KOCAELİ ÜNİVERSİTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

BLM 307: YAZILIM LAB. II 2020-2021 BAHAR- PROJE 1

WEB İNDEKSLEME UYGULAMASI

Tayfun KUŞÇU 170201042

Sinan BALCIOĞLU 130202041

ÖZET

Verilen bir URL'deki web sayfa içeriğine göre diğer birden fazla web sayfasını benzerlik bakımından indeksleyip sıralayan web tabanlı bir uygulama geliştirmek. Böylece bu proje sayesinde web indeksleme yöntemleri hakkında bilgi edinilmesini ve web tabanlı uygulama yazma becerisinin geliştirilmesi amaçlanmaktadır.

1.Giriş

Projede web indeksleme alanındaki isterler 5 başlık altında incelenmiştir;

- Sayfada Geçen Kelimelerin Frekanslarını Hesaplama
- Anahtar Kelime Çıkarmak
- İki Sayfa (URL) Arasındaki Benzerlik Skoru Hesaplama
- Site İndeksleme ve Sıralama
- Semantik Analiz

Her başlık ile alakalı işlemlerin sonucu ilgili sayfanın sonuç sayfasında gösterilmektedir.

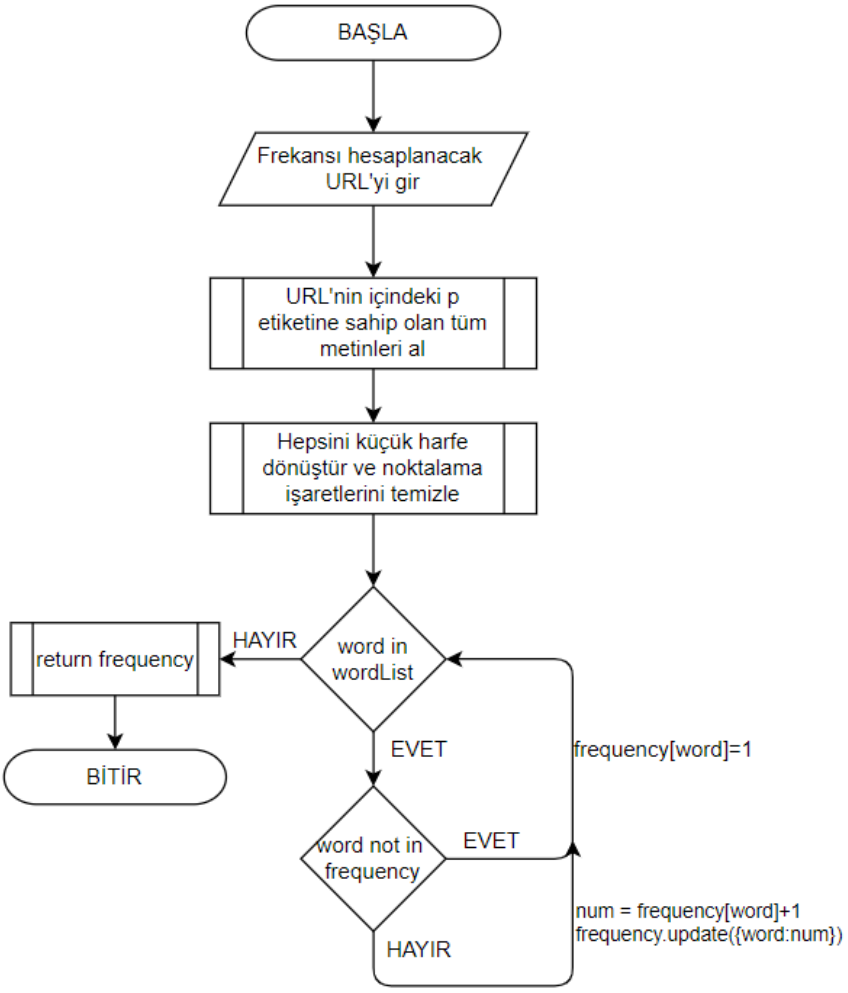
2.Temel Bilgiler

Uygulama Python dilinde Django framework'ü kullanılarak geliştirilmiştir. Verilen URL'lere istek gönderilmesi için *request* modülü, URL içeriklerine ulaşmak ve ayıklamak için *BeautifulSoup* modülü, sayfa tasarımları için ise *Bootstrap* kütüphanesi kullanılmıştır.

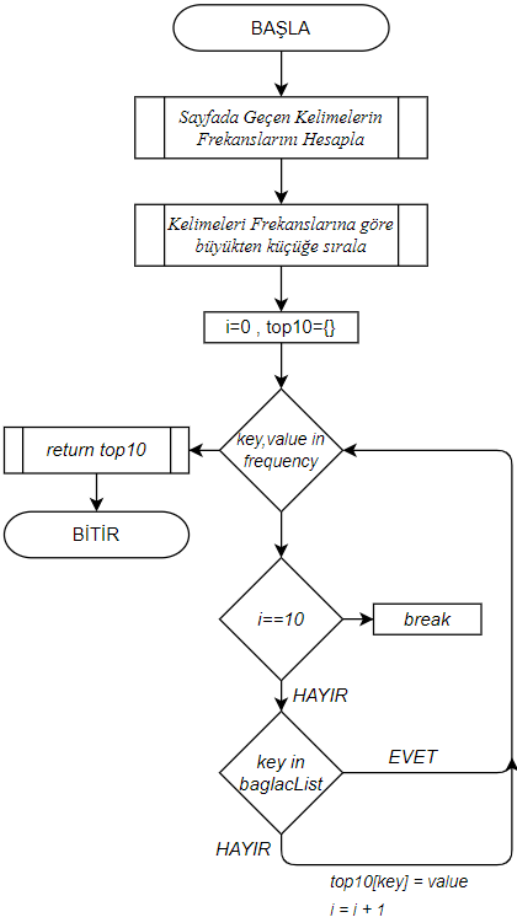
3.Tasarım

Web İndeksleme Uygulamasının yapım aşamaları altta belirtilen başlıklar altında açıklanmıştır.

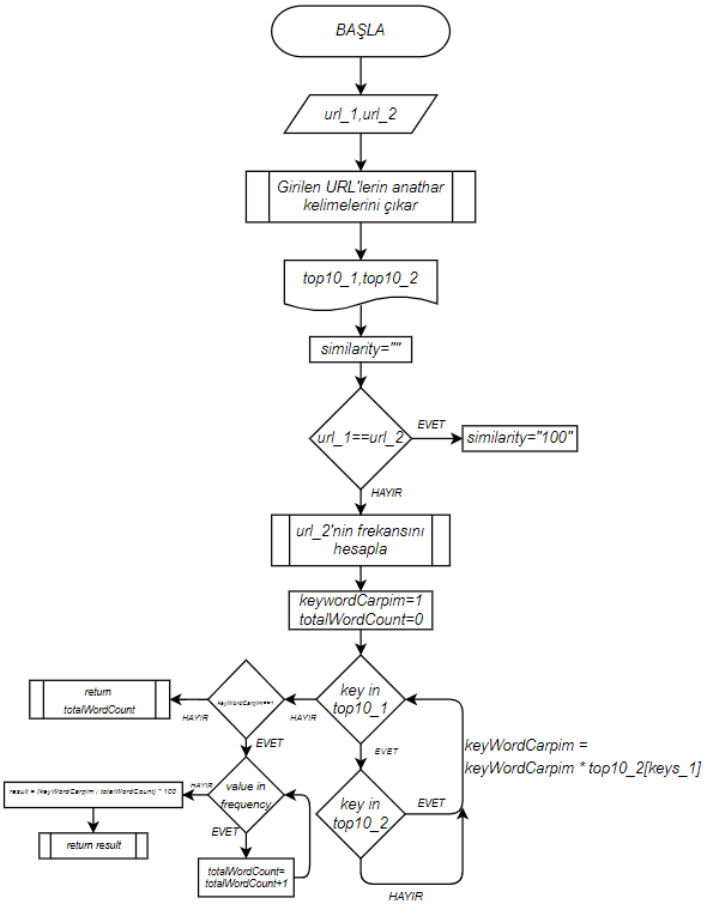
3.1 Algoritma



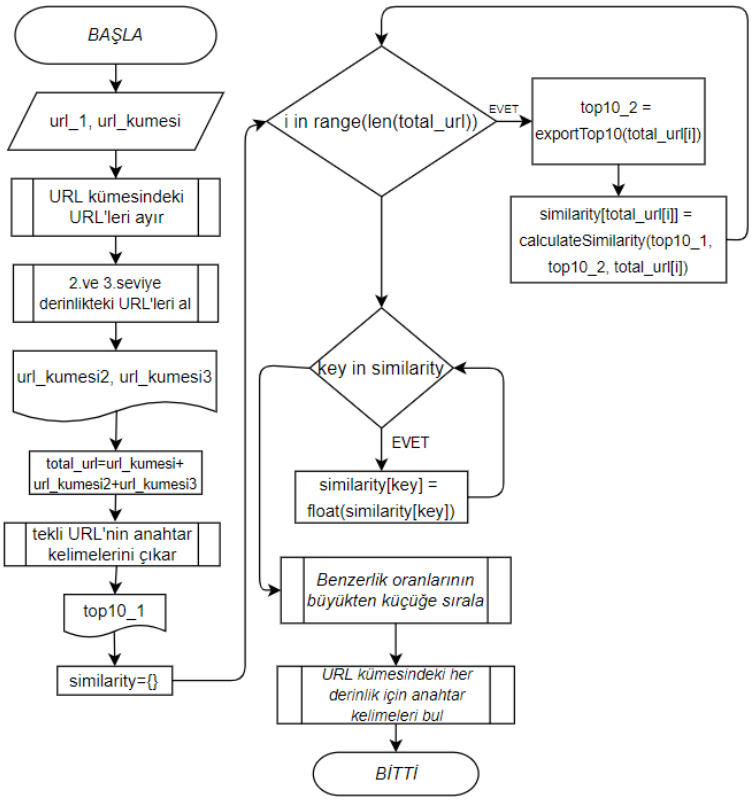
Şekil 1: *Sayfada Geçen Kelimelerin Frekanslarını Hesaplama*



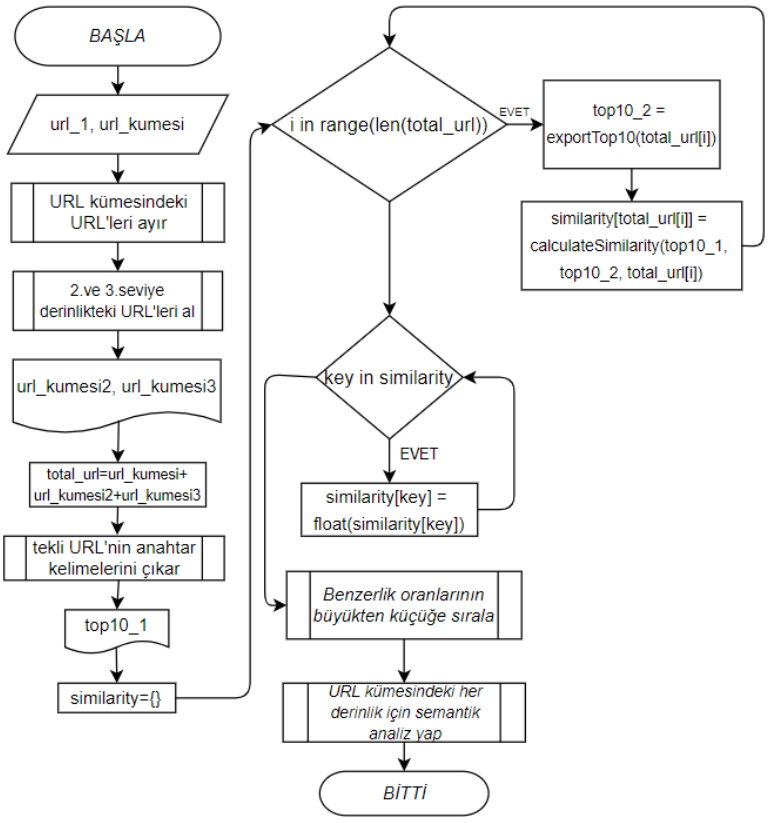
Şekil 2: *Anahtar Kelime Çıkarmak*



Şekil 3: *İki Sayfa (URL) arasındaki benzerlik skoru hesaplama*



Şekil 4: *Site İndeksleme ve Sıralama*



Şekil 5: *Semantik Analiz*

3.2 Karşılaşılan Problemler

URL Kümesinin bulunduğu işlemlerde URL kümesindeki eleman sayısının çok arttığı durumlarda hesaplama süresi de katlanarak artmakta ve uzun sürelere çıkmaktadır. Algoritma karmaşıklığı büyük kümeler üzerinde daha hızlı çalışmak üzere geliştirilebilir.

3.3 Kullanılan Fonksiyonlar

index(request): Uygulamanın başlangıç sayfası olan *'index.html'* sayfasına yönlendirir.

frekans(request): *'frekans.html'* sayfasına yönlendirir.

frekansResult(request):*'calculateFrequency(url)'* fonksiyonunu ile hesaplanan değeri *'frekansResult.html'* sayfasına gönderir.

keyword(request): *'keyword.html'* sayfasına yönlendirir.

keywordResult(request): *'exportTop10(url)'* fonksiyonunu ile hesaplanan değeri *'keywordResult.html'* sayfasına gönderir.

similarityScore(request): *'similarityScore.html'* sayfasına yönlendirir.

similarityScoreResult(request):*'exportTop10(url)'* fonksiyonunu ile hesaplanan değerleri *'calculateSimilarity'* fonksiyonuna gönderir, ardından dönen sonucu *'similarityScoreResult.html'* sayfasına gönderir.

indexingAndSort(request): *'indexingAndSort.html'* sayfasına yönlendirir.

indexingAndSortResult(request): Girilen URL ile URL kümesindeki URL'lerin benzerlik oranını *'calculateSimilarity'* fonksiyonu ile hesaplar ve sıralar, ardından URL kümesindeki her derinlik için anahtar kelimelerin '*URLKumesi_getTop10'* fonksiyonu ile hesaplanır. Hesaplanan sonuç *'indexingAndSortResult.html'* sayfasına gönderir.

semantic(request): *'semantic.html'* sayfasına yönlendirir.

semantikResult(request): Girilen URL ile URL kümesindeki URL'lerin benzerlik oranını *'calculateSimilarity'* fonksiyonu ile hesaplar ve sıralar, ardından URL kümesindeki her derinlik için semantik analiz sonucu '*multipleSemanticAnalysis'* fonksiyonu ile hesaplanır. Hesaplanan sonuç *'semanticResult.html'* sayfasına gönderilir.

scapeUrl(url): Alınan URL BeautifulSoup modülü kullanılarak içindeki tüm p etiketlerinin metinlerini içeren bir liste döndürür.

splitWords(allContentList): Girdi olarak alınan site metinlerini önce küçük harfe dönüştürür, noktalama işaretlerini temizler ve kelimelere ayırır. Sitenin sahip olduğu kelime listesini döndürür.

calculateFrequency(url): Kelime listesini inceleyerek içindeki kelimelerin kaç kere tekrar ettiğini hesaplar ve *dictionary* türünden frekans verisini döndürür.

exportTop10(url): *'calculateFrequency'* fonksiyonu ile gelen frekans verisinin içindeki bağlaçları ayıklayarak listede en çok geçen maksimum 10 kelimeyi döndürür.

calculateSimilarity(top10_1, top10_2, url_2): *'calculateFrequency'* fonksiyonu ile url_2'nin frekans verisi hesaplanır. Top10_1'in anahtar değerlerinde dolaşmak üzere döngüye girilir ve top10_2 anahtarları ile ortak olan kelimelerin top10_2 deki geçme değerleri çarpılarak keyWordCarpim değeri hesaplanır. url_2'nin frekans verisi de döngüye sokularak toplam kelime sayısı hesaplanır.

result = (keyWordCarpim / totalWordCount)*100

formülü ile iki URL arasındaki benzerlik oranı hesaplanır.

URLParser(url_kumesi): Verilen URL kümesinin virgüllere göre ayırır ve URL listesi döndürür.

subLink(url): Verilen URL'nin alt linklerine ulaşarak içinde '*http'* içeren ilk linki döndürür.

URLKumesi_getTop10(url_kumesi): Verilen küme içerisinde döngüye girerek *'exportTop10'* fonksiyonu aracılığı ile her URL'nin anahtar kelimelerini hesaplar ve *dictionary* formatında döndürür.

multipleSemanticAnalysis(url_kumesi): Verilen küme içerisinde döngüye girerek *'exportTop10'* fonksiyonu aracılığı ile her URL'nin semantik analizini gerçekleştirir ve *dictionary* formatında döndürür.

fileOperations(): Semantik analiz işlemini gerçekleştirebilmek için *'kelime_esanlamlisi.txt'* dosyasını okuyan ve her kelimenin eş anlamlılarını tek metin halinde toplayarak *dictionary* formatında eş anlamlı kelimeler verisini döndürür.

frequencyWithoutBaglac(url): *'calculateFrequency'* fonksiyonu ile hesaplanan kelime frekans listesinin içindeki bağlaçları temizler ve *dictionary* formatında bağlaçsız frekans verisi döndürür.

semanticAnalysis(url): Verilen URL'nin bağlaçsız frekans verisi ve anahtar kelimeleri alınır. Anahtar kelimeler döngüye sokulur, içindeki kelime *'kelime_esanlamlisi.txt'* den alınan veriler doğrultusunda eğer *es_anlamlilar* içinde bulunuyorsa kelimeye karşılık gelen string virgüllerle ayrılır ve tüm eş anlamlı kelimeleri bir listede depolanır. Depolanan liste döngüye sokulur ve eğer listenin elemanı URL'nin tüm kelimeleri içinde var ise kelime ve tekrar etme miktarı *semantik* isimli *dictionary* tipindeki değişkende saklanır.

4. Uygulamanın Çalıştırılması

Gerekli modüllerin bulunması durumunda proje klasörünün bulunduğu dizinde aşağıdaki komut çalıştırılarak yerel sunucuda başlatılabilir.

py .\manage.py runserver

Şekil 6: *Giriş Ekranı*

Şekil 7: *URL Girdi Ekranı*

KAYNAKÇA

[1] https://docs.djangoproject.com/en/3.1/

[2]https://www.crummy.com/software/BeautifulSoup/bs4/doc/

[3] https://getbootstrap.com/docs/5.0/getting-started/introduction/

[4] Aidas Bendoraitis, Jake Kronika,2020, Django 3 Web Development Cookbook: Actionable solutions to common problems in Python web development,

ISBN: 1838987428, 9781838987428

[5] Michael Heydt , 2018, Python Web Scraping Cookbook

ISBN: 9781787285217, 1787285219

[6] Joerg Krause, 2020, INTRODUCING BOOTSTRAP 4 : create powerful web applications using bootstrap 4.5.

ISBN: 9781484262023, 1484262026