Yazılım Lab. I Proje II
Academic Search

EMRE KAYA
Öğrenci No
210201029

I. ÖZET

Bu doküman Kocaeli Üniversitesi 20022-2023 bahar dönemi Mühendislik fakültesi Bilgisayar Mühendisliği Yazılım Lab Laboratuvarı II Proje I için hazırlanmıştır. Rapor: Özet, Giriş, Yöntem Sonuç ve Kaynakça alt başlıklarından oluşmaktadır.

II. GİRİŞ

Academic Search projesi, Google Akademik akademik arama motoru üzerinden uygun sayfalara yönlendirme yapılarak web scraping (web kazıma) yöntemiyle aratılan akademik yayınlara ait bilgilerin elde edilip ardından kaydedilebildiği bir Mongo DB veritabanıyla birlikte bu bilgilerin webden aratılması, görüntülenmesi ve istenilen özelliklere göre sorguların yapılmasına imkan sağlayacak bir web arayüzü geliştirmelerini içermektedir.

III. YÖNTEM

A. Teknolojiler

Web frontEnd için javascript, html, css kullanılmıştır.
Bu projenin veritabanı sistemi olarak, Proje ihtiyaçları ve isterleri göz önünde bulundurularak NoSQL bir veritabanı olan MongoDB kullanmıştır. Mongo DB'nin projeye sağladığı özellikler şunlardır:

- Ölçeklenebilirlik :
  MongoDB, veritabanını kolayca yatay olarak ölçeklendirilme imkanı sunar. Bu, büyük miktarda veri ve yüksek trafikle başa çıkabilme yeteneği anlamına gelir. Projede beklenen büyüme ve genişleme dikkate alındığında, bu özellik kritik öneme sahiptir.

- Esnek Şema :
  MongoDB, dinamik şema tasarımına izin verir. Bu, veritabanı şemasının uygulama geliştirme sürecinde kolayca değiştirilebilmesi ve geliştirilebilmesi anlamına gelir. Projenin evrildikçe ihtiyaç duyabileceği değişiklik ve genişletmelere hızlı bir şekilde uyum sağlaması için bu esneklik büyük avantaj sağlar.

- Veri Modelleme Avantajları :
  MongoDB, JSON-benzeri belgeler kullanarak verileri depolar. Bu yapısı, karmaşık hiyerarşilere ve veri türlerine sahip verilerin doğal bir şekilde ifade edilmesini sağlar, böylece uygulamanın veri modelleme ihtiyaçlarına doğrudan hitap eder.

- Performans ve Hız :
  MongoDB, özellikle okuma ve yazma işlemleri için optimize edilmiştir. Büyük veri setlerinde hızlı sorgulama ve veri erişimi sağlar, bu da real-time uygulamalar için idealdir.

Bu projede Yazılım dili olarak Python tercih edilmiştir. Bu tercihin yapılmasında kullanılan genel Kriterler şunlardır :

- Kolay Okunabilirlik ve Dil Tasarımı:
  Python, okunabilir ve sade bir dil tasarımına sahiptir.

- Geniş Kütüphane Desteği:
  Python, zengin bir standart kütüphane ve birçok üçüncü taraf kütüphane sunar. projenin farklı gereksinimlerini karşılamak için veritabanı işlemleri için PyMongo, Web Backend için Django kütüphanelerinin kullanımına imkan sağlamıştır.

- Çapraz Platform Desteği:
  Python, farklı işletim sistemlerinde (Linux, Windows, macOS) çalışabilir. Bu kriter, projenin Linux da geliştirilmesine imkan sağlamıştır.

B. Kütüphaneler

- PyMongo :
  Pymongo, Python ile MongoDB veritabanı arasında iletişim kurmayı sağlayan bir kütüphanedir. Projede, MongoDB'yi temel veri depolama aracı olarak kullanırken, Pymongo sayesinde verilere erişim, yönetim ve sorgulama işlemleri kolayca gerçekleştirilir.
  Projenin temel veri depolama ve yönetim sistemi olarak MongoDB'yi kullandığımız göz önünde bulundurulduğunda, Pymongo kütüphanesi projemizin ana bileşenlerinden biridir. Pymongo'nun sunduğu temel yetenekler arasında belirtilen işlemleri gerçekleştirmenin yanı sıra, veri tabanı ile bağlantı kurma, veri tabanı ve koleksiyonları oluşturma, indeksleme ve veri tabanı işlemleri için güvenli bağlantı yönetimi gibi işlevler de bulunur.

- Django :
  Django, Python ile web uygulamaları geliştirmeyi sağlayan bir web çatısıdır. Projede, Django kullanılarak web uygulaması geliştirilirken, kullanıcı arayüzü oluşturulması, URL yönlendirmesi, veritabanı etkileşimi ve güvenlik gibi temel işlevler kolaylıkla sağlanır.
  Projenin Django kullanımıyla, web arayüzü ve kullanıcı etkileşimi sağlanırken, veri yönetimi ve iş mantığıyla entegrasyon sağlanır. Django'nun sunduğu modüler yapısı, projenin genişletilebilirliğini ve ölçeklenebilirliğini artırır.

C. API

Bu projede API yapısı olarak bir web servisinin veya API'nin standart bir mimarisi olarak bilinen REST (Representational State Transfer) mimarisi tercih edilmiştir. Yapıda temel HTTP metotları olan :
GET: Kaynağı almak için kullanılır.
POST: Yeni bir kaynak oluşturmak için kullanılır.
PUT: Bir kaynağı güncellemek için kullanılır.
DELETE: Bir kaynağı silmek için kullanılır.
HTTP metotları kullanılmıştır.

D. Proje Akışı

Proje Aşağıda belirtilen 3 ana başlıktan oluşmaktadır.

- Web Scraping:

  Isterler gereğince bir akademik arama motorundan "Google Akademik" sayfasından web scraping uygulanılarak kullanıcıdan web sayfası üzerindeki bir input alanından alınacak anahtar kelimelere göre, kelimeler google akademik url sine uygun biçimde parametre verilerek istenilen arama sayfasına erişim sağlanmıştır. erişim sağlanmasından sonra sonuç olarak listelenen ilk sayfadaki ilk 10 akademik yayının bilgileri, tarafımızca oluşturulan web sayfasında görüntülenmektedir. Kullanıcının arama yapmak için kullanacağı anahtar kelimeler oluşturulan web sayfasındaki "/search" altındaki bir text alanı üzerinden girilmektedir. Web scraping işlemi için siteye request isteği yapılarak istenen veriye erişilmektedir. (Erişilecek siteye yönelik hazır web API ler kullanılmamaktadır.) İstenen yayına ilişkin bilgilerin bir kısmı doğrudan akademik arama motorunun sayfasından çekilmektedir. İsterler kontrol edildiğinde her bilginin Google Akademik'den elde edilemediği görülmüştür. Bu yüzden Springer.com, nature.com, IEEE resmi yayın sayfası gibi web sitelere yönlendirme veren yayınların erişilemeyen detaylı bilgilleri arama motoru sayfasındaki yayın linki üzerinden yönlendirilecek diğer bir web sayfasından da elde edilmektedir. İstenen her yayın için istendiği durumda pdf dosyası indirilmektedir.

- Veritabanı

  Web scraping ile elde edilen veriler Kurulan sistemdeki bir docker kontainerinde bulunan MongoDb kullanılarak veritabanına kaydedilmektedir. Veritabanında tutulan bilgiler şunlardır:
  - Yayın id
  - Yayın adı
  - Yazarların isimleri
  - Yayın türü (araştırma makalesi, derleme, konferans, kitap vb.)
  - Yayımlanma tarihi
  - Yayıncı adı (yayının yayımlandığı konferans ismi; dergi veya kitap yayınevi)
  - Anahtar kelimeler (Arama motorunda aratılan)
  - Anahtar kelimeler (Makaleye ait)
  - Özet
  - Referanslar
  - Alıntı sayısı
  - Doi numarası
  - URL adresi
  - Yayın PDF URL adresi
- Web Sayfası:

  Url Paternleri :

  - path("", views.index, name="index") :
    sitenin root nodesi dir. server genellikle 8000 portundan açılır, http://localhost:8000/ e gelindiğinde ana sayfayı döndüren url paternidir.
    path("scrap/",views.scrap, name="scrap") :
    sitenin scraping i tetikleyen get, post gibi işlemlerin yönlendirildiği url dir.

  - path("searcheds",views.viewsearcheds, name="searcheds") :
    web sayfası isterlerinden veritabanında bulunan verilerin ana sayfada görüntülenmesini sağlayan url paternidir.

  - path("dropC",views.dropcol, name="dropC") :
    web sayfasında veritabanı testleri yapılırken MongoDb'nin temizlenmesi işleminin çok kez yapılması gerekmiştir. Kolaylık sağlaması için tüm veritabanını bir adımda silnmesini sağlayan bir path ouşturulmuştur.

  - path('carddetails¡path:cardvariable¿', views.carddetails, name='carddetails') :
    her kartın detaylarının görselleştirileceği ayrı bir sayfaya ihtiyaç duyulmuştur, bu ihtiyaca bakılarak bir path atannmıştır. Gelen isteklere cartın detaylarını dönmektedir.

  Erişilen yayınların bilgilerinin kullanıcıya gösterilmesi

için bir web sayfası oluşturulmuştur.

Web sayfasında aratılacak yayınlar için bir text alanı oluşturulmuştur ve bu text alanı girilecek anahtar kelimeler üzerinden Google Akademik arama motorunun yayınları aratıp bilgilerini web sayfasına getirmesi sağlayan sistem geliştirilmiştir.

Web sayfası ilk açıldığında istenmesi halinde veritabanında bulunan tüm kayıtlar sayfaya getirilmektedir.

Listeleme işleminde yayınların isimleri sırasına uygun biçimde listelenmektedir. Listelenen bir makale isminin altında bulunan butona tıklandığında o makaleye özgü bilgiler ayrı bir sayfada gösterilmektedir.

Web sayfasından herhangi bir çalışmaya yönelik dinamik arama işlemi yapılabilmektedir. Ayrıca arama sırasında yazım yanlışı olması durumunda sistem taradından açıldığı sürece yazım düzeltmesi yapılarak arama yapılmaktadır. Örnek: deep learrning – yazımı düzeltilmiş kelimelerden elde edilen sanuçlar görüntülenmektedir: deep learning şeklinde geliştirilmiştir.

Web sayfasında en son veya en önce yayımlanma tarihine göre sıralama yapılabilmektedir ayrıca yine atıf sayısına göre de en az veya en çok olacak şekilde sıralama yapılabilmektedir.

IV. **SONUÇ**

Projede elde edilen sonuçlar, akademik araştırmalara yönelik bilgi erişiminde ve yönetiminde kullanılabilir. Gelecekteki çalışmalarda, web scraping işlemi için daha karmaşık algoritmaların kullanılması ve web arayüzünde daha fazla kullanıcı dostu özelliklerin eklenmesi önerilmektedir

REFERENCES

[1] https://scholar.google.com/schhp?hl=tr
[2] https://www.mongodb.com/
[3] https://www.docker.com/
[4] https://getbootstrap.com/docs/5.3/forms/checks-radios/
[5] https://www.w3schools.com/html/html_form_input_types.asp
[6] https://www.youtube.com/results?search_query=youtube+video+embed+in+nodejs
[7] https://stackoverflow.com/questions/11722400/programmatically-change-the-src-of-an-img-tag
[8] https://stackoverflow.com/questions/11722400/programmatically-change-the-src-of-an-img-tag
[9] https://www.w3schools.com/howto/howto_css_next_prev.asp
[10] https://www.w3schools.com/css/css3_object-fit.asp

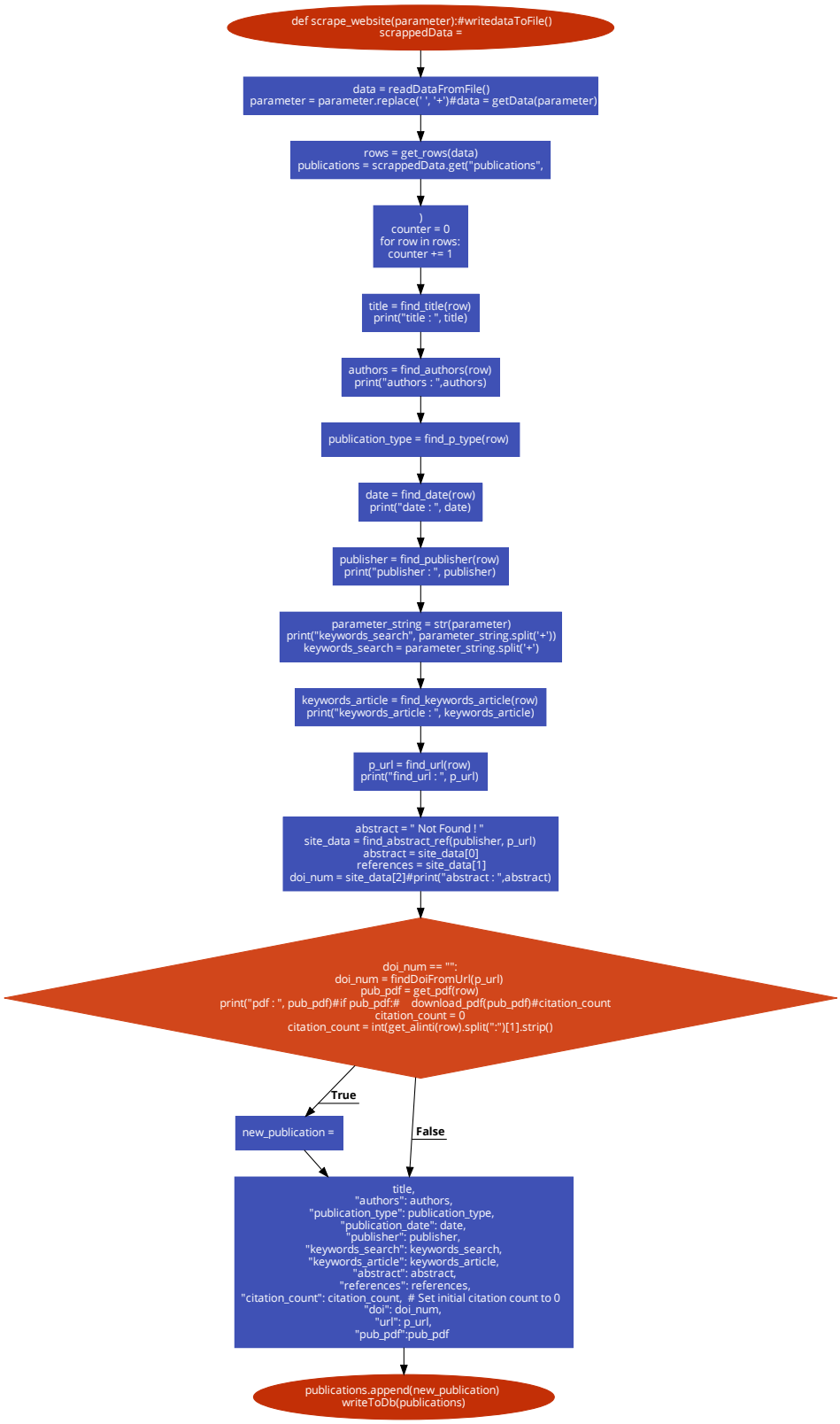

Fig. 1. Scraping algoritması Akış Diyagramı