LAPORAN WEB SCRAPING

FITRI SRI WAHYUNI

(2211082036)

TRPL 3B

PROGRAM STUDI TEKNOLOGI REKAYASA PERANGKAT LUNAK

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI PADANG

2025

A. Dasar Teori

Web Scraping adalah proses pengambilan sebuah dokumen semi-terstruktur dari internet, umumnya berupa halaman halaman web dalam bahasa markup seperti HTML atau XHTML, dan menganalisis dokumen tersebut untuk diambil data tertentu dari halaman tersebut untuk digunakan bagi kepentingan lain, serta banyak penelitian yang menggunakan tools scraping untuk mengumpulkan datanya dari web. Dalam kondisi ideal, kita tidak perlu melakukan web scraping karena semua situs web menyediakan API untuk berbagi data. Namun, kenyataannya tidak semua situs punya API, dan kalaupun ada, sering kali dibatasi jumlah data dan frekuensi aksesnya. API juga bisa diubah atau dihapus oleh pemilik situs kapan saja. Karena itu, kita perlu mempelajari web scraping untuk tetap bisa mengambil data dari internet.

B. Hasil Praktek

1. Membuat kode untuk melakukan parsing (penguraian) terhadap teks HTML menggunakan modul BeautifulSoup dari pustaka bs4, lalu menampilkan isi HTML dalam berbagai format.

```
from bs4 import BeautifulSoup
htmltxt = '''
<!DOCTYPE html>
<html>
<head></head>
<body>
<h1>Web Scraping</h1>
<a herf="Webku.html">Link ke Webku</a>
</body>
</html>
'''

Soup = BeautifulSoup(htmltxt, 'lxml')
print("Hasil Pertama : ")
print(Soup)
print('Hasil Kedua : ')
print(Soup.text)
print('Hasil Ketiga : ')
print(Soup.text.strip())
```

Hasilnya :

```
Hasil Pertama :
<!DOCTYPE html>
<html>
<head></head>
<body>
<h1>Web Scraping</h1>
<a herf="Webku.html">Link ke Webku</a>
</body>
</html>

Hasil Kedua :



Web Scraping
Link ke Webku



Hasil Ketiga :
Web Scraping
Link ke Webku
```

Fungsi dari `.text` adalah menghilangkan bagian lain selain text yang ada di file html tersebut. Sehingga hanya menyisakan text dan bagian kosong yang terletak pada sebelum dan setelah file text.

Fungsi dari `strip()` adalah memotong/ menghilangkan bagian kosong tersebut sehingga yang tersisa hanyalah text tanpa bagian kosong.

2. Membuatkode untukmengambilelemen HTML tertentu secara spesifik (seperti tag dan) dari sebuah teks HTML menggunakan BeautifulSoup, lalu menampilkan isi tag dan atributnya.

```
from bs4 import BeautifulSoup
htmltxt = '''
<!DOCTYPE html>
<html>
<head></head>
<body>
<h1>Web Scraping</h1>
<a href="/content/web1.html">Link ke WebKu</a>
</body>
</html>
'''

soup = BeautifulSoup(htmltxt, 'lxml')
print(soup.h1)
print(soup.h1.text)
print(soup.a)
print(soup.a.text)
print(soup.a['href'])
```

- `print(soup.h1)` = menampilkan tag `<h1>`
- beserta text di dalamnya
- `print(soup.h1.text)` = menampilkan hanya
- text di dalam h1 tanpa tag `<h1>`
- `print(soup.a)` = menampilkan tag `<a href>`
- beserta text di dalamnya
- `print(soup.a.text)` = menampilkan hanya
- text di dalam a href tanpa tag `<a href>`
- `print(soup.a['href'])` = menampilkan link
- yang dituju di dalam `<a href>`

Hasilnya :

```
<h1>Web Scraping</h1>
Web Scraping
<a href="/content/web1.html">Link ke WebKu</a>
Link ke WebKu
/content/web1.html
```

3. Membuat kode untuk membaca file HTML bernama web1.html, lalu menggunakan BeautifulSoup untuk mengambil data dari tag tertentu seperti <a> dan <p>. Berikut ini merupakan isi dari file web1.html berikut:

```html
<div class="satu">
  <p>Paragraph Pertama</p>
  <p>Paragraph Kedua</p>
  <a href="#">Ini Link Di CLass Satu</a>
</div>
<div class="dua">
  <p>Paragraph Ketiga</p>
  <p>Paragraph Ke Empat</p>
  <a href="#">Ini Link di Class Dua</a>
</div>
```

```python
from bs4 import BeautifulSoup

html = open("/content/web1.html", "r")
s = BeautifulSoup(html, "lxml")

print("Ambil Text dari link : ")
print(s.find('a').text)
print("Ambil Text dari Paragraf : ")
print(s.find('p').text)
print("Ambil Data Paragraf : ")
print(s.find_all('p'))
```

- `find()` = menemukan dan mengambil hanya data yang ditarget
- saja •`find_all()` = menemukan dan mengambil seluruh data

Hasilnya:

```
Ambil Text dari link :
Ini Link Di CLass Satu
Ambil Text dari Paragraf :
Paragraph Pertama
Ambil Data Paragraf :
[<p>Paragraph Pertama</p>, <p>Paragraph Kedua</p>, <p>Paragraph Ketiga</p>, <p>Paragraph Ke Empat</p>]
```

4. Membuat kode untuk melakukan web scraping dari file HTML lokal (web1.html), dan secara spesifik mengambil teks dari elemen tertentu berdasarkan class dan tag HTML, menggunakan pustaka BeautifulSoup.

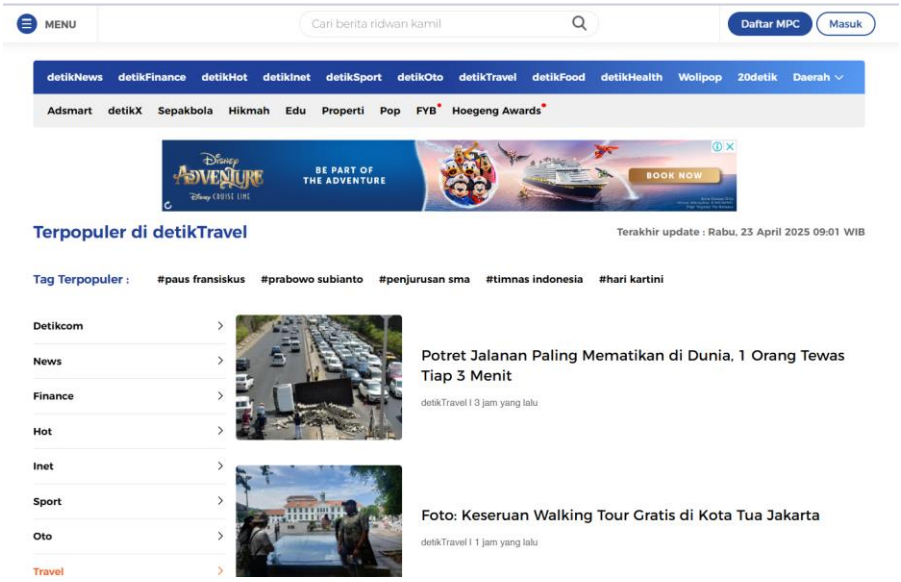
```
html = open("/content/web1.html", "r")

soup = BeautifulSoup(html, "lxml")

print("Ambil Text dari link di Class dua")
d = s.find("div", attrs={'class':'dua'})

#d sekarang berisi script di dalam Class dua
link = d.find('a')
print(link.text)

print()
print("Ambil Text Semua Paragraf :")
all_p = s.find_all('p')
for p in all_p:
    print(p.text)

print()
print()

print("Ambil Text Paragraf di Class Satu :")
d = s.find("div", attrs={'class':'satu'})
all_p = d.find_all('p')
for p in all_p:
    print(p.text)
```

Masih menggunakan file web1.html. Kita memasukkan script dari div class “dua” dengan deklarasi variabel d beserta valuenya yaitu dengan baris kode `d = s.find("div", attrs={'class':'dua'})`. Setelah data didapatkan, kita mengambil text yang ada pada tag `` yaitu dengan kode `link = d.find('a')` dan menampilkannya di output dengan kode `print(link.text)`. Lalu kita mengambil seluruh text pada seluruh paragraf dengan `all_p = s.find_all('p')` kemudian kita lakukan perulangan supaya tampilannya rapi dan urut yaitu dengan `for p in all_p:` lalu di dalam perulangan tersebut kita tampilkan text dengan `print(p.text)`. Kedua kita ambil data dari div class satu dengan baris kode `d = s.find("div", attrs={'class':'satu'})` kemudian kita ambil semua paragraf dalam div class satu dengan `all_p = d.find_all('p')` lalu kita lakukan perulangan sekaligus menampilkan text paragraf dengan `for p in all_p: dan print(p.text)`.

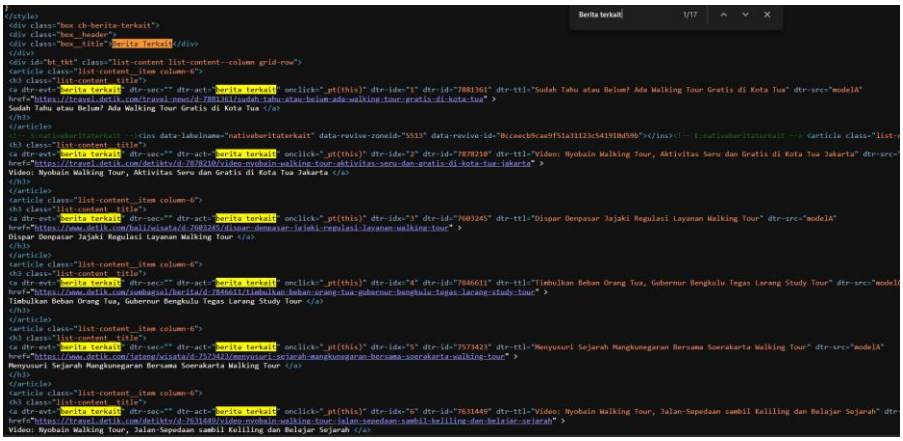
Hasilnya:

```
Ambil Text dari link di Class dua
Ini Link di Class Dua

Ambil Text Semua Paragraf :
Paragraph Pertama
Paragraph Kedua
Paragraph Ketiga
Paragraph Ke Empat


Ambil Text Paragraf di Class Satu :
Paragraph Pertama
Paragraph Kedua
```

5. Membuka Alamat web https://www.detik.com/ , dan meng-inspect Element pada bagian “Terpopuler”



6. Melihat View Page Source



7. Membuat kode untuk melakukan web scraping ke situs Detik.com, kemudian mengambil dan menampilkan judul berita terpopuler yang terdapat dalam elemen HTML tertentu.

```
from bs4 import BeautifulSoup
import requests

url = "https://www.detik.com/"
#ambil isi url, masukkan ke variabel web
web = requests.get(url).text
s = BeautifulSoup(web, "lxml")
#ambil class box cb-mostpop, karena hanya ada 1, maka cukup pakai find
b = s.find('div', attrs={'class':'box cb-mostpop'})
#di dalam class tsb, temukan semua class='media__title'
judul = b.find_all('h3', attrs={'class':'media__title'})
#baca tiap bagian dari judul, dan tampilkan
print("Terpopuler dari Detik.com")
for j in judul:
    print('Judul : ',j.text)
```

Pada kode di atas, kita hanya mengambil judul pada halaman “Terpopuler”. Pertama kita masukkan data dari `class box cb-mostpop`. Kemudian kita temukan semua class pada `media__title` dengan `judul = b.find_all('h3', attrs={'class':'media__title'})`. Terakhir, kita iterasi semua judul dan tampilkan dengan `for j in judul: lalu print('Judul : ',j.text)`.

Hasilnya:

```
Terpopuler dari Detik.com
Judul :

                        Dokter Ungkap Tanda Ginjal Bermasalah yang Kerap Tak Disadari

Judul :

                        Gerindra Heran PAW Digugat: Aneh Parpol Tak Berwenang ke Anggota Legislatif

Judul :

                        Bos PPI Nilai Serangan Deddy Sitorus ke Gibran Buntut Konflik PDIP dan Jokowi

Judul :

                        MA Rombak Besar-besaran Hakim dan Ketua Pengadilan Negeri

Judul :

                        Awal Mula Wanita Tangsel Idap Gangguan Liver di Usia 21, Ini Gejala yang Dikeluhkan
```

8. Membuat kode untuk mengambil dan menampilkan data tertentu dari sebuah tabel HTML yang tersimpan dalam file tabel.html. Data yang diambil adalah nilai dari kolom "Jabatan" dan "Tunjangan" dari setiap baris dalam tabel.

   Berikut merupakan isi dari file tabel1.html

```html
<!DOCTYPE html>
<html>
<head>
  <title></title>
</head>
<body>
<h1>Data Pegawai</h1>
<a href='jabatan-baru.php'>Tambah Data</a>
<hr/>
<table border='1' width='100%'>
<thead>
<tr style='background:blue;color:yellow'>
    <th>NO</th><th>KODE</th><th>NAMA JABATAN</th><th>TUNJANGAN</th>
</tr>
</thead>
<tbody>
  <tr style='background:lightgreen'>
    <td>1</td><td>01</td><td>DIREKTUR</td><td style='text-align:right'>1,500,000</td>
  </tr>
  <tr style='background:pink'>
    <td>2</td><td>02</td><td>KABAG</td><td style='text-align:right'>1,000,000</td>
  </tr>
  <tr style='background:lightgreen'>
    <td>3</td><td>04</td><td>KASUBAG</td><td style='text-align:right'>250,000</td>
  </tr>
  <tr style='background:pink'>
    <td>4</td><td>03</td><td>SUPERVISOR</td><td style='text-align:right'>500,000</td>
  </tr>
  <tr style='background:lightgreen'>
    <td>5</td><td>05</td><td>SEKERTARIS</td><td style='text-align:right'>100,000</td>
  </tr>
</tbody>
</table>

</body>
</html>
```

Pada kasus di dibawah ini, kita mengambil data dari setiap baris atau _row_ pada tabel di dalam file tabel.html. Setelah mengambil data-data text tersebut, kita tampilkan hasilnya sedemikian rupa dengan baris kode `print("Jabatan : ", td[2].text,"Tunjangan :", td[3].text)`.

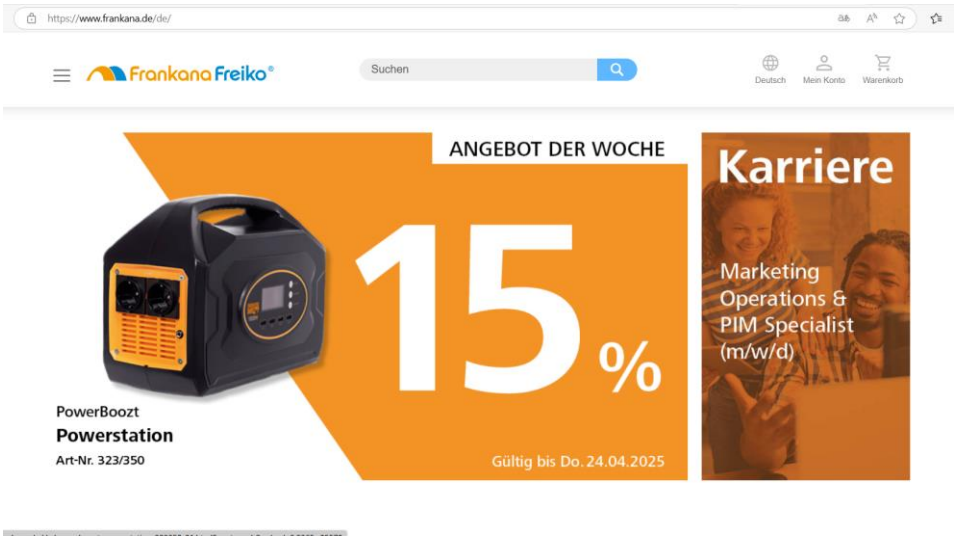
```python
from bs4 import BeautifulSoup as bs

html = open("tabel.html", "r")

s = bs(html, "lxml")

tb = s.find("table")

tr = tb.find_all("tr")
for row in tr:
    td = row.find_all("td")
    if td:
        #baris pertama tr berisi th, jadi nilai td=false
        #maka perlu ada if td, agar yang ditampilkan
        #hanya yg ada isinya
        #tampilkan jabatan dan tunjangan saja
        print("Jabatan : ",td[2].text,"Tunjangan :",td[3].text)
```

Hasilnya :

```
Jabatan :  DIREKTUR Tunjangan : 1,500,000
Jabatan :  KABAG Tunjangan : 1,000,000
Jabatan :  KASUBAG Tunjangan : 250,000
Jabatan :  SUPERVISOR Tunjangan : 500,000
Jabatan :  SEKERTARIS Tunjangan : 100,000
```

Scraping Data Online Shop

1. Membuka website online shop https://www.frankana.de/



2. Membuat kode untuk melakukan web scraping terhadap website e-commerce https://www.frankana.de/de/multimedia.html untuk mengambil data produk dan harganya, lalu menyimpannya dalam file Excel (.xls) menggunakan library xlwt.

```python
# import the library used to query a website
import requests
import xlwt
from bs4 import BeautifulSoup as bs
from urllib import request
import urllib.request

result = xlwt.Workbook()
sheet = result.add_sheet("Product Info")
sheet.write(0, 1, "Product Name")
sheet.write(0, 2, "Price")

wiki_url = "https://www.frankana.de/de/multimedia.html"
page = urllib.request.urlopen(wiki_url)
print(type(page))
soup = bs(page, "html.parser")
print(type(soup))
all_products = soup.find_all("li", {"class": "item product product-item"})
index = 0
for product in all_products:
    productname = product.find("div", {"class": "product details product-item-details"}).find("a").get("product-item-link")
    index = index + 1
    sheet.write(index, 1, productname)
    price = product.find("div", {"class": "price-box"}).find_all("span", {"class": "price"})[-1].text
    sheet.write(index, 2, price)
result.save("product list.xls")
```

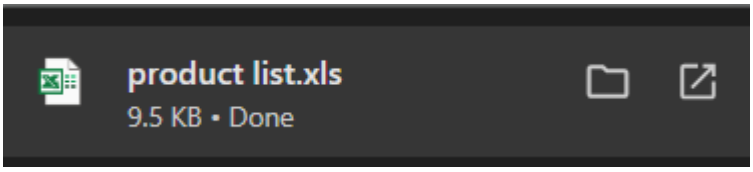
Secara garis besar, prosesnya mencakup 4 hal:

1) Mengimport library yang digunakan untuk query website yaitu requests, xlwt, BeautifulSoup, dan urllib

2) Menentukan url yang dituju yaitu https://www.frankana.de/de/multimedia.html

3) Parsing html dalam variabel “page” dan simpan dalam format Beautifulsoup

4) Simpan file yang sudah jadi ke dalam format excel Hasil akan disimpan dengan nama file product list dalam bentuk excel. File disimpan dalam satu folder sama dengan nama file python kita.



Hasilnya :

Product Name	Price
TV SLA-Linie	579,00 €
Standfuß schwarz für TV Reflexion LDDX27iBT	24,90 €
Standfuß schwarz für TV Reflexion LDDX32iBT	24,90 €
Sat-Anlage Megasat Campingman Portable 4	919,00 €
Sat-Anlage CombiSat WiFi 65	1.655,00 €
Sat-Anlage Onelight@ 65	1.399,00 €
Router ALDEN I-NET 151	599,00 €
WiFi-Booster Megasat Camper Connected	199,00 €
Routerset Megasat Camper Connected 5G Ready	299,00 €
Sat-Anlage CombiSat WiFi 65 Twin	1.825,00 €
Sat-Anlage CombiSat WiFi 65	1.825,00 €
Router CAMPERNET 2.0 EVO	649,00 €
LTE / 5G / WLAN Dachantenne Campernet für Vans	159,00 €
LTE / 5G / WLAN Dachantenne Campernet für Vans	159,00 €
Routerset Pioneer DCT-WR204-SH	599,00 €
Routerset Caratec Electronics CET305R.2	1.099,00 €
Routerset Caratec Electronics CET205R	799,00 €
Autoradio / CD-Spieler Kenwood DPX-7300DAB	219,00 €
Autoradio / CD-Spieler Kenwood KDC-BT960DAB	219,00 €
Aktive FM/DAB+ Kombiflex-Dachantenne, 12 Volt	116,90 €
Aktive FM/DAB+/GPS Kombiflex-Dachantenne für Reisemobile mit GfK-Dach, 12 Volt	349,00 €
Aktive FM/DAB+ Kombiflex-Dachantenne für Reisemobile mit GfK-Dach, 12 Volt	299,00 €
Aktive FM/DAB+ Kombiflex-Dachantenne für Reisemobile mit GfK-Dach, Einkabel, 12 Volt	279,00 €
Aktive FM/DAB+/GPS Kombiflex-Schrägdachantenne, 12 Volt	189,00 €
Aktive FM/DAB+/GPS Kombiflex-Dachantenne, 12 Volt	159,00 €
Radioblende N-XFDUC-1D8 für Fiat Ducato ab Bj. 2021/09	49,00 €
Moniceiver mit Navigations-App Pioneer SPH-EVO107 DAB-C	1.249,00 €
Reflexion LDDXiBT, schwarz	389,00 €
AS4 80 SKEW Ultrawhite inkl. TV A.I.O. Smart	2.649,00 €

C. Referensi

[1] D. D. A. Yani, H. S. Pratiwi, dan H. Muhardi, "Implementasi Web Scraping untuk Pengambilan Data pada Situs Marketplace," *Jurnal Sistem dan Teknologi Informasi*, vol. 7, no. 4, hlm. 257, 2019. doi: 10.26418/justin.v7i4.30930
[2] Modul Web Scraping