

BAB 3

3.1 Capaian Praktikum Pertemuan 3

Pada praktikum pertemuan ke-3 ini mahasiswa diharapkan mampu mengintegrasikan fungsi pembuatan folder dan file dengan request URL, menggunakan BeautifulSoup dan requests untuk scraping data, serta menganalisis dan menyimpan hasil scraping tersebut ke dalam file secara terstruktur.

3.2 Indikator Capaian

- Mahasiswa melakukan integrasi fungsi folder dan request.
- Mahasiswa melakukan integrasi fungsi pada beautifulsoup dan request.
- Mahasiswa melakukan scraping pada website dengan menggunakan request & beautifulsoap serta menyimpannya dalam file.

3.3 Landasan Teori

Menurut Dr. Charles Severance, Web scraping adalah teknik dasar untuk mengambil data dari website ketika API tidak tersedia. Menggunakan tools seperti BeautifulSoup dan requests memungkinkan mahasiswa untuk berinteraksi langsung dengan konten web secara nyata dan meningkatkan logika pemrograman mereka.

Menurut Ryan Mitchell, Web scraping melatih mahasiswa untuk memahami struktur HTML, permintaan HTTP, dan proses parsing data. Ini bukan hanya tentang menulis kode, tapi juga tentang pemecahan masalah dan pemahaman etika pengambilan data.

Menurut Eric Matthes, dengan menggunakan BeautifulSoup dan requests, mahasiswa mendapatkan pengalaman nyata dalam berinteraksi dengan internet dan mengekstrak data yang berguna. Ini adalah salah satu cara terbaik untuk membuat Python terasa praktis dan memberdayakan.

3.4 Pelaksanaan Praktikum

3.4.1 Percobaan Pertama

Pada percobaan pertama ini, mahasiswa diperintahkan untuk membuat pengaturan pada sisi program utama (main.py) agar dapat membaca data dari suatu URL secara otomatis, serta memungkinkan pengguna untuk menentukan URL dan direktori penyimpanan secara manual atau dinamis. Dalam proses scraping pertama kali, program diatur agar mengambil dan menampilkan struktur data dari website (tidak dilakukan berulang-ulang), serta menampilkan waktu dan tanggal saat data diambil. Selain itu, program juga diatur agar dapat menyimpan hasil scraping ke dalam file dengan format yang memuat informasi tanggal dan waktu pengambilan data secara lengkap.

a. Script / Setting Program

Main.py

```
from bs4 import BeautifulSoup
import requests
import fungsi # pastikan modul ini berisi fungsi create_directory, create_new_file,
write_to_file, etc.

def main_scraper(url, directory):
    fungsi.create_directory(directory)

    response = requests.get(url)
    if response.status_code != 200:
        print(f"Gagal mengakses halaman: {url}")
        return

    soup = BeautifulSoup(response.text, "html.parser")

    # Ambil judul utama halaman
    title_tag = soup.select_one("#firstHeading")
    title = title_tag.text.strip() if title_tag else "Tidak ada judul"

    # Ambil paragraf pertama dari isi halaman
    content_div = soup.select_one("div.mw-parser-output")
    if not content_div:
        print("Konten tidak ditemukan.")
        return

    first_paragraph = ""
    for element in content_div.find_all(["p", "div"], recursive=False):
        if element.name == "p" and element.text.strip():
            first_paragraph = element.text.strip()
            break
```

```

# Ambil semua subjudul h2 dan h3
subheadings = []
for heading in content_div.find_all(["h2", "h3"]):
    span = heading.find("span", class_="mw-headline")
    if span:
        subheadings.append(span.text.strip())

# Format hasil scraping
article_format = (
    f"Judul Halaman: {title}\n\n"
    f"Ringkasan Awal:\n{first_paragraph}\n\n"
    f"Daftar Subjudul:\n"
)

for i, sub in enumerate(subheadings, 1):
    article_format += f"{i}. {sub}\n"

article_format += f"- *40\nSumber: {url}\n"

# Simpan ke file
file_path = f"{directory}/artikel.txt"
if not fungsi.does_file_exist(file_path):
    fungsi.create_new_file(file_path)

fungsi.write_to_file(file_path, article_format)
print("Scraping selesai. Data disimpan di folder:", directory)

# Jalankan
main_scraper("https://id.wikipedia.org/wiki/Sejarah_Islam", "Hasil")

```

Fungsi.py

```

import os

def create_directory(path):
    if not os.path.exists(path):
        os.mkdir(path)

def create_new_file(path):
    f = open(path, 'w', encoding='utf-8')
    f.write("")
    f.close()

def does_file_exist(path):
    return os.path.exists(path)

def write_to_file(path, text):
    with open(path, 'a', encoding='utf-8') as f:
        f.write(text + "\n")

```

b. Langkah Uji Coba

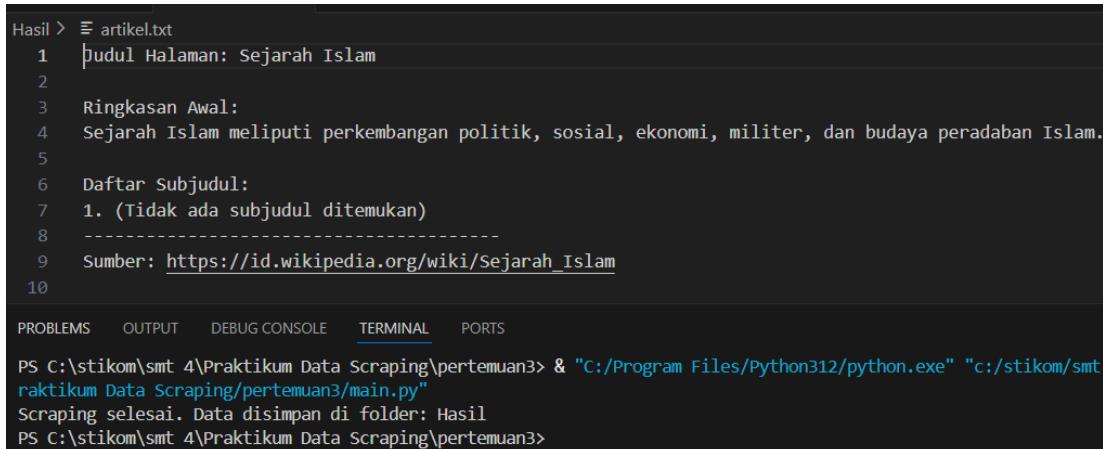
Menjalankan main.py

- **BeautifulSoup**: Untuk mem-parsing dan mengekstrak elemen HTML.
- **requests**: Untuk melakukan permintaan HTTP (GET) ke URL target.
- **main_scraper(url, directory)**, bertugas melakukan seluruh proses scraping dan penyimpanan hasil.
- **fungsi.create_directory(directory)**, mengecek dan membuat folder (misalnya Hasil/) jika belum ada.
- **response = requests.get(url), if response.status_code != 200: print(f"Gagal mengakses halaman: {url}")**, mengirim permintaan ke URL yang diberikan. Jika gagal (status_code ≠ 200), fungsi akan dihentikan.
- **soup = BeautifulSoup(response.text, "html.parser")**, parsing HTML dari halaman menjadi objek yang bisa ditelusuri dengan BeautifulSoup.
- **title_tag = soup.select_one("#firstHeading"), title = title_tag.text.strip() if title_tag else "Tidak ada judul"**, mengambil elemen id="firstHeading" yang berisi judul utama artikel Wikipedia.
- **div.mw-parser-output**, kontainer utama isi artikel Wikipedia.
- **for heading in content_div.find_all(["h2", "h3"]): span = heading.find("span", class_="mw-headline")**, mencari semua elemen heading (h2 dan h3) dalam isi artikel yang memiliki span.mw-headline, yang merupakan subjudul artikel.
- **file_path = f"{directory}/artikel.txt"**, mengecek apakah file artikel.txt sudah ada. Jika belum, dibuat file kosong.
- **print("Scraping selesai. Data disimpan di folder:", directory)**, menampilkan notifikasi ke terminal bahwa scraping berhasil dilakukan.
- **main_scraper("https://id.wikipedia.org/wiki/Sejarah_Islam", "Hasil")**, menjalankan fungsi scraping dengan: URL Artikel Wikipedia tentang Sejarah Islam.

Menjalankan fungsi.py

- `def create_directory(path): if not os.path.exists(path): os.mkdir(path)`, mengecek apakah folder pada path sudah ada dengan `os.path.exists(path)`. Jika belum ada, akan dibuat folder baru dengan `os.mkdir(path)`.
- `def create_new_file(path): f = open(path, 'w', encoding='utf-8') f.write("") f.close()`, membuka file dengan mode '`w`' (**write**), artinya akan menulis dari awal dan menghapus isi lama jika file sudah ada. Menuliskan string kosong `""` ke dalam file (untuk memastikan file kosong saat dibuat). Ditutup dengan `f.close()`.
- `def write_to_file(path, text): with open(path, 'a', encoding='utf-8') as f: f.write(text + "\n")`, membuka file dalam mode '`a`' (**append**) menambahkan teks di akhir file tanpa menghapus isi lama. Menulis teks yang diberikan ke dalam file, diikuti newline (`\n`) agar teks baru dimulai di baris berikutnya.

c. Hasil Uji Coba



```
Hasil > artikel.txt
1 |Judul Halaman: Sejarah Islam
2 |
3 Ringkasan Awal:
4 Sejarah Islam meliputi perkembangan politik, sosial, ekonomi, militer, dan budaya peradaban Islam.
5 |
6 Daftar Subjudul:
7 1. (Tidak ada subjudul ditemukan)
8 -----
9 Sumber: https://id.wikipedia.org/wiki/sejarah\_Islam
10

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\stikom\smt 4\Praktikum Data Scraping\pertemuan3> & "C:/Program Files/Python312/python.exe" "c:/stikom/smt
raktikum Data Scraping/pertemuan3/main.py"
Scraping selesai. Data disimpan di folder: Hasil
PS C:\stikom\smt 4\Praktikum Data Scraping\pertemuan3>
```

Gambar 3.1 Hasil Scraping dari halaman Wikipedia Sejarah Islam

- Hasil uji coba diatas menunjukkan bahwa program berjalan dengan baik, hanya saja subjudul tidak ditemukan karena kemungkinan struktur halaman tidak memiliki tag `` yang biasa digunakan pada Wikipedia.

d. Analisa Hasil

- Program sudah berfungsi dengan baik untuk scraping judul dan paragraf pertama artikel Wikipedia.
- Hanya saja terdapat kekurangan pada bagian pencarian subjudul karena struktur HTML yang tidak konsisten.
- Perlu dilakukan penyesuaian selector agar lebih fleksibel jika digunakan di berbagai halaman Wikipedia dengan struktur berbeda.

3.4.2 Percobaan Kedua

Pada percobaan kedua ini, mahasiswa diminta mengembangkan program untuk mengekstrak data terstruktur dari halaman web, seperti **judul**, **tanggal posting**, **ringkasan cerita**, dan **URL lanjutan**. Program ini bertujuan agar hasil scraping lebih informatif dan sesuai dengan format artikel yang umum digunakan. Data yang diperoleh kemudian disimpan dalam file teks dengan format yang rapi dan sistematis, serta disertai sumber URL sebagai dokumentasi.

a. Script / Setting Program

Tugas.py

```
import requests
from bs4 import BeautifulSoup
import fungsi
import os
from datetime import datetime

def scrape_article(url):
    headers = {
        "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64)"
    }
    response = requests.get(url, headers=headers)
    if response.status_code != 200:
        print("Gagal mengakses:", url)
        return None

    soup = BeautifulSoup(response.text, "html.parser")
    title_tag = soup.select_one("#firstHeading")
    title = title_tag.text.strip() if title_tag else "-"

    waktu = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
    content_div = soup.select_one("div.mw-parser-output")
    first_para = ""
```

```

if content_div:
    for p in content_div.find_all("p", recursive=False):
        if p.text.strip():
            first_para = p.text.strip()
            break

    subs = []
    if content_div:
        for heading in content_div.find_all(["h2", "h3"]):
            span = heading.find("span", class_="mw-headline")
            if span and span.text.strip():
                subs.append(span.text.strip())

    data = (
        f"Judul : {title}\n"
        f"Tanggal Scraping: {waktu}\n"
        f"Ringkasan Cerita: {first_para}\n"
        f"URL Lanjutan : {url}\n"
        + ("Daftar Subjudul:\n" + "\n".join([f"- {s}" for s in subs]) + "\n" if subs
    else "")
        + "-"*40 + "\n"
    )
    return data

def main_scraper_wiki_all(directory):
    fungsi.create_directory(directory)
    kategori_url = "https://en.wikipedia.org/wiki/Category:Indonesian_folklore"

    headers = {
        "User-Agent": "Mozilla/5.0"
    }

    response = requests.get(kategori_url, headers=headers)
    if response.status_code != 200:
        print("Gagal mengakses halaman kategori.")
        return

    soup = BeautifulSoup(response.text, "html.parser")

    # Ambil semua link artikel dari daftar kategori
    links = soup.select("div.mw-category a")
    print(f"Ditemukan {len(links)} artikel dalam kategori.")

    for link in links:
        relative_url = link.get("href")
        if relative_url is None or not relative_url.startswith("/wiki/"):
            continue

        full_url = "https://en.wikipedia.org" + relative_url
        data = scrape_article(full_url)
        if data:
            print(f"Menyimpan: {link.text}")
            file_path = os.path.join(directory, "artikel_wiki.txt")
            if not fungsi.does_file_exist(file_path):
                fungsi.create_new_file(file_path)

```

```

        fungsi.write_to_file(file_path, data)

    print("\n☑ Scraping semua artikel selesai. Cek folder:", directory)

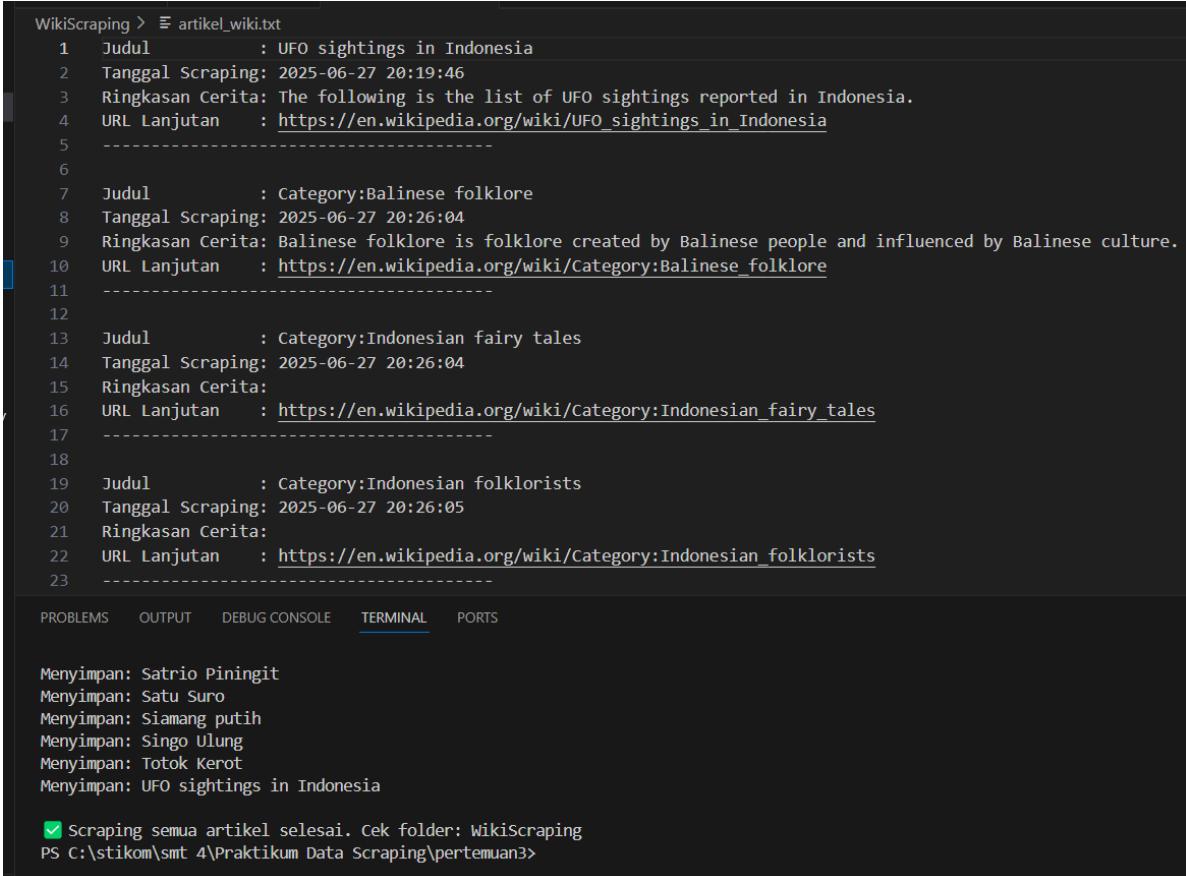
if __name__ == "__main__":
    main_scraper_wiki_all("WikiScraping")

```

b. Langkah Uji Coba

- **kategori_url** = https://en.wikipedia.org/wiki/Category:Indonesian_folklore, mengakses halaman kategori Wikipedia Indonesian folklore, Mengambil semua link artikel dalam kategori tersebut, Membuka setiap artikel satu per satu.
- Semua hasil disimpan ke dalam file teks **artikel_wiki.txt** dalam folder **WikiScraping**.
- **scrape_article(url)**, mengambil satu artikel Wikipedia berdasarkan URL, format data disusun menjadi teks rapi.
- **(div.mw-parser-output p)**, memproses paragraph pertama.
- **main_scraper_wiki_all(directory)**, membuat folder WikiScraping jika belum ada.
- Mengambil semua **elemen <a>** dari bagian **div.mw-category**.
- Melakukan perulangan: mengambil atribut **href** dari link, menggabungkan menjadi URL lengkap, memanggil **scrape_article()** untuk diambil datanya, menyimpan ke file **article_wiki.txt**.

c. Hasil Uji Coba



```
WikiScraping > artikel_wiki.txt
1 Judul : UFO sightings in Indonesia
2 Tanggal Scraping: 2025-06-27 20:19:46
3 Ringkasan Cerita: The following is the list of UFO sightings reported in Indonesia.
4 URL Lanjutan : https://en.wikipedia.org/wiki/UFO\_sightings\_in\_Indonesia
5 -----
6
7 Judul : Category:Balinese folklore
8 Tanggal Scraping: 2025-06-27 20:26:04
9 Ringkasan Cerita: Balinese folklore is folklore created by Balinese people and influenced by Balinese culture.
10 URL Lanjutan : https://en.wikipedia.org/wiki/Category:Balinese\_folklore
11 -----
12
13 Judul : Category:Indonesian fairy tales
14 Tanggal Scraping: 2025-06-27 20:26:04
15 Ringkasan Cerita:
16 URL Lanjutan : https://en.wikipedia.org/wiki/Category:Indonesian\_fairy\_tales
17 -----
18
19 Judul : Category:Indonesian folklorists
20 Tanggal Scraping: 2025-06-27 20:26:05
21 Ringkasan Cerita:
22 URL Lanjutan : https://en.wikipedia.org/wiki/Category:Indonesian\_folklorists
23 -----
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Menyimpan: Satrio Piningit
Menyimpan: Satu Suro
Menyimpan: Siamang putih
Menyimpan: Singo Ulung
Menyimpan: Totok Kerot
Menyimpan: UFO sightings in Indonesia

 Scraping semua artikel selesai. Cek folder: WikiScraping
PS C:\stikom\smt 4\Praktikum Data Scraping\pertemuan3>
```

Gambar 3.2 Hasil Scraping dari halaman Wikipedia

- Program berhasil scraping banyak artikel dari halaman Wikipedia.
- Namun, Wikipedia menyimpan halaman kategori halaman khusus, bukan artikel biasa. Karena tidak cocok untuk di scrape seperti artikel biasa.
- Halaman kategori umumnya tidak memiliki paragraph utama, sehingga beberapa ringkasan cerita kosong.

d. Analisa Hasil

- Program berhasil scraping artikel dari kategori Indonesia folklore di Wikipedia.
- Data yang diambil meliputi judul, tanggal, ringkasan, subjudul, dan URL.
- Looping artikel dan proses penyimpanan berjalan lancar.

- Namun, ada beberapa ringkasan kosong pada halaman kategori karena tidak memiliki paragraf utama.

3.5 Kesimpulan

3.5.1 Kesimpulan Percobaan 1

Mahasiswa telah berhasil melakukan praktikum pada artikel Sejarah Islam, menunjukkan bahwa program scraping berhasil dijalankan dengan baik untuk mengambil data dari halaman Wikipedia. Informasi yang diperoleh meliputi judul halaman, ringkasan paragraf pertama, daftar subjudul, serta URL sumber artikel, program juga mencatat waktu scraping sebagai bagian dari dokumentasi data. Percobaan ini membuktikan bahwa integrasi antara modul requests, BeautifulSoup, dan fungsi bantu dari file fungsi.py telah berjalan efektif untuk melakukan ekstraksi data dari situs web berbasis HTML secara otomatis.

3.5.2 Kesimpulan Percobaan 2

Mahasiswa telah berhasil melakukan praktikum pada artikel Wikipedia kategori Indonesian folklore yang menunjukkan bahwa program mampu melakukan scraping terhadap banyak artikel dalam satu halaman kategori. Program berhasil mengambil data berupa judul, ringkasan paragraf pertama, subjudul, tanggal scraping, dan URL artikel secara otomatis, lalu menyimpannya ke dalam file teks. Hasil yang diperoleh cukup lengkap dan sesuai format, meskipun masih ditemukan beberapa halaman kategori yang ikut terbaca sebagai artikel, sehingga bagian ringkasannya kosong. Hal ini menunjukkan bahwa program sudah berjalan dengan baik dalam melakukan scraping massal, namun masih perlu penyempurnaan dalam penyaringan jenis halaman agar hanya artikel yang relevan saja yang diambil.

Mengetahui:
Dosen Pengampu Mata Kuliah

Arif Hadi Sumitro , M.Kom
NIKP. xxx

DAFTAR PUSTAKA

1. Python for Everybody – www.py4e.com
2. Mitchell, R. (2018). *Web Scraping with Python* (Edisi ke-2). O'Reilly Media.
3. Matthes, E. (2019). Python Crash Course. No Starch Press.

