

**UNIVERSITAS GUNADARMA
PEMROGRAMAN JARINGAN**



**MANUAL BOOK
“Aplikasi Scraping Buku dari Google Books “**

Disusun Oleh :

Zahwa Genoveva (51421554)

**FAKULTAS TEKNOLOGI INDUSTRI JURUSAN
TEKNIK INFORMATIKA UNIVERSITAS
GUNADARMA
2025**

DAFTAR ISI

DAFTAR ISI.....	i
KATA PENGANTAR.....	ii
BAB I.....	1
1.1 Latar Belakang.....	1
BAB II	3
2.1 Pemrograman Jaringan	3
2.2 Konsep HTTP Request dan Response	3
2.3 Screen Scraping dan API Scraping.....	3
2.4 Google Books API.....	4
BAB III.....	6
3.1 Desain dan Perancangan Program	6
BAB IV.....	11
4.1 Kesimpulan.....	11
4.2 Saran.....	11
DAFTAR PUSTAKA.....	12
LAMPIRAN.....	13

KATA PENGANTAR

Puji dan syukur saya panjatkan ke hadirat Tuhan Yang Maha Esa atas limpahan rahmat dan karunia-Nya, sehingga saya dapat menyelesaikan penyusunan *manual book* ini dengan baik dan tepat waktu. *Manual book* ini disusun sebagai bagian dari penyelesaian tugas pada mata kuliah **Pemrograman Jaringan**, dengan judul:

“Aplikasi Scraping Buku Menggunakan Google Books API”

Pada kesempatan ini, saya ingin mengucapkan terima kasih kepada semua pihak yang telah memberikan dukungan dan bantuan selama proses penyusunan, khususnya kepada **dosen pengampu mata kuliah Pemrograman Jaringan** yang telah memberikan arahan dan bimbingan hingga manual ini dapat diselesaikan.

Saya menyadari bahwa *manual book* ini masih jauh dari sempurna dan masih dapat dikembangkan lebih lanjut. Oleh karena itu, saya sangat mengharapkan masukan berupa kritik dan saran yang membangun dari para pembaca sebagai bahan perbaikan ke depannya.

Akhir kata, semoga *manual book* ini dapat memberikan manfaat bagi siapa pun yang membacanya. Terima kasih.

Depok, 29 April 2025

Penulis

BAB I

PENDAHULUAN

1.1 Latar Belakang

Di era digital saat ini, akses terhadap informasi menjadi semakin mudah dan cepat, termasuk dalam hal pencarian referensi buku. Google sebagai salah satu penyedia layanan teknologi terbesar di dunia, menyediakan API (Application Programming Interface) yang memungkinkan pengembang untuk mengakses data buku dari Google Books. API ini sangat berguna untuk membangun aplikasi yang dapat mencari, mengolah, dan menampilkan data buku secara real-time sesuai kata kunci yang dimasukkan pengguna.

Dalam praktiknya, proses pencarian buku secara manual melalui situs web bisa memakan waktu, terutama jika pengguna memerlukan data dalam jumlah besar atau ingin mengolah data tersebut lebih lanjut. Oleh karena itu, dibutuhkan sebuah solusi yang dapat mengotomatisasi proses pencarian dan pengumpulan data buku, serta menyajikannya dalam format yang rapi dan mudah dipahami. Salah satu solusi tersebut adalah dengan membuat aplikasi scraping menggunakan Google Books API.

Aplikasi ini dibangun menggunakan bahasa pemrograman Python dengan bantuan pustaka requests untuk mengambil data dari API, json untuk memproses data, serta ipywidgets untuk membangun antarmuka interaktif di Jupyter Notebook atau Google Colab. Aplikasi ini tidak hanya menampilkan hasil pencarian secara visual, tetapi juga menyediakan fitur untuk menyimpan data dalam format .csv dan .json agar dapat digunakan lebih lanjut.

1.2 Tujuan

Tujuan dari penyusunan *manual book* ini adalah:

- a. Memberikan panduan lengkap dalam menggunakan aplikasi scraping data buku dari Google Books.
- b. Menjelaskan cara kerja aplikasi mulai dari pengambilan data, tampilan hasil, hingga penyimpanan data.
- c. Menyediakan dokumentasi yang dapat digunakan oleh pengguna awam maupun pengembang untuk memahami dan memanfaatkan aplikasi ini secara maksimal.

1.3 Rumusan Masalah

Rumusan masalah yang dibahas dalam *manual book* ini meliputi:

- a. Bagaimana cara membangun aplikasi yang dapat melakukan pencarian data buku dari Google Books API secara otomatis?
- b. Bagaimana menampilkan hasil pencarian buku dalam format yang interaktif dan mudah dibaca oleh pengguna?
- c. Bagaimana menyimpan hasil scraping data buku ke dalam format file .csv dan .json secara praktis?

BAB II

PEMBAHASAN

2.1 Pemrograman Jaringan

Pemrograman jaringan adalah cabang ilmu komputer yang mempelajari bagaimana dua atau lebih perangkat komputer dapat saling berkomunikasi melalui jaringan komputer dengan menggunakan protokol komunikasi tertentu. Dalam konteks aplikasi berbasis internet, pemrograman jaringan melibatkan pemahaman mengenai protokol jaringan seperti **HTTP/HTTPS**, **TCP/IP**, serta penggunaan **library** atau **API** untuk mengakses dan bertukar data dengan server.

Mata kuliah Pemrograman Jaringan membekali mahasiswa dengan kemampuan untuk mengembangkan aplikasi yang dapat melakukan komunikasi melalui jaringan, baik secara sinkron maupun asinkron. Salah satu penerapan nyatanya adalah pada proses **pengambilan data melalui internet** menggunakan protokol HTTP, seperti yang dilakukan pada proyek aplikasi scraping buku ini. Proyek ini menerapkan prinsip pemrograman jaringan untuk mengambil data dari server Google Books melalui permintaan HTTP (HTTP Request) yang dilakukan oleh program Python.

2.2 Konsep HTTP Request dan Response

HTTP (Hypertext Transfer Protocol) adalah protokol komunikasi utama yang digunakan dalam jaringan internet untuk pertukaran data antara klien dan server. Dalam aplikasi scraping ini, **HTTP GET request** digunakan untuk mengambil data dari endpoint Google Books API. Data dikirim oleh server dalam bentuk **response** berformat **JSON (JavaScript Object Notation)**.

Secara umum, struktur HTTP request meliputi:

- URL (Uniform Resource Locator)
- Metode request (GET, POST, PUT, DELETE)
- Header
- Parameter (query string)

Contoh penggunaan HTTP GET dalam kode program:

```
response = requests.get('https://www.googleapis.com/books/v1/volumes', params=params)
```

Setelah permintaan dikirim, respons dari server diproses dalam bentuk JSON:

```
data = response.json()
```

2.3 Screen Scraping dan API Scraping

Screen scraping adalah proses mengekstrak data dari tampilan antarmuka pengguna (biasanya halaman web) secara otomatis, tanpa menggunakan antarmuka resmi yang disediakan. Metode ini sering digunakan untuk mengambil data dari situs yang tidak menyediakan API.

Namun, dalam proyek ini digunakan pendekatan **API scraping**, yaitu mengambil data secara

langsung dari API resmi (dalam hal ini, **Google Books API**), bukan dari halaman web. API scraping lebih direkomendasikan karena:

- Lebih stabil dan tidak tergantung pada struktur HTML yang bisa berubah-ubah.
- Lebih cepat dan efisien.
- Lebih aman karena biasanya menggunakan autentikasi dan dokumentasi resmi.

API scraping juga memberikan hasil dalam bentuk JSON, yang memudahkan pengolahan data menggunakan Python.

2.4 Google Books API

Google Books API adalah layanan publik dari Google yang memungkinkan pengembang mengakses informasi tentang buku seperti judul, penulis, penerbit, tanggal terbit, deskripsi, dan gambar sampul. API ini sangat bermanfaat untuk membangun sistem pencarian buku, katalog digital, atau aplikasi edukasi berbasis pustaka online.

Endpoint yang digunakan dalam program:

```
https://www.googleapis.com/books/v1/volumes
```

Parameter penting yang digunakan:

- q : kata kunci pencarian
- startIndex : posisi awal data
- maxResults : jumlah maksimum data yang diambil per request
- langRestrict : membatasi hasil berdasarkan bahasa

Contoh parameter dalam kode:

```
params = {  
    'q': keyword,  
    'startIndex': start,  
    'maxResults': 40,  
    'printType': 'books',  
    'langRestrict': 'id'  
}
```

Dengan API ini, pengguna dapat melakukan pencarian hingga ratusan buku secara otomatis dan menampilkannya dalam bentuk yang interaktif melalui antarmuka pengguna berbasis ipywidgets.

2.5 JSON (JavaScript Object Notation)

adalah format pertukaran data yang ringan dan mudah dibaca oleh manusia maupun mesin. Dalam konteks aplikasi ini, semua data hasil scraping dari Google Books API dikembalikan dalam format JSON. Format ini kemudian diproses dan diubah menjadi struktur Python

(dictionary dan list) untuk ditampilkan dan disimpan.

Contoh struktur JSON dari hasil API:

```
{  
  "items": [  
    {  
      "volumeInfo": {  
        "title": "Judul Buku",  
        "authors": ["Nama Penulis"],  
        "publishedDate": "2020",  
        "publisher": "Nama Penerbit",  
        "description": "Deskripsi...",  
        "imageLinks": {  
          "thumbnail": "URL Gambar"  
        }  
      }  
    }  
  ]  
}
```

2.6 Penyimpanan Data: Format CSV dan JSON

Aplikasi ini menyediakan fitur untuk menyimpan data hasil scraping ke dalam dua format:

- **CSV (Comma Separated Values):** Format sederhana untuk menyimpan data tabular, dapat dibuka di aplikasi seperti Microsoft Excel atau Google Sheets.
- **JSON:** Format yang fleksibel dan cocok untuk digunakan kembali dalam aplikasi berbasis web atau API.

Fungsi penyimpanan ini menggunakan modul csv dan json dari Python, dan memungkinkan pengguna untuk menyimpan hingga 100 data buku yang telah diambil.

2.7 Antarmuka Interaktif dengan ipywidgets

ipywidgets adalah library Python yang memungkinkan pembuatan antarmuka pengguna interaktif di Jupyter Notebook. Dalam aplikasi ini, ipywidgets digunakan untuk:

- Menampilkan input teks untuk kata kunci pencarian.
- Menyediakan tombol untuk memicu pencarian.
- Menampilkan output berupa daftar buku dalam tampilan HTML yang menarik.

Penggunaan ipywidgets sangat cocok untuk prototipe aplikasi karena tidak memerlukan instalasi GUI tambahan dan langsung dapat digunakan di Google Colab.

BAB III

ANALISA DAN PERANCANGAN

Analisis kode program dilakukan untuk memahami struktur, logika, dan komponen-komponen utama yang terkandung dalam implementasi. Untuk membangun aplikasi scraping buku berbasis Google Books API, diperlukan pemahaman yang baik terhadap konsep pemrograman jaringan, API, serta antarmuka interaktif menggunakan ipywidgets. Aplikasi ini bertujuan untuk mempermudah pengguna dalam mencari dan mengoleksi data buku berdasarkan kata kunci tertentu, yang kemudian dapat disimpan ke dalam file berformat .csv dan .json.

3.1 Desain dan Perancangan Program

a. Flowchart Proses Aplikasi

```
[ Mulai ]
  ↓
[ Input Kata Kunci ]
  ↓
[ Kirim HTTP Request ke Google Books API ]
  ↓
[ Ambil & Parsing Data JSON ]
  ↓
[ Tampilkan Buku dalam Format HTML ]
  ↓
[ (Opsional) Simpan ke CSV / JSON ]
  ↓
[ Selesai ]
```

b. Struktur Data yang Digunakan

Hasil dari API akan dikonversi ke dalam bentuk dictionary Python yang memiliki struktur sebagai berikut:

```
book = {
    'judul': 'Judul Buku',
    'penulis': 'Nama Penulis',
    'tanggal': '2022',
    'penerbit': 'Penerbit Buku',
    'deskripsi': 'Deskripsi ringkas...',
    'sampul': 'https://link.to/thumbnail.jpg'
}
```

Data dikumpulkan dalam list:

```
books = [book1, book2, book3, ..., book100]
```

3.2 Struktur Proyek

Struktur file program dalam proyek ini adalah sebagai berikut:

📁 Struktur Folder

📝 Keterangan

- `scraping_buku.ipynb` → Notebook utama untuk scraping data.
- `hasil_scraping.csv` → Data buku hasil scraping dalam format CSV.
- `hasil_scraping.json` → Data buku dalam format JSON.
- `image/` → Folder opsional jika ingin menyimpan gambar buku secara lokal.

3.3 Komponen Utama Program

a. Fungsi Pengambilan Data (Scraping API)

Fungsi `get_books_data(keyword)` akan melakukan 1–3 kali request ke API Google Books (dengan paginasi) untuk mengumpulkan maksimal 100 data buku.

b. Tampilan Data HTML

Fungsi `tampilkan_buku_rapi(buku_list)` merender daftar buku dalam format HTML yang stylish, lengkap dengan gambar sampul dan deskripsi ringkas.

c. Antarmuka Pengguna

Menggunakan ipywidgets, terdiri dari:

- Text input untuk kata kunci
- Tombol pencarian
- Tombol simpan (CSV dan JSON)
- Area tampilan hasil

d. Penyimpanan Data

Fungsi `simpan_csv()` dan `simpan_json()` menyimpan list buku ke dalam file eksternal sesuai format yang dipilih.

3.4 Cara Menjalankan Program

Berikut adalah langkah-langkah menjalankan program:

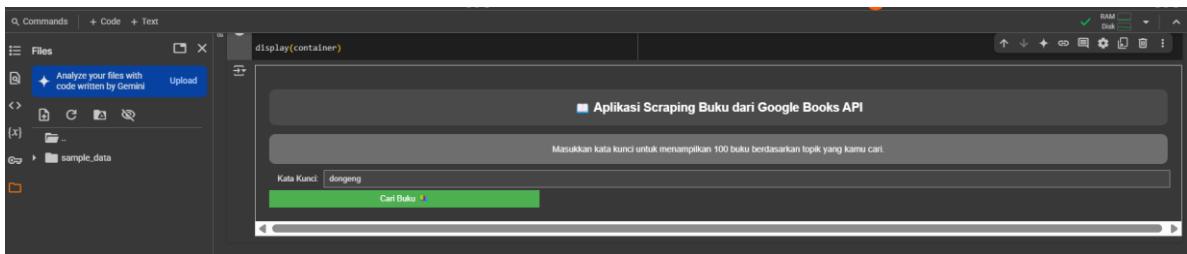
Langkah 1: Siapkan Lingkungan



- Gunakan **Jupyter Notebook** atau **Google Colab**.
- Tidak perlu instalasi tambahan karena requests, json, dan ipywidgets sudah tersedia.

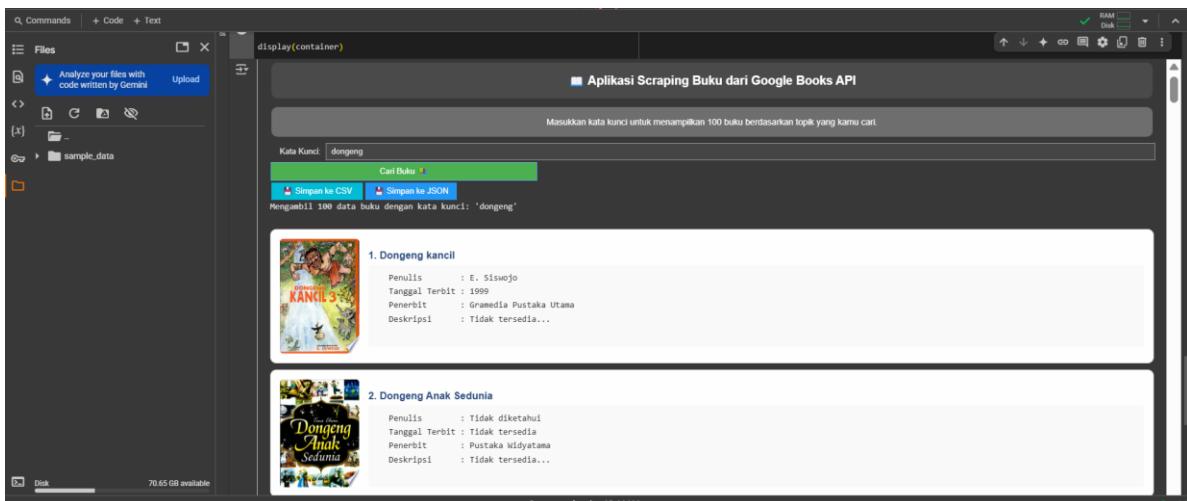
Langkah 2: Jalankan Notebook

1. Buka file scraping_buku.ipynb.
2. Jalankan semua sel (Ctrl + F9 atau Run all).
3. Akan muncul antarmuka dengan:



- Input kata kunci
- Tombol "Cari Buku"
- Tampilan daftar buku
- Tombol simpan data

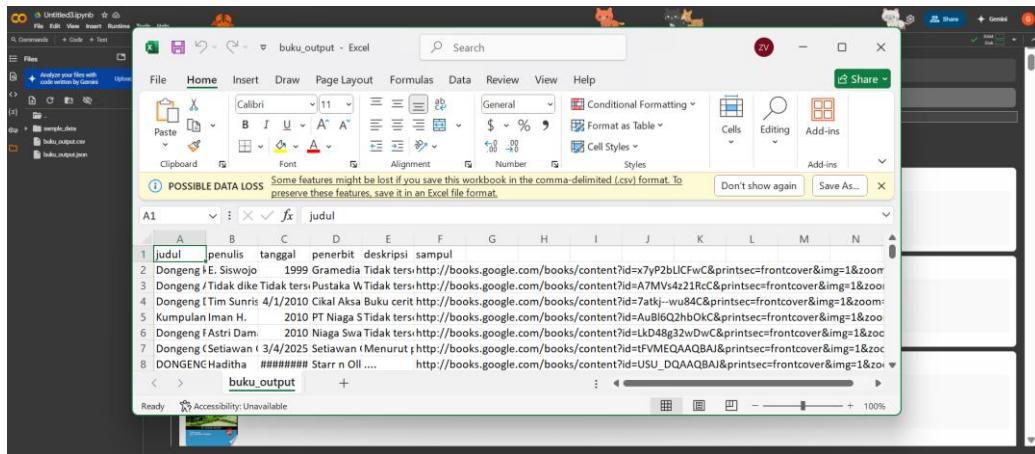
Langkah 3: Mulai Pencarian



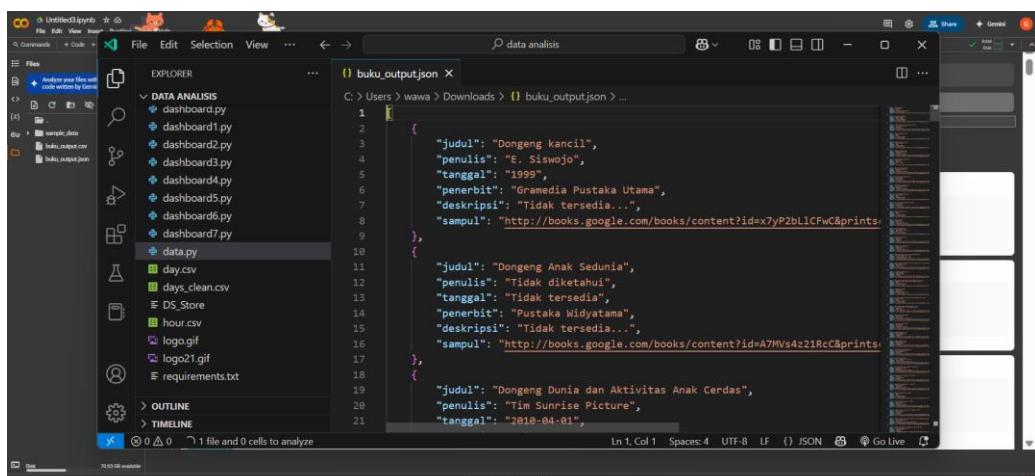
- Masukkan kata kunci, misalnya: filsafat, cerita anak, sains, dll.
- Klik tombol "Cari Buku"
- Tunggu proses pengambilan data selesai (1–2 detik)

Langkah 4: Simpan Data

- Klik tombol **Simpan CSV** untuk menyimpan hasil_scraping.csv



- Klik tombol **Simpan JSON** untuk menyimpan hasil_scraping.json



File akan langsung terbuat di folder kerja notebook yang sedang digunakan.

3.5 Pengujian Program

Program diuji menggunakan kata kunci populer dalam Bahasa Indonesia seperti:

- sejarah

- dongeng
- teknologi
- psikologi anak

Semua hasil menunjukkan pengambilan data maksimal sesuai batas (100 buku), dengan format tampilan HTML yang bersih dan valid, serta file CSV dan JSON berhasil disimpan tanpa error.

Contoh uji coba pada pencarian buku terkait teknologi:

1. Teknologi informasi & Komunikasi 1

Penulis	: Tidak diketahui
Tanggal Terbit	: Tidak tersedia
Penerbit	: Yudhistira Ghalia Indonesia
Deskripsi	: Tidak tersedia...

2. Ilmu, Teknologi Dan Etika

Penulis	: Tidak diketahui
Tanggal Terbit	: Tidak tersedia
Penerbit	: BPK Gunung Mulia
Deskripsi	: Tidak tersedia...

3.6 Kelebihan dan Kekurangan Program

Kelebihan:

- Terintegrasi dengan API resmi (Google Books)
- Tampilan data menarik dan rapi
- Antarmuka interaktif tanpa perlu aplikasi desktop
- Proses scraping cepat (karena langsung dari API)
- Fitur simpan data sangat berguna untuk analisis lanjutan

Kekurangan:

- API tidak selalu mengembalikan 100 hasil, tergantung kata kunci
- Tidak menyimpan gambar ke lokal
- Tidak ada pencarian berdasarkan filter (misal tahun, penulis)

3.7 Video Demonstrasi

Terdapat video demonstrasi penggunaan program yang dapat diakses melalui tautan berikut :

<https://drive.google.com/file/d/1aelzSWQia6zbc3Z8k0-ylc5ftM0UF1je/view?usp=sharing>

BAB IV

PENUTUP

4.1 Kesimpulan

Program scraping buku yang dibuat menggunakan Google Books API telah berhasil diimplementasikan dengan baik dan mencerminkan penerapan konsep dasar pemrograman jaringan. Program ini mampu mengirim permintaan ke API, mengolah data JSON yang diterima, dan menampilkannya dalam antarmuka interaktif menggunakan ipywidgets. Fitur penyimpanan data ke dalam format .csv dan `.json` juga berjalan lancar, memberikan kemudahan bagi pengguna untuk menyimpan dan menggunakan data secara fleksibel. Penggunaan API dalam aplikasi ini menunjukkan pendekatan yang legal dan efisien dibanding scraping halaman web secara langsung. Secara keseluruhan, program ini praktis, mudah dijalankan di Jupyter Notebook atau Google Colab, dan memberikan pengalaman penggunaan yang interaktif.

4.2 Saran

Untuk pengembangan selanjutnya, disarankan agar fitur pencarian ditingkatkan dengan opsi filter seperti tahun terbit atau nama penulis agar hasil lebih relevan. Tampilan antarmuka juga bisa dibuat lebih menarik dengan tambahan visualisasi data. Selain itu, aplikasi dapat dikembangkan ke bentuk web menggunakan framework seperti Flask atau Streamlit agar lebih fleksibel. Menambahkan sistem logging dan penanganan error juga akan membuat aplikasi lebih andal saat digunakan secara luas. Ekspansi ke API lain seperti Open Library juga bisa dipertimbangkan untuk memperkaya data buku yang tersedia.

DAFTAR PUSTAKA

Python Software Foundation. (2024). *Requests: HTTP for Humans.* Diakses dari <https://docs.python-requests.org>

Google Developers. (2024). *Google Books API Documentation.* Diakses dari <https://developers.google.com/books/docs/v1/using>

Project Jupyter. (2024). *ipywidgets Documentation.* Diakses dari <https://ipywidgets.readthedocs.io/en/latest/>

Python Software Foundation. (2024). *csv — CSV File Reading and Writing.* Diakses dari <https://docs.python.org/3/library/csv.html>

Python Software Foundation. (2024). *json — JSON encoder and decoder.* Diakses dari <https://docs.python.org/3/library/json.html>

Rouse, M. (2021). *What is an API (Application Programming Interface)?.* TechTarget. Diakses dari <https://www.techtarget.com/searchapparchitecture/definition/application-programming-interface-API>

Prayudi, Y. (2022). *Pemrograman Jaringan dengan Python.* Yogyakarta: Deepublish.

Wahyuni, I. (2020). *Web Scraping untuk Pengumpulan Data Otomatis.* Bandung: Informatika.

LAMPIRAN

A. Kode Program

```
✓ 0s import requests
import csv
import json
import ipywidgets as widgets
from IPython.display import display, clear_output, HTML

# Variabel global untuk menyimpan data buku hasil scraping
hasil_buku_global = []

# Ambil data dari Google Books API
def get_books_data(keyword):
    books = []
    try:
        for start in range(0, 100, 40): # Ambil dalam batch
            params = [
                'q': keyword,
                'startIndex': start,
                'maxResults': 40,
                'printType': 'books',
                'langRestrict': 'id'
            ]
            response = requests.get('https://www.googleapis.com/books/v1/volumes', params=params)
            data = response.json()
            items = data.get('items', [])
            for item in items:
                volume = item.get('volumeInfo', {})
                image_links = volume.get('imageLinks', {})
                book = {
                    'judul': volume.get('title', 'Tidak tersedia'),
                    'penulis': ', '.join(volume.get('authors', ['Tidak diketahui'])),
                    'tanggal': volume.get('publishedDate', 'Tidak tersedia'),
                    'penerbit': volume.get('publisher', 'Tidak tersedia'),
                    'deskripsi': volume.get('description', 'Tidak tersedia')[::300] + '...',
                    'sampul': image_links.get('thumbnail', 'https://via.placeholder.com/128x180?text=No+Image')
                }
                books.append(book)
            if len(books) >= 100:
                break
    except Exception as e:
        books = [{"judul": f"Terjadi kesalahan: {e}"}]
    return books[:100]
```

```
✓ 0s # Format HTML per buku
def tampilan_buku_rapi(buku_list):
    html_output = ""
    for i, b in enumerate(buku_list, 1):
        html_output += f"""
        <div style="display: flex; border: 1px solid #ccc; border-radius: 10px; padding: 15px; margin: 10px 0; background-color: #fefefe; box-shadow: 2px 2px 8px rgba(0,0,0,0.05);>
            
            <div style="flex: 1;>
                <h3 style="margin-bottom: 5px; color: #2b4ace; font-size: 14px; line-height: 1.6; color: #333; background-color: #f9f9f9; padding: 10px; border-radius: 5px; border: 1px solid #ccc; width: 100%;">{b['judul']}</h3>
                <pre style="font-size: 14px; line-height: 1.6; color: #333; background-color: #f9f9f9; padding: 10px; border-radius: 5px; border: 1px solid #ccc; width: 100%;>{b['penulis']}</pre>
                Tanggal Terbit : {b['tanggal']}
                Penerbit : {b['penerbit']}
                Deskripsi : {b['deskripsi']}
            </div>
        </div>
        """
    return html_output

# Fungsi untuk simpan ke CSV
def simpan_csv(b):
    if hasil_buku_global:
        with open('buku_output.csv', 'w', newline='', encoding='utf-8') as f:
            writer = csv.DictWriter(f, fieldnames=hasil_buku_global[0].keys())
            writer.writeheader()
            writer.writerows(hasil_buku_global)
        print("✓ Data berhasil disimpan sebagai 'buku_output.csv'.")

# [3] # Fungsi untuk simpan ke JSON
def simpan_json(b):
    if hasil_buku_global:
        with open('buku_output.json', 'w', encoding='utf-8') as f:
            json.dump(hasil_buku_global, f, indent=4, ensure_ascii=False)
        print("✓ Data berhasil disimpan sebagai 'buku_output.json'.")

# [4] # Komponen GUI
keyword_input = widgets.Text(
    value='dongeng',
    placeholder='Masukkan kata kunci pencarian buku...',
    description='Kata Kunci:',
    layout=widgets.Layout(width='100%')
)

output_area = widgets.Output()
button_area = widgets.HBox() # Akan menampilkan tombol simpan setelah pencarian
```

```

[5] # Fungsi pencarian
def saat_diklik(b):
    global hasil_buku_global
    with output_area:
        clear_output()
        print(f"Mengambil 100 data buku dengan kata kunci: '{keyword_input.value}'\n")
        buku = get_books_data(keyword_input.value)
        hasil_buku_global = buku # simpan untuk keperluan export
        html = tampilkan_buku_rapi(buku)
        display(HTML(html))
        tampilkan_tombol_simpan()

# Fungsi menampilkan tombol simpan
def tampilkan_tombol_simpan():
    tombol_csv = widgets.Button(description="CSV Simpan ke CSV", button_style='info')
    tombol_json = widgets.Button(description="JSON Simpan ke JSON", button_style='primary')
    tombol_csv.on_click(simpan_csv)
    tombol_json.on_click(simpan_json)
    button_area.children = [tombol_csv, tombol_json]

# Tombol cari
tombol_cari = widgets.Button(description="Cari Buku 🔎", button_style='success', layout=widgets.Layout(width='30%'))
tombol_cari.on_click(saat_diklik)

```

```

# Desain ulang GUI akhir
title = widgets.HTML("""
<h2 style='
    color: white;
    background-color: #4a4a4a;
    padding: 15px;
    border-radius: 10px;
    text-align: center;
'>
    Aplikasi Scraping Buku dari Google Books API
</h2>
""")

subtext = widgets.HTML("""
<p style='
    font-size: 14px;
    color: white;
    background-color: #6e6e6e;
    padding: 10px;
    border-radius: 10px;
    text-align: center;
'>
    Masukkan kata kunci untuk menampilkan 100 buku berdasarkan topik yang kamu cari.
</p>
""")

# Membungkus elemen GUI
container = widgets.VBox(
    [title, subtext, keyword_input, tombol_cari, button_area, output_area],
    layout=widgets.Layout(
        border='1px solid #ccc',
        padding='20px',
        border_radius='10px',
        background_color='#2e2e2e',
        box_shadow='0px 4px 10px rgba(0, 0, 0, 0.2)'
    )
)

display(container)

```