T.C.
SAKARYA ÜNİVERSİTESİ
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ

BSM 498 BİTİRME ÇALIŞMASI

İÇERİK ÖNERİ SİSTEMİ

B151210036 – Yusuf Taha ÖZTÜRK
B161210018 – Nuh YURDUSEVEN
B171210375 – Murat PEKUZ

Bölüm : BİLGİSAYAR MÜHENDİSLİĞİ
Danışman : Doç. Dr. Devrim Akgün

2019-2020 Bahar Dönemi

T.C.
SAKARYA ÜNİVERSİTESİ
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ

İÇERİK ÖNERİ SİSTEMİ

BSM 498 - BİTİRME ÇALIŞMASI

Yusuf Taha ÖZTÜRK

Nuh YURDUSEVEN

Murat PEKUZ

Fakülte Anabilim Dalı : BİLGİSAYAR MÜHENDİSLİĞİ

Bu tez .. / .. / … tarihinde aşağıdaki jüri tarafından oybirliği / oyçokluğu ile kabul edilmiştir.

…….	…….	…….
Jüri Başkanı	**Üye**	**Üye**

ÖNSÖZ

Bilgisayar Mühendisliği Tasarımı dersi kapsamında hazırlamış olduğumuz projemizde, bazı global şirketlerin ürünlerinde kullandığına benzer bir teknoloji geliştirilmesi amaçlanmıştır. Geliştirilen teknoloji ile görseller arası benzerlik, kullanıcı ilgi alanları tespiti ve kullanıcıya ilgi duyabileceği içerik tavsiyesi yapılması istenmektedir.

Projede emeği geçen arkadaşlarımıza, kullandığımız kaynak ve içerikleri edindiğimiz web sayfalarına (Kaynakça bölümünde belirtilmiştir), sunucu masraflarımızı karşılamak üzere maddi destekte bulunan Github'a ve katkılarından dolayı Devrim Akgün hocamıza teşekkür eder, saygılarımızı sunarız.

İÇİNDEKİLER

ŞEKİLLER LİSTESİ

ÖZET

Anahtar kelimeler: Mobil Uygulama, web uygulaması, İçerik öneri sistemleri, sosyal medya, istatistik, nesne tesbiti

Günümüzde birçok sosyal medya uygulaması, e-ticaret web siteleri veya benzer uygulama kullanıcıların ilgisini çekecek, ihtiyacı olan ürüne veya içeriğe kolayca ulaşmasını sağlayan büyük veri analiz sistemleri ve içerik öneri sistemleri gibi kompleks sistemler kullanmaktadır. Mobil ve web uygulamaları ise sosyal medyalar ve e-ticaret web sitelerine ulaşmak için hazırlanan ve kullanılan istemci uygulamalarıdır. Bir ürüne benzeyen bir içeriğin tespit edilmesi için birçok metot vardır. Bu proje içerisinde kullanıcı bilgileri kullanılmayıp ürün bazlı ve istatistik tabanlı metotlar kullanılacaktır.

Görsel içerisindeki nesnelerin tespit edilmesi bilgisayar görmesi içerisinde ayrı bir araştırma alanıdır. Nesne tesbiti için derin öğrenme metotlarının kullanılması ise son yıllarda oldukça popüler bir hale gelmiştir. Farklı nesneler ile eğitilen bir model farklı açılardan görünen nesneleri tanıma açısından oldukça başarılı sonuçlar verebilmektedir.

Proje içerisinde gerekli materyal ve metotlar belirlenerek kullaılmak üzere hazırlanmıştır. Gerekli IaaS servisi, nesne tespit modeli ve eğitilmiş nesne veritabanı, flask freamwork'ü, ftp server mongoDb veri tabanı yönetim sistemi, istemci uygulamalarını geliştirmek amacıyla fiziksel olarak uygun cihazlar hazırlanmıştır.

İstemci uygulamalarından suncuya görsel dosyası yükledikten sonra diğer dosyalar ile benzerlik ilişkisi kurulması sağlanıyor. Kullanıcı arayüzde herhangi bir görsel ile etkilişime girdiğinde bu görsele benzer olan dosyaların kullanıcıya gösterilmesi sağlanmaktadır.

Projede birçok sosyal içerik sağlayıcısının, e-ticaret sağlayıcılarının kullandığı teknolojilerden biri için basit bir alternatif hazırlmanmaya çalışılmıştır. Süreç içerisinde geliştirilmesi, yüksek doğruluk ve hız oranlarının sağlanması , kapsam olarak daha fazla veri ile çalışılabilmesinin sağlanması ve aynı anda birçok kulllanıcıya hizmet verilebilmesi için geliştirmeler yapılabilir.

BÖLÜM 1. GİRİŞ

Günümüzde internet ortamında verilen hizmetleri kullanıcı bazında kişiselleştirilmiş olması oldukça yaygındır. Bunu sağlayan şirketlerin kullanıcı memnuniyetini daha kolay kazanabildiklerini biliyoruz. Bu projede de görsel içerik sunumlarının kullanıcı yönelimlerine göre kişiselleştirilip, kullanıcının uygulama içerisinde daha keyifli vakit geçirmesi amaçlanmaktadır.

Bu gibi çalışmaları sosyal medya , e-ticaret, arama motorları, reklam servisleri gibi birçok alanda görebilmekteyiz. Teknoloji şirketlerinin hemen hemen hepsi bu gibi çalışmalara bütçe ve kaynak ayırmaktadır. Hatta bu gibi çalışmalar için danışmanlık hizmeti veren şirketler bile bulunmaktadır.

Örnek olarak son zamanlarda bütün dünya da yankı uyandıran Amerikan seçmenlerinin manipüle edilmesini verebiliriz. Cambridge Analytica adlı şirketin Facebook'tan elde ettiği verileri kullanarak seçmenlerin davranışlarını ve duygu durumlarını tespit edip seçimlerde belli bir aday lehine bu analizleri kullanmıştı.
Benzer çalışmaları Facebook, Instagram, Google Ads, Pinterest gibi şirketlerin içerik sunma algoritmalarında görebiliyoruz

1.1. Projenin Amacı ve Önemi

Şu anda birçok şirket kullanıcıya özel uygulamalar gelişitirerek müşterilerinin ilgilerini çekmeyi amaçlamaktadır. Kişiye özel uygulamalar insanlara kendilerini önemli hissettirmekte ve ürüne olan bağlılıklarını artırmaktadır. Bu sayede şirketler daha fazla kullanıcıyı elinde tutup daha fazla kar edebilmektedir. Bu sebeple kişiye özel deneyimler sunmak inovasyona önem veren şirketler için son derece önemlidir.

Kullanıcılar için son yıllarda kullanımı artan mobil cihazlar ve uzun yıllar boyunca kullanılan web platformu için geliştirilecek istemci uygualamaları ile birbirine benzer görsel içieriklere hızlı ve kolay bir şekilde ulaşması proje için temel amaç olarak nitelendirilebilir.

İçerik öneri sistemleri için kullanılan farklı algoritmalar ve sistemler mevcuttur . Bu projede görsel içeirisindeki nesnelerin isim ve sayıları kullanılarak merkezde istatistik tabanlı bir benzerlik tespit metodu ile benzer içerikler sunulması amaçlanmıştır.

1.2. Yapılacak İşler ve Kullanılacak Yöntemler

İlk aşamada kullanıcılar tarafından yüklenen görseller içerisindeki objeler tespit edilecek, uygun bir formatta veri tabanına kaydı yapılacak. Objenin ismi, benzerlik oranı, görsel içerisinde kapladığı alan gibi veriler tutulabilir. Bu tespit işlemi görüntü işleme kütüphaneleri yardımıyla yapılacak. Darknet Yolo V3 kütüphanesi ya da Python dilindeki Opencv, Matplotlib, Tensorflow yardımıyla veya bunları kullanan freamworkler(imageAi) kullanılarak yapılabilir. Yöntemler arasından en uygun olan yöntem seçilecek ve sunucu tarafında çalıştırılmak üzere bir web servisine dönüştürülecek. Tespit edilecek objeler ile birlkte önceden eğitilmiş bir model kullanılacak. Model seçiminde hız ve doğruluk oranı kriterlerine en fazla dikkat edilmesi gereken kriterler olacaktır.

Bu dağınık verinin depoplanması için no-sql bir veritabanı kullanılacak. Daha sonra görsellerin belirli benzer özellikleri içerdikleri objeler ve bu objelerin sayıları gibi parametreler göz önünde bulundurularak girdi olarak verilen bir görselin karşılığında veri tabanında tutulan benzer görsellerin kullanıcıya sunulması sağlanacak. Bunları sağlayabilecek bir benzerlik algoritması tasarlanacak ve testi yapılacak.

Görsellerin kullanıcıyla sunulabilmesi için bir arayüz tasarlanacak. .Net core kullanılarak bir web uygulaması ile birlikte native bir Android uygulama geliştirilecek ve kullanıcıların mümkün olduğunca akış içerisinde kalmasını sağlayacak responsive bir arayüz ile birlikte backend işlemlerinin entegrasyonu sağlanacak.

Sunucu üzerindeki servisler RESTful olacak şekilde Flask freamwork'ü kullanılarak tasarlanacak, istemci uygulamalarından gelen http isteklerine karşılık cevap verilerek sistemin ,stabilitesi, devamlılığı sağlacak.

1.3. İş-zaman Çizelgesi

Yapılacak İşler	H3	H4	H5	H6	H7	H8	H9	H10	H11	H12	H13
1. Benzerlik tesbiti ve içerik önerimi için araştırma	X	X	X								
2. Sunucu hizmeti kiralanması ve gerekli kurulumlar		X	X	X	X	X					
3. Yazılım mimarisi tasarımı			X	X	X	X	X	X			
4. API oluşturulması ve sunucu entegrasyonu			X	X	X	X	X	X			
5. Android mobil uygulama yazılması				X	X	X	X	X			
6. .Net core web uygulaması,				X	X	X	X	X			
7. Benzerlik tesbiti işlemi ve servislerin yazılması						X	X				
8. Yapılan işlemlerin uygulama tarafına entegre edilmesi kullanılması							X	X	X		
9. Test ve düzeltme işlemleri									X	X	X
10. Ar-ge										X	X

Grup üyelerinin iş-zaman çizelgesindeki görevleri şöyledir:

Yusuf Taha Öztürk: 1, 2, 3,5,7,8,9,10 numaralı,

Nuh Yurduseven: 1, 2,3,4 ,7,9,10 numaralı,

Murat Pekuz: 1, 2,3, 6,8,9,10 numaralı iş paketlerinde görev almıştır.

- **Maliyet analizi**

Aylık 10$'lık ücreti olan sunucu için proje geliştirme süresince "github developer pack" tarafından öğrenciler için sağlanan 50$ lık hibe kullanılcaktır.

50$/10$= 5 ay

5 ay boyunca yeterli olacaktır. 50$lık hibe kullanıldıktan sonra ikinci bir Digitalocean hesabı açılıp yeni bir öğrenci hesabı ile tekrar 50$ lık hibe kullanılacaktır.

50$/10$ =5 ay

Toplamda 100$ lık hibe, 10 aylık sunucu maliyetini karşılamaktadır.

- Sunucu maliyeti 10ay*10$ = 100$

Sunucu maliyeti dışında proje için ek maliyet oluşturacak bir durum yoktur.

BÖLÜM 2. MATERYAL VE METOT

Yapay Zekanın önemli alanlarından biri Computer Vision'dur. "Computer Vision, görüntüleri ve sahneleri tanıyabilen ve anlayabilen bilgisayar ve yazılım sistemleri bilimidir"[1]. Computer Vision ayrıca görüntü tanıma, nesne algılama, görüntü oluşturma ,çözünürlük ayarlama gibi alanlar ve daha fazlasından oluşur. Nesne algılama, bilgisayarlı görmenin en derin ve en çok üzerinde çalışılan alanıdır..

Nesne algılama, yazılım sistemlerinin bir görüntü ya da sahne içindeki nesneleri bulma ve her nesneyi tanımlama yeteneğidir. Nesne algılama, yüz algılama, araç algılama, yaya sayma , güvenlik sistemleri ve sürücüsüz otomobiller için yaygın olarak kullanılmaktadır. Nesne tespitinin birçok uygulama alanında da kullanılabileceği birçok yol vardır.

İçerik öneri sistemlerinin kullandığı birbirinden farklı birçok yöntem ve metot vardır. Kullanıcıya sunulmak istenen içeriğin türü, cevap verme hızı, yazılım ölçeklendirilmesi ile ilgili ihtiyaçlar veya kaynak yönetimi gibi kriterler kullanılmak istenen metot seçimi için göz önünde bulundurulur. İçerik öneri motorları ilgili ve özelleşmiş aramalar yapmak için yapay zeka, makine öğrenmesi, derin öğrenme yöntem ve metotlarını kullanan algoritmalardır. Genellikle işbirlikçi filtreleme, içerik tabanlı filtreleme, demografik tabanlı filtreleme, yardımcı program tabanlı filtreleme, bilgi tabanlı filtreleme ve hibrit filtreleme modellerini kullanır.

2.1. ImageAI

“ImageAI, geliştiricilere en yeni Yapay Zeka özelliklerini yeni ve mevcut uygulamalarına ve sistemlerine kolayca entegre etmeleri için güç veren kullanımı kolay bir Computer Vision Python kütüphanesidir.”[2]

ImageAI, görüntü nesnesi algılama ve çıkarma işlemlerini gerçekleştirmek için çok güçlü ve kullanımı oldukça kolay sınıflar ve işlevler sağlayan bir Python kütüphanesidir. ImageAI, bunların hepsini RetinaNet, YOLOv3 ve TinyYOLOv3 gibi derin öğrenme algoritmalarıyla gerçekleştirilmesini sağlar. ImageAI ile algılama görevlerini çalıştırılabilir ve görüntüler analiz edebilir.

Bu tespit sınıfı, COCO veri kümesinde eğitilmiş önceden eğitilmiş modelleri kullanarak herhangi bir görüntü veya görüntü kümesi üzerinde nesne algılama gerçekleştirme işlevi sunar. Desteklenen modeller RetinaNet, YOLOv3 ve TinyYOLOv3’dir.Bu modelleri kullanılarak , 80 farklı türdeki gündelik nesneyi algılanabilir.[3]
Bu modellerden biri indirildikten sonra Python programlama dilinde Şekil 2.1.0 deki gibi import işlemi gerçekleştirirlir.

```
from imageai.Detection import ObjectDetection

detector = ObjectDetection()
```

Şekil-2.1.0 ObjectDetection sınıfının import edilmesi [1]

.setModelTypeAsRetinaNet(): Bu metot, RetinaNet modeline oluşturduğunuz nesne algılama örneğinin modelinin türünü ayarlamak için kullanılır. Şekil 2.1.1 de Python ortamındaki uygulaması görülmektedir.

```
detector.setModelTypeAsRetinaNet()
```

Şekil-2.1.1 Nesne algılama modelinin ayarlanması[1]

.setModelPath (), Bu işlev, indirdiğiniz model dosyasının yolu olması gereken ve nesne algılama örneği için ayarlanılan model türüne karşılık gelen bir dizeyi kabul eder. Şekil 2.1.2 te örnek uygulama görülmektedir.

```
detector.setModelPath( os.path.join(execution_path , "resnet50_coco_best_v2.0.1.h5"))
```

Şekil-2.1.2 Model dosyasının yolu

.loadModel ( ), Bu metot modeli metot çağrısında belirtilen yoldan yüklemek için kullanılır.(bkz. Şekil 2.1.3)

```
detector.loadModel()
```

Şekil-2.1.3. Modelin yüklenmesi[1]

.detectObjectsFromImage (), Model yüklendikten sonra nesne algılama görevini gerçekleştiren işlevdir. Herhangi bir sayıda görüntüdeki nesneleri algılamak için birçok kez çağrılabilir. (bkz Şekil 2.1.4)

```
detections =detector.detectObjectsFromImage(input_image="image.jpg",
output_image_path="imagenew.jpg")
```

Şekil-2.1.4. Görselin metoda parametre olarak gönderrilmesi[1]

Örnek bir nesne tespit uygulaması şekil 2.1.5’ teki gibidir.

```
from imageai.Detection import ObjectDetection
import os

execution_path = os.getcwd()

detector = ObjectDetection()
detector.setModelTypeAsRetinaNet()
detector.setModelPath( os.path.join(execution_path , "resnet50_coco_best_v2.0.1.h5"))
detector.loadModel()
detections = detector.detectObjectsFromImage(input_image=os.path.join(execution_path , "image.jpg"),
output_image_path=os.path.join(execution_path , "imagenew.jpg"))

for eachObject in detections:
    print(eachObject["name"] , " : " , eachObject["percentage_probability"] )
```

Şekil-2.1.5. Obje tespit eden kod

Şekil 2.1.5 dekii kod içerisinde resnet50_coco_best_v2.0.1.h5 modeli kullanılmıştır.

2.2. RetinaNet modeli

RetinaNet eğitim için Feature Pyramid Network (FPN) ve Focal Loss kullanan tek aşamalı bir algılayıcıdır. FPN çok ölçekli nesne algılama için kullanılan bir yapıdır. Sonuç olarak, hem sınıflandırıcı hem de regresör ağlarına yardımcı olan ağdaki birden çok düzeyde farklı ölçekte özellik haritaları üretir.

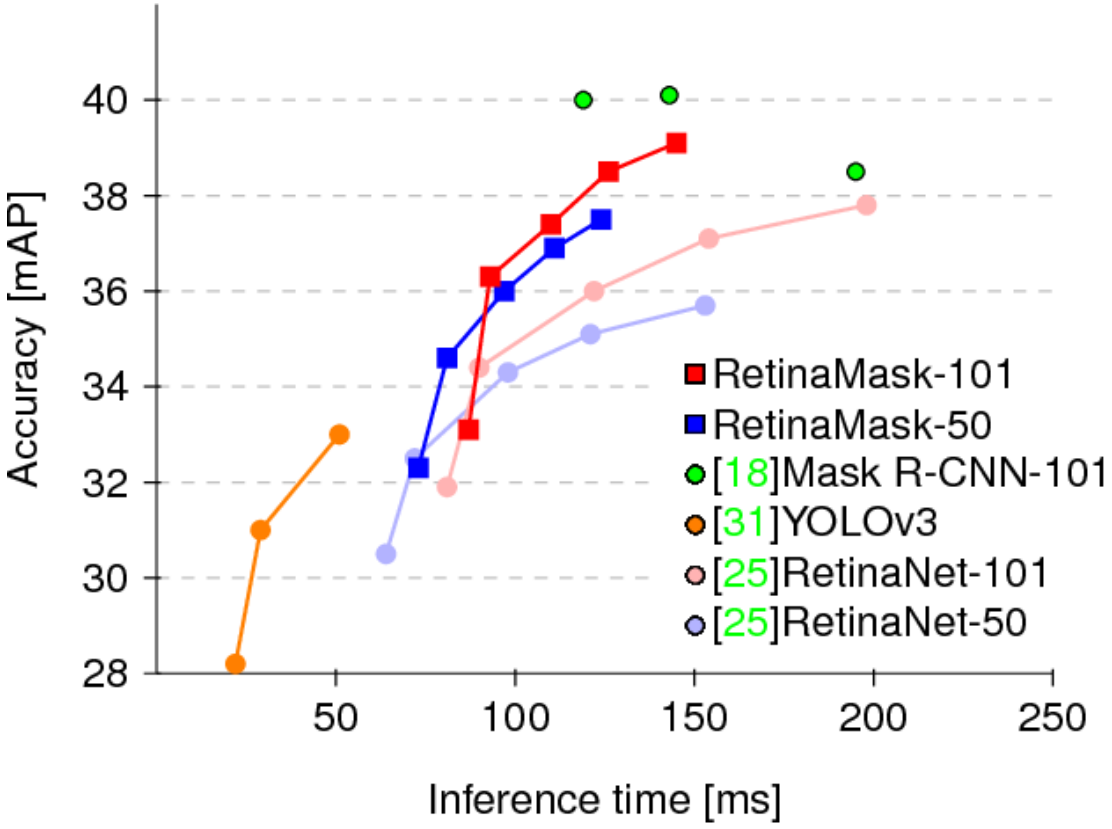
Focal Loss, çok sayıda olası arka plan sınıfının ve sadece birkaç ön plan sınıfının olduğu dengesizlik ile tek aşamalı nesne algılama sorunlarını ele almak için tasarlanmıştır. Çoğu yerin yararlı bir sinyale katkıda bulunmayan kolay negatifler olduğu ve bu olumsuz örneklerin büyük miktarının eğitimi etkilediği ve model performansını düşürdüğü için eğitimin verimsiz olmasına neden olur. Focal Loss, çapraz entropi kaybına dayanmaktadır ve gama parametresini ayarlayarak, iyi sınıflandırılmış örneklerden kayıp katkısını azaltabiliriz.

RetinaNet, YOLOv3 gibi veri seti olarak COCO veri kümesini kullanmaktadır. RetinaNet ve YOLO versiyonlarının COCO veri kümesi üzerindeki performans grafikleri aşağıdaki gibidir.

“Focal Loss, çok sayıda olası arka plan sınıfının ve sadece birkaç ön plan sınıfının olduğu dengesizlik ile tek aşamalı nesne algılama sorunlarını ele almak için tasarlanmıştır. Çoğu yerin yararlı bir sinyale katkıda bulunmayan kolay negatifler

olduğu ve bu olumsuz örneklerin büyük miktarının eğitimi etkilediği ve model performansını düşürdüğü için eğitimin verimsiz olmasına neden olur. Focal Loss, çapraz entropi kaybına dayanmaktadır ve gama parametresini ayarlayarak, iyi sınıflandırılmış örneklerden kayıp katkısını azaltabiliriz".[4]

RetinaNet, YOLOv3 gibi veri seti olarak COCO veri kümesini kullanmaktadır. RetinaNet ve YOLO versiyonlarının COCO veri kümesi üzerindeki performans grafikleri ise şekil 2.2.0'daki gibidir.

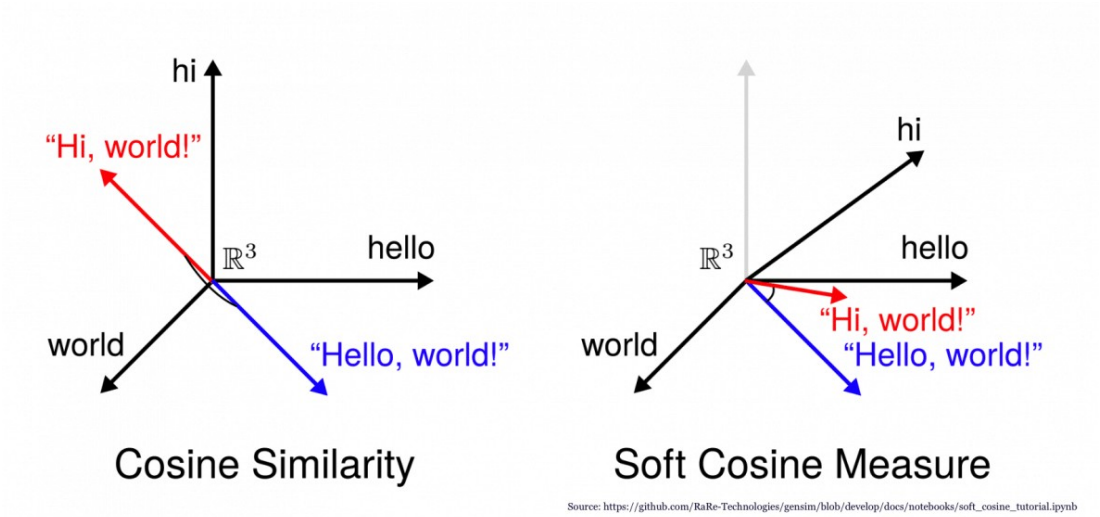


Şekil-2.2.0. Model performanlarının karşılaştırılması[2]

2.3. Kosinüs Benzerliği

İçerik bazlı öneri sistemlerinde sıklıkla kullanılan kosinüs benzerliği yöntemi birbiri ile aynı yöne bakan vektörlerin cos(x,y)=1 sonucunu üretmesinden ve vektörler

arasındaki açının kosinüsünün aralarındaki ilişkiyi ortaya koyabilmesinden yararlanan bir metotdur. Daha öce belirtildiği gibi aynı yönü gösteren vektörlerin kosinusü “1”, birbirine dik olan vektölerin kosinüsü ”0” ve birbiri ile zıt yönü gösteren vektörlerin kosinüsü ise “-1”’dir. Elde edilecek çıktı +1 ile – 1 arasında değerler alır.



Şekil-2.3.0. Cosine Similarity[3]

Doküman benzerliğinde vektörleri oluşturmak için özel kelimelerin frekansları vektöleri oluşturmak için kullanılan temel metriklerden biridir ve sıklıkla kullanılır.

Şekil 2.2.0 da görüldüğü gibi kelimeler vektörel olarak ifade edildikten sonra uzaklık vektörleri arasındaki açının cosinus’ü, iki doküman arasında benzerlik oranını -1 ve 1 arasında bir değer olarak temsil edebilmeyi sağlar.

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

Şekil-2.3.1. Cosine Similarity matematiksel model[4]

Cosine similarity yöntemi için matematiksel model gösterimi ise yukarıda anlatılan temel mantığa göre şekil 2.3.1 de gösterildiği gibidir.

Cosine_similarity metodu python modüllerinden “sklearn” modülü içerisinde hazır olarak bulunmaktadır. (şekil 2.3.2)

```
sklearn.metrics.pairwise.cosine_similarity(X, Y=None, dense_output=True)
```

Şekil-2.3.2. Sklearn cosine similarity method[5]

Parametre olarak n boyutlu iki adet liste alan ve bunları yukarıdaki Şekil-2.3.2 de belirtilen matematiksel forrmülü kuullanarak işleyen, bolean tipinde liste döndüren bir metottur.

2.4. Flask API

Flask, Python'da yazılmış bir “mikro” web freamwork’ü olarak tanımlanabilir. Araç ya da kütüphane gerektirmediği için bir mikro freamwork olarak sınıflandırılmıştır. Herhangi bir veritabanı soyutlama katmanı, form doğrulaması veya önceden var olan üçüncü taraf kitaplıklarının ortak işlevler sağladığı diğer bileşenleri yoktur.” . Nesne-ilişkisel eşleştiriciler, yükleme yönetimi, form doğrulama, çeşitli açık kimlik doğrulama teknolojileri ve çerçeveyle ilgili birkaç ortak araç için uzantılara sahiptir.

Flask freamwork’ü kullanılarak “/” route’u içerisinde çalışan basit bir uygulama programı Şekil-2.3.0 gösterilmektedir.

```python
from flask import Flask
app = Flask(__name__)

@app.route("/")
def hello():
    return "Hello World!"

if __name__ == "__main__":
    app.run()
```

Şekil-2.4.0. Flask hello world[6]

Şekil-2.3.0 da verilen program ,localhost 5000 numaralı portu kullanan ve "development server" üzerinde çalışan bir servisi çalıştırır.

"Flask çerçevesini kullanan uygulamalar arasında Pinterest ve LinkedIn bulunmaktadır."[5]

2.5. Dosya Transfer Protokolü(FTP)

Açılımı File Transfer Protocol olan FTP'nin Türkçe karşılığı Dosya Transfer Protokolü'dür. Ftp internete bağlı iki bilgisayar arasında dosya transferini sağlamak için kullanılan bir protokoldür ve bu işleme hizmet eden uygulamaya verilen isimdir. Kullanım alanına örnek verecek olursak, günümüzde web sitelerinde yer alması istenen dosyalar FTP aracılığı ile sunuculara aktarılmaktadır.
FTP ilk geliştirilen internet protokollerinden biridir. FTP protokolü ile;

- Bir başka bilgisayardan bir başka bilgisayara dosya aktarımı yapılırken, o bilgisayar ile etkileşimi aynı anda bağlantı kurulur.
- Protokol ile sağlanan bir dizi komutlar yardımıyla iki bilgisayar arasında dosya alma/gönderme işlemleri yapılır.

"Ftp'yi kullanmak için;

- Bağlanacağımız bilgisayarın internet adresi,
- Bağlanacağımız bilgisayarda dosyalara ulaşmak istediğimiz hesapla ilgili kullanıcı numarası, varsa şifresi,

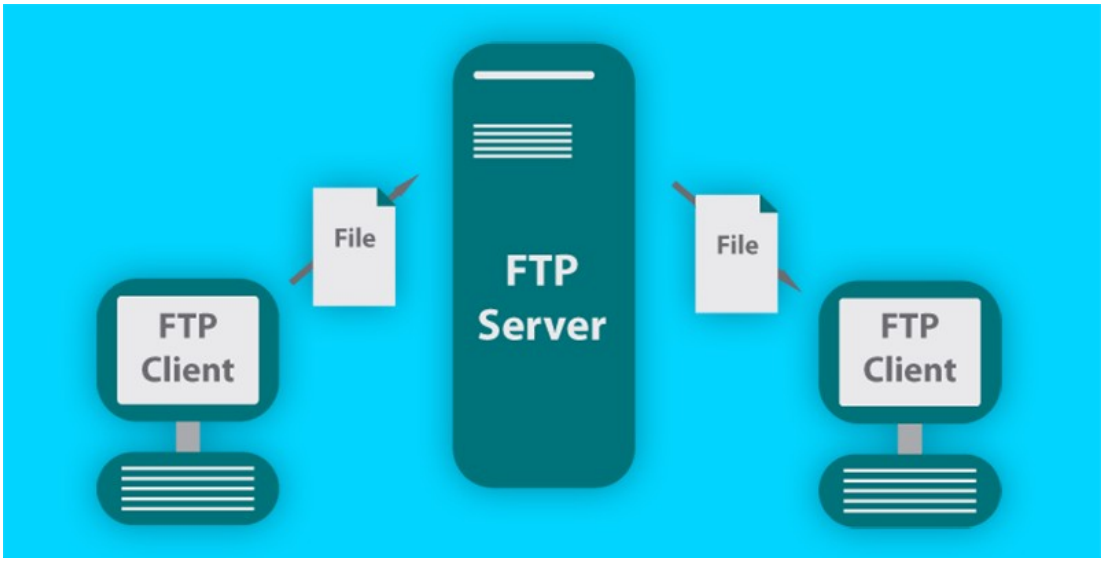
- İnternet erişimi olan, üzerinde FTP yazılımı bulunan bilgisayar
- Bağlanacağımız bilgisayarda, FTP protokol komutlarını yorumlayacak çalışır

durumda bir FTP servis programı yani FTP sitesi gereklidir.”[6]

2.6. FTP server

FTP sunucusu bir dosya aktarma protokolü ilişkisinin yarısıdır. FTP, verileri bir istemci-sunucu ilişkisi üzerinden aktaran bir protokoldür. FTP sunucusu kullanıcıların verilerini depolar ve bilgisayarınıza yüklemeniz gereken dosyaları almak veya başka bir bilgisayara dosya göndermek için daha sonra FTP üzerinden erişimini sağlar. Bir FTP sunucusu, bir dosya aktarma protokolü (FTP) adresine sahip ve bir FTP bağlantısı almayı sağlayan
bir uygulamadır.

“Bir FTP sunucusunun çalışabilmesi için bir TCP / IP ağına ihtiyacı vardır. Bir veya daha fazla FTP istemcisiyle ayrılmış sunucuların kullanımına bağlıdır. Bağlantıların istemcilerden her zaman kurulabilmesini sağlamak için, genellikle bir FTP sunucusu açılır. Bir FTP sunucusu, FTP mimarisinde önemli bir bileşendir ve internet üzerinden dosya alışverişinde yardımcı olur. Bir FTP sunucusu ayrıca bir FTP sitesi olarak da bilinir.”[7]



Şekil-2.6.0 Ftp sembolik gösterimi[7]

Şekil 2.6.0 da ftp server'ı temsil eden bir ilustrasyon görülmektedir.

"FTP sunucusunun bazı özellikleri aşagıdaki gibi sıralanabilir:

- Kullanıcının FTP sunucusuyla bağlantı kurması için, kullanıcı adı ve şifre USER ve PASS komutları kullanılarak gönderilir. FTP sunucusu tarafından kabul edildikten sonra müşteriye bir onay gönderilir ve oturum başlayabilir.
- Bir FTP bağlantısı, daha önce başarıyla tamamlanmadıysa indirme işlemine devam etmek mümkündür.
- FTP sunucusu, dosyaların indirilmesine ve yüklenmesine izin verir. FTP sunucusu yöneticisi tarafından, farklı dosyaları indirmek için ve FTP sunucusunda bulunan farklı klasörlerden erişim için kısıtlamalar koyulabilir.
- FTP sunucusu, kullanıcılara giriş kimlik bilgilerine ihtiyaç duymadan bağlantı sağlayabilir; ancak, FTP sunucusu bunlara yalnızca sınırlı erişime izin verebilir.

- FTP sunucularında bulunan dosyalar ortak web tarayıcıları tarafından alınabilir, ancak FTPS gibi protokol uzantılarını desteklemiyor olabilir.
- FTP sunucuları adsız erişim sağlayabilir. Bu erişim, kullanıcıların dosyalarını sunuculardan anonim olarak indirmelerini sağlar ancak FTP sunucularına dosya yüklemelerini yasaklar.
- Tüm dosya aktarım protokolü site adresleri ftp: // ile başlar." [7]

2.7. NoSQL

“ NoSQL, ilk olarak 1998 yılında Carlo Strozzi tarafından ortaya çıkarılan bir kavramdır. Sql arayüzü olmayan bu veri depolama sistemine, tasarımcısı Carlo Strozzi, ilişkisel olmayan anlamında (İng. No Relation) tarzında bir isim konulması gerektiğini söyler.”[8]

“ NoSQL kavramı 2009 yılı başlarında, Rackspace şirketinin çalışanlarından Eric Evans tarafından, açık kaynak dağıtık veri tabanları sistemlerinin görüşüleceği bir toplantı düzenlenmek istendiği dönemlerde tekrardan kullanılmaya başlandı. .”[8]

NoSQL , ilişkisel veritabanı sistemlerindeki problemlere çözüm olarak ortaya çıkan, yatay olarak ölçeklendirilen bir veri depolama sistemidir. Günümüzde kullanılan sistemlerde kullanıcıların aktif rol alması ve internetin hızla yayılması, diğer herşey de olan değişim gibi veritabanlarında da değişikliğe gidilmesi gerekliliğini ortaya çıkardı. Sıklıkla tercih edilen ilişkisel veritabanlarındaki hiyerarşi, ilk olarak tabloları ve her tabloya ait sütunları oluşturmak sonrasında ise bilgileri satır satır eklemektir.
Zamanla gelişen yazılım ve sistemlerde karşılaştığımız bir problem ise tanımı olmayan alana bilgi kaydetmektir. Bu problemi çözmek için yapmamız gereken öncelikle tabloyu tekrardan ele alıp yeni sütunlar eklemektir. Bu işlemi yaparken diğer bağlantılı tablo ve ilişkileri etkileyecek durumlar da ortaya çıkabilmektedir. Tabloya yeni bir alan eklemek ve tabloyla ilişkili diğer tabloları güncellemek çok maliyetli olacaktır.

Özetle İlişkisel Veritabanı Sistemlerinde zamanın ihtiyaçlarına bağlı olarak maliyetin yükselmesi, kullanıcıları yapacakları proje için uygun bir yapı bulmaya ve yeni bir veri depolama sistemi arayışına yönlendirmiştir . Fakat NoSQL veritabanları , bilgilerin kayıt altına alınması adına bu yapıyı sonradan kurmak ek bir maliyet getirmemekte ya da az bir maliyet getirmektedir. Bunun sebebi ise NoSQL sistemlerde tablo ve sütun kavramının olmamasıdır. NoSQL sistemlerde yeni bir alana ihtiyacınız olduğu durumlarda kaydı direkt olarak eklemek yeterli olmaktadır. NoSQL sistemler kullanıcı yerine yeni bir alan oluşturur ve girilen değeri kaydeder.
Kayıtların kullanıcıya maliyet getirmemesinin sebebi ise verilerin tablo ve sütunlarda saklanması yerine JSON ve XML formatına benzer yapılarda saklanmasıdır.

Tüm bu anlatımlardan “NoSQL veritabanları, ilişkisel veritabanlarından çok daha iyidir” gibi bir sonucu çıkarmamamız gerekmektedir. Çünkü ikisininde kendine has kullanım alanları mevcuttur. Örneğin: ilişkisel olayların çok yoğun gerçekleştiği bir bankacılık uygulamasında NoSQL bir veritabanı kullanmamız uygun değildir. NoSQL, Fire and Forget prensibi ile çalıştığı için bankacılık, alışveriş gibi para üzerinden işlem yapılan kritik uygulamalarda kullanılmamalıdır. Aksine veri odaklı olunmayan durumlarda kullanımı daha uygundur.

“NoSQL sistemlerin avantajları;

- NoSQL veritabanı sistemlerinde verilere erişmek ilişkisel veritabanlarına göre oldukça kolaydır.

- NoSQL veritabanı sistemleri okuma ve yazma performansları olarak göreceli olarak ilişkisel veritabanı sistemlerine göre daha performanslı olabilirler.

- NoSQL veritabanı sistemleri yatay olarak genişletilebilirler. Binlerce sunucu bir arada küme olarak çalışabilir ve çok büyük veri üzerinde işlem yapabilirler.

- NoSQL veritabanı sistemleri esnek yapılarından dolayı programlama ve bakım anlamında kolaylık sağlarlar.

- NoSQL veritabanı sistemlerinde farklı özelliklere sahip birçok implementasyon arasından seçim yapma şansınız vardır.

- NoSQL veritabanı sistemleri birçok açık kaynak kodlu projelere ve bulut ilişim teknolojilerine uygun olduğu için maliyet olarak ilişkisel veritabanı yönetim sistemlerine göre daha avantajlıdır."[8]

2.7.1. MongoDB

MongoDB 2009 yılında geliştirilmiş açık kaynak kodlu bir NoSQL veritabanıdır. Bugün piyasada Cassandra, BigTable, DynamoDB, Redis gibi birçok NoSQL veritabanı bulunmaktadır.

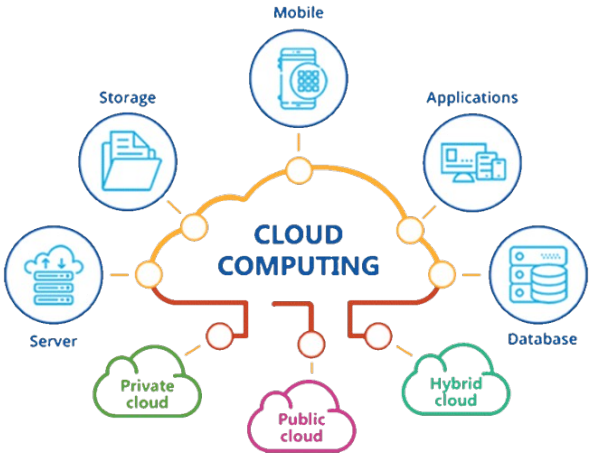
MongoDB'de kayıtlar doküman veri tipinde tutulur. Ve MongoDB'detutulan bu dökümanlar json formatında saklanır. İlişkisel veritabanlarında kullanılan table yapısını burada collection, row yapısını document, column yapısını ise field alır.

Günümüzde yazılımda aktif olarak kullanılan pek çok programlama dili için driver desteği bulundurması bakımından bugün NoSQL veritabanı sistemleri en çok tercih edilen sistemlerden biridir. Bu projede python programlama dili ile "pymongo" modülü kullanılacaktır.

2.8. Bulut Mimarisi ve Hizmetleri

Bulut bilişim, bir ağ üzerinden, bir hizmet sunmak için donanım ve yazılım kullanımıdır. Bulut bilişim sayesinde, kullanıcılar dosyalara veya cihaz arayüzüne erişebilir ve internete erişebilen herhangi bir cihazdan uygulamaları kullanabilir.(Bkz Şekil 2.8.0)

Uygulamanın gereksinimlerine göre kullanacağınız bulut bilişim sistemi değişkenlik gösterir. İşinizle ilgili hangi bulut sistem sizin için uygunsa onu seçmelisiniz. Diğer tüm sistemlerde olduğu gibi bulut bilişim sistemleri de kullanıcıların sorunlarına çözüm getirmek için farklı model, servis ve tip geliştirmiştir.



Şekil 2.8.0. Bulut Bilişim **[8]**

(a) Bulut Bilişim Teknolojisi Dağıtım Türleri

“Genel Bulut Sistemi: Genel bulut dağıtım sistemi, servis sağlayıcının kaynağına bağlı, standart bulut modelidir. Ücretsiz veya kullandığın kadar öde şeklinde tarifesi bulunur.”[9]

“Özel Bulut Sistemi: Özel bulut sistemi, sadece tek bir işletmenin veya organizasyonun kullandığı bulut bilgi işlem kaynaklarıdır. Bu sistem fiziksel olarak firmanın bilgi merkezinde bulunabilir. Bazı firmalar sistemlerini üçüncü taraf ile paylaşarak gelir elde ederler. Bu sistemde hizmetler ve altyapı özel bir ağda tutulur.”[9]

“Hibrit Bulut Sistemi: Hibrit bulut sistemi, benzersiz oluşumlardır. İki ya da daha fazla bulut bilişim alt yapısını veri ve uygulama taşınabilirliği sağlayan standart veya tescilli teknoloji ile birbirine bağlar. Diğer bir deyişle, uygulamanızın bir bölümünü genel bir bulutta çalıştırmak ve özel bir bulutta farklı bir bölüme sahip olmak istiyorsanız, iyi bir sonuç için bu iki bulutu birleştirin. Hibrit bulut her iki ortamı da dengeleyerek bağlı kalan hibrit bulut organizasyonlarınızın benzersiz ihtiyaçları için harika bir çözümdür.**”[9]**

(b) Bulut Bilişim Teknolojisi Hizmet Türleri

“Bulut bilişim dağıtım türleri dışında, bulut bilişim bilgi sistem türlerini de bilmek gerekir. Dört çeşit bulut bilişim bilgi sistemi bulunur. Bunlar; Platform hizmeti (PaaS), yazılım hizmeti (SaaS), altyapı hizmeti (IaaS) ve sunucusuz bilgi işlem. Bu bilgi sistemlerine, bulut bilişim bilgi yığını da deniyor. Çünkü bu sistemler birbirleri üzerine kurulur. Bu bulut bilgi sistemlerinin ne olduğunu ve birbirleri arasındaki farkları bilmek, firma hedeflerinize ulaşmanız da kolaylık sağlar.”[9]

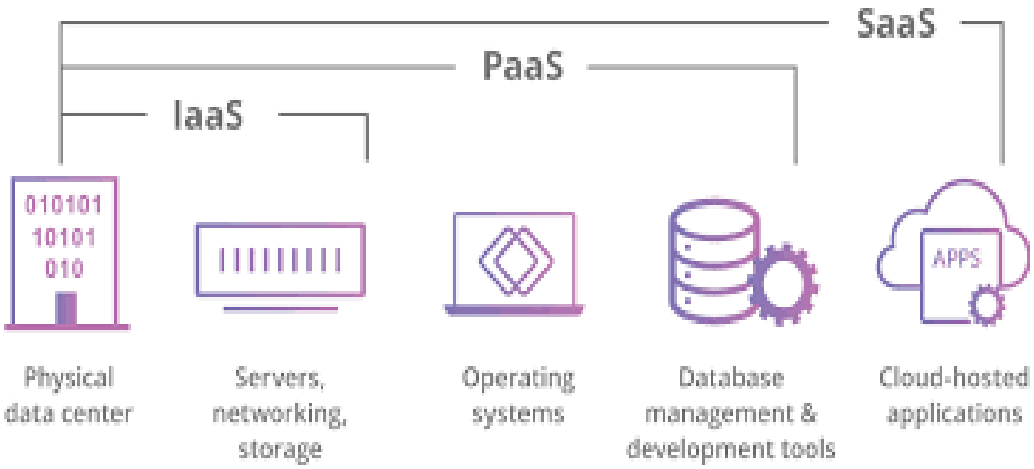
“PaaS - Platform Hizmeti: Uygulama geliştirenlere projelerini geliştirebilmeleri için yazılım ve donanım katmanları sunar. Veri tabanı, programlama dili, işletim sistemi gibi platformlar bu işletim sisteminin size sunduğu birkaç örnek platformdur. Bu sistemde sadece verileri ve uygulamaları yönetirsiniz. Çünkü, Sistem yönetimini hizmet sağlayıcı gerçekleştirir. Örneğin PHP kullanarak yazılım kodlaması yaptınız. Bu yazılımın web sunucu altyapısına zaman harcamazsınız. PaaS, yalnızca yazılımın çalıştığı platformları sunar”.[9]

“SaaS – Yazılım Hizmeti: Firmaların kullandıkları programlara çalışanların diledikleri zaman diledikleri yerde erişim sağlamasına yardımcı olur. Firmanıza program maliyeti anlamında yararlı olacaktır. Fazladan yazılım gideri olmasını önler. Kullandığınız ERP sisteminini yada muhasebe programına dünyanın başka bir ucunda bulut bilişimin bu hizmeti ile erişebilirsiniz.”[9]

“IaaS – Altyapı Hizmeti: Bu hizmet bulut bilişim teknolojisinin sağladığı temel hizmettir. Altyapı hizmeti ile sanal sunucu oluşturularak, kullanıcıya bulut bilişim hizmeti sağlanır. IaaS kullanıcılara esnek altyapı sağlar. Altyapıdan faydalanma durumu ihtiyaca bağlı, dönemsel olarak kaynak kullanımı azaltılıp çoğaltılabilir.”[9]

“Sunucusuz Bilgi İşlem: Bu hizmet PaaS ile örtüşür. Sunucusuz bilgi işlem, müşterinin arka uç kodunu çalıştırmak için sunucu tedarik etmek zorunda olmadığı, ancak gerektiğinde hizmetlere eriştiği bir tür bulut bilgi işlemdir. Bunun yerine, bulut sağlayıcısı, istekler geldiğinde hizmet sağlayıcısı tarafından faturalandırıldığı sırada bir konteyner platformunu bir hizmet olarak başlatır ve durdurur.”[9]

Şekil 2.8.1. de bukut suncu hizmetleri arasındaki ilişki görülebilmektedir.



Şekil l 2.8.1. Bulut Blişim hizmet modellleri[9]

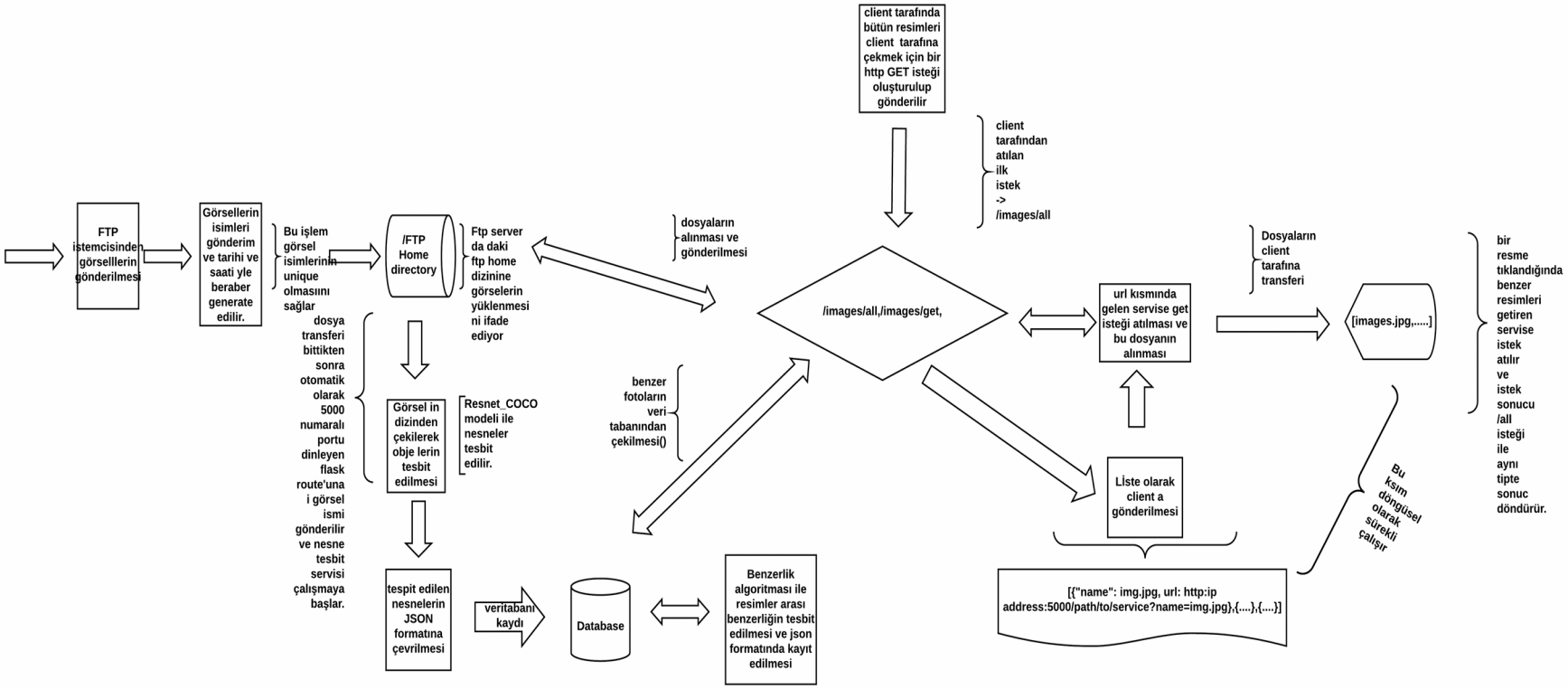
DigitalOcean ve Amazon AWS gibi bir çok bulut hicmet sağlayıcısı bulunmaktadır.DigitalOcean üzerinde droplet denilen İşlemci, bellek ve depolama kaynakları önceden seçilebilen, kullanılacak işletim sistemi, Kullanılacak veritabanı ve http server gibi uygulamaların önceden yüklenip oldukça hızlı bir şekilde ayağa kaldırılan bir IaaS hizmeti sunmaktadır.

Github Developer Pack üzerinden öğrenci e-mail hesabı ile kayıt olduktan sonra bir çok uygulama ve servis sağlayıcısında geçerli belirili miktarlarda kupon paketleri mevcuttur. DigitalOcean üzerinde kullanılmak için 50$ değerinde kupon kullanılacaktır.

2Gb ram, 2vcpu ve 60 Gb SSD depolama alanına sahip bir sanal bilgisayar aylık 15$ lık bir servis ücretine sahiptir. Ubuntu server 18.06 LTS oldukça bilinen ve güven veren stabil bir server işletim sistemidir. Bu sebeple uzak sunucuda Ubuntu server kullanılmasına karar verilmişrtir.

BÖLÜM 3. TASARIMIN GERÇEKLEŞTİRİLMESİ

3.1. Blok Diagramı



Şekil 3.1.0 Blok diagramı

Proje mimarisinin şematize edilmiş hali bu işlemlerin daha açık bir şekilde anlaşımalsını sağlayabilir. Tasarlanan yazılım mimarisinin şema üzerinde gösterimini Şekil 3.1.0 da görülebilir.

Blok diagramı ve Proje mimarisi yapılan işlemler hakkında bilgi vermektedir. Hazırlanan blok diagramında ve geliştirilen uygulamanın çalışma sistemi hakkında ön fikir elde edilebilir.

Proje içindeki en önemli gereksinim olan görsel dosyalarının alınması ve sunucuya kaydedilmesi, kulllanıcı tarafında hazırlanan uygulamalar yardımı ile olacaktır. Daha sonra yapılacak olan nesne tesbiti ve benzer görsel tesbiti işlemleri ise blok diagramında karar sembolü olan vce anahtarlama görevi yapacak olan, servis route'larını içerisinde barındıran ana servise bağlı olarak gerçekleştiriliyor. Bu durum proje mimarisi üzerinde daha iyi anlaşılabilir.

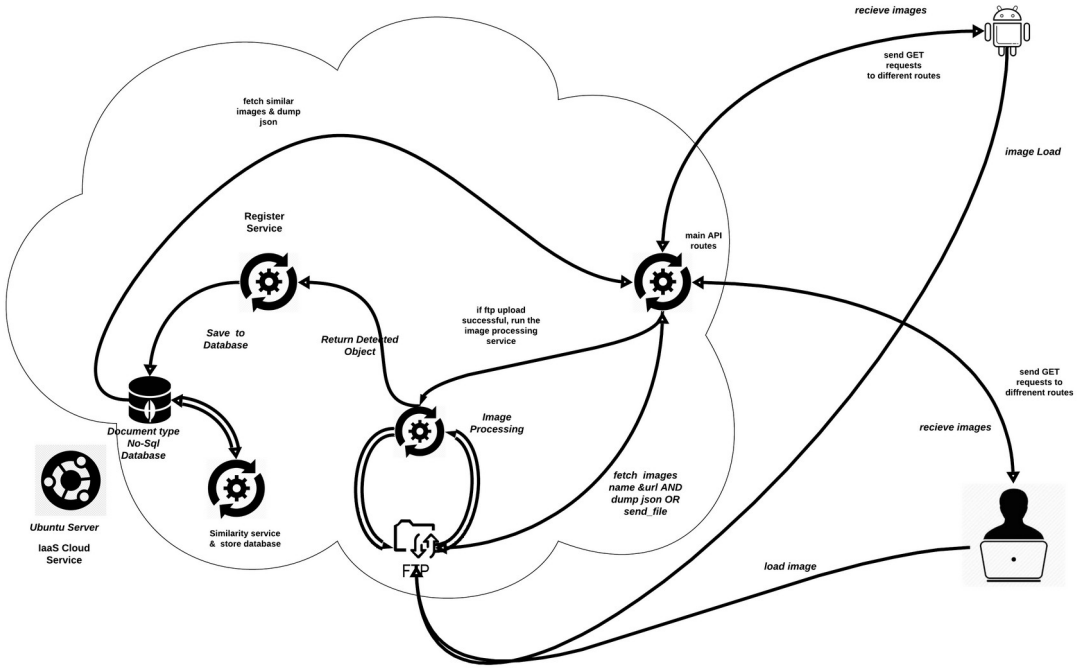
Projemiz için ilk olarak yapılanlar şu şekildedir:

- DigitalOcean üzerinde bir sunucu kiralanması.
- Kiralanan sunucuda gerekli kurulumlar tamamlandı. Bu kurulumlar MongoDB, FTP, Python3 , imageAI ve bağımlı olduğu modüller gibi ihtiyaçlardır.
- Kurulumlar tamamlanırken diğer taraftan test amacı ile bir C# Form uygulaması yazıldı. Bu uygulama içerisinde bir görsel seçilip, seçilen görselin seçilme zamanı datetime formatında alınarak sunucuya upload işlemi FTP ile yapıldı.
- Sunucu tarafında FTP'den gelen görselleri içerdikleri nesneleri tespit edecek bir servis Pyhton dilinde yazıldı. Görüntü işleme sonrası elde edilen nesneler ve görselin bu nesneleri içerme sayısı veritabanına yazılmaya hazır formata getirildikten sonra MongoDB veritabanına kayıt yapıldı.
- Form uygulamasında FTP işlemi ile transfer tamamlandıktan sonra dosyanın ismi parametre olarak verilerek üst maddede bahsi geçen servisin çalıştırlması sağlanarak görüntünün işlenmesi sağlandı.

- Native bir Android uygulaması ve .Net core ile bir web uygulaması geliştirilmeye , ön yüz tasarımları yapılmaya başlandı.
- Basit bir Flask API hazırlanıp rastgele ftp dizininde bulunan dosyaların isimlerini “json” formatına dönüştürüp GET servis route’u na eklenen bir metot hazırlandı.
- Benzer dosyaları alabilmek için tek bir görsel dosyası ismini parametre olarak yine htttp GET isteği alan bir route hazırlandı.
- Android ve Web uygulamaları üzerinden gönderilen get isteklerine karşılık görsel dosyalarının istemcilere ulaştırılması sağlandı.

Proje için temel hazırlaklar ve kontroller tamamlandıktan sonra blok diagramı ve proje mimarisi altındaki tasarım kurulmaya başlanmıştır. Tüm route’ları içerinde barındıran ve http isteklerine karşılık veren bir ana servis hazırlandı. Blok diagramında baklava dilimi ile gösterilmektedir.

3.2. Proje Mimarisi



Şekil 3.2.0 Proje mimari şeması

Projenin temel mimarisi Şekil 3.2.0 de görüldüğü şekildedir. Mobil veya web kullanıcısı tarafından bulut sunucuda bulunan FTP sunucusuna bir görsel yükleniyor. Yükleme tamamlandıktan hemen sonra dosya ismi Get isteği olarak ilgili route ‘a gönderiliyor ve saveImageInfo servisi görsel içerisindeki nesnelerin isimlerini ve sayılarını json formatına çevirerek MongoDB No-SQL veri tabandaki ilgili collection’a kayıt ekleniyor. (FTP dizininden aynı isimli görseli bulup imageai kütüphanesi ile içerisinde bulunana objeleri tespit ederek bir obje dizisi olarak çıktı veriyor. Bu obje dizisi save2Db isimli metoda parametre olarak göderilip Json formatına çevriliyor ve MongoDB No-SQL veri tabanına kayıt gerçeklşiyor.) eğer kayıt başarılı ise benzerlik tespit işlemi başlatılır ve benzer olan resimler veri tabanına kaydedilir. Uygulama başlatıldığında kullanıcıya benzer görsel kaydına sahip olan görseller gönderilir. İstemci tarafından bir görsele tıklandığında görsel ismi veri tabanında aranır ve benzer olan görseller istemciye gönderilir.

3.3. Use-Case Diagramı

Şekil 3.3.0 Use-Case Diagramı

Şekil 3.2 de görülen Use-Case diagramında “Aktor” olarak belirtilen kullanıcı arayüz görüntüleme, görsel görüntüleme, sisteme görsel yükleme ve gezinti sonucu veri üretme aksiyonlarını gerçekleştirir. Sistem yöneticisi ise kullam istatistikleri verilerini görüntüleyebilme ve üzerinde işlem yapabilme yetkisine sahiptir. “Case” ler arasında birbirine bağlı olma durumları mevcuttur(“include”). Görsellerin görüntülenebilmesi için önce arayüzün görüntülenmesi “include ” durumunu ifade eder.

3.4. FTP İstemci C# Form Uygulaması(Test süreci içerisinde kullanılmıştır)

FTP istemcisi, FTP protokolü üzerinden FTP Sunucu'ya bağlanarak dosya transferi gerçekleştiren yazılımlara denir. Bizim yaptığımız projede de dosya transferi FTP istemcisi ile sağlanıyor. Projemizde kullanıcının form uygulamasına yüklediği görseli ftp ile sunucuya atıyoruz .

```
using (var client = new WebClient())
{
    string fileNameDateTime = System.DateTime.Now.ToString("ddMMyyyyHHmmss");
    string extension = Path.GetExtension(ofd.FileName);
    extension = extension.ToLower();

    if (extension == ".png" || extension == ".jpg" || extension == ".jpeg") {

        uploadfile("ftp://46.101.154.46/" + fileNameDateTime + extension, dizin);

        GetAsync("http://46.101.154.46:5000/saveImageInfo?filename=" + fileNameDateTime + extension);

    }
    else{
        label1.Text = "Hatalı uzantı";
    }


}
```

Şekil 3.4.0. Ftp İstemci 1

fileNameDateTime değişkeni ile sunucuya atılacak olan dosyanın unique bir id alanına sahip olması için, dosyanın atıldığı tarihin günü, ayı, yılı, saati, dakikası ve saniyesi ile birlikte kaydedilir. Extension değişkeni ise image dosyasının uzantısını tutan değişken olarak kullanılmıştır. Uygulama sadece png, jpg ve jpeg uzantılı image dosyalarını sunucuya atmak için tasarlandı. Bu formata uyan dosyalar uploadFile fonksiyonu ile ilk olarak sunucuya yollanır. İşlem bittikten hemen sonra ise nesne tespit servisi için bir Get isteği ile dosya isimi gönderilir. (bkz. 3.4.0)

cd

```
private void uploadfile(string url, string file)
{
    FtpWebRequest request = (FtpWebRequest)WebRequest.Create(url);
    request.Credentials = new NetworkCredential("yusufnuh", "aysegul");
    request.Method = WebRequestMethods.Ftp.UploadFile;

    using (Stream fileStream = File.OpenRead(@file))
    using (Stream ftpStream = request.GetRequestStream())
    {
        byte[] buffer = new byte[10240];
        int read;
        while ((read = fileStream.Read(buffer, 0, buffer.Length)) > 0)
        {
            ftpStream.Write(buffer, 0, read);
            label1.Text = "Uploaded "+ fileStream.Position+" bytes";
        }
    }
}
```

Şekil 3.4.1 Ftp İstemci 2

uploadFile fonksiyonu ile image dosyasını ftp aracılığı ile sunucuya atıyoruz. Ftp ye bağlanmak için öncelikle user ve password bilgilerini girmemiz gereklidir. Bu bilgiler girilmeden sunucuya erişilemez. Sunucuya dosya aktarım işleminden sonra gönderilen dosyanın boyutu ekrana yazdırılıyor.(bkz. Şekil 3.4.1)

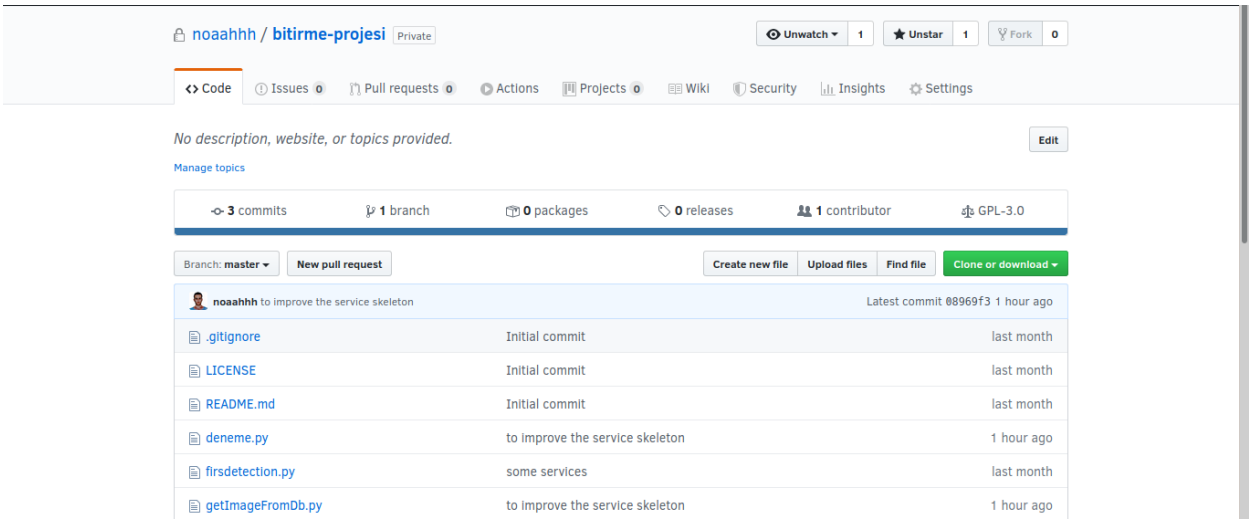
```
1 başvuru
public async Task<string> GetAsync(string uri)
{
    HttpWebRequest request = (HttpWebRequest)WebRequest.Create(uri);
    request.AutomaticDecompression = DecompressionMethods.GZip | DecompressionMethods.Deflate;

    using (HttpWebResponse response = (HttpWebResponse)await request.GetResponseAsync())
    using (Stream stream = response.GetResponseStream())
    using (StreamReader reader = new StreamReader(stream))
    {
        return await reader.ReadToEndAsync();
    }
}
```

Şekil 3.4.2 Ftp İstemci 3

Şekil 3.4.2 de görülen metot ise nesne tespit servisini çalıştırmak için bir http Get isteği oluşturmak için kullanılmıştır.

3.5. Versiyon yönetimi kullanımı



Şekil 3.5.0 Github repository

Versiyon yönetemi github private repository açıldı ve sunucuda yer alan servisler commit edildi.(bkz. Şekil 3.5.0)

Servislerin uzak bir sunucuda güvenle tutulması ve servislere ekip üyelerinin müdahale etmesi sağlanmış oldu.

repo link: https://github.com/noaahhh/image-recommender-system

3.6. Uygulama servisleri

Uygulama servisleri flask freamwork'ü kullanılarak oluşturulmuştur. Yönlendiriciler "main.py" servisi altında istemci isteklerine cevap üretebilecek şekilde kodlanmıştır. Diğer servisler kullanıcak metotların tanımlandığı, nesne tespit servisi, benzerlik tespit eden servis ve mongoDB veritabanı bağlantı ve kayıt metotlarını içeren servislerden oluşmaktadır.

Şekil 3.6.0 de servisine ait bir ekran görüntüsünde /api/bitirme//images/all yönlendiricisine ait fonsiyonun içeriğini görebilmekteyiz.

```python
@app.route('/api/bitirme/images/all',methods=['GET'])
def getAll():

#    random.shuffle(entries)
    entries = getAllImage()
    for entry in entries:
        frame={"name" : "null" , "url" : "null"}
        url = "http://138.197.195.224:5000/api/bitirme/images?name=" + entry
        frame.update({"name" : entry})
        frame.update({"url": url})
        images.append(frame)
        #if len(images) == 5:
            # break
    jsonobject=json.dumps(images)
    images.clear()
    return jsonobject
```

Şekil 3.6.0. /api/bitirme//images/all route

Servis, manageDb modülü içerisindeki getAllImage metodu ile birbibine benzeyen görsellerin isimlerini veri tabanından çekerek entries isimli listeye kopyalar. Daha sonra görsel isimleri name kısmına ve bu fotoğrafları gönderen servisi url kısmına ekleyen bir JSON obje dizisini döndürür.

Şekil 3.6.1 de ise /api/bitirme//images/get?img= isimli bir parametre alan yönlendiici fonksiyonuna ait kod bloğu mevcuttur.

```python
@app.route('/api/bitirme/images/get',methods=['GET'])
def getByImageName():

    if 'img' in request.args:
        img =request.args['img']
    else:
        return "Error: No e field provided. Please specify an name."

    result=getSimilarImages(img)
    for entry in result:
        frame={"name" : "null" , "url" : "null"}
        url = "http://138.197.195.224:5000/api/bitirme/images?name=" + entry
        frame.update({"name" : entry})
        frame.update({"url": url})
        images.append(frame)

    jsonobject=json.dumps(images)
    images.clear()

    return jsonobject
```

Şekil 3.6.1. /api/bitirme//images/get route

Bu foksiyon bir adet fotoğraf ismi parametre alarak kontrol eder. ManageDb modülü içerisindeki getSimilarImages metoduna parametre olarak aldığı ismi gönderir ve bu görsel ile benzer olan görselleri bir liste olarak dödürür.

Daha sonra görsel isimini ve URI olarak görsel dosyasını gönderen servisi url kısmına ekleyerek objeler oluşturur. Json formatına dönüştürerek istemciye geri gönderir.

```
53  @app.route('/api/bitirme/images',methods=['GET'])
54  def getSingle():
55
56      if 'name' in request.args:
57          name =request.args['name']
58      else:
59          return "Error: No name field provided. Please specify an name."
60
61      return send_file('/home/yusufnuh/files/'+name)
62
```

Şekil 3.6.2 . send_file fonksiyonu

Şekil 3.6.2 da */api/*bitirme/images?name= isimli bir parametre alan bir route fonksiyonunu görmekteyiz. Burada parametre olarak verilen dosya ismini geri döndüren Flask fonksiyonu oln send_file ile isteğe karşılık dosya gönderilmektedir.

```
64  @app.route('/api/bitirme/saveImageInfo',methods=['GET'])
65  def saveImageInfo():
66
67      if 'img' in request.args:
68          img=request.args['img']
69      else:
70          return "Error: No img field provided. Please specify an img."
71
72      results=saveImageInfo2Db(img)
73      return results
```

Şekil 3.6.3 . nesne tesbitinin başlatılması

Şekil 3.6.3 ‘ te nesne tespit servisini çalıştırmak için ftp upload işleminden hemen sonra isyemci tarafından gönderilen dosya ismini parametre olarak alan ve manageDb modülü içerisindeki saveImage2Db metoduna parametre olarak gönderen bir API route’unu görmekteyiz. İsteğe karşılık olarak nesne tespit sonuçlarını gere döndürür.

Diğer servis foksiyonları ise şu şekildedir:

- **similarityService()**

```
92                      result=cosine_similarity([a],[b])
93                      if result >= 0.5:
94                          similar.append(name)
95                      print(result)
96                      a.clear()
97                      b.clear()
98              print("iterasyon sonu")
99              names_1.clear()
100             print(similar)
101
102             bolean=mycollection2.find({"name": query}).count()
103             print(bolean)
104             if bolean is 0:
105         #         print("buradayım non")
106                 mycollection2.insert_one({"name":query, "similar_images":similar})
107             else:
108                 mycollection2.update_one({"name": query},{ "$set" :{"similar_images":similar}})
109             similar.clear()
```

Şekil 3.6.4 . Benzerlik tesbiti ve veri tabanı kaydı

Şekil 3.6.4'te similarity.py içerisinde bulunan similarityService() metodunun bir bölümü görülmektedir. Metot, veri tabanına yeni bir nesne tespit kaydı eklendiğinde tüm kayıtları tarar. Nesne isimlerini ve sayılarını kullanarak karşılaştırılmak üzere iki liste oluşturur. Ve listelerin cosinus benzerliği 0.5 ten büyük ise "similar" isimli listeye ekler. Daha sonra similar_image collection'ı içinde kayıt varsa güncelleme ,yoksa yeni kayıt ekleme işlemi yapar.

- **SaveImageInfo()**

```
7   def saveImageInfo2Db(filename):
8
9       #filename = request.args.get("filename")
10      execution_path = "/home/yusufnuh/services"
11      imagefile_path ="/home/yusufnuh/files"
12      outputfile_path="/home/yusufnuh/out"
13      detector = ObjectDetection()
14      detector.setModelTypeAsRetinaNet()
15      detector.setModelPath( os.path.join(execution_path , "resnet50_coco_best_v2.0.1.h5"))
16      detector.loadModel()
17      detections = detector.detectObjectsFromImage(input_image=os.path.join(imagefile_path , filename), output_image
18
19      output = "SONUCLAR: | "
20      detectedObject=[]
21      for eachObject in detections:
22          output = output + "  |  " + eachObject["name"] + " : " + str(eachObject["percentage_probability"] )
23          detectedObject.append(eachObject["name"])
24
25      print(detectedObject)
26      resultObject=save2Db( filename, detectedObject)
27      if resultObject.acknowledged is True:
28          similarityService()
29
30      return output
```

Şekil 3.6.5 . Nesne tespit servisi

Şekil 3.6.5'teki servis foksiyonu dosya ismini aldıktan sonra ftp klasöründen resmi alarak Resnet50 modeli ve ImageAI metotları ile nesneleri ve olabilirlik yüzdesini parametre olarak döner. Daha sonra bu sonucu veri tabanı kaydı için save2Db metoduna dosya ismi ile berarber parametre olarak gönderir. Metot kayıt yaptıktan sonra dönen bilgilendirme mesajı "True" ise "similarityService" i çalıştırır ve benzerlik tesbiti işlemi başlar.

- **Veritabanı metotları**

```
9   def save2Db (filename,detectedObject):
10      mycollection=connection("image_properties")
11      jsondata = {}
12      jsondata['_id'] = filename
13      jsondata['objects'] = []
14      for eachObjectName in detectedObject:
15          objectDict={}
16          value=detectedObject.count(eachObjectName)
17          objectDict['name']=eachObjectName
18          objectDict['count']=value
19          if objectDict not in jsondata['objects']:
20              jsondata['objects'].append(objectDict)
21
22
23      result_object=mycollection.insert_one(jsondata)
24      return result_object
25
26  def getSimilarImages(img):
27
28      mycollection=connection("similar_images")
29      doc=mycollection.find({"name":img})
30      for key in doc:
31          array=key["similar_images"]
32      print(array)
33      return array
34
35  def getAllImage():
36      array=[]
37      mycollection=connection("similar_images")
38      doc=mycollection.find({"similar_images": { "$exists" : 1, "$ne": [] } })
39      for key in doc:
40          array.append(key["name"])
41      print(array)
42      return array
```

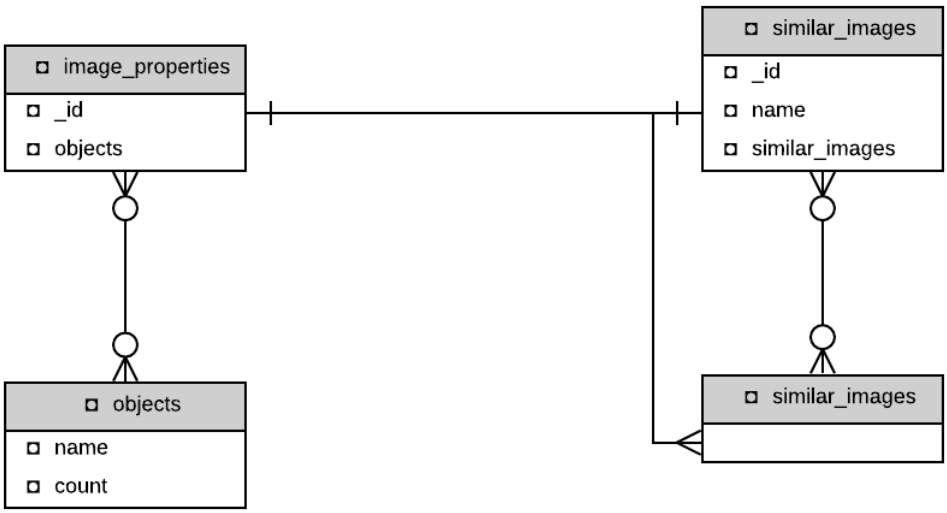
Şekil 3.6.6 . veri tabanı metotları

Şekil 3.6.6 da ise veri tabanı işlemleri için kullanılan metotlar bulunmaktadır. Save2Db() metodu tespit edilen objeleri istenilen formata çevirip doküman olarak veri tabanına kaydeder. GetSimilarImages() metodu ismi verilen dosyanın kaydını veri tabanında bulup benzer olduğu resimlerin isimlerini döndürür.getAllImage() metotu ise sadece benzer görsel kaydı bulunan kayıtlardaki isimleri döndürür.

3.7. Veri tabanı kayıtları

Veri tabanı kaydı yapılan "image_properties" ve "similar_images" olmak üzere iki adet "collection" 'dan oluşmaktadır.

Veri tabanı kullanılırken sadece hızlı bir şekilde depolama ve yapılan sorgulara hızlı cevap alma amacı güdülmüştür. Bu sebeple collection'lar arasında bir ilişki kurulmamış veya veritabanı detaylandırılmamıştır. Veritabanına ait Nosql veri model diagramı Şekil 3.7.0 da görüldüğü gibidir.



Şekil 3.7.0 . Nosql veri modeli

image_properties adlı collection, "_id" ve "object" listesinden oluşur. Objects listesi ise "name" ve "count" olmak üzere iki alandan oluşur. Aralarında çok-çok ilişkisi vardır. Bir görselin hiçbir ojesi olmayacağı gibi bir obje de hiçbir görsele ait olmayabilir.

Similar_images adlı collection "_id", "name", "similar_images" isimli alanlara sahiptir. similar_images bir listedir. Ve içerisinde "key-value" şeklinde kayıt turmaz. Görsel isimlerini değer olarak alır.

image_properties "_id" alanı ile similar_images "name" alanı aynıdır. Ve aralarında birebir ilişki vardır. Kayıt sayısı aynı olmalıdır.

Bu “collection”larda bulunan kayıtlar ve formatları sorasıyla şekil 3.7.1 ve Şekil 3.7.2 de görülebilmektedir.

```
/* 1 */
{
    "_id" : "23052020180618.jpeg",
    "objects" : [
        {
            "name" : "cup",
            "count" : 1
        },
        {
            "name" : "fork",
            "count" : 1
        },
        {
            "name" : "spoon",
            "count" : 1
        },
        {
            "name" : "dining table",
            "count" : 1
        }
    ]
}
```

Şekil 3.7.1. “image_properties” ‘e ait bir kayıt

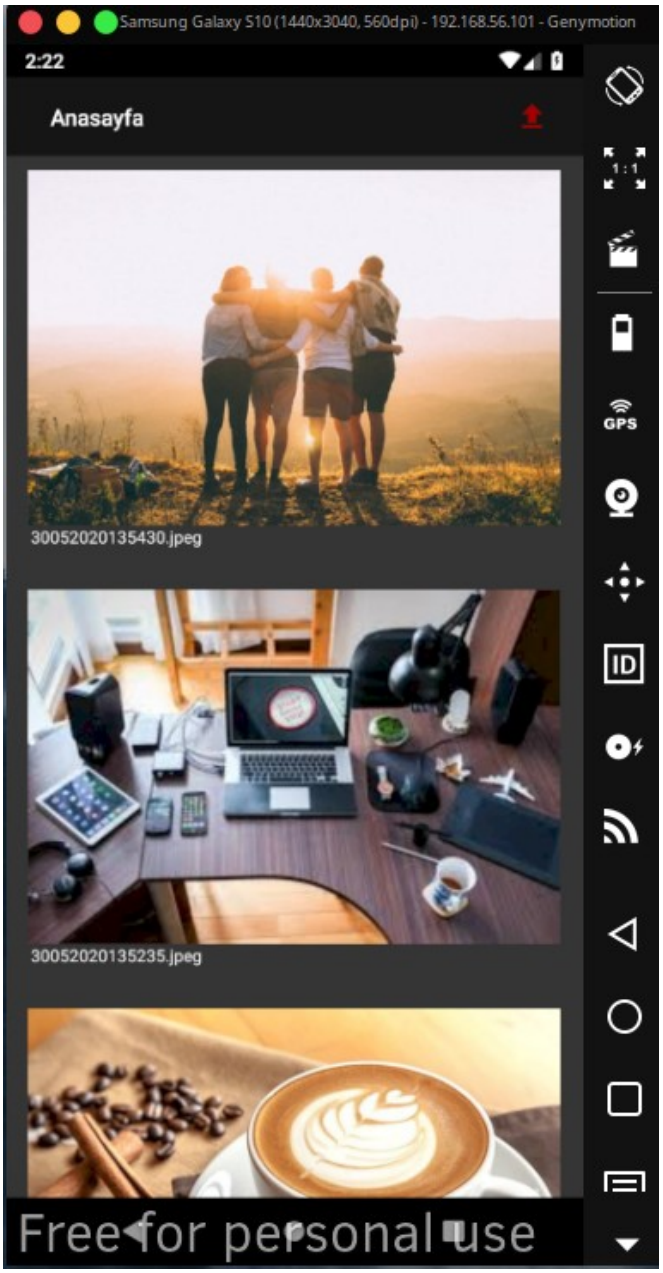
```
{
    "_id" : ObjectId("5ececbb6b739928b01acb2aa"),
    "name" : "23052020182954.jpg",
    "similar_images" : [
        "23052020181833.jpg",
        "23052020183739.png",
        "27052020235914.jpeg"
    ]
}

/* 4 */
{
    "_id" : ObjectId("5ececbb6b739928b01acb2ab"),
    "name" : "23052020183739.png",
    "similar_images" : [
        "23052020181833.jpg",
        "23052020182954.jpg",
        "27052020235914.jpeg"
    ]
}
```

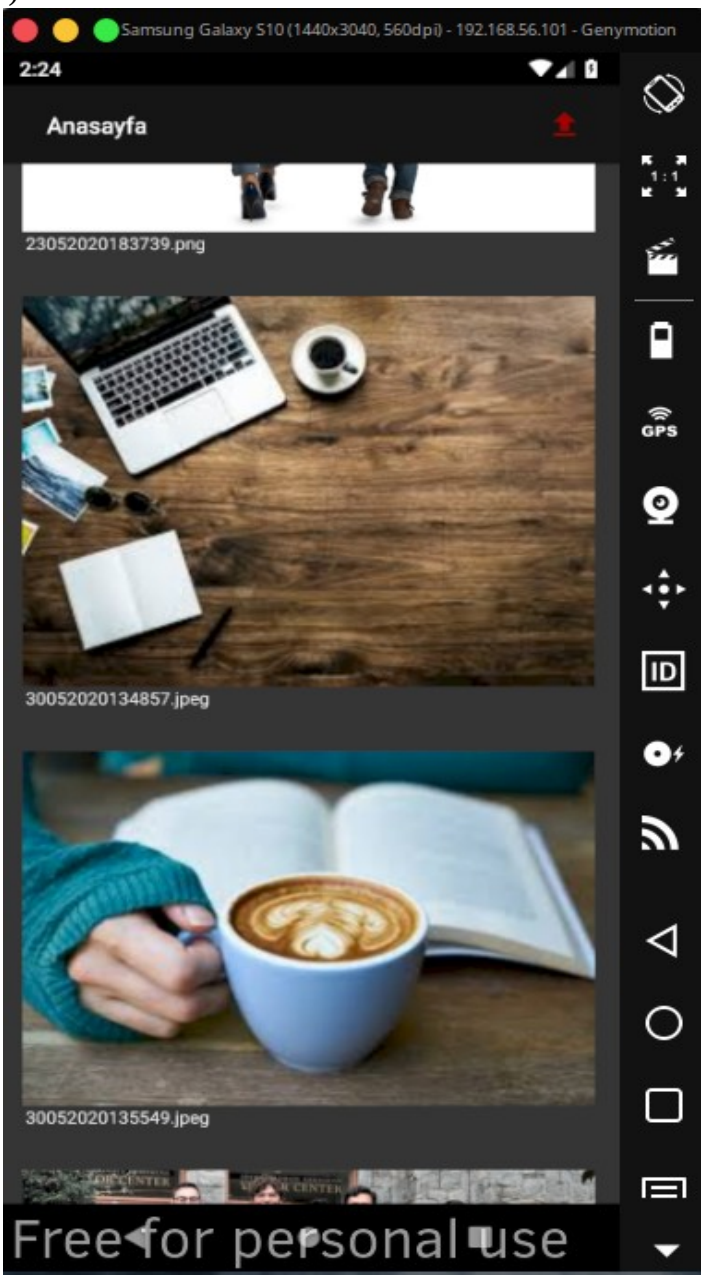
Şekil 3.7.2. “image_properties” ‘e ait bir kayıt

3.8. Android client uygulamasının geliştirilmesi

Android uygulaması içerisinde uygulama başlatıldığında yulkarıda belirtilen servis fonksiyonlarından ilk route a istek atılır ve rastgele fotoğraflar arayüzde gösterilmektedir.(bkz Şekil 3.8.0 ve Şekil 3.8.1)

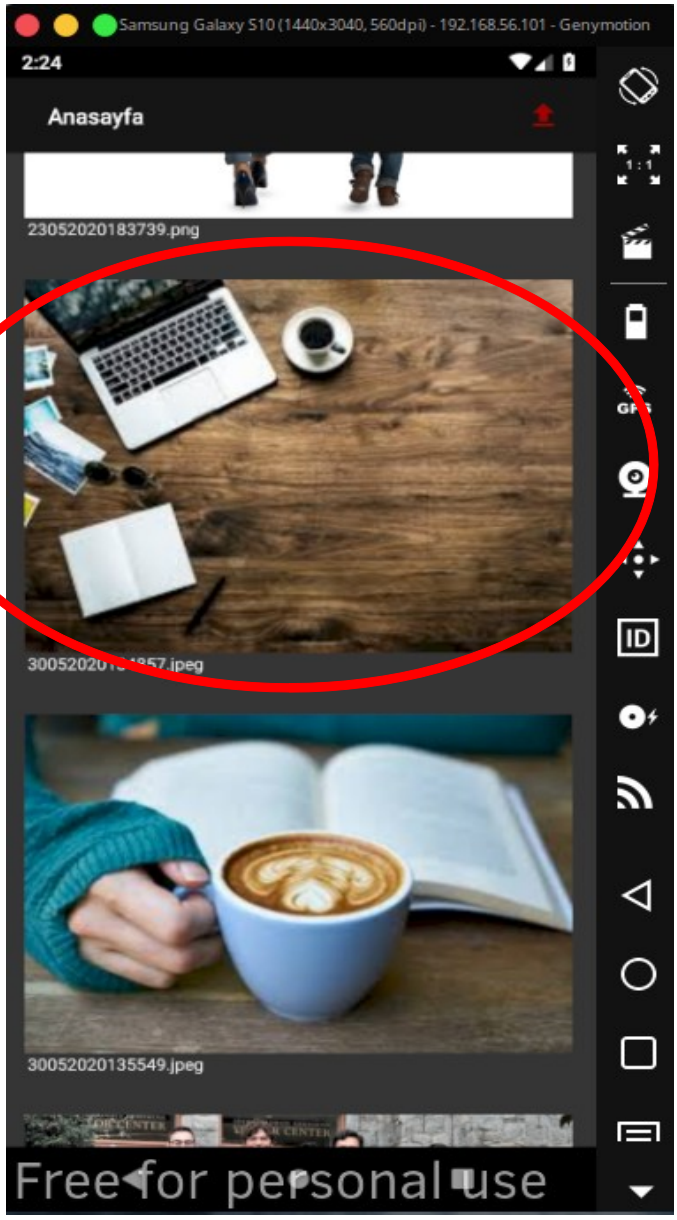


Şekil 3.8.0. anansayfa 1

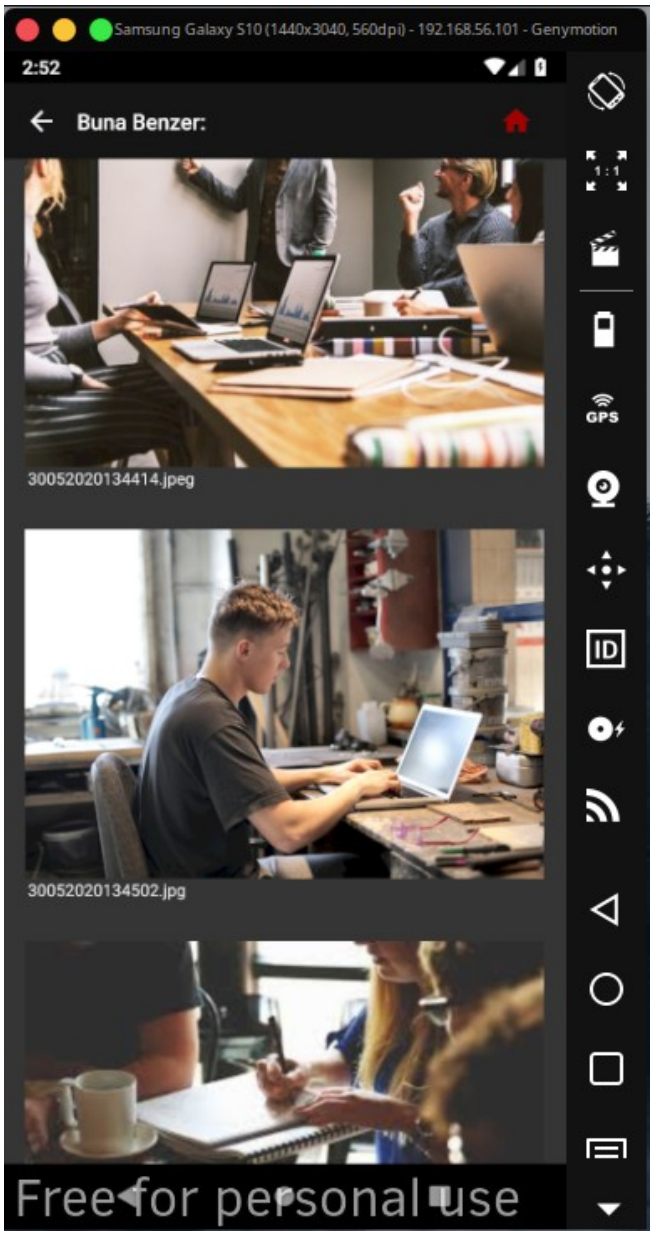


Şekil 3.8.1. anansayfa 2

Resimlerden birine tıklandığında ise yeni bir sayfada “buna benzer:” resim ler listelenmektedir.



Şekil 3.8.2. Seçilen görsel



Şekil 3.8.3. Benzer görseller

Şekil 3.8.2 de ve Şekil 3.8.3 de seçmeden önce ve seçtikten sonraki değişimleri ve sonuçları görebilmekteyiz.

3.9. Web client uygulamasının geliştirilmesi

Web uygulaması içerisinde uygulama başlatıldığında yulkarıda belirtilen servis fonksiyonlarından ilk route a istek atılır ve rastgele fotoğraflar arayüzde gösterilmektedir.(bkz. Şekil 3.9.0)

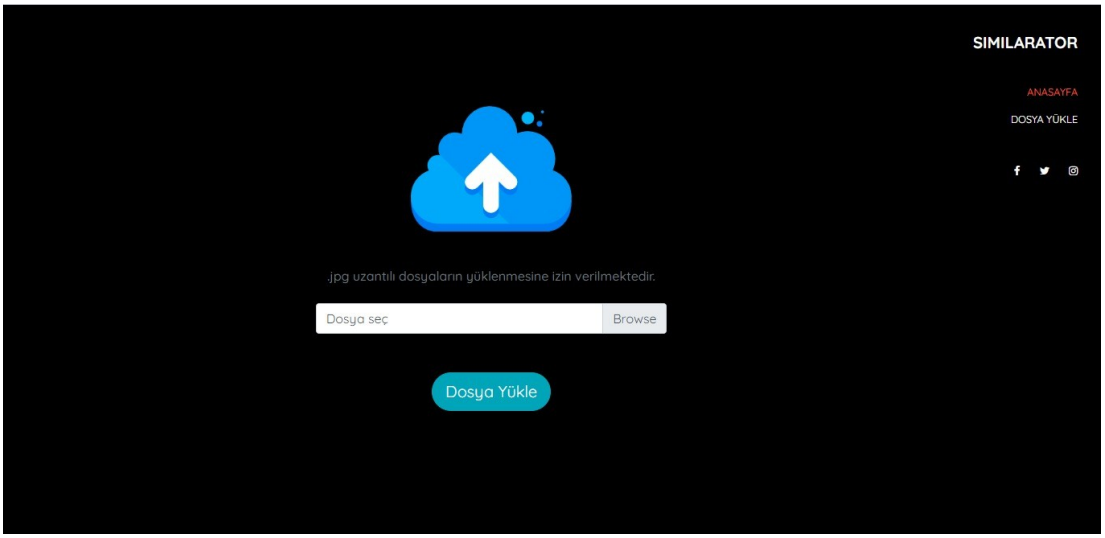
Şekil 3.9.0. Anasayfa (web)

Daha sonra herhangi bir fotoğrafa tıklandında bu fotoğrafa benzer görseller kullanıcıya sunulmaktadır. Şekil 3.9.0’ de seçilen kahve bardağı görseline benzer olan görseller Şekil 3.9.1. , Şekil 3.9.2 görülmektedir.

Şekil 3.9.1. Benzer görseller (web)

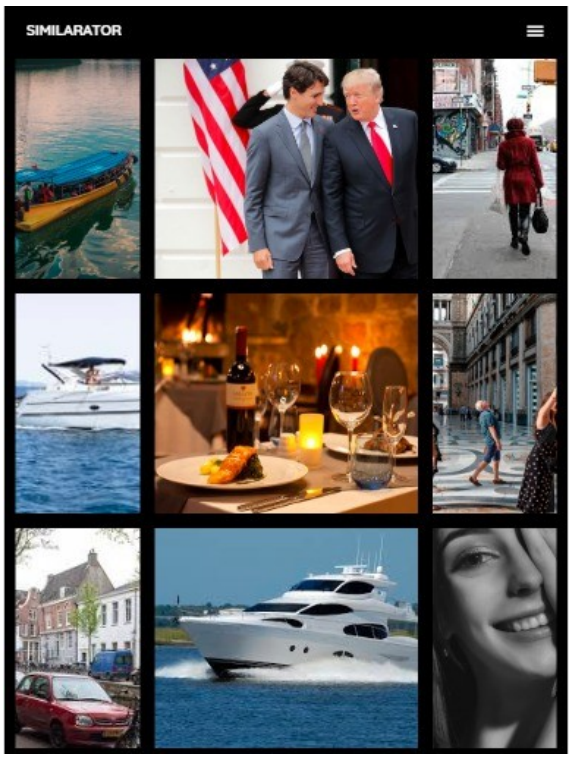
Şekil 3.9.2. Benzer görseller 2 (web)

Şekil 3.9.2'de uygulama içindeki dosya yükleme modülü görülmektedir. Dosyalar sunucudaki ftp dizinine gönderilmektedir.

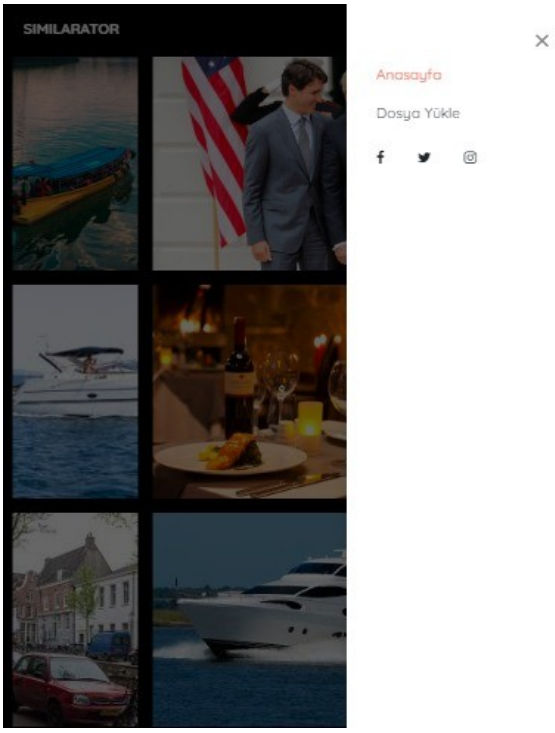


Şekil 3.9.3. Ftp upload sayfası (web)

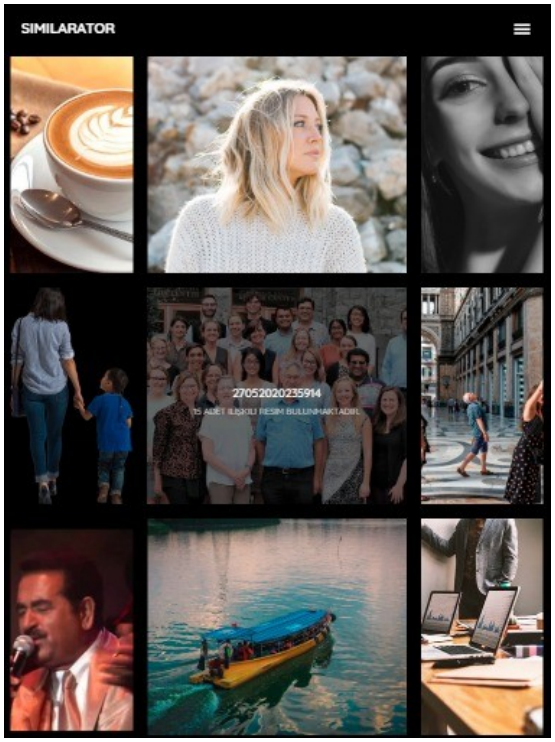
Web uygulamana “responsive” özelliği eklenmiş olup ekran görüntüleri görüldüğü gibidir.



Şekil 3.9.4. Anasayfa 1(Responsive)



Şekil 3.9.5. Anasayfa 2(Responsive)



Şekil 3.9.6. Anasayfa 3(Responsive)

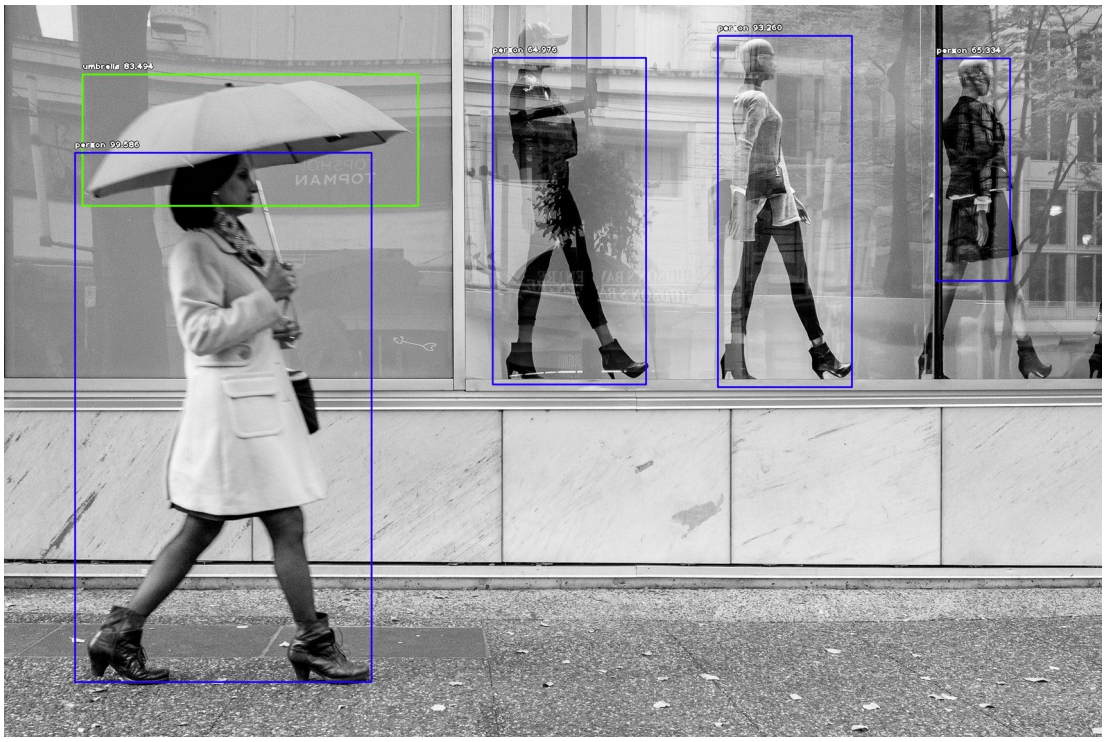
Şekil 3.9.6 da seçilen görsele benzeyen görseller ise Şekil 3.9.7 de görüldüğü gibidir. Obje bazlı olarak benzerlik tesbitini burada görebilmekteyiz.

Şekil 3.9.7. Benzer görseller (Responsive)

Benzerlik algoritması şekil 3.9.7'de görülen görseldeki gibi insan olan fotoları benzer olarak sınıflandırmıştır. Bir isnan fotoğrafı seçildiğinde içeriisinde insan bulunan fotoğraflar kullanıcıya sunulmaktadır ve bu işlem rekürsiftir.

3.10. Nesne Tespit Testleri

Nesne tespiti ile yapılan testlerde imageai kütüphanesinin oldukça başarılı sonuçlar verdiği kaydedildi. Görseller arasındaki benzerlik tesbitinde yeterli karakteristiğe sahip çıktıların üretilmesini sağladığı gözlemlenmiştir.



Şekil 3.9.0. Tespit Edilmiş Nesneler

Şekil 3.9.0 ‘da görüldüğü gibi görsel içerisindeki birçok nesneyi tespit edibilmekte oldukça başarılı sonuçlar vermiş ve test sonuçlarında bu açıkça görülebilmektedir.

SONUCLAR: | | cell phone : 53.288471698760986 | potted plant : 82.01721906661987 | laptop : 69.60868835449219 | book : 63.27598690986633 | book : 81.18999600410461 | book : 75.5860984325409

Şekil 3.9.1 . nesne tespit sonuçları

Şekil 3.9.1 de bir resim servise verildikten sonra yapılan işlemlerden sonra çıktı olarak tespit edilen nesnelerin isimleri ve nesne sınıfına ait olabilirliklerini görebilmekteyiz.

BÖLÜM 4. SONUÇLAR VE ÖNERİLER

4.1.Proje Genel Özeti

Proje tasarımında sosyal medya ,e-ticaret siteleri, arama motorları gibi ilgili içeirğe benzer içeriklerin sunulması, Android için bir mobil uygulama, web içinde .net core kullanılan bir web uygualaması isteöcileri ile bu hizmete erişilebilmesi amaçlanmış, tasarımı yapılmış ve uygulanmıştır.

Tasarımın uygulanması için bir adet Iaas sunucu hizmeti kiralanarak, yazılım projesinin kullanacağı kaynaklar, servisler, sunucu uygulamaları, veri tabanı kurulumları yapılmıştır. Flask freamwork'ü kullanılarak hazırlanan API ve route'lar hazırlanarak istemci uygulamalarına hizmet verilmesi sağlanmıştır.

İçerik öneri sistemi oluşturulurken görsellerin içerdiği obje isimleri ve sayılarını kullanarak iki görsel arasında cosinus benzerliği hesabı yapan bir algoritma kullanılmıştır.

4.2. Proje ile Sunulan Çözümler

Tasarımını yapmış olduğumuz projede kullanıcının ilgi alanına göre ayrıca bir sorgu yapmaması amaçlanmış olup, kullanıcı ilgilendiği içerikle ilgili görsele tıkladığı zaman , benzer içerikteki diğer görseller getirilmesi ve böylelikle kullanıcının istediği görsellere kolay ve zahmetsiz ulaşmasına yönelik geliştirilmiş bir projedir.

Genellikle doküman benzerliği için kullanılan cosinus benzerliği yönteminin görsel içerikler arasındaki benzerliği tespit etme ve kullanıcıya sunulması için kullanılması amaçlanmıştır.

4.3. Tasarımın Uygulanmasında Karşılaşılan Problemler ve Çözüm Yolları

- Hazırlanan servislerin servislerin çalıştırılması için ve görsel dosyalarının depolanması için kullanılacak olan işlemci, bellek ve depolama kaynaklarının bir maliyeti vardır. Proje geliştirilme sürecinde “Github developer pack” tarafından sunulan hibeler maliyeti karşılamıştır.

- Görsel dosyası sayısı arttıkça depolama alanı ihtiyacı doğacaktır. Kullanıcı sayısı arttıkça da hizmet süresinde aksama ve uzama gerçekleşebilir. Çözüm olarak donanım özelliklerini yükseltmek sorunu çözecektir.

- Uygulamanın geliştirme sunucusu üzerinde kullanılması ilerde süreç içerisinde aksaklıklar oluşturulabilir. Bunun için Apache server ile birleştirilmesi ve “production server ” üzerine kurulması problemi çözecektir.

- Benzerlik tespiti için doküman benzerliği içinde kullanılan bir metot kullanıldığı için çok sayıda nesnenin bulunduğu görseller içeirindeki benzerliği tespit etmede başarılıdır. Az sayıda veya bir adet nesne içeren görsellerde sayı faktörünün etkisi oldukça azdır.

- Belirli sayıdaki nesnelelrin tespit edilebilmesi sistem performansını azaltmaktadır. Çözüm olarak farklı nesneler ile eğitilebilen bir nesne tespit modeli ve uygulaması kullanılabilir. Böylece tanınan nesne veritabanı artar ve algoritma performansı da gelitirilebilir.

KAYNAKLAR

[1]	Object Detection with 10 lines of code, MOSES OLAFENWA, https://towardsdatascience.com/object-detection-with-10-lines-of-code-d6cb4d86f606 -Object Detection with 10 lines of code
[2]	Detection Classes , http://imageai.org/ , Erişim Tarihi: 29.12.2019
[3]	https://imageai.readthedocs.io/en/latest/detection/index.html ,Erişim Tarihi : 28.12.2019
[4]	Pedestrian Detection in Aerial Images Using RetinaNet ,Priyanka Kochhar, https://www.kdnuggets.com/2019/03/pedestrian-detection-aerial-imagesretinanet.html
[5]	Bulut Bilişim Nedir? , TUĞBA AYVAZ , https://www.mediaclick.com.tr/blog/bulut-bilisim-nedir, Erişim Tarihi: 29.12.2019
[6]	FTP Nedir ve Nasıl Kullanılır?, https://www.hosting.com.tr/bilgi-bankasi/ftp-nedir-ve-nasil-kullanilir/ Erişim Tarihi: 26.12.2019
[7]	FTP Sunucusu Nedir?, TUĞBA ATAKAN , https://blog.isimtescil.net/ftp-sunucusu-nedir/ , Erişim Tarihi: 26.12.2019
[8]	NoSQL Nedir Avantajları ve Dezavantajları Hakkında Bilgi, SAVAŞ DAVAZ, https://kodcu.com/2014/03/nosql-nedir-avantajlari-ve-dezavantajlari-hakkinda-bilgi/ ,Erişim Tarihi: 26.12.2019
[9]	Nedir Bu MongoDB ?, BERKE KURNAZ , https://medium.com/@berkekurnaz/nedir-bu-mongodb-994a94a9d1df, Erişim Tarihi: 26.12.2019

Kaynaklar(Şekilller)

[1]	Şekil 2.1.0, Şekil 2.1.1,Şekil 2.1.2, Şekil 2.1.3 https://imageai.readthedocs.io/en/latest/detection/index.html, Erisim tarihi :14.04.2020
[2]	Şekil 2.2.0 https://d3i71xaburhd42.cloudfront.net/9d8747468f0fed8e335656d7fe9737e4dc21c798/1-Figure1-1.png , Erişim tarihi : 14.04.2020
[3]	Şekil 2.3.0 https://www.machinelearningplus.com/wp-content/uploads/2018/10/soft-cosine.png,Erişim tarihi: 15.05.2020
[4]	Şekil 2.3.1 https://wikimedia.org/api/rest_v1/media/math/render/svg/1d94e5903f7936d3c131e040ef2c51b473dd071d,Erişim tarihi : 16.05.2020
[5]	Şekil 2.3.2 https://scikit-learn.org/stable/modules/generated/sklearn.metrics.pairwise.cosine_similarity.html, Erişim tarihi : 16.05.2020
[6]	Şekil 2.4.0 https://en.wikipedia.org/wiki/Flask_(web_framework), Erişim tarihi : 21.05.2020
[7]	Şekil 2.6.0 https://nesildc.com/Assets/filemanager/source/blog/9aOyX9.png, Erişim tarihi : 21.05.2020
[8]	Şekil 2.8.0 https://miro.medium.com/max/640/1*ngkHgQq7ij1NBNr62er3zA.png, Erişim tarihi : 22.01.2020
[9]	Şekil 2.8.1 https://encrypted-tbn0.gstatic.com/images?q=tbn%3AANd9GcSWShoRA2bVPqUBowhD-wxgLtSdusN0J1DHZTR7HRlxqM0rH7W9&usqp=CAU, Erişim tarihi : 25.05.2020

BSM 498 BİTİRME ÇALIŞMASI
DEĞERLENDİRME VE SÖZLÜ SINAV TUTANAĞI

KONU : İçerik Öneri Sistemi
ÖĞRENCİLER (Öğrenci No/AD/SOYAD): b161210018/Nuh Yurduseven
b151210036/Yusuf Taha Öztürk
b171210375/Murat Pekuz

Değerlendirme Konusu	İstenenler	Not Aralığı	Not
Yazılı Çalışma			
Çalışma klavuza uygun olarak hazırlanmış mı?	x	0-5	
Teknik Yönden			
Problemin tanımı yapılmış mı?	x	0-5	
Geliştirilecek yazılımın/donanımın mimarisini içeren blok şeması (yazılımlar için veri akış şeması (dfd) da olabilir) çizilerek açıklanmış mı?	x		
Blok şemadaki birimler arasındaki bilgi akışına ait model/gösterim var mı?	x		
Yazılımın gereksinim listesi oluşturulmuş mu?	x		
Kullanılan/kullanılması düşünülen araçlar/teknolojiler anlatılmış mı?	x		
Donanımların programlanması/konfigürasyonu için yazılım gereksinimleri belirtilmiş mi?			
UML ile modelleme yapılmış mı?	x		
Veritabanları kullanılmış ise kavramsal model çıkarılmış mı? (Varlık ilişki modeli, noSQL kavramsal modelleri v.b.)	x		
Projeye yönelik iş-zaman çizelgesi çıkarılarak maliyet analizi yapılmış mı?	x		
Donanım bileşenlerinin maliyet analizi (prototip-adetli seri üretim vb.) çıkarılmış mı?			
Donanım için gerekli enerji analizi (minimum-uyku-aktif-maksimum) yapılmış mı?			
Grup çalışmalarında grup üyelerinin görev tanımları verilmiş mi (iş-zaman çizelgesinde belirtilebilir)?	x		
Sürüm denetim sistemi (Version Control System; Git, Subversion v.s.) kullanılmış mı?	x		
Sistemin genel testi için uygulanan metotlar ve iyileştirme süreçlerinin dökümü verilmiş mi?			
Yazılımın sızma testi yapılmış mı?			
Performans testi yapılmış mı?			
Tasarımın uygulamasında ortaya çıkan uyumsuzluklar ve aksaklıklar belirtilerek çözüm yöntemleri tartışılmış mı?			
Yapılan işlerin zorluk derecesi?	x	0-25	
Sözlü Sınav			
Yapılan sunum başarılı mı?	x	0-5	
Soruları yanıtlama yetkinliği?	x	0-20	
Devam Durumu			
Öğrenci dönem içerisindeki raporlarını düzenli olarak hazırladı mı?	x	0-5	
Diğer Maddeler			
Toplam			

DANIŞMAN (JÜRİ ADINA):

DANIŞMAN İMZASI: