

LAPORAN
APLIKASI IMAGE COMPRESSION
BERBASIS WEBSITE

Disusun untuk memenuhi persyaratan
Mata Kuliah Pengelolaan Citra
Program Studi Sarjana Teknologi Informasi



Disusun Oleh:

Aditya Putra	212310025
Adrian Adhari	212310035
Nagasa Anandes Putra Ramadhan	212310048

A. Pendahuluan

Perusahaan menghadapi kebutuhan untuk mengurangi ukuran file citra tanpa kehilangan banyak kualitas visual, guna menghemat ruang penyimpanan dan bandwidth saat transmisi data. Solusi yang diusulkan adalah membuat aplikasi kompresi citra menggunakan metode JPEG. Aplikasi ini akan menerima file citra sebagai input, kemudian mengompresinya dengan pengaturan kualitas yang dapat disesuaikan. Hasil akhir dari aplikasi ini adalah citra terkompresi yang secara visual mendekati citra asli, namun dengan ukuran file yang lebih kecil.

B. Tujuan

Adapun tujuan dari pembuatan aplikasi ini yaitu :

- 1) Mengimplementasikan kompresi citra menggunakan metode JPEG.
- 2) Menampilkan citra asli dan citra terkompresi untuk perbandingan.
- 3) Menghitung dan menampilkan perbedaan ukuran file antara citra asli dan terkompresi.

C. Teknologi yang digunakan

- 1) Python version 24.2 : Python dipilih karena mudah digunakan, memiliki sintaksis yang intuitif, dan didukung pustaka yang luas, cocok untuk aplikasi pengolahan citra. Versi 24.2 menyediakan fitur terkini yang dapat meningkatkan performa.
- 2) React.js : React.js dipilih sebagai framework untuk UI karena memungkinkan pembuatan komponen yang dinamis dan interaktif. React.js juga menawarkan kemudahan dalam manajemen state, membuat pengalaman pengguna lebih responsif.
- 3) Tailwind : Tailwind digunakan untuk mempercepat pengembangan dan styling UI. Dengan sistem utility-first, Tailwind memungkinkan styling yang cepat dan konsisten di seluruh aplikasi tanpa perlu menulis CSS dari awal.
- 4) Flask : Flask digunakan untuk membuat backend aplikasi ini, menangani permintaan HTTP, dan menyediakan API endpoint untuk menerima gambar dari frontend serta mengirim hasil kompresi citra kembali ke frontend.

- 5) Flask CORS (Cross-Origin Resource Sharing) : Library Flask CORS dibutuhkan ketika frontend (misalnya aplikasi React) dan backend (Flask) berada pada domain atau port yang berbeda.
- 6) OpenCV (cv2) : Library *OpenCV* digunakan untuk membuka file gambar dan melakukan kompresi dengan pengaturan kualitas JPEG. Kompresi JPEG diatur menggunakan parameter `cv2.IMWRITE_JPEG_QUALITY`, yang dapat menurunkan ukuran file citra tanpa terlalu banyak mengurangi kualitas visual.
- 7) `os` : Module `os` digunakan untuk menyimpan dan mengakses ukuran file gambar asli serta file terkompresi dalam satuan kilobyte (KB). Fungsi ini memungkinkan penghitungan perbedaan ukuran sebelum dan sesudah kompresi, yang disampaikan kembali kepada pengguna melalui API.

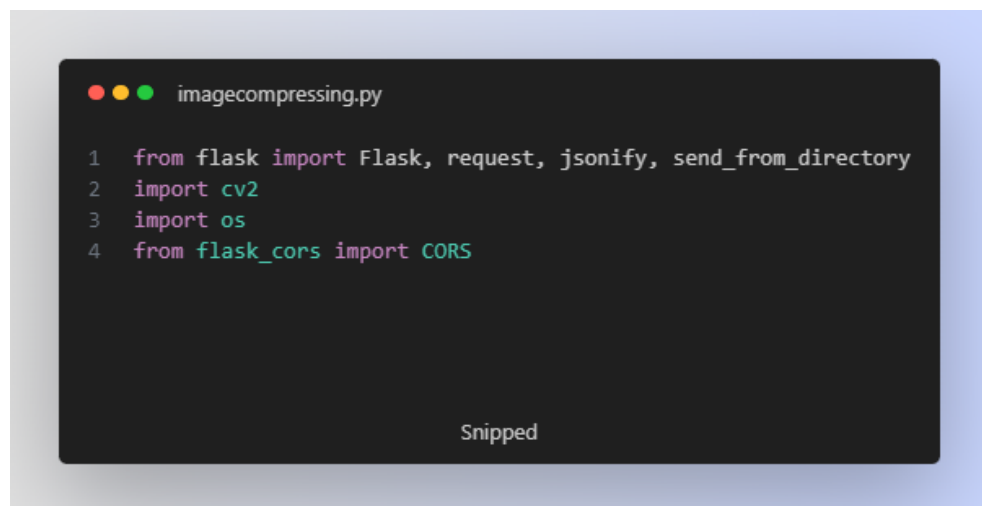
D. Langkah Pembuatan Aplikasi

1) Instalasi

Instal pustaka yang diperlukan dengan perintah berikut:

`pip install pillow matplotlib`

2) Import Pustaka

A screenshot of a code editor window titled 'imagecompressing.py'. The editor has a dark background with light-colored text. It shows four lines of Python code: line 1: 'from flask import Flask, request, jsonify, send_from_directory', line 2: 'import cv2', line 3: 'import os', and line 4: 'from flask_cors import CORS'. The code is syntax-highlighted. At the bottom right of the editor, the word 'Snipped' is visible.

```
1 from flask import Flask, request, jsonify, send_from_directory
2 import cv2
3 import os
4 from flask_cors import CORS
```

Snipped

3) Konfigurasi Flask dan CORS

```
imagecompressing.py

6  app = Flask(__name__)
7  CORS(app)
8  UPLOAD_FOLDER = 'static/'
9  app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
10 app.config['MAX_CONTENT_LENGTH'] = 16 * 1024 * 1024

Snipped
```

Penjelasan :

- `UPLOAD_FOLDER`: Mengatur folder untuk menyimpan gambar yang diunggah.
- `MAX_CONTENT_LENGTH`: Membatasi ukuran maksimum file yang dapat diunggah ke server (16 MB).

4) Buat Endpoint untuk Upload dan Kompresi Gambar

```
imagecompressing.py

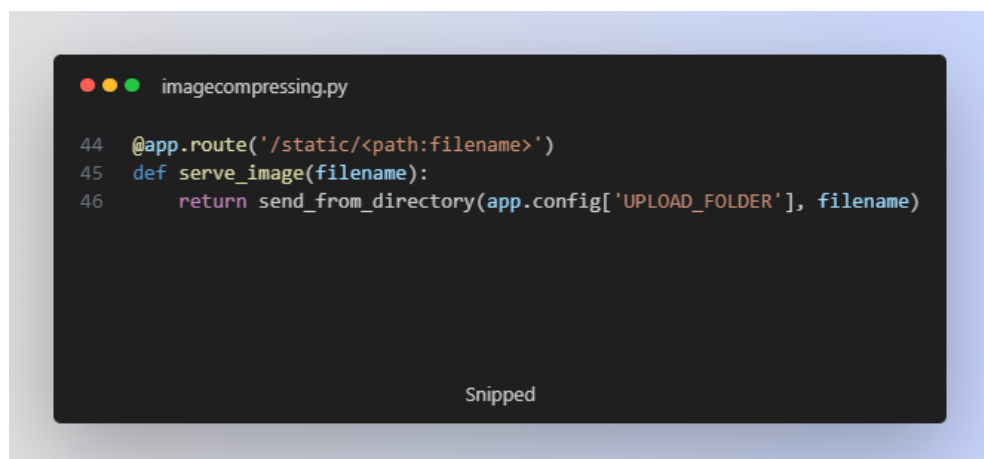
13 @app.route('/upload', methods=['POST'])
14 def upload_image():
15     if 'image' not in request.files:
16         return jsonify({"error": "No image provided"}), 400
17
18     file = request.files['image']
19     if file.filename == '':
20         return jsonify({"error": "Empty filename"}), 400
21
22     # Simpan gambar yang di-upload
23     input_path = os.path.join(app.config['UPLOAD_FOLDER'], file.filename)
24     file.save(input_path)
25
26     # Proses kompresi menggunakan OpenCV
27     output_path = os.path.join(app.config['UPLOAD_FOLDER'], 'compressed_' + file.filename)
28     image = cv2.imread(input_path)
29     cv2.imwrite(output_path, image, [int(cv2.IMWRITE_JPEG_QUALITY), 50]) # Kompres dengan kualitas 50
30
31     # Hitung ukuran file asli dan terkompresi
32     original_size = os.path.getsize(input_path) / 1024 # dalam KB
33     compressed_size = os.path.getsize(output_path) / 1024 # dalam KB
34
35     # Return informasi ukuran dan path gambar
36     return jsonify({
37         "original_size": f"{original_size:.2f} KB",
38         "compressed_size": f"{compressed_size:.2f} KB",
39         "original_image": f"/static/{file.filename}",
40         "compressed_image": f"/static/compressed_{file.filename}"
41     }), 200

Snipped
```

Penjelasan :

- `@app.route('/upload', methods=['POST'])`: Membuat route POST yang menangani upload file.
- `request.files`: Digunakan untuk mengambil file yang diunggah dari permintaan HTTP.
- `if 'image' not in request.files`: Memeriksa apakah file telah diunggah atau belum.
- `file.filename == ''`: Memeriksa apakah nama file kosong.
- `input_path`: Menentukan path tempat menyimpan file yang diunggah.
- `file.save(input_path)`: Menyimpan file yang diunggah ke path `input_path`.
- `cv2.imread(input_path)`: Membaca file gambar dari path.
- `cv2.imwrite(output_path, image, [int(cv2.IMWRITE_JPEG_QUALITY), 50])`: Menyimpan gambar terkompresi di path `output_path` dengan kualitas 50.
- `os.path.getsize(input_path)`: Mengambil ukuran file asli dan terkompresi dalam byte, lalu dikonversi ke KB.
- `jsonify`: Mengembalikan respons dalam format JSON, termasuk ukuran file dan path gambar.

5) Buat Endpoint untuk Mengakses File Gambar

A screenshot of a code editor window titled 'imagecompressing.py'. The code is as follows:

```
44 @app.route('/static/<path:filename>')
45 def serve_image(filename):
46     return send_from_directory(app.config['UPLOAD_FOLDER'], filename)
```

At the bottom of the code block, the word 'Snipped' is visible.

Penjelasan

- `send_from_directory`: Mengirim file dari direktori `static/`, sehingga frontend dapat mengakses gambar yang diunggah dan dikompres.

6) Jalankan Aplikasi



Penjelasan :

- `app.run(debug=True)`: Menjalankan server di mode debug, sehingga perubahan dapat langsung terlihat selama pengembangan.