LAPORAN UAS

Data Scraping

Disusun untuk Memenuhi UAS Data Sraping
Dibimbing oleh Bapak Pelsri Ramadar N.S. M.Kom

Oleh:

Rhegysa Alvyanthi Juniartha (1123102098)

Imas Nabellia Venda (1123102116)

PROGRAM STUDI S1 TEKNIK INFORMATIKA
SEKOLAH TINGGI ILMU KOMPUTER PGRI BANYUWANGI
2025

Scraping Website Pariwisata

Mengambil semua topik beserta detail data pada website CNN, Kompas, Detik dengan tema pariwisata dan menampilkan hasilnya pada website yang dibuat. Detail artikel meliputi gambar artikel, judul artikel, kategori (apabila ada), dan waktu unggah.

a. Script / Setting Program

app.py

```
from flask import Flask, render_template
import requests
from bs4 import BeautifulSoup
from urllib.parse import urljoin

app = Flask(__name__)

@app.route("/")
def Home():
    return render_template("index.html")

@app.route("/cnn-wisata")
def cnn_wisata():
    html_doc = requests.get("https://www.cnnindonesia.com/tag/wisata")
    soup = BeautifulSoup(html_doc.text, "html.parser")
    populer_area = soup.find(attrs={'class': 'flex flex-col gap-5'})

    articles = []
    if populer_area:
        items = populer_area.findAll("article")
        for item in items:
            image = item.find("img")
            link = item.find("a")
            title = item.find("h2")
            category = item.find("span", {"class": "text-cnn_red"})
            time = item.find("span", {"class": "text-cnn_black_light3"})

            if image and link and title and category and time:
                articles.append({
                    "img_src": image["src"],
                    "img_alt": image["alt"],
                    "link": f"/detail/cnn/{link['href']}",
                    "title": title.text.strip(),
                    "category": category.text.strip(),
                    "time": time.text.strip()
                })

    return render_template("cnn.html", articles=articles)

@app.route("/kompas-wisata")
def kompas_wisata():
    html_doc = requests.get("https://travel.kompas.com/travel-ideas")
    soup = BeautifulSoup(html_doc.text, "html.parser")
    populer_area = soup.find(attrs={'class': 'latest--news mt2 clearfix'})
```

```
    articles = []
    if populer_area:
        items = populer_area.findAll("div", {"class": "article__list clearfix"})
        for item in items:
            image = item.find("img")
            link = item.find("a")
            title = item.find("h3")
            category = item.find("div", {"class": "article__subtitle article__
                       subtitle--inline"})
            time = item.find("div", {"class": "article__date"})

            if image and link and title and category and time:
                articles.append({
                    "img_src": image["data-src"],
                    "img_alt": image["alt"],
                    "link": f"/detail/kompas/{link['href']}",
                    "title": title.text.strip(),
                    "category": category.text.strip(),
                    "time": time.text.strip()
                })

    return render_template("kompas.html", articles=articles)

@app.route("/detik-wisata")
def detik_wisata():
    html_doc = requests.get("https://travel.detik.com/travel-news/indeks")
    soup = BeautifulSoup(html_doc.text, "html.parser")
    populer_area = soup.find(attrs={'class': 'grid-row list-content'})

    articles = []
    if populer_area:
        items = populer_area.findAll("article", {"class": "list-content__item"})
        for item in items:
            image = item.find("img")
            link = item.find("a")
            title = item.find("h3")
            time = item.find("div", {"class": "media__date"})

            if image and link and title and time:
                articles.append({
                    "img_src": image["src"],
                    "img_alt": image["alt"],
                    "link": f"/detail/detik/{link['href']}",
                    "title": title.text.strip(),
                    "time": time.text.strip()
                })

    return render_template("detik.html", articles=articles)

@app.route("/detail/<source>/<path:url>")
def article_detail(source, url):

    def fetch_content(url):
        """Fetch content and next page link from the article page."""
        try:
            html_doc = requests.get(url)
            html_doc.raise_for_status()
```

```
        except requests.exceptions.RequestException as e:
            return None, None, None, None

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

        if source == "cnn":
            title = soup.find("h1", class_="text-[32px]").get_text(strip=True) if
                    soup.find("h1", class_="text-[32px]") else "Judul tidak
                    ditemukan"
            content_div = soup.find("div", class_="detail-text")
            content = content_div.get_text(strip=True, separator="\n") if content_
                      div else "Konten tidak tersedia"
            next_page = soup.find("a", {"class": "inline-block py-2 px-4 text-sm
                        border border-cnn_red"})
            next_page_url = urljoin(url, next_page["href"]) if next_page and next_
                            page.get("href", "").startswith("http") else None
            image = soup.find("img", class_="w-full")
            image_src = image["src"] if image else None
        elif source == "kompas":
            title = soup.find("h1").text.strip() if soup.find("h1") else "Judul
                    tidak ditemukan"
            content_div = soup.find("div", {"class": "read__content"})
            content = content_div.get_text(strip=True, separator="\n") if content_
                      div else "Konten tidak ditemukan"
            next_page = soup.find("a", {"class": "paging__link"})
            next_page_url = None
            if next_page:
                href = next_page.get("href", "")
                if href.startswith("http") or href.startswith("/"):
                    next_page_url = urljoin(url, href)
            image = soup.find("div", {"class": "cover-photo -gallery"})
            if image:
                img_tag = image.find("img")
                image_src = img_tag["src"] if img_tag and "src" in img_tag.attrs
                            else None
            else:
                image_src = None
        elif source == "detik":
            title = soup.find("h1").text.strip() if soup.find("h1") else "Judul
                    tidak ditemukan"
            content_div = soup.find("div", {"class": "detail__body-text"})
            content = content_div.get_text(strip=True, separator="\n") if
                      content_div else "Konten tidak ditemukan"
            next_page = soup.find("a", {"class": "detail__btn-next"})
            next_page_url = urljoin(url, next_page["href"]) if next_page and
                            "href" in next_page.attrs else None
            image_section = soup.find("div", {"class": "detail__media"})
            if image_section:
                img_tag = image_section.find("img")
                image_src = img_tag["src"] if img_tag and "src" in img_tag.attrs
                            else None
            else:
                image_src = None

        else:
            return None, None, None, None

        return title, content, next_page_url, image_src
```

```
    full_content = ""
    next_url = url
    title = None
    image_src = None
    while next_url:
        current_title, content, next_url, current_image_src = fetch_content
        (next_url)
        if current_title:
            title = title or current_title
            image_src = image_src or current_image_src
            full_content += content + "\n"
        else:
            break
    detail = {
        "title": title,
        "content": full_content.strip(),
        "image_src": image_src
    }

    return render_template("detail.html", detail=detail)

if __name__ == "__main__":
    app.run(debug=True)
```

index.html

```
<div class="collapse navbar-collapse" id="navbarResponsive">
   <ul class="navbar-nav ms-auto py-4 py-lg-0">
      <li class="nav-item"><a class="nav-link px-lg-3 py-3 py-lg-4" href="/">
       Home</a></li>
      <li class="nav-item"><a class="nav-link px-lg-3 py-3 py-lg-4" href="cnn-
       wisata">CNN</a></li>
      <li class="nav-item"><a class="nav-link px-lg-3 py-3 py-lg-4" href="kompas-
       wisata">KOMPAS</a></li>
      <li class="nav-item"><a class="nav-link px-lg-3 py-3 py-lg-4" href="detik-
       wisata">DETIK</a></li>
   </ul>
</div>
```

cnn.html

```
{% if articles %}
   {% for article in articles %}
   <div class="post-preview">
      <a href="{{ article.link }}" target="_blank">
         <img src="{{ article.img_src }}" alt="{{ article.img_alt }}" class="img-
          fluid mb-3">
         <h2 class="post-title">{{ article.title }}</h2>
         <h3 class="post-subtitle">{{ article.category }}</h3>
      </a>
      <p class="post-meta"> Diposting pada {{ article.time }} </p>
   </div>
   <hr class="my-4" />
   {% endfor %}
{% else %}
   <p class="text-center">Tidak ada artikel ditemukan.</p>
{% endif %}
```

kompas.html

```
{% if articles %}
   {% for article in articles %}
   <div class="post-preview">
      <a href="{{ article.link }}" target="_blank">
         <img src="{{ article.img_src }}" alt="{{ article.img_alt }}" class="img-
          fluid mb-3">
         <h2 class="post-title">{{ article.title }}</h2>
         <h3 class="post-subtitle">{{ article.category }}</h3>
      </a>
      <p class="post-meta"> Diposting pada {{ article.time }} </p>
   </div>
   <hr class="my-4" />
   {% endfor %}
{% else %}
   <p class="text-center">Tidak ada artikel ditemukan.</p>
{% endif %}
```

detik.html

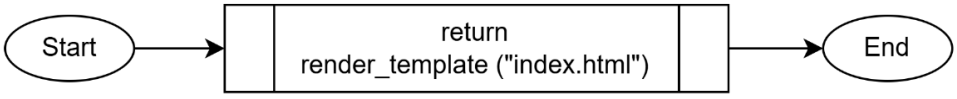
```
{% if articles %}
   {% for article in articles %}
   <div class="post-preview">
      <a href="{{ article.link }}" target="_blank">
         <img src="{{ article.img_src }}" alt="{{ article.img_alt }}" class="img-
          fluid mb-3">
         <h2 class="post-title">{{ article.title }}</h2>
      </a>
      <p class="post-meta"> Diposting pada {{ article.time }} </p>
   </div>
   <hr class="my-4" />
   {% endfor %}
{% else %}
   <p class="text-center">Tidak ada artikel ditemukan.</p>
{% endif %}
```

detail.html

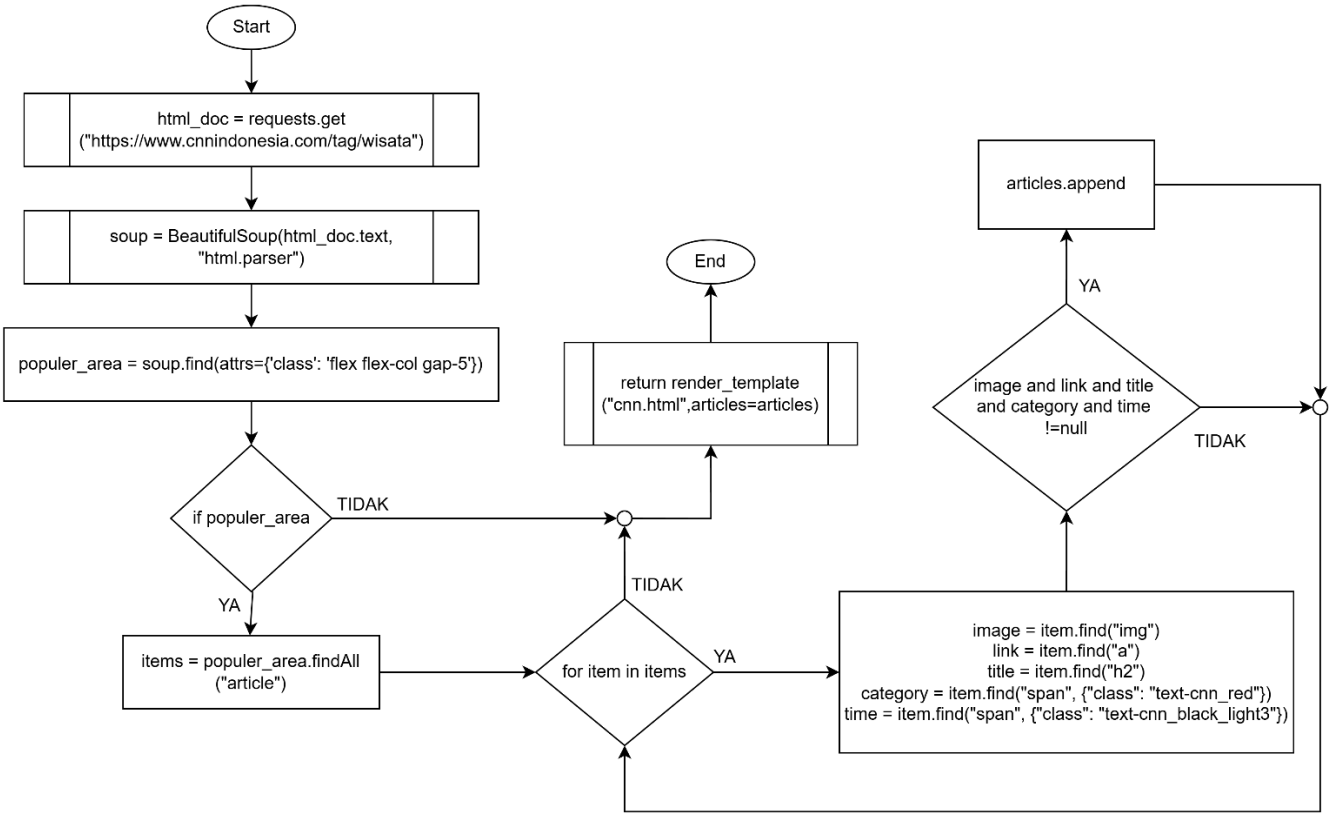
```
<header class="masthead" style="background-image: url('{{ url_for('static',
filename='assets/img /home-bg.jpg') }}');">
   <h1>{{ detail.title }}</h1>
   <span class="subheading">Detail Berita Pariwisata</span>
</header>

<article class="mb-4">
   {% if detail.image_src %}
      <img class="img-fluid rounded" src="{{ detail.image_src }}" alt="Gambar
       Utama">
   {% endif %}
   <p class="mt-4">{{ detail.content }}</p>
</article>
```

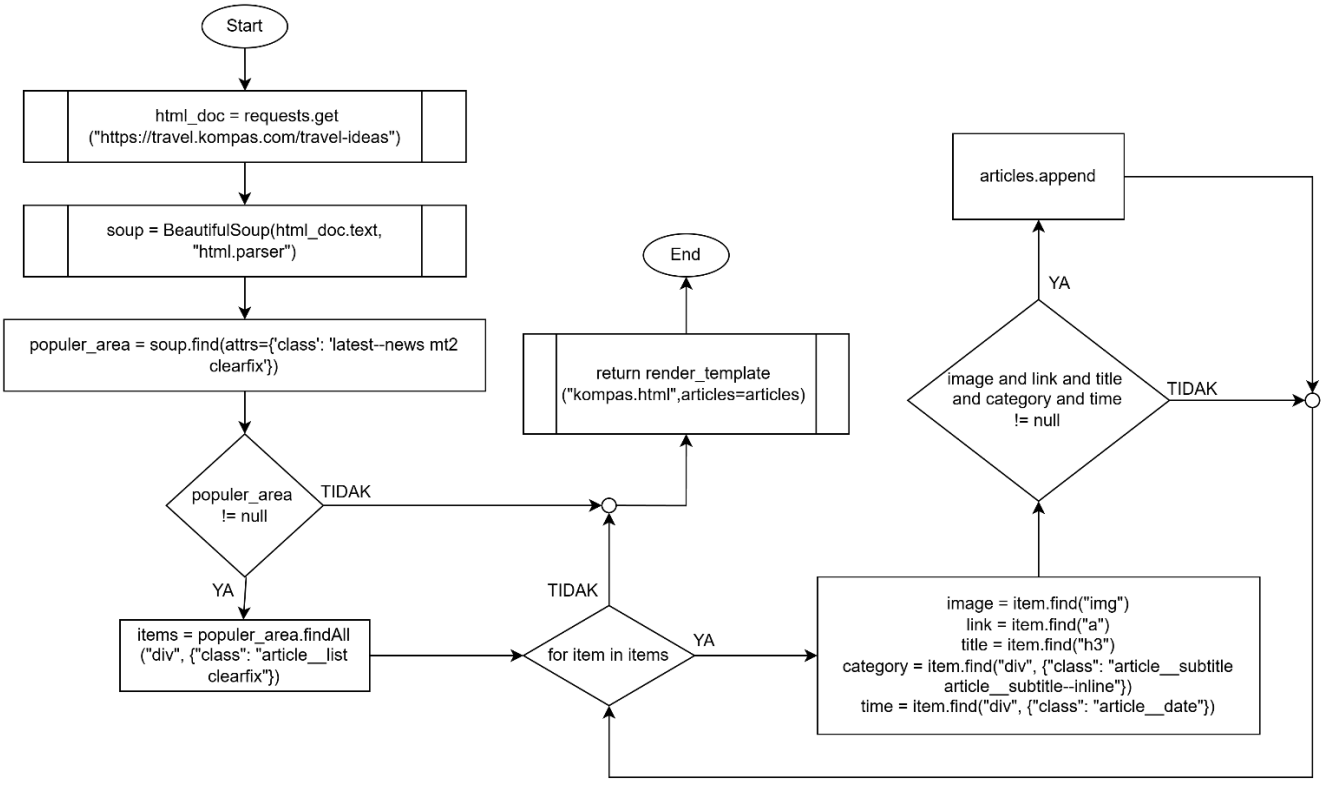
b. Flowchart



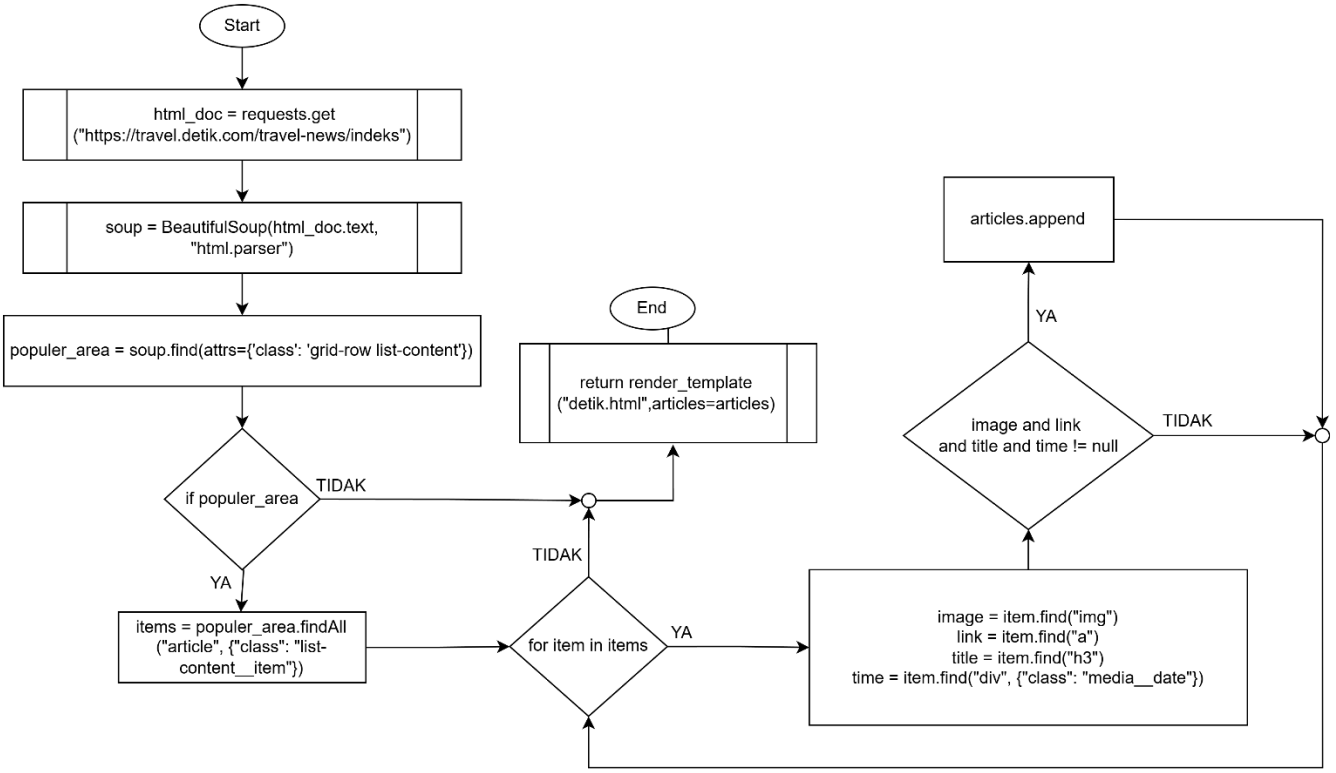
Flowchart Fungsi home()



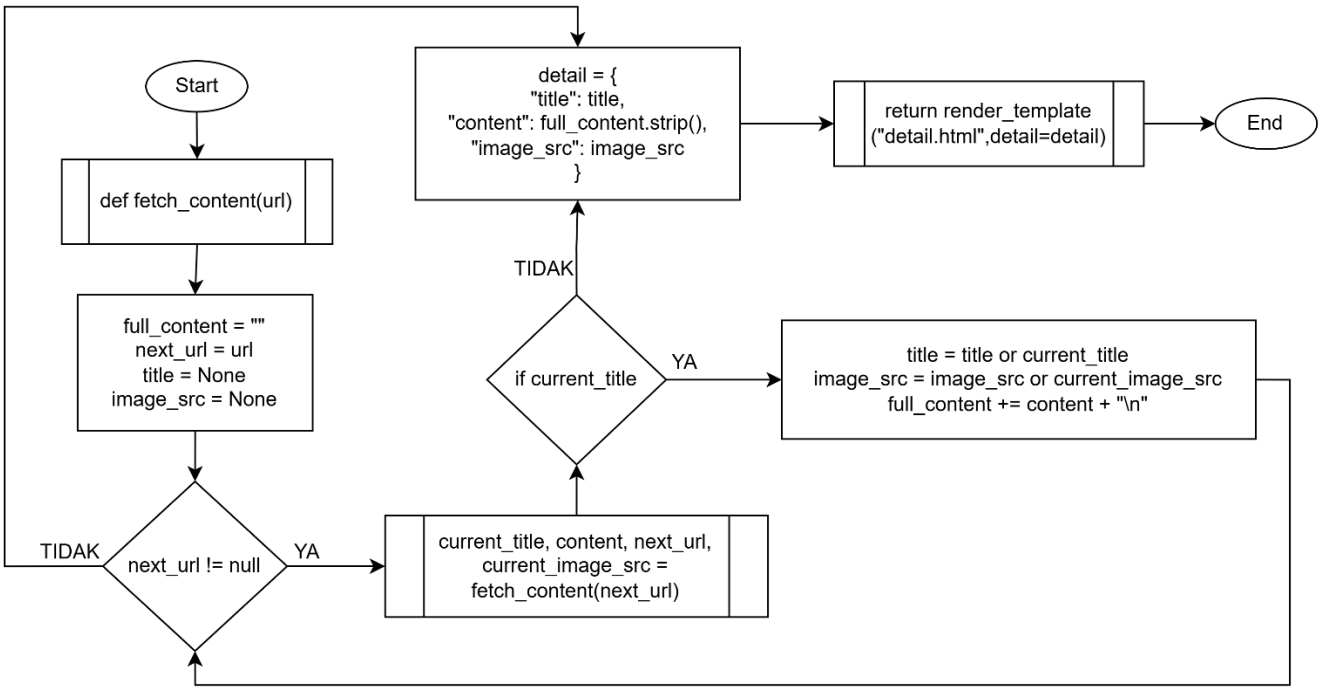
Flowchart Fungsi cnn_wisata()



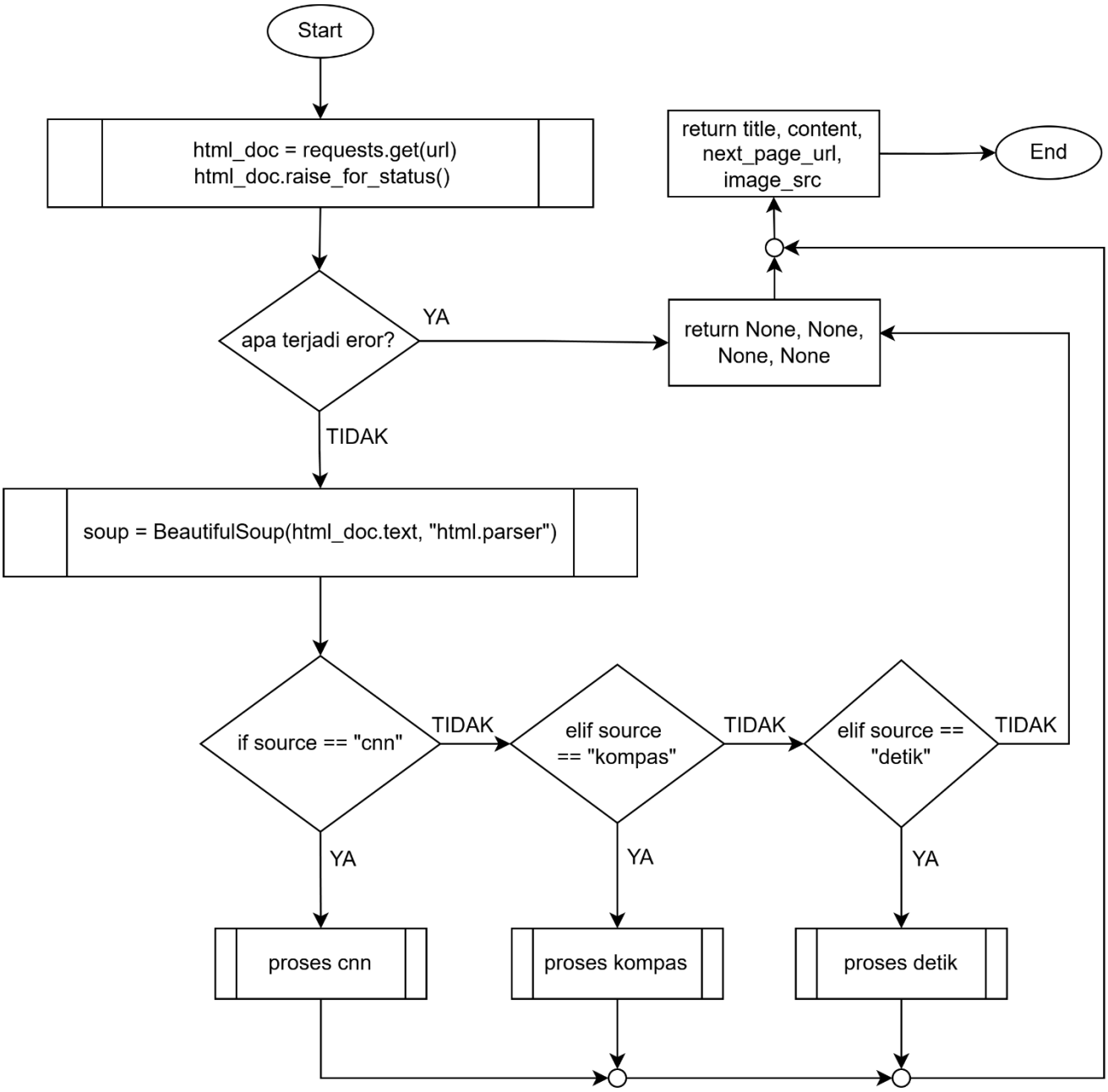
Flowchart Fungsi kompas_wisata()



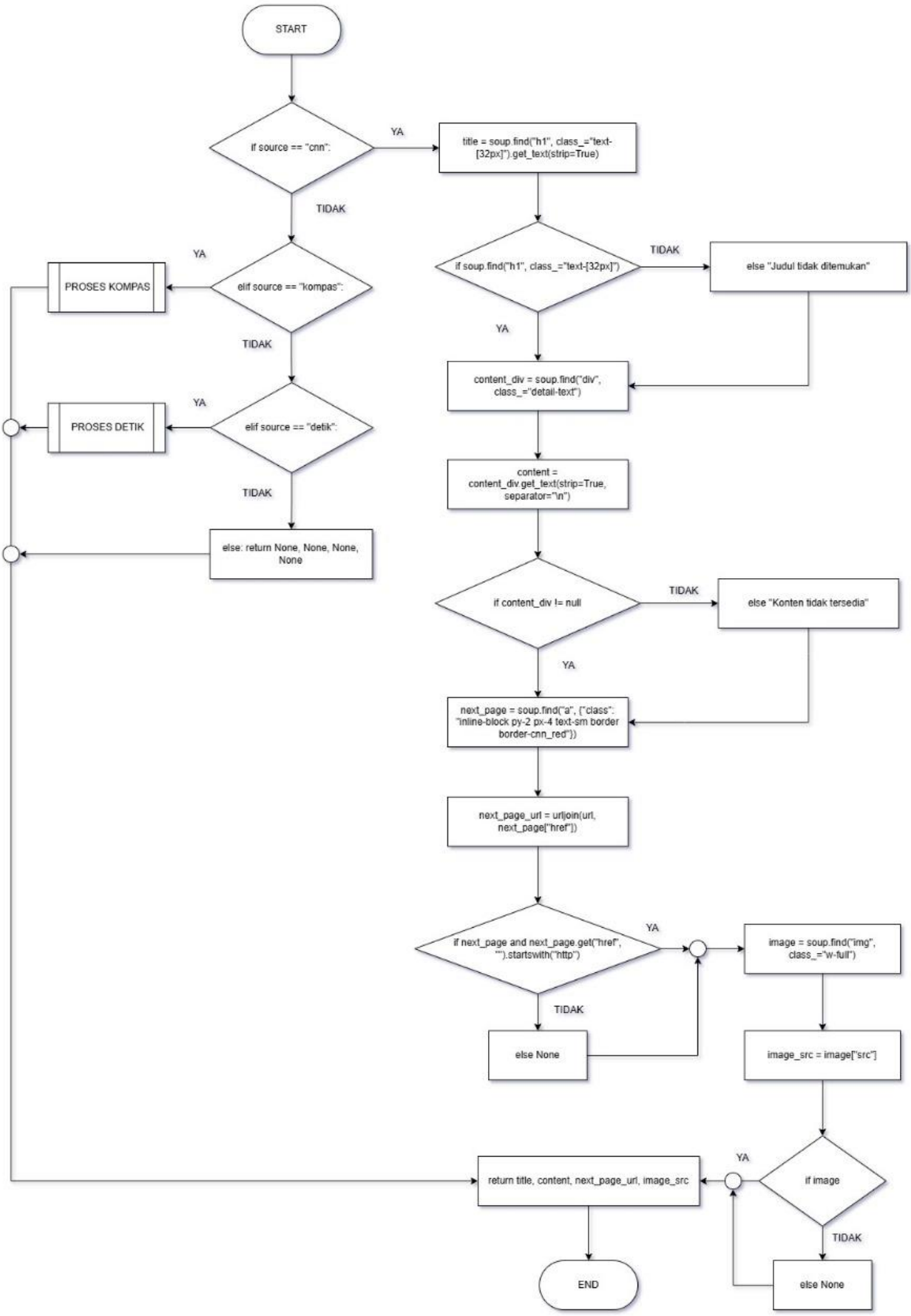
Flowchart Fungsi detik_wisata()



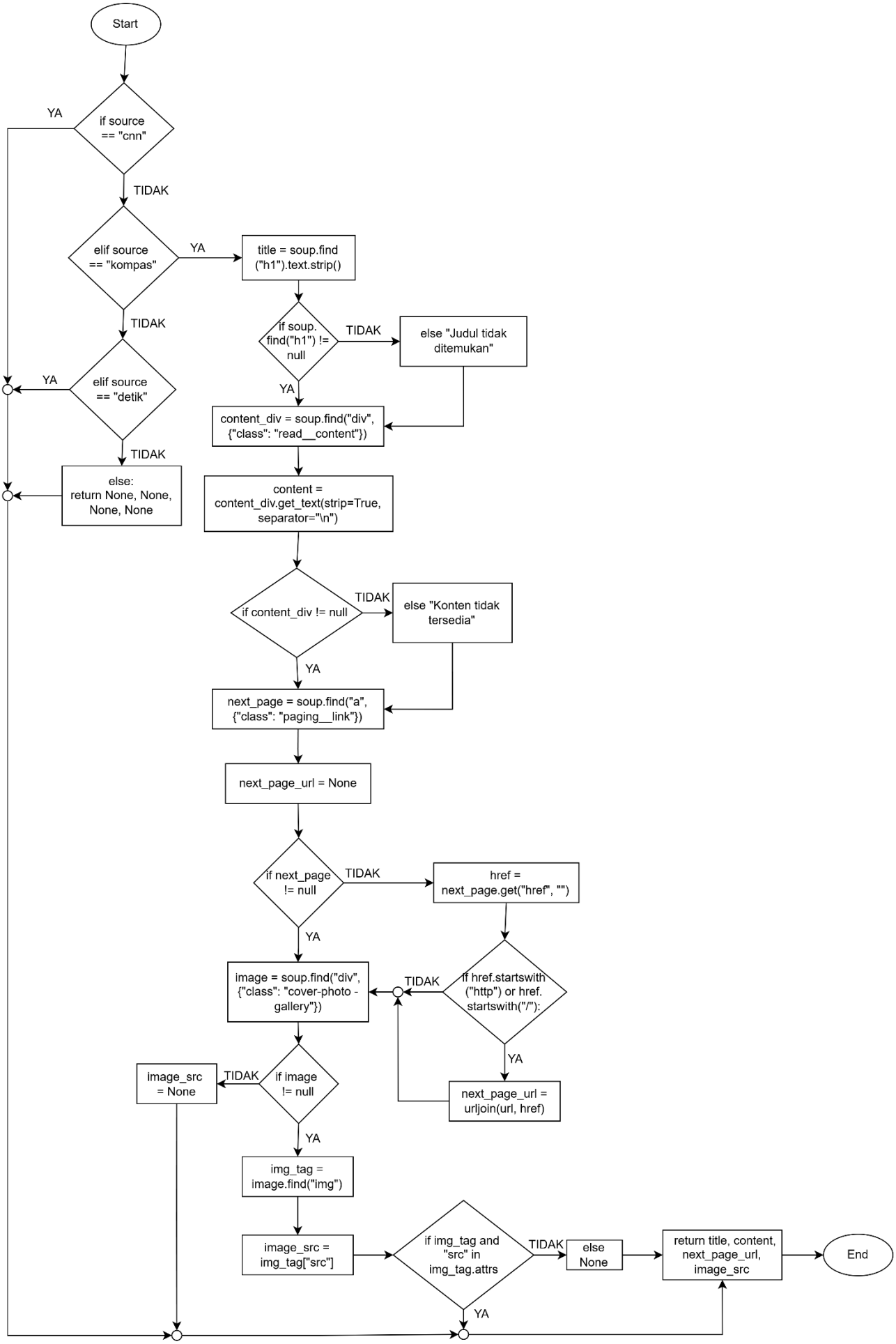
Flowchart Fungsi article_detail()



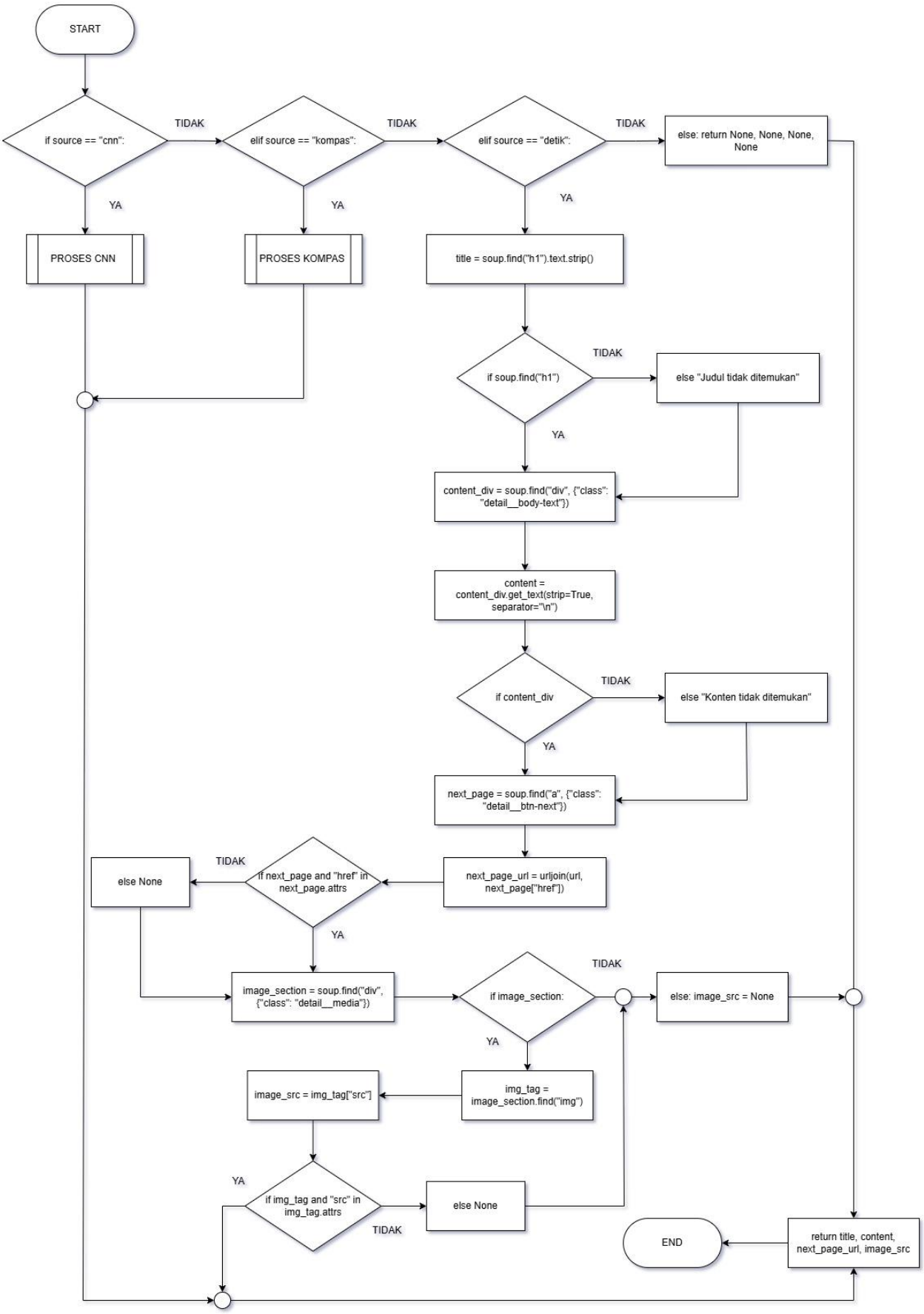
Flowchart Fungsi fetch_content(url)



Flowchart proses source==cnn



Flowchart proses source==kompas



Flowchart proses source==detik

c. Penjelasan Kode

1. App.py

- Import library Flask, render_template, requests, Beautifulsoup, urljoin dan inisialisasikan Flask melalui kode “app = Flask(__name__)”.
- Fungsi home() berisi routing untuk home dimana render_template akan dipanggil dan merender indeks.html.
- Fungsi cnn_wisata() digunakan untuk mengambil data dari website CNN dengan tag wisata. Fungsi ini akan dipanggil saat pengguna mengakses /cnn-wisata.
  - Requests get akan dikirim ke URL halaman wisata di CNN Indonesia dan akan di parsing menggunakan Beautifulsoup. Elemean class dengan atribut 'flex flex-col gap-5' akan dicari dan disimpan ke dalam variabel populer_area.
  - Terdapat list articles=[] kosong yang nantinya akan digunakan untuk menyimpan data artikel yang ditemukan. Akan dilakukan pengecekan apakah populer_area ditemukan. Tag article dengan class “list-content__item” dalam populer_area akan dicari dan disimpan ke dalam variabel items.
  - Dilakukan iterasi untuk setiap tag yang ditemukan. Tag img yang berisi gambar, tag a yang bersisi detail artikel, h3 yang berisi judul, tag span dengan class “text-cnn_red” yang bersisi kategori artikel, dan tag span dengan class “text-cnn_black_light3” yang berisi waktu publikasi akan dicari. Jika semua elemen tersebut ada / tidak kosong akan ditambahkan ke dalam list articles.
  - File cnn.html akan dirender dan variabel articles dikirim sebagai data ke template untuk ditampilkan.

- Fungsi kompas_wisata() digunakan untuk mengambil data dari website Kompas dengan tag travel ideas. Fungsi ini akan dipanggil saat pengguna mengakses /kompas-wisata.
  - Requests get akan dikirim ke URL halaman travel di Kompas dan akan di parsing menggunakan Beautifulsoup. Elemean class dengan atribut “latest--news mt2 clearfix” akan dicari dan disimpan ke dalam variabel populr_area.
  - Terdapat list articles=[] kosong yang nantinya akan digunakan untuk menyimpan data artikel yang ditemukan. Akan dilakukan pengecekan apakah populer_area ditemukan. Tag div dengan class “article__list clearfix” dalam populer_area akan dicari dan disimpan ke dalam variabel items.
  - Dilakukan iterasi untuk setiap tag yang ditemukan. Tag img yang berisi gambar, tag a yang bersisi detail artikel, h3 yang berisi judul, tag div dengan class "article__subtitle article__subtitle--inline" yang bersisi kategori artikel, dan tag div dengan class "article__date" yang berisi waktu publikasi akan dicari. Jika semua elemen tersebut ada / tidak kosong akan ditambahkan ke dalam list articles.
  - File kompas.html akan direnderdan variabel articles dikirim sebagai data ke template untuk ditampilkan.
- Fungsi detik_wisata() digunakan untuk mengambil data dari website Detik dengan tag travel news. Fungsi ini akan dipanggil saat pengguna mengakses /kompas-wisata.
  - Requests get akan dikirim ke URL halaman travel di Kompas dan akan di parsing menggunakan Beautifulsoup. Elemean class dengan atribut “grid-row list-content” akan dicari dan disimpan ke dalam variabel populer_area.
  - Terdapat list articles=[] kosong yang nantinya akan digunakan untuk menyimpan data artikel yang ditemukan. Akan dilakukan pengecekan

apakah populer_area ditemukan. Tag article dengan class "list-content__item" dalam populer_area akan dicari dan disimpan ke dalam variabel items.

- Dilakukan iterasi untuk setiap tag yang ditemukan. Tag img yang berisi gambar, tag a yang bersisi detail artikel, h3 yang berisi judul, dan tag div dengan class "media__date" yang berisi waktu publikasi akan dicari. Jika semua elemen tersebut ada / tidak kosong akan ditambahkan ke dalam list articles.
- File detik.html akan direnderdan variabel articles dikirim sebagai data ke template untuk ditampilkan.

- Fungsi article_detail digunakan untuk melihat detail artikel. Dalam fungsi ini juga terdapat fungsi fetch_content yang digunakan untuk mengambil detail data halaman artikel dari URL yang diberikan.
  - String kosong full_content digunakan untuk menggabungkan semua konten artikel dari halaman utama dan halaman berikutnya. Next_url diinisialisasi dengan url, yaitu URL halaman artikel pertama. Title diinisialisasi sebagai None dan akan diisi dengan judul artikel pertama yang ditemukan. Image_src diinisialisasi sebagai None dan akan diisi dengan URL gambar dari artikel pertama yang ditemukan.
  - Perulangan next_url dimulai dan tidak akan berhenti selama next_url tidak kosong. Fungsi fetch_content akan dipanggil dan mengembalikan current_title, content, next_url, current_image_src. Akan dilakukan pengecekan apakah current_title ditemukan. Jika ditemukan title yang belum memiliki nilai akan diisi dengan current_title dari halaman saat ini. Img_src yang belum memiliki nilai diisi dengan current_image_src dari halaman saat ini. Apabila current_title tidak ditemukan maka perulangan akan berhenti (break).

- Dictionary detail digunakan untuk menyimpan data title, content, dan img_src. Keseluruhan fungsi article_detail akan di render ke detail.html.

- Pada fungsi fetch_content blok try berisi requests.get ke URL yang diberikan, status kesalahan apabila permintaan gagal juga akan ditampilkan. Jika terjadi kesalahan fungsi akan menampilkan nilai (None, None, None, None). Konten HTML akan diparsing menggunakan BeautifulSoup.
  - Jika URL yang diberikan berasal dari CNN elemen H1 dengan atribut class="text-[32px]" akan disimpan sebagai judul artikel. Jika elemen ditemukan akan mengambil teks dalam elemen H1 dan menghapus spasi tambahan di awal/akhir. Apabila elemen tidak ditemukan title akan diatur menjadi “Judul Tidak Ditemukan”.
  - Elemen div dengan atribut class="detail-text" akan dijadikan sebagai konten utama artikel. Jika elemen ditemukan spasi tambahan yang ada di awal/akhir akan dihapus dan ditambahkan baris pemisah dija terdapat elemen yang berisi teks terpisah di dalam div. Apabila konten tidak ditemukan content akan diatur menjadi “Konten tidak tersedia”.
  - Elemen a dengan class="inline-block py-2 px-4 text-sm border border-cnn_red" akan digunakan sebagai link ke halaman berikutnya. Atribut href dari elemen a akan diambil dan dipastikan bahwa nilai href adalah URL lengkap. URL dasar dengan URL relative dari atribut href akan digabungkan melalui kode urljoin(url, next_page["href"]).
  - URL gambar diambil dari elemen img dengan class ="w-full". Jika elemen ditemukan akan mengambil nilai dari atribut src, apabila tidak ditemukan img_crs akan diatur ke None.
  - Apabila URL yang diberikan berasal dari Kompas blok elif pertama akan dijalankan. Apabila URL yang diberikan berasal dari Detik blok elif kedua akan dijalankan. Pada dasarnya fungsi kode untuk pengambilan detail

artikel sama dengan CNN hanya saja pada kode Kompas mendukung URL relatif(atribut dimulai dengan /) dan absolut(atribut dimulai dengan http) yang terdapat pada kode “href = next_page.get("href", "") if href.startswith("http") or href.startswith("/")”. Selain dari ketiga kondisi tersebut akan menampilkan nilai (None, None, None, None).

- Kode “return title, content, next_page_url, image_src” akan mengembalikan judul, konten, URL halaman berikutnya, dan URL gambar.

2. Index.html

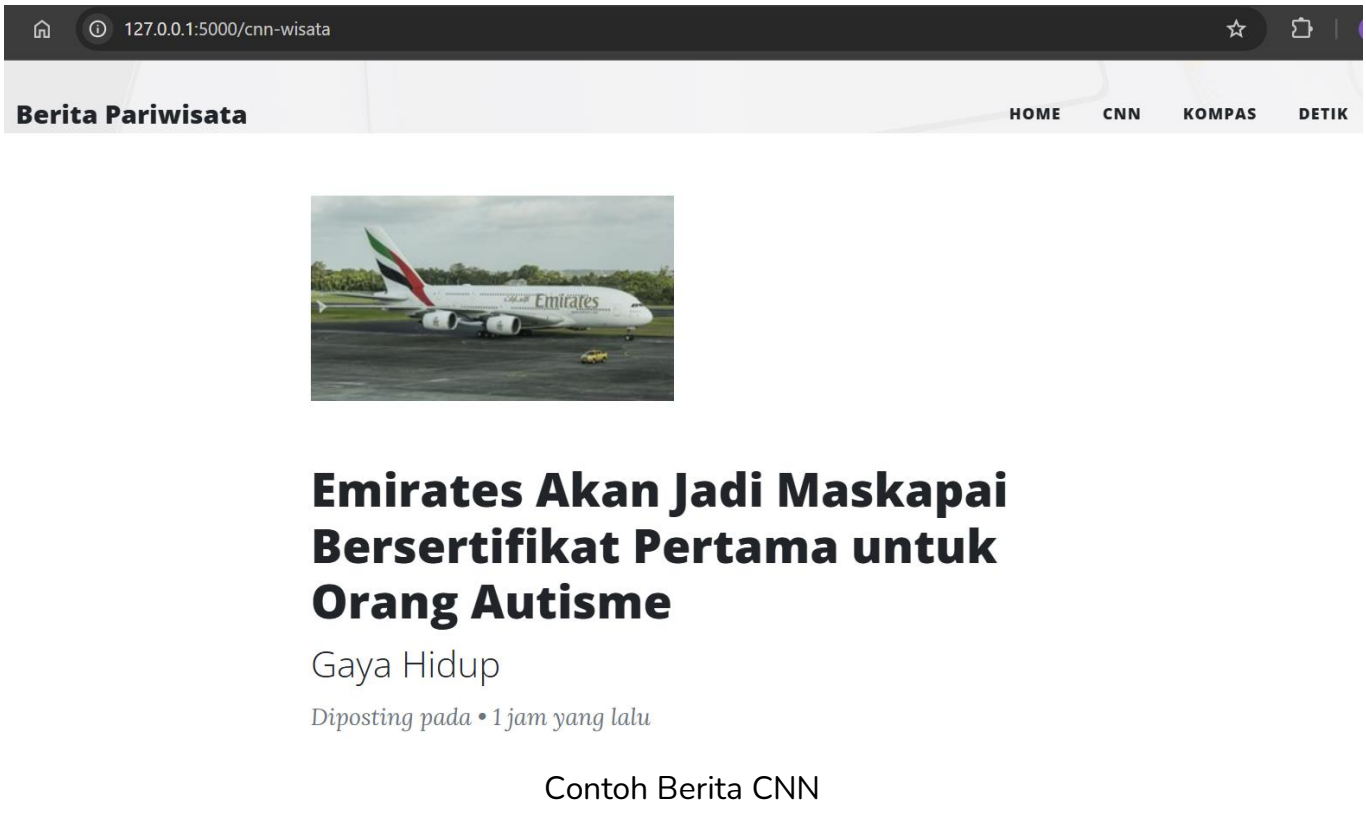
- Di halaman utama terdapat navigasi yang akan mengarahkan ke masing-masing website yaitu CNN, Kompas, dan Detik.

3. Cnn.html, Kompas.html, Detik.html

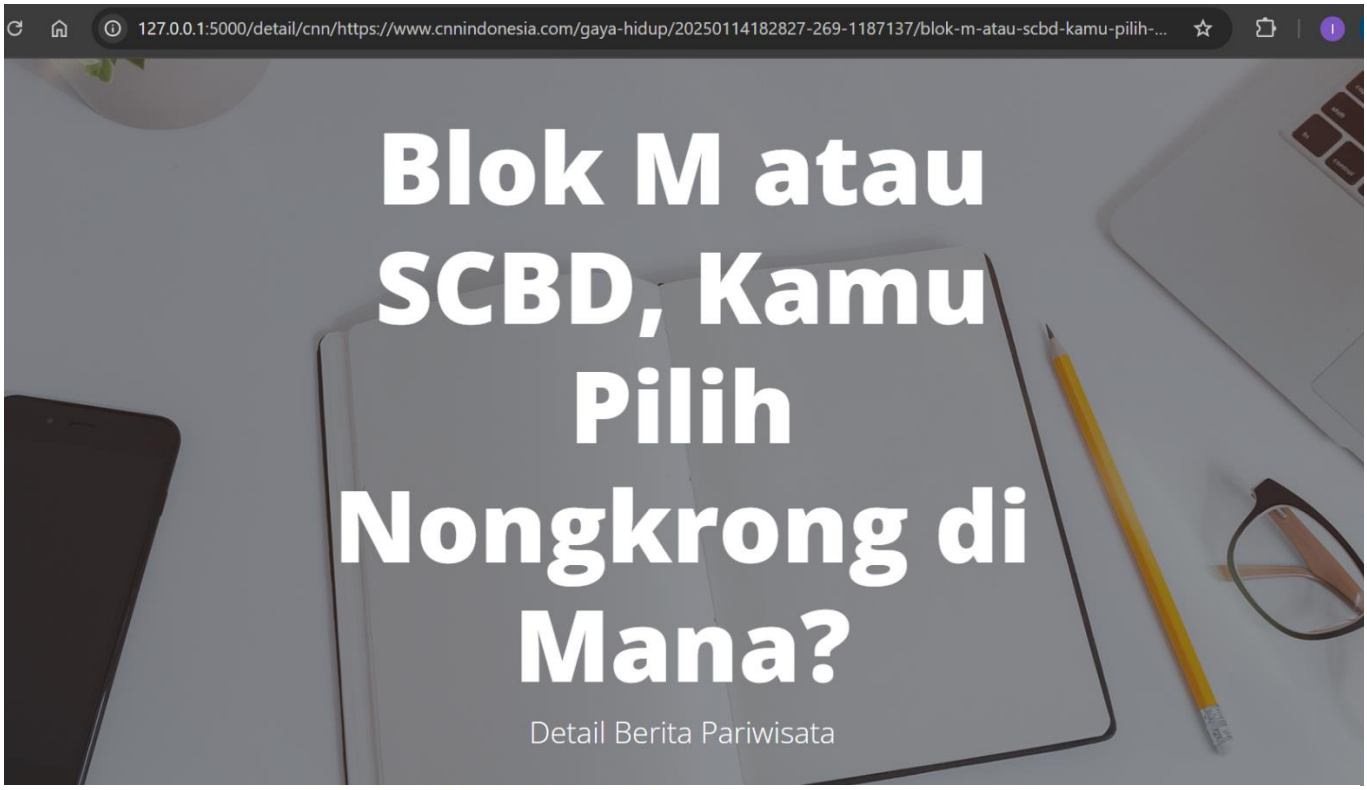
- Akan dilakukan pengecekan variabel articles apakah ada dan tidak kosong. Kemudian dilakukan iterasi pada setiap item yang ada di list articles.
  - Tautan untuk setiap artikel diambil dari article.link dan apabila tautan diklik akan membuka tautan tersebut di tab baru.
  - URL gambar diambil dari article.img_src, apabila gambar tidak dapat ditampilkan akan muncul deskripsi alternatif gambar yang diambil dari article.img_alt.
  - Judul artikel diambil dari artikel.title.
  - Kategori artikel diambil dari article.category. Tidak semua website ada kategori untuk setiap artiklenya. Dalam scraping ini website yang memiliki kategori hanya CNN Indonesia dan Kompas.
  - Waktu publikasi arikel diambil dari article.time.
- Apabila articles kosong blok else akan dijalankan, dimana blok tersebut berisi paragraf “Tidak ada artikel ditemukan.”

d. Hasil

Halaman Utama



Contoh Berita CNN



Jakarta, CNN Indonesia -- Warga Jakarta dan sekitarnya pasti sudah tak asing dengan dua kawasan ternama di kota metropolitan ini, yakni Blok M dan Sudirman Cental Business District (SCBD). Kedua kawasan ini merupakan pusat bisnis sekaligus ramah dikunjungi anak-anak muda untuk sekadar makan atau nongkrong bareng teman. Dua kawasan terkenal ini juga sama-sama berlokasi di Jakarta Selatan. Setiap hari

Contoh Detail Artikel