŞ 7](#_Toc168953244)

[2. Literatür 8](#_Toc168953245)

[3. Materyal ve Yöntem 13](#_Toc168953246)

[3.1 Python Programlama Dili 13](#_Toc168953247)

[3.2 Flet Uygulaması 15](#_Toc168953248)

[3.2.1 Uygulamamızı Geliştirirken Kullanılan Kütüphaneler 20](#_Toc168953249)

[3.3 Uygulamanın Ana Menüsü 23](#_Toc168953250)

[3.3.1 Resim Analizi 24](#_Toc168953251)

[3.3.2 PDF Analizi 31](#_Toc168953252)

[3.3.3 Kamera 36](#_Toc168953253)

[3.3.4 Excel Analizi 41](#_Toc168953254)

[4. Sonuçlar 45](#_Toc168953255)

[4.1 Görüntü Analizin Sonuçları 46](#_Toc168953256)

[4.2 PDF Analizin Sonuçları 48](#_Toc168953257)

[4.3 Kamera Bölümün Sonuçları 50](#_Toc168953258)

[4.4 Excel Analizin Sonuçları 51](#_Toc168953259)

[Kaynaklar 52](#_Toc168953260)

[ÖZGEÇMİŞ 53](#_Toc168953261)

[Zubair Najim Dost 53](#_Toc168953262)

[Huzeyfe Aktaş 53](#_Toc168953263)

**Sayfa**

**TABLO LİSTESİ**

**Sayfa**

[Tablo 1 Satış Tablousu 51](#_Toc168654966)

**ŞEKİL LİSTESİ**

**Sayfa**

[Şekil 1 Bilgisayar Bilimleri bölümlerinin sayısı ve giriş derslerini öğretmek için kullanılan diller 9](#_Toc168953093)

[Şekil 2 Flet Sohbet Uygulamasının Temel Mimarisi 16](#_Toc168953094)

[Şekil 3 17](#_Toc168953095)

[Şekil 4 Basit bir Flet sohbet penceresi 20](#_Toc168953096)

[Şekil 5 AI konusunda Google'in önderliği 21](#_Toc168953097)

[Şekil 6 AI Modelleri Eğitmek İçin Tahmini Maliyet 21](#_Toc168953098)

[Şekil 7 Ana Menü 23](#_Toc168953099)

[Şekil 8 Renk Seçim Menüsü 27](#_Toc168953100)

[Şekil 9 Üç Nokta Şekli Buton 30](#_Toc168953101)

[Şekil 10 Prompt Mühendisliği Yapılmış Butonlar 30](#_Toc168953102)

[Şekil 11 PDF analizin arayüzü 33](#_Toc168953103)

[Şekil 12 PDF Dosyanın Yükleme Sistemi 34](#_Toc168953104)

[Şekil 13 PDF Analizi İçin Prompt Butonları 35](#_Toc168953105)

[Şekil 14 Kamera Analiz Penceresi 41](#_Toc168953106)

[Şekil 15 Yüklenmiş Sayıların Görüntüsü 46](#_Toc168953107)

[Şekil 16 Görüntü İşlemeden Sonra Sadece Kırmızı Renkler 47](#_Toc168953108)

[Şekil 17 Yeşilin Görüntü İşlemesi 47](#_Toc168953109)

[Şekil 18 Görüntü İşlemeden Sonra Kırmızı Rengin Algılaması 48](#_Toc168953110)

[Şekil 19 Yorumlama Butonu 49](#_Toc168953111)

[Şekil 20 Çıkan Sonuç 49](#_Toc168953112)

[Şekil 21 OpenCv İle Kamera Açma ve Kaydetme 50](#_Toc168953113)

[Şekil 22 Kamera Görüntüsünü Analiz Etme Butonları 50](#_Toc168953114)

[Şekil 23 Dosyanın Sözel Analizi 51](#_Toc168953115)

# GİRİŞ

Hızlı teknolojik ilerleme çağında, gerçek zamanlı iletişim araçlarına olan ihtiyaç çok önemli hale geldi. Bu tez, Gemini API'sini bir dil öğrenme sağlayıcısı (LLP) olarak kullanan ve Python'u temel programlama dili olarak entegre eden yenilikçi bir gerçek zamanlı sohbet uygulamasının geliştirilmesini sunmaktadır. Kusursuz etkileşim ve sezgisel bir kullanıcı deneyimi sağlamak için Python tabanlı bir GUI çerçevesi olan Flet kullanılarak ön uç işlevselliği kolaylaştırılmıştır.

Proje, akademik araştırma sohbet yardımcısının yanı sıra PDF, resim, veri ve tablo tanıma gibi çok yönlü yetenekler sunmak üzere tasarlandı. Uygulama, bu özellikleri kullanarak yalnızca iletişimi geliştirmekle kalmaz, aynı zamanda bilgi işleme ve alma için güçlü araçlar da sağlar. Uygulamanın her bileşeni Python kullanılarak titizlikle hazırlanmış olup tutarlı ve uyumlu bir geliştirme ortamı sağlar.

Bu projenin önemli bir yönü, uygulamanın akıllı işlevlerinin omurgasını oluşturan Google Cloud'un önceden eğitilmiş büyük dil modellerinin kullanılmasıdır. Gemini API aracılığıyla erişilen bu modeller, gelişmiş dil anlayışı ve oluşturma yetenekleri sunarak sohbet uygulamasının son derece doğru ve bağlamsal olarak anlamlı yanıtlar sunmasını sağlar.

Bu sohbet uygulamasının uygulanması birkaç temel hedefe dayanmaktadır:

1. Gelişmiş Gerçek Zamanlı İletişim: Hızlı yanıt veren ve etkileşimli bir sohbet deneyimi sağlamak için Gemini API'sinden yararlanılıyor.

2. Kapsamlı Veri Tanıma: Bilgi işlemeyi kolaylaştırmak için PDF'ler, resimler, veriler ve tablolar için gelişmiş tanıma özelliklerinin entegre edilmesi.

3. Akademik Araştırma Yardımı: Akademik araştırmaya yardımcı olacak araçlar geliştirmek, uygulamayı öğrenciler ve araştırmacılar için değerli bir kaynak haline getirmek.

4. Tutarlı Geliştirme Ortamı: Uygulamanın her bölümünde Python kullanılarak tekdüzelik ve bakım kolaylığı sağlanır.

5. Bulut Entegrasyonu: Önceden eğitilmiş dil modellerinin gücünden yararlanmak ve uygulamanın yapay zekâ yeteneklerini geliştirmek için Google Cloud hizmetlerinden yararlanılıyor.

Ön uç kolaylaştırıcı olarak Flet'in seçilmesi, uygulamanın sunduğu çeşitli işlevlerin yönetilmesi için çok önemli olan sağlam ve esnek bir kullanıcı arayüzüne olanak tanır. Flet'in Python ile entegrasyonu, ön uç geliştirmenin arka uç süreçleri kadar akıcı ve verimli olmasını sağlar.

Özetle bu proje, en son teknolojiye sahip yapay zekâ teknolojilerinden ve bulut hizmetlerinden yararlanan, çok yönlü ve güçlü bir gerçek zamanlı sohbet uygulamasının geliştirilmesini sergiliyor. Gemini API, Python ve Google Cloud'un önceden eğitilmiş modellerini entegre eden uygulama, gerçek zamanlı iletişim ve bilgi işleme araçları için yeni bir standart belirliyor. Bu tez, uygulamanın tasarımını, uygulanmasını ve değerlendirilmesini detaylandıracak ve uygulamanın dijital ortamda etkileşim kurma ve bilgiyi yönetme biçimimizde devrim yaratma potansiyelini vurgulayacaktır.

# Literatür

“Python for Data Analytics, Scientific and Technical Applications,” IEEE Conference Publication araştırmasında [1], Bilgisayarların veya makinelerin icadından bu yana, çeşitli görevleri yerine getirme yetenekleri katlanarak arttı. Günümüzde bilgisayar gücünün büyük ölçüde artması, büyük miktarda verinin varlığı ve Veri Analitiği, Yapay Zekâ, Makine alanındaki tekniklerin daha iyi anlaşılması nedeniyle bilgisayar biliminin bir dalı olan veri bilimi ve analitiği yeniden canlanmıştır. Öğrenme, Derin Öğrenme vb. Dolayısıyla teknoloji sektörünün vazgeçilmez bir parçası haline gelmişler ve birçok zorlu sorunun çözümünde kullanılıyorlar. Birçok veri bilimi uygulamasının geliştirilebileceği iyi bir programlama dili arayışında Python, eksiksiz bir programlama çözümü olarak ortaya çıktı. Python'un düşük öğrenme eğrisi ve esnekliği nedeniyle en hızlı büyüyen dillerden biri haline geldi. Python'un sürekli gelişen kitaplıkları, onu Veri analitiği için iyi bir seçim haline getiriyor. Makalede Python programlama dilinin özellikleri ve özellikleri anlatılıyor ve daha sonra Python’un en hızlı büyüyen programlama dillerinden biri olarak kabul edilmesinin ardındaki nedenler ve neden veri bilimi uygulamaları, araştırma ve geliştirmede ön sıralarda yer aldığı tartışılıyor.

A graph of different languages

Description automatically generated

Şekil 1 Bilgisayar Bilimleri bölümlerinin sayısı ve giriş derslerini öğretmek için kullanılan diller

Projemde daha önce, [2] A. Vilkomir, "An Empirical Exploration of Python Machine Learning API Usage." adı altında yayınlanmış makaleden yararlanıp, Python'da web ve masaüstü uygulamaları oluşturmaya yönelik bir çerçeve olan Flet'i kullanarak bir Python API geliştirmek için makine öğrenimi API kullanımına ilişkin araştırmamdan elde edilen bulguları kullandım. Kaggle'daki makine öğrenimi projelerinden oluşan bir derlemenin analizinden elde edilen bilgilerden yararlanarak, özellikle bu alanda yeni olan geliştiriciler için makine öğrenimi işlevlerini uygulamalara entegre etme sürecini basitleştiren bir API oluşturmayı hedefledim.

Araştırmada belirlenen temel zorluklardan biri pandalar ve scikit-learn gibi popüler makine öğrenimi API'lerinin yeni başlayanlar için göz korkutucu olabilen karmaşıklığıydı. Bu sorunu çözmek için kullanıcı dostu olma ve erişilebilirliğe önem veren Flet tabanlı API'yi tasarladım. API, daha karmaşık ayrıntılardan bazılarını soyutlayarak ve net, sezgisel arayüzler sağlayarak, geliştiricilerin ortak makine öğrenimi görevlerini minimum sorunla gerçekleştirmesine olanak tanır. Örneğin API, iş akışını kolaylaştıran ve hata olasılığını azaltan veri ön işleme, model eğitimi ve değerlendirme için önceden yapılandırılmış yöntemler içerir.

Araştırma ayrıca makine öğrenimi projelerinde kullanılan kitaplıkların ortak kalıplarını ve kombinasyonlarını da vurguladı. Bu, en sık kullanılan kitaplıklarla sorunsuz bir şekilde bütünleşmesini sağlayarak API'nin tasarımına bilgi sağladı. Örneğin API, panda veri çerçeveleri ve scikit-learn modelleri için yerleşik destek sağlayarak geliştiricilerin kapsamlı standart kod yazmaya gerek kalmadan verileri kolayca değiştirmesine ve makine öğrenimi algoritmalarını uygulamasına olanak tanır. Ek olarak API, kendisiyle oluşturulan uygulamaların genel kalitesini ve performansını artırmaya yardımcı olan verimli veri işleme ve model doğrulama teknikleri gibi derlem analizinden belirlenen en iyi uygulamaları içerir.

Ayrıca çalışma, kütüphanelerin ithal edildiği ancak hiç kullanılmadığı durumları ortaya çıkardı; bu da daha iyi rehberlik ve belgelendirmeye ihtiyaç duyulduğunu gösteriyor. Yanıt olarak Flet tabanlı API'ye, sağlanan işlevlerin nasıl etkili bir şekilde kullanılacağını gösteren kapsamlı belgeler ve örnekler ekledim. Bu yalnızca yeni programcıların hızlı bir şekilde başlamalarına yardımcı olmakla kalmaz, aynı zamanda çeşitli makine öğrenimi araçlarının doğru kullanımını göstererek iyi kodlama uygulamalarını da teşvik eder.

Araştırmamdan elde ettiğimiz bulguları Flet tabanlı Python API'sinin tasarımına ve geliştirilmesine dahil ederek geliştiriciler için güçlü ancak erişilebilir bir araç yaratmayı hedefledim. Bu API, yalnızca makine öğrenimi uygulamaları oluşturma sürecini basitleştirmekle kalmaz, aynı zamanda temel kavramların ve en iyi uygulamaların daha derinlemesine anlaşılmasını da teşvik ederek, sonuçta makine öğrenimini herkes için daha ulaşılabilir hale getirme yönündeki daha geniş hedefe katkıda bulunur.

Projemizde, sanal asistanlar ve Soru-Cevap (QnA) chatbotları üzerine yaptığımız araştırmalardan [3] elde ettiğimiz bilgileri kullanarak, maliyet etkin ve verimli QnA chatbotlarının oluşturulmasını ve dağıtılmasını kolaylaştırmak için Flet tabanlı bir Python API geliştirdik. Bu girişimin ana motivasyonu, pahalı bulut tabanlı çözümlere bir alternatif sunarak, gelişmiş sanal asistan yeteneklerini sınırlı kaynaklara sahip bireyler ve organizasyonlar için erişilebilir hale getirmekti.

Veri ve müşteri etkileşimlerinin katlanarak artması nedeniyle sanal asistanlara olan talebin arttığını anlayarak, QnA chatbotlarının geliştirilme sürecini basitleştiren bir API oluşturmayı hedefledik. Araştırmamız, sohbet botları, QnA chatbotları ve Etkileşimli Sesli Yanıt Sistemleri (IVRS) gibi çeşitli sanal asistan türlerine dikkat çekti ve bu hizmetlerin Google Cloud, Microsoft Azure ve Amazon Web Services gibi platformlarda barındırılmasının yüksek maliyetlerini vurguladı. Bu araştırmayı kullanarak, bu maliyetleri düşürürken yüksek işlevselliği koruyan bir API geliştirmeyi amaçladık.

Geliştirdiğimiz Flet tabanlı Python API, kullanıcıların QnA chatbotlarını etkin bir şekilde anlayıp yanıt verebilecek şekilde oluşturmalarına olanak tanır. Temelde bir QnA chatbot, insan konuşmasını simüle eder ve kullanıcıların dijital cihazlarla gerçek bir kişiyle iletişim kuruyormuş gibi etkileşimde bulunmalarını sağlar. API'miz hem basit, tek satırlık yanıt veren chatbotları hem de öğrenip evrimleşerek kişiselleştirme seviyesini artıran daha sofistike sanal asistanları destekler.

QnA chatbotlarının işlevselliğinin anahtarı, kullanıcı sorgularını anahtar kelimelere ve bağlama göre anlayabilme yetenekleridir. Bu, birkaç önemli adımı içerir: girdi sorgusunun ön işlenmesi, bir veri seti ile karşılaştırılması, olası yanıtların değerlendirilmesi ve en uygun yanıtın sonlandırılması. API, bu adımları sorunsuz bir şekilde yöneterek chatbotların kullanıcı girdisine dayalı olarak önceden tanımlanmış görevleri yerine getirebilmesini sağlar.

Ön işleme, bu sürecin önemli bir bileşenidir. Araştırmamız, noktalama işaretleri, ekstra boşluklar, tekrarlanan kelimeler, tek karakterler ve özel karakterler gibi gereksiz öğelerin temizlenmesinin ve filtrelenmesinin önemini vurguladı. API'ye sağlam ön işleme teknikleri ekleyerek, kullanıcı sorgularının doğru bir şekilde yorumlanmasını sağladık, bu da daha kesin ve ilgili yanıtlar verilmesine yol açar.

Ön işleme sonrasında, API temizlenmiş sorguyu en iyi eşleşmeyi belirlemek için bir veri seti ile karşılaştırır. Daha sonra olası yanıtları, uygunluk ve doğruluğa göre değerlendirir ve yanıtı sonlandırır. Bu, chatbotun uygun ve yardımcı yanıtlar vermesini sağlar, kullanıcı deneyimini artırır.

Bu araştırma bulgularını Flet tabanlı Python API'ye entegre ederek, geliştiricilere pahalı bulut hizmetlerine ihtiyaç duymadan QnA chatbotları oluşturup dağıtma imkânı tanıyan bir araç yarattık. Bu API, yalnızca gelişmiş sanal asistan teknolojisine erişimi demokratikleştirmekle kalmaz, aynı zamanda geliştiricilerin kullanıcılarının ihtiyaçlarını karşılayan verimli ve duyarlı chatbotlar oluşturmasını kolaylaştırır.

Projemizde, bu araştırmadan [4] yararlanarak bir Flet tabanlı Python API geliştirdik ve bu API'yi, yazılım uygulamaları için alternatif kullanıcı arayüzleri oluşturmak amacıyla kullandık. Yazılım uygulamaları kullanılırken, kullanılabilecek kullanıcı arayüzleri arasında komut satırı, grafiksel kullanıcı arayüzü (GUI), menü tabanlı, form tabanlı, doğal dil gibi seçenekler bulunur. Ana akım kullanıcı arayüzleri GUI ve web tabanlı olsa da zaman zaman alternatif bir kullanıcı arayüzüne ihtiyaç duyulmaktadır. Sohbet botu tabanlı bir konuşma arayüzü bu alanda yerini alır.

Sohbet botları, sohbet platformlarında var olan bir bot sınıfıdır. Kullanıcılar, grafik arayüzler veya widget'lar aracılığıyla onlarla etkileşime girebilir ve trend bu yöndedir. Genellikle durumsal bir hizmet sunarlar, yani uygulama her oturumun verilerini kaydeder. Bir üniversite web sitesinde, kullanıcılar belirli bir tür bilgiye nereden ulaşacaklarını bilemeyebilirler. Öğrenci veya çalışan olmayan biri için bilgi edinmek zor hale gelebilir. Bu sorunu çözmek için bir üniversite sorgu sohbet botu geliştirdik; bu bot, üniversite web sitesinin kullanıcı deneyimini geliştiren ve kullanıcıya etkili bilgi sağlayan hızlı, standart ve bilgilendirici bir widget sunar.

Sohbet botları, yapay zeka (AI) ve doğal dil işleme (NLP) algoritmaları kullanılarak geliştirilen zeki sistemlerdir. Bu botlar, etkili bir kullanıcı arayüzüne sahip olup sınav birimi, kabul, akademik bilgiler, kullanıcıların devam durumu ve not ortalaması, yerleştirme birimi ve diğer çeşitli etkinliklerle ilgili soruları yanıtlar. Araştırmamız, bu tür bir sohbet botu geliştirmek için gerekli olan AI ve NLP algoritmalarının yanı sıra bilgi tabanı, lemmatizasyon, anlamsal cümle benzerliği ve WordNet gibi kavramları içerir.

Bu araştırma bulgularını Flet tabanlı Python API'mize entegre ederek, kullanıcıların yazılım uygulamalarında sohbet botları aracılığıyla etkileşim kurmalarını sağladık. API'miz, sohbet botlarının kolayca oluşturulmasını ve dağıtılmasını mümkün kılarak, üniversite web siteleri gibi platformlarda kullanıcı deneyimini önemli ölçüde iyileştirir. Ayrıca, bilgiye hızlı ve etkili bir şekilde erişim sağlayarak kullanıcı memnuniyetini artırır.

# Materyal ve Yöntem

Bitirme çalışmamızda Python dilini ana programlama dili olarak kullandık, bu projemizin bir özgün tarafı da bütün arayüzü ve back-end mühendisliğini Python diliyle yapmamız sayılır. Bu yüzden Python’a giriş yapıp sonradan Python’daki API oluşumun nasıl olduğunu açıklayarak Flet kütüphanesinden ve diğer kullandığımız kütüphanelerden bahsedeceğiz.

## Python Programlama Dili

Projemizi zorlaştıran ama bir yandan programın hızlı çalışmasını da sağlayan Python dili; yüksek seviyeli bir programlama dilidir ve ilk olarak 1991 yılında Guido van Rossum tarafından geliştirildi. Python’un popülerliğini borçlu olduğu birkaç temel özellik vardır:

* **Kolay Okunabilirlik ve Yazılabilirlik**

Python, okunabilir ve yazılabilir bir sözdizimi ile tasarlanmıştır. Bu, özellikle yeni başlayanlar için öğrenmeyi ve kullanmayı kolaylaştırır. Karmaşık yapılandırmalar ve gereksiz semboller yerine, Python kodu neredeyse doğal bir dilde yazılmış gibidir.[5]

* **Yüksek Seviyeli Dil**

Python, soyutlamaların çoğunu halleder ve programcıların düşük seviyeli ayrıntılarla uğraşmak zorunda kalmadan karmaşık işlemler yapmasını sağlar. Bu, geliştirme sürecini hızlandırır ve yazılım geliştirmeyi daha verimli hale getirir. [6]

* **Dinamik Tip**

Python, dinamik olarak yazılmış bir dildir. Bu, değişkenlerin veri türlerinin önceden belirtilmesine gerek olmadığı anlamına gelir. Değişkenler, çalışma zamanında türlerine göre yorumlanır. [7]

* **Büyük Standart Kütüphane**

Python, geniş ve kapsamlı bir standart kütüphane ile gelir. Bu kütüphane, dosya işlemleri, web hizmetleri, veri işleme, matematiksel hesaplamalar ve daha birçok görev için önceden yazılmış birçok modül ve paket içerir. [8]

* **Çoklu Paradigma Desteği**

Python, çeşitli programlama paradigmalarını destekler. Bu, nesne yönelimli, işlevsel ve prosedürel programlama gibi farklı yaklaşımlar kullanarak yazılım geliştirmeyi mümkün kılar.

* **Platform Bağımsızlık**

Python, platform bağımsız bir dildir. Python kodu, Windows, MacOS, Linux ve diğer birçok işletim sistemi üzerinde çalışabilir. Bu, taşınabilirlik ve esneklik sağlar.

* **Topluluk ve Ekosistem**

Python, büyük ve aktif bir topluluğa sahiptir. Bu, geniş destek ve çok sayıda üçüncü taraf kütüphanesi ve çerçevesi anlamına gelir. İster web geliştirme (Django, Flask, Flet), veri bilimi (Pandas, NumPy, SciPy), makine öğrenimi (TensorFlow, Keras, scikit-learn) olsun, Python ekosistemi geniş ve çeşitlidir.

* **Kolay Entegrasyon**

Python, diğer diller ve platformlarla kolayca entegre edilebilir. C, C++, Java ve .NET gibi dillerle birlikte kullanılabilir. Bu, mevcut sistemlerle entegrasyonu kolaylaştırır ve Python'u çok yönlü bir araç haline getirir.

* **Hızlı Prototipleme**

Python, hızlı prototipleme ve geliştirme için mükemmeldir. Bu, geliştiricilerin fikirlerini hızlı bir şekilde test etmelerini ve üretime geçmeden önce prototipler oluşturmalarını sağlar.

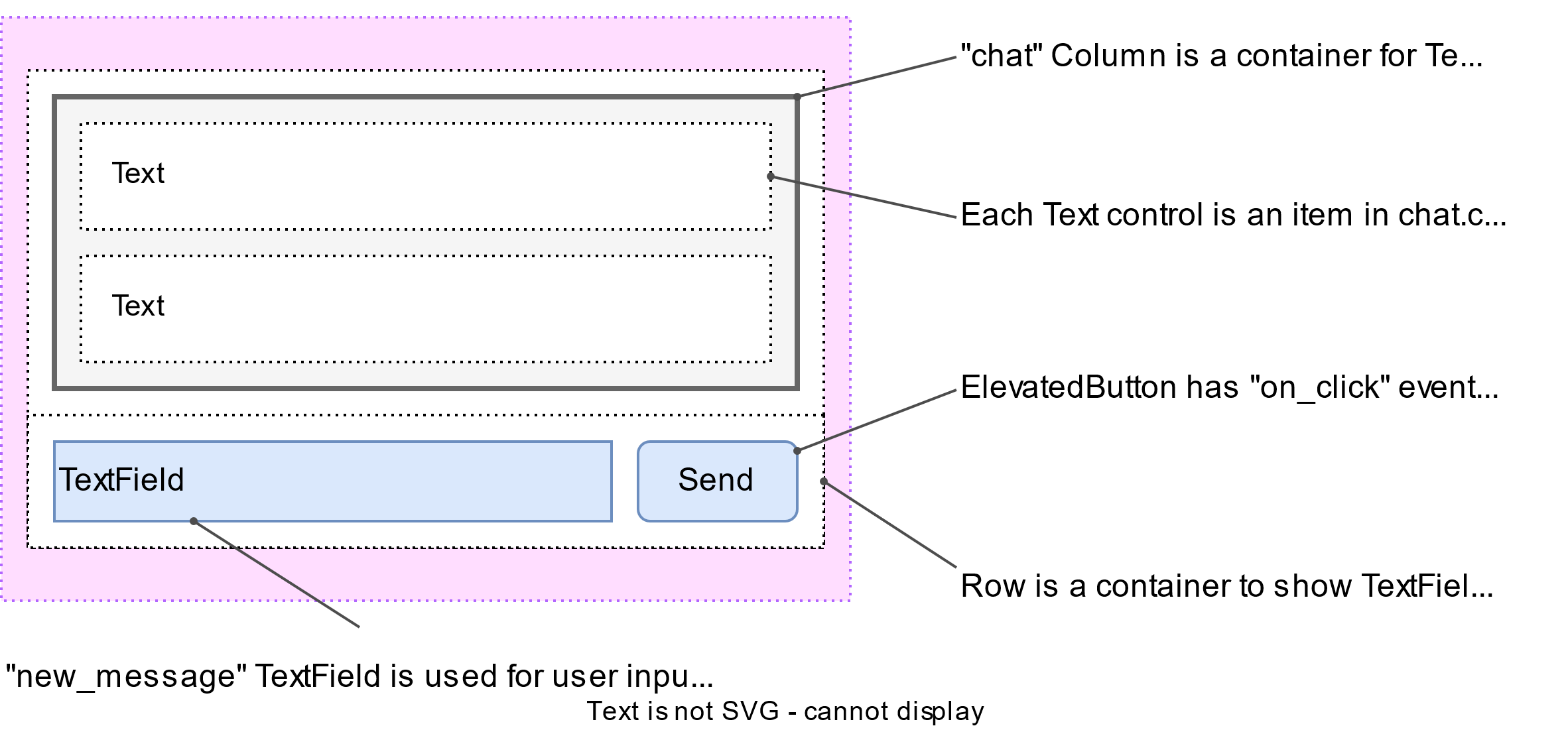
* **Eğitim ve Akademi**

Python, eğitim ve akademik alanlarda da yaygın olarak kullanılır. Basitliği ve okunabilirliği sayesinde, programlama öğrenenler için ideal bir dil olarak kabul edilir. Python’un bu özellikleri, onu hem yeni başlayanlar hem de deneyimli geliştiriciler için popüler bir seçim haline getirmiştir. Geniş uygulama alanları ve güçlü özellikleriyle Python, günümüzde en yaygın kullanılan programlama dillerinden biridir.

## Flet Uygulaması

Flet, ön uç geliştirme konusunda önceden deneyiminiz olmadan Python'da web, masaüstü ve mobil uygulamalar oluşturmaya olanak tanıyan bir çerçevedir.

Google'ın Flutter'ını temel alan Flet kontrolleriyle programımız için bir kullanıcı arayüzü oluşturabiliriz. Flet, Flutter widget'larını sarmanın ötesine geçer. Daha küçük widget'ları birleştirerek, karmaşıklıkları basitleştirerek, en iyi kullanıcı arayüzü uygulamalarını uygulayarak ve mantıklı varsayılanları uygulayarak kendi dokunuşunu katar. Bu, bizim tarafınızdan ek tasarım çabası gerektirmeden uygulamalarınızın şık ve gösterişli görünmesini sağlar.



Şekil 2 Flet Sohbet Uygulamasının Temel Mimarisi

Bu düzeni uygulamak için şu Flet kontrollerini kullanacağız:

* Column- sohbet mesajlarını (Metin kontrolleri) dikey olarak görüntülemek için bir kap.
* Text- sohbet sütununda görüntülenen sohbet mesajı.
* TextField- kullanıcıdan yeni mesaj girişi almak için kullanılan giriş kontrolü.
* ElevatedButton- Sohbet sütununa yeni mesaj ekleyecek "Gönder" düğmesi.
* Row- TextField ve ElevatedButton'u yatay olarak görüntülemek için bir kap.

import flet as ft

def main(page: ft.Page):

    chat = ft.Column()

    new\_message = ft.TextField()

    def send\_click(e):

        chat.controls.append(ft.Text(new\_message.value))

        new\_message.value = ""

        page.update()

    page.add(

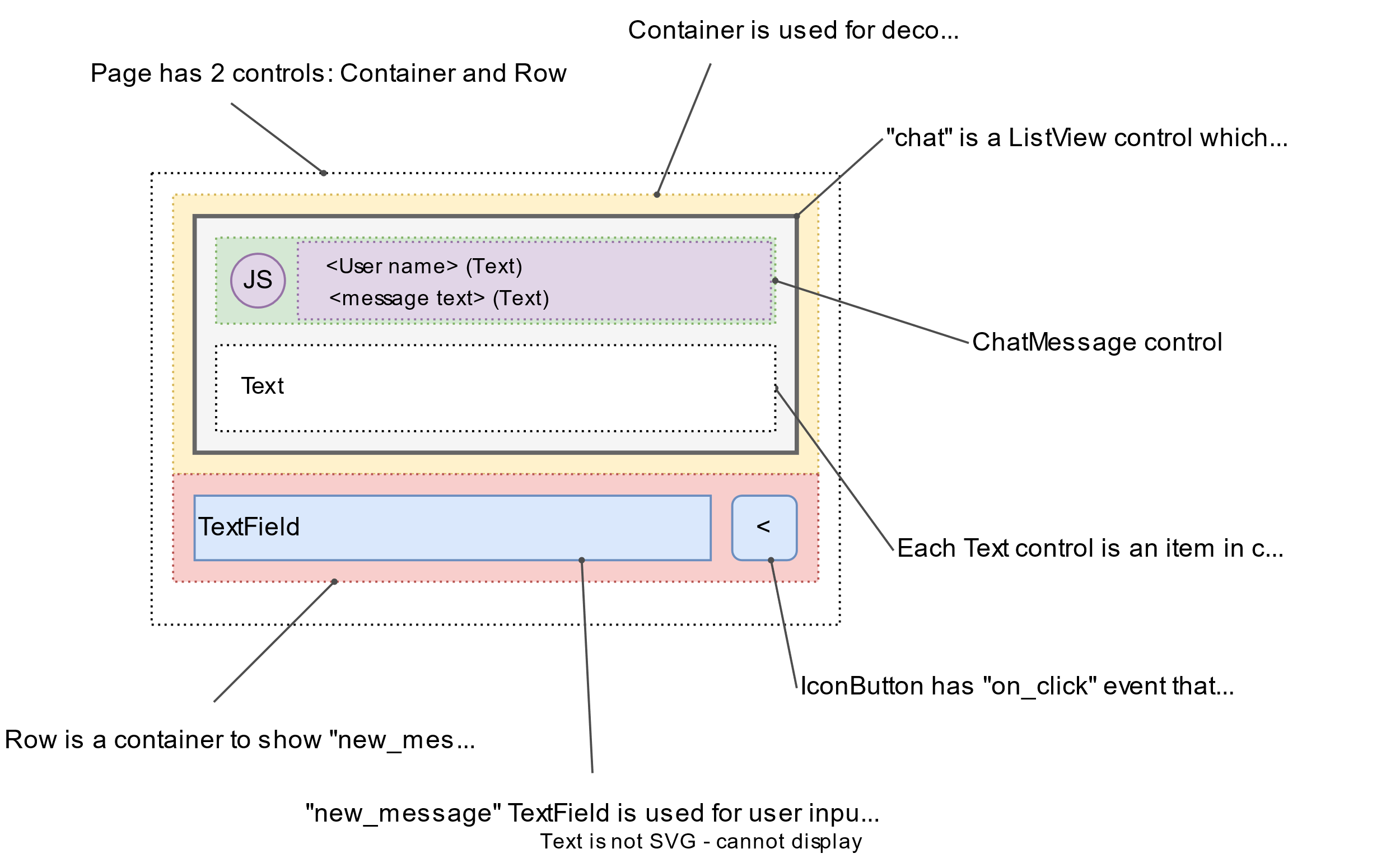
        chat, ft.Row(controls=[new\_message, ft.ElevatedButton("Send", on\_click=send\_click)])

    )

ft.app(main, view=ft.AppView.WEB\_BROWSER)

Kullanıcı "Send" butonuna tıkladığında send\_click yöntemini çağıran on\_click olayını tetikler. send\_click daha sonra Sütun kontrolleri listesine yeni Metin kontrolü ekler ve new\_message TextField değerini temizler.

Şimdi mimarimizi biraz daha geliştirip artık sohbet uygulaması için daha iyi bir düzen oluşturmak amacıyla yepyeni ChatMessage'ı kullanabiliriz: [7]



Şekil 3

On\_message yönteminde düz Metin sohbet mesajları yerine ChatMessage örnekleri oluşturulacaktır:

def on\_message(message: Message):

    if message.message\_type == "chat\_message":

        m = ChatMessage(message)

    elif message.message\_type == "login\_message":

        m = ft.Text(message.text, italic=True, color=ft.colors.BLACK45, size=12)

    chat.controls.append(m)

    page.update()

Yeni düzende önerilen diğer iyileştirmeler şunlardır:

* Mesajları görüntülemek için Column yerine ListView, daha sonra mesajlar arasında gezinebilmek için
* ListView çevresinde kenarlığı görüntülemek için kapsayıcı
* Mesaj göndermek için ElevatedButton yerine IconButton
* Kullanılabilir alanı doldurmak amacıyla kontroller için genişletme özelliğinin kullanılması

Bu düzeni şu şekilde uygulayabiliriz:

# Sohbet mesajları listesi

chat = ft.ListView(

    expand=True,  # ListView'in mevcut alanı doldurması

    spacing=10,  # ListView'deki öğeler arasındaki boşluk

    auto\_scroll=True,  # Yeni mesaj eklendiğinde otomatik olarak en alta kaydırma

)

# Yeni mesaj girişi formu

new\_message = ft.TextField(

    hint\_text="Bir mesaj yaz...",  # Giriş alanı boşken görünen yer tutucu metin

    autofocus=True,  # Sayfa yüklendiğinde giriş alanına otomatik odaklanma

    shift\_enter=True,  # Shift+Enter tuşlarına basarak yeni satır oluşturma

    min\_lines=1,  # Giriş alanında minimum görünen satır sayısı

    max\_lines=5,  # Giriş alanında maksimum görünen satır sayısı

    filled=True,  # Giriş alanının arka planını doldurma

    expand=True,  # Giriş alanının yatayda mevcut alanı doldurması

    on\_submit=send\_message\_click,  # Kullanıcı metni gönderdiğinde çağrılacak fonksiyon

)

# Her şeyi sayfaya ekle

page.add(

    ft.Container(

        content=chat,  # Container'ın içeriği olarak chat ListView

        border=ft.border.all(1, ft.colors.OUTLINE),  # 1 birim genişliğinde ve OUTLINE renginde kenarlık ekleme

        border\_radius=5,  # Kenar köşelerini yuvarlama

        padding=10,  # Container içinde 10 birim dolgu

        expand=True,  # Container'ın mevcut alanı doldurması

    ),

    ft.Row(

        [

            new\_message,  # Giriş alanı

            ft.IconButton(

                icon=ft.icons.SEND\_ROUNDED,  # Gönderme simgesi

                tooltip="Mesaj gönder",  # Fareyle üzerine gelindiğinde görünen bilgi metni

                on\_click=send\_message\_click,  # Butona tıklandığında çağrılacak fonksiyon

            ),

        ]

    ),

)

Son olarak basit bir Flet sohbet uygulamasını elde etmiş oluruz aşağıdaki şekil gibi:

A screenshot of a computer

Description automatically generated

Şekil 4 Basit bir Flet sohbet penceresi

### Uygulamamızı Geliştirirken Kullanılan Kütüphaneler

- `pathlib` modülü, Python 3.4 ile tanıtılmış ve dosya ve dizin yolları ile çalışmayı daha kolay ve sezgisel hale getiren bir kütüphanedir. `pathlib` modülü, nesne yönelimli bir yaklaşım kullanarak dosya sistemi yollarını temsil eder ve yönetir. Bu modül, platformdan bağımsız olarak çalışmakta ve dosya yollarını işlemek için daha okunabilir ve bakımı kolay bir yöntem sunmaktadır.

- `google.generativeai` ve `google.ai.generativelanguage` kütüphaneleri, Google tarafından sağlanan yapay zeka ve doğal dil işleme (NLP) araçlarını içermektedir. Bu kütüphaneler, metin üretme, dil modeli oluşturma ve diğer generatif yapay zeka uygulamaları için kullanılabilir. Özellikle dil modelleri ve doğal dil üretimi konularında gelişmiş fonksiyonlar sağlarlar. Bu araçlar, araştırma ve endüstriyel uygulamalarda doğal dil işlemi için kullanışlıdır.[9]

A graph of a company

Description automatically generated

Şekil 5 AI konusunda Google'in önderliği

A graph of a training cost and compute model

Description automatically generated

Şekil 6 AI Modelleri Eğitmek İçin Tahmini Maliyet

- `cv2`, açık kaynaklı bir bilgisayarla görme kütüphanesi olan OpenCV'nin bir Python bağlamasıdır. OpenCV, bilgisayarla görme, görüntü işleme ve makine öğrenimi konularında geniş bir yelpazede fonksiyonlar sunar. `cv2` ile görüntüleri yükleme, işleme, dönüştürme ve analiz etme gibi işlemler yapılabilir. OpenCV, gerçek zamanlı uygulamalar için optimize edilmiş ve yaygın olarak kullanılan bir kütüphanedir.

- `os` modülü, Python'un standart kütüphanesinde bulunan ve işletim sistemi ile etkileşim kurmayı sağlayan bir modüldür. `os` modülü, dosya ve dizin işlemleri, çevre değişkenleri, işlem yönetimi ve diğer işletim sistemi görevleri için çeşitli fonksiyonlar sağlar. Bu modül, platformdan bağımsız bir şekilde çalışır ve Python programlarının işletim sistemi düzeyinde işlevsellik kazanmasını sağlar.

- `reportlab` kütüphanesi, PDF dosyaları oluşturmak için kullanılan güçlü bir araçtır. `reportlab.lib.pagesizes`, PDF sayfa boyutlarını tanımlamak için kullanılan sabitler içerir, örneğin `letter` gibi standart sayfa boyutları. `reportlab.pdfgen.canvas` ise PDF dosyaları üzerinde çizim yapmayı sağlar. `canvas` sınıfı, PDF üzerinde metin, şekil, resim gibi öğeleri yerleştirme ve düzenleme işlevleri sunar. Bu kütüphane, PDF belgeleri oluşturma ve manipüle etme konusunda oldukça esnek ve güçlü bir çözümdür.

- `reportlab.pdfbase.pdfmetrics`, PDF belgelerinde kullanılan yazı tipleri ve metin metriklerini yönetmek için kullanılır. `reportlab.pdfbase.ttfonts.TTFont` sınıfı ise TrueType yazı tiplerini PDF belgelerinde kullanmak için kaydetmeye olanak tanır. Bu araçlar, PDF dosyalarında özelleştirilmiş yazı tipleri kullanarak estetik ve okunabilir belgeler oluşturmayı sağlar.

- `PyPDF2`, Python ile PDF dosyalarını okumak, yazmak ve manipüle etmek için kullanılan bir kütüphanedir. Bu kütüphane, PDF dosyalarını bölme, birleştirme, sayfa ekleme ve çıkarma, metin çıkarma ve şifreleme gibi çeşitli işlemleri gerçekleştirmeyi sağlar. PyPDF2, PDF belgeleri ile çalışırken ihtiyaç duyulan birçok temel işlevi sağlar ve kullanıcıların PDF belgelerini programatik olarak işlemesine olanak tanır.

- `numpy`, Python programlama dilinde bilimsel hesaplamalar için temel bir kütüphanedir. `numpy`, yüksek performanslı çok boyutlu dizi ve matris işlemleri için veri yapıları sağlar ve bu veri yapıları üzerinde matematiksel ve mantıksal işlemleri gerçekleştirmek için geniş bir fonksiyon yelpazesi sunar. `numpy`, özellikle sayısal hesaplamalar ve veri analizi ile ilgili alanlarda yaygın olarak kullanılır ve diğer birçok bilimsel kütüphane için temel oluşturur.

Bu kütüphaneler, Python programlamasında geniş bir uygulama yelpazesi için gerekli olan çeşitli işlevsellikleri sağlar ve birçok farklı alanda yazılım geliştirme sürecini destekler.

## Uygulamanın Ana Menüsü

Uygulamamızın ilk açıldığında bize sunduğu bir ana menüsü de mevcuttur, bu kısımda uygulamada yapılabilen bütün işlemlerin kılavuzu olarak geçiyor. Bu mimaride Resim Analizi, PDF Yükleme, Kamera ve Excel Analizi yer alıyor.

A screenshot of a computer

Description automatically generated

Şekil 7 Ana Menü

### 3.3.1 Resim Analizi

Bu bölümde herhangi bir resim yüklendiğinde gereken analizleri yapılıyor, örnek olarak şekil tespiti, plaka ve barkod tespiti, eşya tespiti ve resimdeki objeler hakkında yorumlamayı da gerçekleştiriliyor. Artı olarak akademisyenler için kolaylık sağlayan sınav kağıttaki kırmızı renge sahip olan sayıları toplama ve tespit etme gibi bir özgün yanı eklemiş olduk.

Bir resimde renk tespiti yapılmak istenildiğinde yapay zekâ, yazı renkleri algılarken doğru çıktı verememektedir. Bu durum, farklı renklerde sayıları karışık olarak yazıp daha sonrasında spesifik bir renk belirtip sonuç istenerek test edilmiştir. Yapay zekâ bu isteğe yanlış cevap vermiştir. Sorunu ortadan kaldırmak için yapay zekaya resim göndermeden önce bir ön işlemden geçirilmiştir.

Rengin Tespit Edilmesi

İlgili Fonksiyonu Oluşturma:

def process\_image(image\_path, color=None):

    if color is None or color == "original":

        return image\_path

Görüntü Yükleme : cv2.imread(image\_path) ile belirtilen yoldan görüntü yüklenir.

    image = cv2.imread(image\_path)

    if image is None:

        print(f"Görüntü yüklenemedi. Yol: {image\_path}")

        return None

HSV Dönüşümü: cv2.cvtColor(image, cv2.COLOR\_BGR2HSV) ile RGB renk uzayındaki görüntü, HSV renk uzayına dönüştürülür. HSV (Hue, Saturation, Value) renk uzayı, renkleri ton, doygunluk ve parlaklık olarak temsil eder. Renk tespiti için daha uygun bir yapı sağlar.

    hsv\_image = cv2.cvtColor(image, cv2.COLOR\_BGR2HSV)

Maske Oluşturma: cv2.inRange() fonksiyonu, belirtilen renk aralığına uyan pikseller için bir maske oluşturur.

1. Kırmızı için: Kırmızı renk için iki ayrı aralık kullanılır çünkü kırmızı renk spektrumunun iki ucunda yer alır.
2. Yeşil ve Mavi için: Belirli renk aralıkları kullanarak yeşil ve mavi renklerin maskeleri oluşturulur.

    if color == "red":

        lower\_color1 = np.array([0, 120, 70])

        upper\_color1 = np.array([10, 255, 255])

        lower\_color2 = np.array([170, 120, 70])

        upper\_color2 = np.array([180, 255, 255])

        mask1 = cv2.inRange(hsv\_image, lower\_color1, upper\_color1)

        mask2 = cv2.inRange(hsv\_image, lower\_color2, upper\_color2)

        mask = mask1 + mask2

    elif color == "green":

        lower\_color = np.array([40, 40, 40])

        upper\_color = np.array([70, 255, 255])

        mask = cv2.inRange(hsv\_image, lower\_color, upper\_color)

    elif color == "blue":

        lower\_color = np.array([100, 150, 0])

        upper\_color = np.array([140, 255, 255])

        mask = cv2.inRange(hsv\_image, lower\_color, upper\_color)

    else:

        print(f"Geçersiz renk seçildi: {color}")

        return None

**Rengi Tespit Etme:** cv2.bitwise\_and() ile orijinal görüntü, maskelenmiş görüntüyle birleştirilir. Böylece sadece belirtilen renk aralığındaki pikseller elde edilir.

    color\_detection = cv2.bitwise\_and(image, image, mask=mask)

    processed\_path = os.path.splitext(image\_path)[0] + \

        f"\_{color}\_processed.jpg"

**İşlenmiş Görüntüyü Kaydetme:** İşlenen görüntü, cv2.imwrite() ile belirtilen yola kaydedilir

    cv2.imwrite(processed\_path, color\_detection

return processed\_path

Bu ön işlem adımları sayesinde bir görüntüdeki kırmızı, yeşil ve mavi renkler tespit edilebilmektedir. Öncelikle kullanıcının belirlediği bir görüntü alınır ve tespit edilmesi istenilen renk algılandıktan sonra algılanan renk dışındaki kısımlar siyah olarak gösterilir. Böylece algılanan renk ön plana çıkarılarak yapay zekanın daha rahat inceleme yapması ve karar vermesi sağlanır.

Renk Tespit Arayüzü

Renk tespit sisteminin çalışabilmesi için Flet arayüzüyle entegresi sağlanmıştır. Bu bütünleşmiş, programın “Resim” menüsünde gerçekleşmektedir.

“Resim” menüsüne giriş yapıldığında resim yüklemek için kullanılan buton farklı seçeneklerle çalışabilmektedir. Bu kısımda tercihe göre alınan resmin orijinal olması da seçilebilir, sadece farklı renklerin algılanması da. Bu seçim yapıldıktan sonra dosya yolundan seçilen resim istenilen şekilde program ekranına yüklenmektedir.

metin, ekran görüntüsü, yazılım, ekran, görüntüleme içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şekil 8 Renk Seçim Menüsü

Renk Seçme Butonu:

detect\_red\_button = ft.IconButton(

    icon=ft.icons.COLORIZE,

    tooltip="Renk Seç",

    selected=False,

    icon\_color=ft.colors.RED,

    on\_click=choose\_color,

)

**detect\_red\_button:**Bu, renk seçim diyaloğunu açmak için kullanılan bir butondur. Flet kütüphanesi ile bir IconButton olarak tanımlanıyor.

**icon=ft.icons.COLORIZE:** Butona renk simgesi ekleniyor.

**tooltip="Renk Seç":** Fare imleci butonun üstünde durduğunda belirecek açıklama metni.

**selected=False:** Butonun varsayılan olarak seçili olmaması için.

**icon\_color=ft.colors.RED:** Buton simgesinin rengi kırmızı.

**on\_click=choose\_color:** Butona tıklandığında çalışacak fonksiyon. Bu durumda choose\_color fonksiyonu çağrılıyor.

Renk Seçme Diyaloğu

color\_dialog = ft.AlertDialog(

    title=ft.Text("Renk Seçin"),

    content=ft.Row(

        [

            ft.Container(

                content=ft.ElevatedButton(

                    "Orijinal",

                    data="original",

                    on\_click=color\_selected,

                    width=100,

                    height=100

                ),

                padding=10

            ),

            # ... Diğer renk seçenekleri ...

        ],

        alignment=ft.MainAxisAlignment.SPACE\_AROUND

    ),

)

**color\_dialog:** Renk seçim seçeneklerinin yer aldığı bir diyalog kutusu. Flet kütüphanesi ile bir AlertDialog olarak tanımlanıyor.

**title=ft.Text("Renk Seçin"):** Diyaloğun başlık metni.

**content=ft.Row(...):** Diyalog içindeki içerikleri düzenlemek için bir satır oluşturuluyor.

**ft.Container(...):** Butonları düzenlemek için konteynerlar oluşturuluyor.

**ft.ElevatedButton(...):** Renk seçim seçenekleri olarak butonlar oluşturuluyor. Her butonun data özniteliği, seçilen rengi temsil ediyor ("original", "red", "green", "blue"). on\_click=color\_selected ile butona tıklandığında color\_selected fonksiyonunu çalıştırıyor.

Renk Seçim Fonksiyonu

def choose\_color(e):

    page.dialog = color\_dialog

    color\_dialog.open = True

    page.update()

**color\_selected(e):** Renk seçim diyaloğunda bir butona tıklandığında çağrılan fonksiyon.

**selected\_color = e.control.data:** Seçilen rengi selected\_color değişkenine atıyor.

**color\_dialog.open = False:** Diyalog kutusunu kapatıyor.

**apply\_color\_to\_image():** Seçilen renge göre görüntüyü işlemek için fonksiyonu çağırıyor.

**page.update():** Arayüzü güncelliyor.

Görüntü İşleme Fonksiyonu

def apply\_color\_to\_image():

    global processed\_image\_path

    if image\_path:

        if selected\_color == "original":

            processed\_image\_path = image\_path

        else:

            processed\_image\_path = process\_image(

                image\_path, selected\_color)

        if processed\_image\_path is not None:

            page.pubsub.send\_all(

                Message("Me", processed\_image\_path, message\_type="chat\_message", is\_image=True))

**apply\_color\_to\_image():** Renk seçildiğinde çağrılan ve görüntüyü işleyen fonksiyon.

**processed\_image\_path = process\_image(image\_path, selected\_color):** process\_image() fonksiyonunu kullanarak seçilen renge göre görüntüyü işliyor ve yeni resmin yolunu processed\_image\_path değişkenine atıyor.

Uygulamamızı daha özgün bir hale getirme sürecinde, her bir resim yüklediğimizde Prompt Mühendisliği butonlarını da eklemek büyük bir rol oynamaktadır. Şöyle ki, kodumuzda uzun bir prompt, örneğin akademik bir çalışma için gereken özellikler ve istediğimiz kaynaklardan yararlanmasını istediğimiz bir şekilde bu promptu yazmadan, sadece bir butona tıklayarak erişebiliyoruz.



Şekil 9 Üç Nokta Şekli Buton

A screenshot of a computer

Description automatically generated

Şekil 10 Prompt Mühendisliği Yapılmış Butonlar

### 3.3.2 PDF Analizi

import PyPDF2  # PDF işleme için PyPDF2 kütüphanesini ekleyin

# ... diğer importlar ...

# ... (Diğer sınıf ve fonksiyon tanımları) ...

    def save\_last\_response\_to\_pdf(pdf\_path): # Son cevabı PDF'ye kaydetme fonksiyonu

        if last\_response\_text:

            c = canvas.Canvas(pdf\_path, pagesize=letter)

            width, height = letter

            c.setFont('TNR', 12)

            text\_y = height - 72

            lines = last\_response\_text.split('\n')

            for line in lines:

                for i in range(0, len(line), 100):

                    c.drawString(

                        36, text\_y, line[i:i+100])

                    text\_y -= 14

                    if text\_y < 36:

                        c.showPage()

                        text\_y = height - 72

            c.save()

    def save\_pdf(e): # PDF kaydetme diyaloğunu açma fonksiyonu

        save\_file\_dialog.save\_file()

    def on\_message(message: Message):

        # ... (diğer kod) ...

    def extract\_text\_from\_pdf(pdf\_path): # PDF'den metin çıkarma fonksiyonu

        with open(pdf\_path, 'rb') as f:

            pdf\_reader = PyPDF2.PdfReader(f)

            num\_pages = len(pdf\_reader.pages)

            text = ""

            for i in range(num\_pages):

                page = pdf\_reader.pages[i]

                text += page.extract\_text()

            return text

    def pick\_pdf(e): # PDF seçme diyaloğunu açma fonksiyonu

        pdf\_file\_dialog.pick\_files()

    # ... (diğer kod) ...

Bu kod parçaları, PDF dosyalarıyla ilgili işlemleri ele almaktadır:

* save\_last\_response\_to\_pdf(pdf\_path): Bu fonksiyon, last\_response\_text değişkeninde saklanan son cevabı belirtilen pdf\_path yoluna bir PDF dosyası olarak kaydeder.
* save\_pdf(e): Bu fonksiyon, kullanıcının cihazında bir PDF dosyası kaydetmek için bir dosya kaydetme diyaloğu açar.
* extract\_text\_from\_pdf(pdf\_path): Bu fonksiyon, belirtilen pdf\_path yolundaki PDF dosyasından metni çıkarır ve çıkarılan metni döndürür.
* pick\_pdf(e): Bu fonksiyon, kullanıcının cihazından bir PDF dosyası seçmek için bir dosya seçme diyaloğu açar.

Normalde kullanılan kütüphane Türkçe karakterleri algılayamıyor, bu fonksiyonun vasıtasıyla karakterleri uygun hale getirip kaydedilmeye hazırlıyor.

Aşağıda PDF okuma bir PDF’e dönüştürme komutların nasıl çalıştığını görsel olarak görebiliyoruz:

A screenshot of a computer

Description automatically generated

Şekil 11 PDF analizin arayüzü

A screenshot of a computer

Description automatically generated

Şekil 12 PDF Dosyanın Yükleme Sistemi

Kodunuzda PDF özetleme süreci şu şekilde işliyor:

1. **PDF Seçimi:** Kullanıcı arayüzünde "PDF Seç" butonuna tıklandığında pick\_pdf fonksiyonu çağrılır. Bu fonksiyon, pdf\_file\_dialog adlı bir dosya seçme diyaloğunu açar ve kullanıcının cihazında bir PDF dosyası seçmesini sağlar.
2. **PDF'den Metin Çıkarımı:** Kullanıcı bir PDF dosyası seçtiğinde, seçilen dosyanın yolu pick\_pdf\_result fonksiyonuna iletilir. Bu fonksiyon, extract\_text\_from\_pdf fonksiyonunu kullanarak seçilen PDF dosyasından metni çıkarır. extract\_text\_from\_pdf fonksiyonu, PyPDF2 kütüphanesini kullanarak PDF dosyasını sayfa sayfa işler ve her sayfadaki metni çıkararak tek bir metin dizesinde birleştirir.
3. **Özetleme İsteği:** Çıkarılan metin, kullanıcı arayüzündeki metin giriş alanına (new\_message) yerleştirilir ve sonuna "\n\nBu yazıyı özetle." ifadesi eklenir. Bu, kullanıcıya özetleme isteğini onaylama ve gerekirse düzenleme imkanı sağlar. Ardından send\_message\_click fonksiyonu otomatik olarak çağrılarak özetleme isteği gönderilir.
4. **Yapay Zeka İle Özetleme:** send\_message\_click fonksiyonu, kullanıcı tarafından girilen metni (bu durumda PDF'den çıkarılan metin ve özetleme isteği) alır ve Google Gemini API'sine gönderir. Gemini API, metni analiz eder ve bir özet oluşturur.
5. **Özetin Gösterimi:** Oluşturulan özet, kullanıcı arayüzünde "AI" kullanıcısı tarafından gönderilmiş bir mesaj olarak görüntülenir. Kullanıcı, özeti okuyabilir ve gerekirse kaydedebilir.

Kısacası, kodunuz PDF dosyasını yükler, metnini çıkarır, bu metni özetlemek için Google Gemini API'sini kullanır ve sonucu kullanıcıya gösterir.

Ayrıca bahsedilmesi gerekir ki bu fonksiyonu kullanarak uygulamamız bütün diğer bölümlerin sohbette verilen son çıktıyı PDF’e dönüştürebiliyor.

Ayrıca PDF analizi için de bir Prompt Mühendisliği yapılmış menümüz de mevcut ki yorumlama, özetleme, güncelleştirme ve erkana yazma özellikleri bize özgün olan prompt komutlarla gerçekleştiriyor.

A screenshot of a computer

Description automatically generated

Şekil 13 PDF Analizi İçin Prompt Butonları

### 3.3.3 Kamera

Kamera açma, Fotoğraf çekme ve kaydetme işlemleri gerçekleştiren kodlar bu şekilde:

import flet as ft

import cv2 # OpenCV kütüphanesini import ediyoruz

# ... (diğer importlar ve kod blokları) ...

camera\_file\_dialog = ft.FilePicker(on\_result=camera\_file\_result) # Dosya seçme diyaloğu oluştur

page.overlay.append(camera\_file\_dialog) # Diyaloğu arayüze ekle

cap = None # Kamera nesnesi için global bir değişken tanımlıyoruz, başlangıçta boş

def open\_camera(e):

    """

    Kamerayı açar ve canlı görüntü akışını başlatır.

    'q' tuşuna basıldığında akış durur ve kamera kapanır.

    """

    global cap

    cap = cv2.VideoCapture(0)  # Varsayılan kamera cihazını (genellikle 0) aç

    if not cap.isOpened():

        print("Kamera açılamadı")

        return

    while True:

        ret, frame = cap.read()  # Kameradan bir görüntü karesi oku

        if not ret:

            print("Kare alınamadı. Akış sonlandı mı? Çıkılıyor...")

            break

        cv2.imshow('frame', frame)  # Okunan kareyi 'frame' penceresinde göster

        if cv2.waitKey(1) & 0xFF == ord('q'): # 1 milisaniye bekleyip 'q' tuşuna basılırsa

            break # Döngüyü sonlandır

    cap.release()  # Kamera kaynağını serbest bırak

    cv2.destroyAllWindows()  # Açık olan tüm OpenCV pencerelerini kapat

def capture\_image(path):

    """

    Açık olan kameradan bir görüntü karesi yakalar ve belirtilen yola kaydeder.

    """

    global cap, image\_path, processed\_image\_path, selected\_color

    if cap and cap.isOpened():

        ret, frame = cap.read()  # Kameradan bir kare oku

        if ret:

            cv2.imwrite(path, frame) # Okunan kareyi belirtilen yola kaydet

            image\_path = path

            processed\_image\_path = process\_image(

                path, selected\_color)

            page.pubsub.send\_all(

                Message("Me", processed\_image\_path, message\_type="chat\_message", is\_image=True))

        else:

            print("Görüntü yakalanamadı")

def capture\_image\_button(e):

    """

    Fotoğraf çekme ve kaydetme işlemini başlatır.

    Kullanıcıdan kaydedilecek dosyanın adını ve konumunu ister.

    """

    camera\_file\_dialog.save\_file() # Dosya kaydetme diyaloğunu aç

# ... (diğer kod blokları) ...

**cv2 Kütüphanesini İçe Aktarma:**

1. import cv2

Bu kod satırı, OpenCV (Open Source Computer Vision Library) kütüphanesini içe aktarır. OpenCV, bilgisayar görüşü görevlerini gerçekleştirmek için kullanılan popüler bir kütüphanedir. Kamera işlevi için temel işlevleri sağlar.

**cap Değişkenini Tanımlama:**

1. cap = None

cap değişkeni, OpenCV'nin VideoCapture nesnesini depolamak için kullanılır. VideoCapture nesnesi, kamerayla etkileşim kurmanıza izin verir.

**open\_camera Fonksiyonu:**

1. def open\_camera(e):
2. global cap
3. cap = cv2.VideoCapture(0)
4. if not cap.isOpened():
5. print("Cannot open camera")
6. return
7. while True:
8. ret, frame = cap.read()
9. if not ret:
10. print("Can't receive frame (stream end?). Exiting ...")
11. break
12. cv2.imshow('frame', frame)
13. if cv2.waitKey(1) & 0xFF == ord('x'):
14. break
15. cap.release()
16. cv2.destroyAllWindows()

**cap = cv2.VideoCapture(0):** Bu satır, varsayılan kamera (genellikle 0 numaralı kamera) ile bir VideoCapture nesnesi oluşturur.

**if not cap.isOpened()::** Bu kontrol, kameranın başarılı bir şekilde açılıp açılmadığını kontrol eder. Açılmamışsa, hata mesajı basılır ve fonksiyon sonlandırılır.

**while True::** Bu döngü, kameradan sürekli olarak kare okur.

**ret, frame = cap.read():** Bu satır, kameradan bir kare okur. ret, kareyi okuma işleminin başarılı olup olmadığını gösterir. frame, okunan kareyi içeren bir NumPy dizisidir.

**cv2.imshow('frame', frame):** Bu satır, okunan kareyi "frame" adında bir pencerede görüntüler.

**if cv2.waitKey(1) & 0xFF == ord('x')::** Bu satır, kullanıcı klavyeden 'x' tuşuna basana kadar bekler. Kullanıcı 'x' tuşuna bastığında döngüden çıkar.

**cap.release():** Bu satır, kameranın kaynaklarını serbest bırakır.

**cv2.destroyAllWindows():** Bu satır, tüm OpenCV pencerelerini kapatır.

**capture\_image Fonksiyonu:**

1. def capture\_image(path):
2. global cap, image\_path, processed\_image\_path, selected\_color
3. if cap and cap.isOpened():
4. ret, frame = cap.read()
5. if ret:
6. cv2.imwrite(path, frame)
7. image\_path = path
8. processed\_image\_path = process\_image(
9. path, selected\_color)
10. page.pubsub.send\_all(
11. Message("Me", processed\_image\_path, message\_type="chat\_message", is\_image=True))
12. else:
13. print("Failed to capture image")

**if cap and cap.isOpened()::** Bu kontrol, kameranın açık olup olmadığını kontrol eder.

**ret, frame = cap.read():** Bu satır, kameradan bir kare okur.

**cv2.imwrite(path, frame):** Bu satır, okunan kareyi belirtilen path yoluna kaydeder.

**image\_path = path:** image\_path değişkeni, kaydedilen görüntünün yolunu depolar.

**processed\_image\_path = process\_image(path, selected\_color):** Bu satır, process\_image fonksiyonunu çağırır ve kaydedilen görüntüyü işler. process\_image fonksiyonu, görüntüyü bir renk filtresinden geçirir (bu durumda selected\_color tarafından belirtilen renge göre filtreler).

**page.pubsub.send\_all(Message("Me", processed\_image\_path, message\_type="chat\_message", is\_image=True)):** Bu satır, işlenmiş görüntüyü sohbet arayüzüne gönderir.

**capture\_image\_button Fonksiyonu:**

1. def capture\_image\_button(e):
2. camera\_file\_dialog.save\_file()

Bu fonksiyon, kullanıcı kamera düğmesine tıkladığında çağrılır.

**camera\_file\_dialog.save\_file():** Bu satır, görüntüyü kaydetmek için bir dosya seçme diyaloğunu açar. Kullanıcı bir dosya seçtiğinde capture\_image fonksiyonu çağrılır ve görüntü o dosyaya kaydedilir.

Bu kod parçası, kamera açma, fotoğraf çekme ve bu fotoğrafı belirtilen bir yola kaydetme işlemlerini yönetir. open\_camera fonksiyonu kamerayı açar ve kareleri görüntüler, capture\_image fonksiyonu fotoğraf çeker ve kaydeder, capture\_image\_button fonksiyonu ise dosya seçme diyaloğunu başlatarak kullanıcının fotoğrafı kaydedebileceği yeri seçmesini sağlar.

A screenshot of a computer

Description automatically generated

Şekil 14 Kamera Analiz Penceresi

Gördüğümüz üzere, Şekil 9. açılışı ve fotoğraf çekme süreci özet olarak göstermektedir. Tabii ki bu sohbet sayfamızda da görüntüyü analiz edip daha sonra sohbetteki geçen bütün diyaloğu kaydedebiliriz.

Tezekkür etmeliyiz ki Flut Python ile direkt bir kamera bağlantısı kuramıyor ve biz ayrıca fonksiyonları tanıtarak bu işi başarmış olduk ki projemizin özgünlüğüne değer katmaktadır.

### 3.3.4 Excel Analizi

Bu bölümde, uygulamamızda bir Excel dosyasını yükleyip veri analizi yapabilmeyi sağlayacak bir program geliştirmek amacıyla kullandığımız kütüphaneler ve fonksiyonlarla, ilk olarak Excel dosyasını Python’un okuyabileceği bir hale getiriyoruz ve ardından yapay zeka modelimize aktararak veri analizi yapıyoruz.

**1. Gerekli Kütüphaneleri İçe Aktarma:**

import pandas as pd

Bu satır, Pandas kütüphanesini içe aktarır. Pandas, Python'da veri analizi ve manipülasyonu için kullanılabilen güçlü bir kütüphanedir. Özellikle, Excel dosyalarını işlemek için kullanacağız.

**2. excel\_path, excel\_data Değişkenlerini Tanımlama:**

excel\_path = None

excel\_data = None

* excel\_path değişkeni, seçilen Excel dosyasının yolunu depolamak için kullanılır.
* excel\_data değişkeni, Pandas DataFrame'inde yüklenen Excel dosyasındaki verileri depolamak için kullanılır.

**3. pick\_excel\_result Fonksiyonu:**

def pick\_excel\_result(e: ft.FilePickerResultEvent):

    global excel\_path, excel\_data

    excel\_path = None

    excel\_data = None

    if e.files is not None:

        excel\_path = e.files[0].path

        try:

            excel\_data = pd.read\_excel(excel\_path)

            page.pubsub.send\_all(

                Message("AI", "Dosya analiz ediliyor...", message\_type="chat\_message"))

            analyze\_excel\_data(excel\_data)

        except Exception as err:

            page.pubsub.send\_all(

                Message("AI", f"Hata: {str(err)}", message\_type="chat\_message"))

* Bu fonksiyon, kullanıcı Excel dosyası seçtiğinde çağrılır.
* **excel\_path = e.files[0].path:** Seçilen Excel dosyasının yolu excel\_path değişkenine kaydedilir.
* **excel\_data = pd.read\_excel(excel\_path):** Bu satır, Pandas'ın read\_excel fonksiyonunu kullanarak Excel dosyasını bir DataFrame'e yükler.
* **try...except Bloğu:** Bu blok, Excel dosyasının yüklenmesi sırasında oluşabilecek hataları ele alır.
* **analyze\_excel\_data(excel\_data):** Yüklenen Excel verileri analyze\_excel\_data fonksiyonuna aktarılır. [8]

**4. analyze\_excel\_data Fonksiyonu:**

def analyze\_excel\_data(excel\_data):

    try:

        summary = excel\_data.describe(include='all')

        question = f"İşte bir Excel dosyasından alınan verilerin özeti:\n\n{summary}\n\nBu veriler hakkında ne söyleyebilirsiniz? Verileri analiz et ve açıkla"

        response = chat\_text.send\_message(question)

        page.pubsub.send\_all(

            Message("AI", response.text, message\_type="chat\_message"))

    except Exception as err:

        page.pubsub.send\_all(

            Message("AI", f"Hata: {str(err)}", message\_type="chat\_message"))

* Bu fonksiyon, yüklenen Excel verilerini analiz eder.
* **summary = excel\_data.describe(include='all'):** Bu satır, Pandas'ın describe fonksiyonunu kullanarak DataFrame'in özet istatistiklerini hesaplar. include='all' parametresi, hem sayısal hem de kategorik sütunlar için özeti içerir.
* **question = ...:** Bu satır, AI modeline sorulacak soruyu oluşturur. Bu soruda, özet istatistikleri AI modeline sunulur ve modelin verileri analiz etmesi ve açıklaması istenir.
* **response = chat\_text.send\_message(question):** Bu satır, oluşturulan soruyu AI modeline gönderir ve yanıtını alır.
* **page.pubsub.send\_all(Message("AI", response.text, message\_type="chat\_message")):** Bu satır, AI modelinin yanıtını sohbet arayüzüne gönderir.

**5. pick\_excel Fonksiyonu:**

def pick\_excel(e):

    excel\_file\_dialog.pick\_files(

        allowed\_extensions=["xlsx", "xls"]

    )

* Bu fonksiyon, kullanıcı "Excel Dosyası Seç" düğmesine tıkladığında çağrılır.
* **excel\_file\_dialog.pick\_files(allowed\_extensions=["xlsx", "xls"]):** Bu satır, Excel dosyalarını seçmek için bir dosya seçme diyaloğunu açar. allowed\_extensions parametresi, sadece .xlsx ve .xls uzantılı dosyaların seçilebileceğini belirtir. Kullanıcı bir dosya seçtiğinde pick\_excel\_result fonksiyonu çağrılır.
* Kod, Pandas kütüphanesini kullanarak Excel dosyalarını yükler ve işler.

Özet olark Excel Analizi için yazdığımız kodları bu işlemleri gerçekleştiriyor:

* analyze\_excel\_data fonksiyonu, DataFrame'in özet istatistiklerini hesaplar ve bu istatistikleri AI modeline sorulacak bir soruda sunar.
* AI modeli, verileri analiz eder ve yorumlar.
* Modelin yanıtı sohbet arayüzüne gönderilir.
* Kullanıcı, "Excel Dosyası Seç" düğmesini tıklayarak bir Excel dosyası seçebilir.

# Sonuçlar

Sonuç olarak tamamen kişisel ayarlarımıza göre ve verileri sadece bizim elimizde olan birçok amaçlı asistan uygulamayı elde etmiş olduk. Python ile entegre edilmesinden sonra server ile daha hızlı bağlantı kurup cevaplar da çok daha hızlı bir şekilde ekranımıza yanıt olarak yazılıyordur, yapay zeka modelinin hata yaptığı yerleri derinleştirme ve farklı Python kütüphaneleri kullanarak güçlendirmesinin ardından takriben hatasız bir modeli elde etmiş oluyoruz.

Ayrıca Flet ile geliştirdiğimiz programla yapay zeka kişisel asistan ve Python bazlı arayüz konusunda projemiz bir ilk sayılabilir. Bunu Github üzerine yapılan araştırmalarımızda da görebiliriz. [10][11]

## 4.1 Görüntü Analizin Sonuçları

Programda “renk seçim” butonuna basıldığında alınacak görüntünün seçimi yapılmaktadır. Programın test edilebilmesi için bu seçim, aynı görüntü üzerinde farklı renk seçimleriyle denenmiştir.

Üzerinde kırmızı,yeşil ve mavi renklerle sayılar yazılmış bir görüntü alınıp her bir seçenek ayrı ayrı uygulandı. Burdaki amaç yazılan sayılardan bir renge ait sayı istenildiğinde o renge ait sayıları verebilmesiydi. Bu istekte bulunulduğunda yapay zekâ, tam da beklendiği gibi yanlış, fazla veya eksik sonuçlar verdi ama bu durum renklere göre ayrılıp denendiğinde %95 oranında daha doğru sonuçlar elde edildi.

Aşağıda sayfa açılıp renk seçim kısmında “orijinal” butonuna tıklandığında ekrana gelen resmi görmekteyiz. Orijinal resmi kullanıp yapay zekaya “resimdeki kırmızı sayıları yaz” dediğimizde doğru sonuç vermemektedir.

A screenshot of a computer

Description automatically generated

Şekil 15 Yüklenmiş Sayıların Görüntüsü

Orijinal resim ekranda bulunurken renk seçim kısmından kırmızı rengi seçtiğimizde diğer kısımları siyah yaparak aşağıdaki gibi ekrana gelmektedir.

A screenshot of a computer

Description automatically generated

Şekil 16 Görüntü İşlemeden Sonra Sadece Kırmızı Renkler

Yeşil veya mavi renk seçildiğinde de aşağıdaki gibi çıktılar alınmaktadır.

A screenshot of a computer

Description automatically generated

Şekil 17 Yeşilin Görüntü İşlemesi

Sadece kırmızı renkli sayıların tespit edildiği resim üzerinden, “resimdeki sayıları yaz” denildiğinde bu sefer yapay zekâ aşağıda görülüğü gibi öncekinin aksine doğru sonuç vermektedir.

A screenshot of a computer

Description automatically generated

Şekil 18 Görüntü İşlemeden Sonra Kırmızı Rengin Algılaması

Tüm bu durumlar değerlendirildiğinde görülmektedir ki yapay zekanın ciddi eksiklik gösterdiği bir konuya, harici olarak bir çözüm bulunmuş ve renk tespiti sayesinde doğru sonuçlar alınmıştır.

## 4.2 PDF Analizin Sonuçları

Bir PDF dosyasını yükledikten sonra uygulamamız analiz için çeşitli araçları sunuyor, şöyle ki hem sohbet kısmında PDF hakkında soruları yazabiliriz veya eğer maksadımız yorumlama, özetleme vs. ise zaten Prompt[[1]](#footnote-1) mühendisliği yapılmış butonlardan da seçebiliriz bu şekilde hem zaman kaybetmemiş oluruz ve hem daha profesyonel bir Prompt ile yapay zeka modelimizden soru sormuş oluruz.

A screenshot of a computer

Description automatically generated

Şekil 19 Yorumlama Butonu

A screenshot of a computer

Description automatically generated

Şekil 20 Çıkan Sonuç

## 4.3 Kamera Bölümün Sonuçları

Şekil 21 OpenCv İle Kamera Açma ve Kaydetme

A screenshot of a computer

Description automatically generatedDaha önce tezekkürde bulunduğumuz gibi Flet tek başına Python bazlı bir kamera sistemi sunmuyor dolaysıyla biz yenilikçi bir yaklaşımla ve yaptığımız araştırmalarla beraber sonunda OpenCv’yi kullanarak bir kamera arayüzünü tamamen fonksiyonel ve çalışılabilir hale getirmiş olduk. [12]

Şekil 22 Kamera Görüntüsünü Analiz Etme Butonları

Diğer bölümler için geliştirdiğimiz prompt mühendisliği yapılmış butonları kamera bölümü için de uygun hale getirip sınav notu hesaplama, akademik açıklama, sanatsal açıklama ve daha da ekleyebileceğimiz butonlara sahip.

## 4.4 Excel Analizin Sonuçları

Elimizde bir veri seti ve ya excel dosyası olduğunda kolayca bu dosyayı yükleyip hiçbir prompt yazmadan uygulama analiz etmeye başlar. [13]

Örnek bir excel dosyasını analiz edelim:

Tablo 1 Satış Tablousu

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sum of Sales** | **Region** |  |  |  |  |
| **Product Container** | **East** | **West** | **Central** | **South** | **Grand Total** |
| Small Box | 236221.87 | 149212 | 148010.96 | 159002.2 | 692447.03 |
| Jumbo Drum | 171206.92 | 160956.79 | 99979.61 | 60293.84 | 492437.16 |
| Jumbo Box | 55375.88 | 108672.21 | 82139.39 | 50630.79 | 296818.27 |
| Large Box | 93160.12 | 79507.33 | 63502.23 | 44916 | 281085.68 |
| Medium Box | 9326.33 | 9442.21 | 29787.03 | 17455.36 | 66010.93 |
| Small Pack | 15605.28 | 11786.93 | 16108.39 | 19256.93 | 62757.53 |
| Wrap Bag | 11275.09 | 7199.1 | 8757.09 | 5550 | 32781.28 |
| **Grand Total** | **592171.49** | **526776.57** | **448284.7** | **357105.12** | **1924337.88** |

A screenshot of a computer

Description automatically generated

Şekil 23 Dosyanın Sözel Analizi

# Kaynaklar

# ÖZGEÇMİŞ

## Zubair Najim Dost

Bandırma Onyedi Eylül Üniversitesi'nde Elektrik Elektronik Mühendisliği bölümünden 2024 yılında mezun olacak bir mühendislik öğrencisidir. Makine öğrenimi ve bilgisayarla görme alanında Alntelia şirketinde staj yapmış olup, otonom araçlar için Python ve MATLAB kullanarak makine öğrenimi modelleri geliştirmiştir. CARLA simülasyon ortamını kullanarak araç algılama ve karar verme algoritmalarını test etmiş ve doğrulamıştır. Bunun yanı sıra, Etimaden Bandırma Fabrikası'nda Elektronik ve Haberleşme Mühendisi olarak staj yapmıştır. Bu süre zarfında otomasyon projeleri ve bakım görevlerinde yer almış ve Siemens ile ABB PLC ve motor sürücüleri kullanarak elektronik ve iletişim sistemlerinde teknik becerilerini geliştirmiştir. Eğitim hayatında çeşitli programlama ve yapay zeka kurslarına katılmış olup, TRAI Academy'de Veri Analizi ve Makine Öğrenimi, AI Business School'da Makine Öğrenimine Giriş ve Gazi Üniversitesi'nde Python, HTML, CSS ve JavaScript gibi ana programlama dillerine giriş konularında eğitimler almıştır. Türkçe, İngilizce, Farsça ve Peştuca dillerinde yüksek seviyede yetkinliğe sahiptir.

## Huzeyfe Aktaş

Bandırma Onyedi Eylül Üniversitesi'nde Elektrik Elektronik Mühendisliği ve Bilgisayar Mühendisliği çift anadal öğrencisi olarak eğitim görmektedir. 2020'den beri Teknofest yarışmalarında aktif olarak yer alan Huzeyfe, Supernova Rocket Team bünyesinde roket tasarımı, drone sistemleri ve görüntü işleme gibi alanlarda tecrübe edinmiştir. İstanbul Doğalgaz Termik Santrali ve Kıyı Emniyeti Genel Müdürlüğü'nde staj yapan Huzeyfe, aynı zamanda Kodland bünyesinde Python, Unity ve matematik alanlarında online eğitmenlik yapmaktadır. C, Java, Python, C# gibi programlama dillerine hakim olan Huzeyfe; OpenCV, AutoCAD, Solidworks, Eagle, Proteus, OpenRocket gibi yazılımları kullanabilmektedir. Ayrıca Festo FluidSim ile pnömatik ve elektropnömatik sistemler konusunda da bilgi sahibidir. İleri seviye İngilizce bilen Huzeyfe, yapay zeka alanında kendini geliştirmeye devam etmektedir.

1. Prompt mühendislik, yapay zekâ modellerine doğru ve etkili girdiler (prompt'lar) vererek istenen çıktıları elde etme sürecidir. Bu süreç, girdilerin net, bağlamsal ve spesifik talimatlarla tasarlanmasını içerir. [↑](#footnote-ref-1)