BİL412 – İNTERNET PROGRAMLAMA
PROJE RAPORU

İBRAHİM BURAK ŞAHİN - 151101049

MERİÇ BALGAMIŞ - 141101047

WEBSITE : ULUDAĞ SÖZLÜK

BİL412 dersi kapsamında gerçekleştirdiğimiz projede ‘uludagsozluk.com’ web sitesini belli bir zaman aralığı içerisinde crawl ettik. Elde ettiğimiz veriyi JSON formatında depoladık ve son aşama olarak elimizdeki verileri PHP kullanarak bir serverda analiz ettik. Sırası ile crawl , store ve analiz kısımlarını açıklayacağız.

CRAWL

Crawl aşamasında beautifulSoup kullanarak bizden istenen tarih aralıkları çerçevesinde Uludağ Sözlük web sitesindeki entryleri crawl ettik. Crawl işleminde ‘www.uludagsozluk.com/?id=3’ URL’ini kullandık. Bu sayede tüm entryleri crawl edebildik.

Web sitesinde her entry’nin kendine ait bir id değeri olduğu için son 6 ay içindeki tüm entryleri sırayla crawl ettik. Son 6 ay içerisindeki entrylerin id değerleri yaklaşık olarak 38.000.000 – 40.000.000 arasında bulunmaktaydı. Aynı zamanda bir tarih kontrolü ekleyerek sadece paramatre olarak verilen tarihten sonraki entryleri crawl ettik.

```
if(datetime.datetime(int(year),int(month),int(day)) < (datetime.datetime(2018,1,1))):
    continue
```

Her entry’i bir obje olarak tanımladık.

```
class EntryObject(object):
    author = ""
    entry_date = datetime
    entry_time = datetime
    entry_text = ""
    number_of_likes = 0;
    number_of_dislikes = 0;
```

Bu entry objelerinin bilgilerini beautifulSoup kullanarak doldurduk.

```
link = soup.find('h1', attrs={'class':'tekentry-baslik'})   #take entry name
if (link is not None):
    title = link.getText().strip()

link = soup.find('div', attrs={'class':'entry-p'})  #take entry text
if (link is not None):
    entry_text = link.getText().strip()

link = soup.find('a', attrs={'href':'#'}) #take entry author
if (link is not None):
    author = link.getText().strip()

link = soup.find('span', attrs={'class':'oylar arti_sayi color-yesil'})     #take entry upvote
if (link is not None):
    number_of_likes = link.getText().strip()

if (number_of_likes == "" or number_of_likes == 0):
    number_of_likes = "0"
```

Bu verileri tek seferde crawl etmek yerine, her crawl işleminde 250.000 entry crawl ettik. Crawl işleminde programın kullanıcı tarafından durdurulmasına ve exception vermesine karşı önlemler aldık. Aşağıda görüldüğü üzere exception atıldığında, en son crawl edilen entry id değerinin bir fazlası ile crawl işlemini yapan metod tekrar çağırılmaktadır.

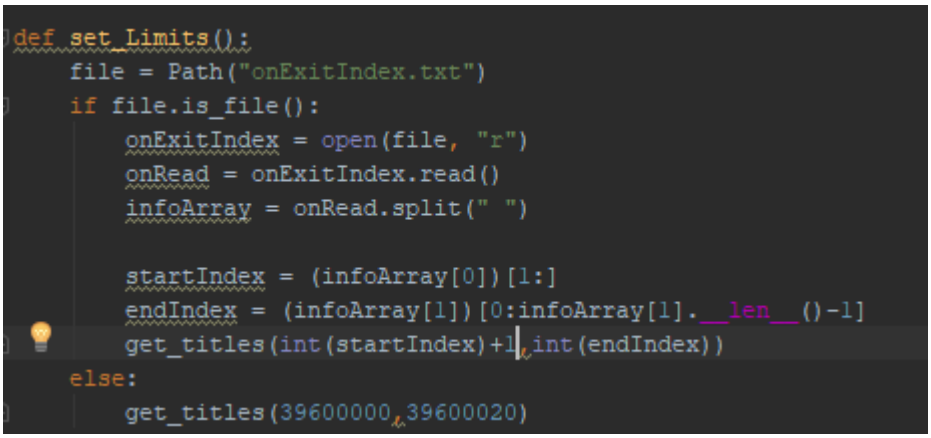
```
except:
    print("Hata.")
    print(i)
    sleep(1)
    startIndex = i+1;
    get_titles(startIndex,endIndex)
```

Eğer program kullanıcı tarafından kapatılırsa, en son crawl edilen entry'nin id değeri bir dosyaya yazılmakta ve o ana kadar çekilen veriler store edilmekte. Aşağıda görüldüğü üzere 'onExitOutput' değişkeni 'onExitJsonDumper' metodu ile '.txt' uzantılı bir dosyaya yazılmaktadır.

```
        onExitOutput = str(i) + " " + str(endIndex)
        onExitJsonDumper(onExitOutput)
```

```
def onExitJsonDumper(startIndex):
    with open('onExitIndex.txt', 'w', encoding='utf-8') as outfile:
        json.dump(startIndex, outfile, sort_keys=True, indent=4, ensure_ascii=False)
```

Program tekrar çalıştırıldığında ise eğer önceden bir durdurma işlemi yapılmışsa, dosyadan en son crawl edilen entry'nin id değeri okunarak bir sonraki entryden crawl işlemi devam etmektedir. Aşağıda görüldüğü üzere program çalıştığında 'onExitIndex.txt' isminde bir dosya search ederek, eğer program kullanıcı tarafından kapatılmışsa o dosyadan kalınan entry id değerini okuyarak, o değerin bir fazlasından çalışmaya devam etmektedir.



```
def set_Limits():
    file = Path("onExitIndex.txt")
    if file.is_file():
        onExitIndex = open(file, "r")
        onRead = onExitIndex.read()
        infoArray = onRead.split(" ")

        startIndex = (infoArray[0])[1:]
        endIndex = (infoArray[1])[0:infoArray[1].__len__()-1]
        get_titles(int(startIndex)+1,int(endIndex))
    else:
        get_titles(39600000,39600020)
```

STORE

Store aşamasında ise bizden istenildiği gibi crawl edilen verileri bir tree yapısı içerisinde JSON formatında sakladık. Aşağıda bir örneği görülen JSON dosyasında bizden istenilen tree yapısına tamamen uygun bir şekilde store işlemini gerçekleştirdik.

-Dictionary
--Category
---Title
----Entries
-----Entry
------Author
------Date/Time
-------Text
-------Number of Likes
-------Number of Dislikes

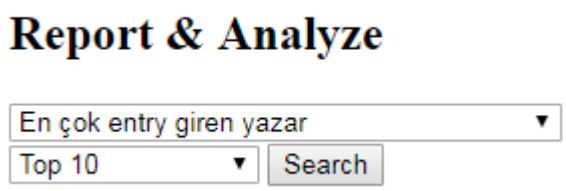

```
{
    "/kategori/anket/": [
        [
            "köpek saldırırsa yapılacaklar",
            {
                "author": "peri10",
                "entry_date": "01.01.2018",
                "entry_text": "Köpeğin size verdiği tepkileri ona verirseniz sizden korkabilirmis.
                "entry_time": "00:27",
                "number_of_dislikes": "0",
                "number_of_likes": "0"
            },
            {
                "author": "geahpisipisi",
                "entry_date": "01.01.2018",
                "entry_text": "panik yapip bagirmali deli gibi saga sola kacmali ve aglamaliyiz.",
                "entry_time": "00:50",
                "number_of_dislikes": "0",
                "number_of_likes": "1"
            }
        ],
        [
            "gecenin metal şarkısı",
            {
                "author": "jimmy tudeski",
                "entry_date": "01.01.2018",
                "entry_text": "mudvayne - dig\nhttps://www.youtube.com/watch?v=G_5UOkUDti8+",
                "entry_time": "00:36",
                "number_of_dislikes": "0",
                "number_of_likes": "0"
            },
            {
                "author": "kivilive",
                "entry_date": "01.01.2018",
                "entry_text": "trivium - through blood and dirt and bone.",
                "entry_time": "00:38",
                "number_of_dislikes": 0,
```

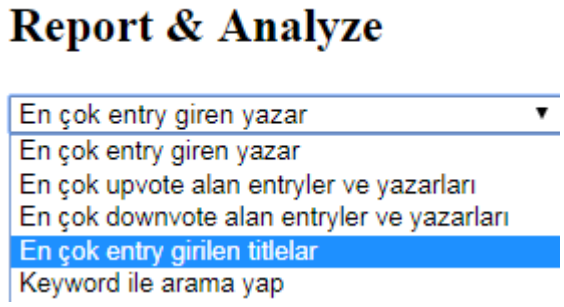
ANALİZ

Analiz aşamasında ise elimizde bulunan JSON dosyalarını kullanarak bir PHP server üzerinde, elimizdeki verilerle çeşitli analizler yaptık. Bu analizler

- En çok entry giren yazar
- En çok upvote alan entry ve yazarı
- En çok downvote alan entry ve yazarı
- En çok title girilen kategori
- Verilen bir kelimenin bulunma sıklığının hesaplanması şeklindedir.

Aşağıdaki gibi görülen analiz sayfamızda kullanıcı görüntülemek istediği analiz türünü seçerek 'search' butonuna bastıktan sonra sonuçları listelenmiş bir şekilde görebilmektedir. Aynı zamanda kullanıcı görüntülenecek sonuçların sayısına da karar verebilmektedir. Aşağıda görülen menü yardımıyla ilk 10,20,50,100 ya da tüm sonuçları görüntüleme seçenekleri kullanıcıya sunulmaktadır.





Kullanıcı 'Keyword ile arama yap' seçeneğine tıklayıp bir kelimenin bulunma sıklığını öğrenmek istediğinde ise aşağıdaki gibi kullanıcıdan bir input beklenilmektedir. Verilen input ile işlemler yapılarak bulunma sıklığı görüntülenebilmektedir.