KOCAELİ ÜNİVERSİTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ
YAZILIM LABORATUVARI

Tuğba ÇEVİKER
210202056

Samethan KAZANBAŞ
210202043

I. ÖZET

Bu döküman Yazılım Laboratuvarının dördüncü projesi olan Web Scraping Akademi Uygulamasının çözümünü açıklamaya yönelik oluşturulmuştur. Dökümanda özet , giriş , deneysel sonuçlar , sonuç , yöntem ve kaynakça kısımlarına yer verilmiştir.

II. GİRİŞ

Google Akademik gibi akademik arama motorları üzerinden web scraping yöntemiyle aratılan akademik yayınlara ait bilgilerin kaydedildiği bir veritabanıyla birlikte bu bilgilerin webden aratılması, görüntülenmesi ve istenilen özelliklere göre sorguların yapılmasını sağlayan bir web arayüzü geliştirilmelidir. Proje sayesinde web scraping ile bir web sayfasından bilgiye erişim sağlama; MongoDB veritabanı ile Elasticsearch sorgu yapılarını kullanma ve web kodlama becerilerinin geliştirilmesi amaçlanmaktadır.

1)WEB SCRAPİNG

En az bir akademik arama motorundan web scraping kullanılarak girilen anahtar kelimelere göre listelenmiş en az ilk 10 akademik yayının bilgileri,oluşturduğunuz web arayüzünde görüntülenmelidir.Kullanıcının arama yapmak için kullanacağı anahtar kelimeler oluşturacağınız kendi web sayfanızdaki bir text alanı üzerinden girilecektir. İstenen yayına ilişkin bilgiler doğrudan akademik arama motorunun sayfasından çekilebileceği gibi arama motoru sayfasındaki link üzerinden yönlendirilecek diğer bir web sayfasından da elde edilebilir. İstenen her yayın için pdf dosyası mutlaka indirilmelidir. Daha sonra tercihe göre yayın bilgileri ya web sayfası üzerinden ya da indirilmiş pdf dosyasının içeriğinden elde edilebilir.

2)VERİTABANI

Web scraping ile elde edilen veriler MongoDb kullanılarak veritabanına kaydedilecektir. Veritabanında tutulması beklenen bilgiler: Yayın id , yayın adı , yazarların isimleri , yayın türü , yayımlanma tarihi , yayıncı adı , anahtar kelimler(aranan), anahtar kelimeler (makale) , özet , referanslar , alıntı sayısı , doi numarası (varsa) , url adresi.

3)WEB SAYFASI

Erişilen yayınların bilgilerinin kullanıcıya gösterilmesi için bir web sayfası oluşturmanız beklenmektedir. Web sayfasında aratılacak yayınlar için bir text alanı oluşturulmalı ve bu text alanı girilecek anahtar kelimeler üzerinden ilgili arama motorunun yayınları aratıp bilgilerini web sayfasına getirmesi sağlanmalıdır.Web sayfası ilk açıldığında veri tabanında tutulan bilgiler ana sayfada gösterilmelidir.

III. YÖNTEM

1)WEB SCRAPİNG

Öncelikle projeyi hangi dilde yapcağımıza karar verdik. Projeyi C dilinde ve .Net Core 6.0 projesi olarak yapmaya karar verdik. Yaptığımız araştırmalarda .Net Core 6.0 da Web Scraping yapabilmemiz için kullanacağımız belli başalı kütüphaneler vardı. Bunlar Selenium ve HtmlAgiliyPazk kütüphaneleriydi. Biz bu kütüphaneler arasından HtmlAgilityPack kütüphanesini kullanmaya karar verdik. Ardından web scraping uygulayacağımız siteye karar vermemiz gerekiyordu. Burda da ”dergipark.com” sitesini kullanmaya karar verdik. HtmlAgilityPack kullanarak dergipark sitesinini herhangi bir uzantısına ulaşmamızı sağlayan kod parçası.

```
String url = "https://dergipark.org.tr/tr/search?q=" + search;
url = url + "&section=articles";
var httpClient = new HttpClient();
var html = httpClient.GetStringAsync(url).Result;
var htmlDocument = new HtmlDocument();
htmlDocument.LoadHtml(html);


HtmlNodeCollection articleCardNodes = htmlDocument.DocumentNode.SelectNodes("//div[contains(@class, 'card-body')]");
HtmlNodeCollection linkNodes = htmlDocument.DocumentNode.SelectNodes("//h5[@class='card-title']/a");
```

WebScraping işlemiyle ilgili siteinin html kodundan istediğimiz yeri istediğimiz şekilde çekebiliyoruz. Başlangıç controllerımızda yazdığımız algoritma sayesinde bir makalenin özelliklerini Proje dökümanında da belirtildiği gibi böldük. Oluşturduğumuz Article nesnesine de bu özellikleri mongoDb veritabanında da kullanılacak bir şekilde düzenleyip atamasını yapabildik.

2)Veritabanı

Veritabanı olarak dökümanda da belirtilen mongoDB veritabanını kullandık. HtmlAgilityPack kütüphanesi sayesinde böldüğümüz her veriyi atadığımız article nesnelerini veritabanına ArticleService sınıfında yazdığımız Create fonksiyonu sayesinde ekleyebildik.

```
_id: ObjectId('65f74fdc5d6f62b7edb995db')
yayin_adi : "OKUL KAVRAMINA İLİŞKİN METAFORLAR"
yazar_adi : "PınarARSLAN"
yayin_turu : "Makaleler"
yayinlanma_tarihi : 2020-04-05T21:00:00.000+00:00
konular : ""
anahtarkelime_motor : "Okul"
anahtarkelime_makale : "Metafor, Okul, Okul Metaforları"
ozet : "Metaforlar, olayların oluşumu, işleyişi ile ilgili olarak bireyin düşü…"
referanslar : "Akkaya, N. (2009). Öğretmen adaylarının öğretmenlik mesleğine yönelik …"
alinti_sayisi : 1
doi_numarasi : ""
url_adres : "https://dergipark.org.tr/tr/pub/ijls/issue/53618/668509"
download : "https://dergipark.org.tr/tr/download/article-file/1038100"
```

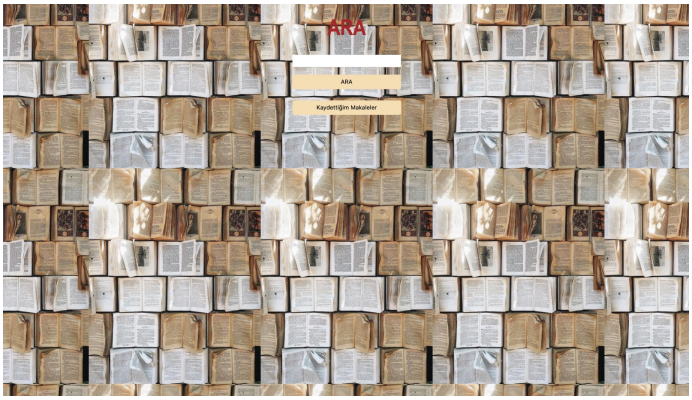
Projemizin veritabanıyla bağlanabilmesi içinde .Net Core 6.0 projemize Connection String eklememiz gerekiyordu.

Bunu da projemizde yer alan app.settings dosyasına aşağıdaki gib ekledik.

```
{
  "WebScrapingDatabaseSettings": {
    "WebScrapingCollectionName": "Article",
    "ConnectionString": "mongodb://localhost:27017",
    "DatabaseName": "WebScraping"
  },
```

3)Web Sayfası

Proje çalıştırıldığında ekrana Arama yapabileceğiniz arama motorumuz geliyor.



Ardından kullanıcı istediği kelimeyi aratıyor ve ekrana arattığı kelimeyle alakalı en fazla 24 sonuç geliyor. Sonuç ekranı da aşağıdaki gibidir.

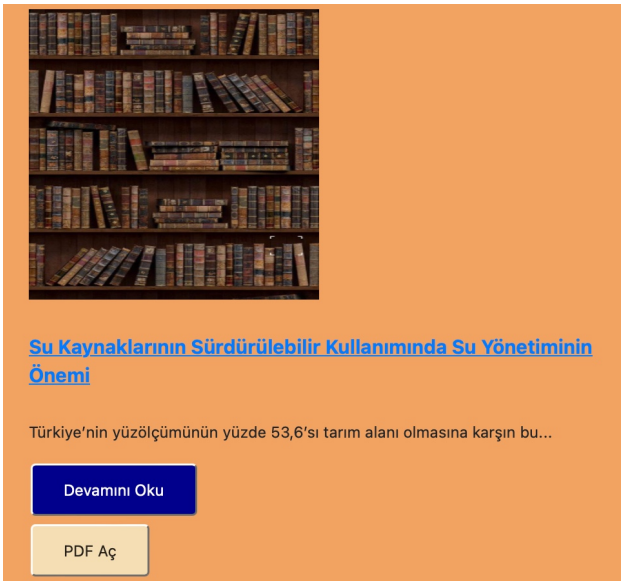
Kullanıcı ilgili makalenin adına basıp direkt dergipark üzerindeki linke ulaşabilir veya makaleye git butonuylailgili sitedeki bilgilerin gösterildiği yeni bir sayfaya geçer.

Kullanıcının makaleye git butonuna basıp gittiği nakaleler arka planda çalışan algoritma ile mongodb veritabanına kaydedilir.

Giriş ekranın altında yer alan kaydettiğim makaleler butonuna basılırsa ekrana kullanıcının makaleleri gelir.

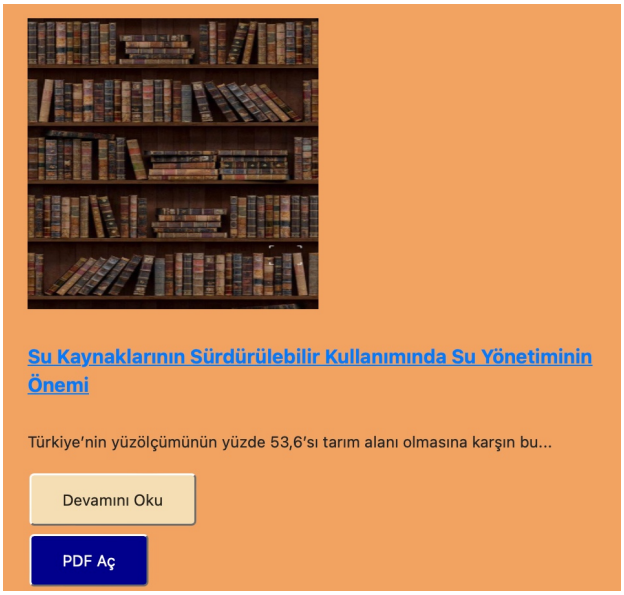
Kullanıcı bu ekrandan makalenin özelliklerine göre dinamik bir filtreleme ve sıralama işlemi yapabilir.

Kaydedilen her bir makalenin altında yer alan Devamını oku butonuyla kullanıcı makalenin bilgi sayfasına ulaşır.



Kaydedilen her bir makalenin altında yer alan Pdf AÇ butonuyla kullanıcı makalenin Pdfini ekranda görüntüleyebilir.

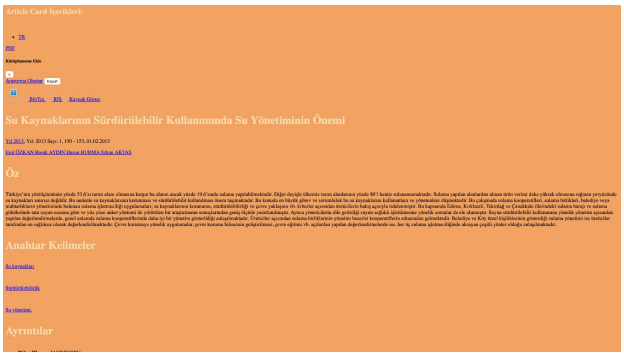
Önamli Notlar !(Eğer bir makalenin pdfi yoksa veya isim bilgisine ulaşılamıyorsa bu makale veritabanına kaydedilmez. Dinamik filtreleme ve sıralama işlemleri ElasticSearch yapısıyla indexleme yöntemi kullanılarak yapılmıştır.)

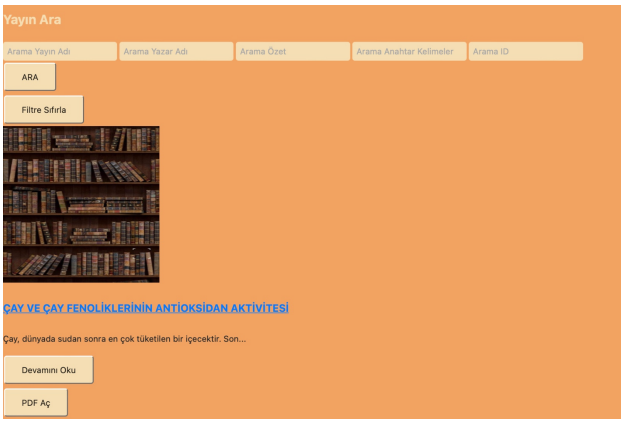


IV. DENEYSEL SONUÇLAR

Arama motoruna istenilen kelime aratıldığında ekrana gelen ilk 24 sonucu listeliyoruz.Aşağıdaki fotoğraflarda görebilirsiniz.Daha sonrasında da makaleye git butonuna basıldığında makaleye ait bilgiler ekrana gelmektedir .Özet, yazar , yayımlandığı yıl vs .Bunu da aşağıdaki fotoğraftan görebilirisiniz.

Alttaki resimde görülen bir filtreleme örneğidir. Çay kelimesi aranmak istenen kelime labelına yazıldığında ve ara butonuna tıklandığında aşağıdaki gibi bir görüntü oluşur.





V. SONUÇ

İlk defa web scraping içeren bir projede bulunduk , edindiğimiz bilgiler ve deneyimler bizi çok geliştirdi. Htmlagiltypack kullanımı hakkında bilgi sahibi olduk.MONGODB ve Elasticsearch yapılarının kullandık ve bir daha bu tarz bir proje içinde bulunursak kullanmamız ve ekip arkadaşlarımıza yardımcı olmamız daha kolay olacak.

VI. KAYNAKÇA

https://dergipark.org.tr/tr/search?q=nemsection=articles
https://www.youtube.com/watch?v=ullxVcavOSY
https://www.youtube.com/watch?v=m9zFq6KS94Y
https://www.youtube.com/watch?v=mh48aGK85LQ
https://www.youtube.com/watch?v=-bt$_y4Loofg$

VII. AKIŞ ŞEMASI SONRAKI SAYFALARDADIR