LAPORAN TUGAS BESAR II
IF2123 ALJABAR LINEAR DAN GEOMETRI
APLIKASI ALJABAR VEKTOR DALAM SISTEM TEMU BALIK GAMBAR

Disusun untuk memenuhi tugas mata kuliah Aljabar Linear dan Geometri pada Semester 1 (satu) Tahun Akademik 2023/2024.

"SangkuLens"

Disusun oleh:

Melati Anggraini	13522035
Raffael Boymian Siahaan	13522046
Shulha	13522087

PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
BANDUNG

2023

DAFTAR ISI

BAB I
DESKRIPSI MASALAH

I.1 Latar Belakang

Dalam era digital, jumlah gambar yang dihasilkan dan disimpan terus meningkat pesat, termasuk foto pribadi, gambar medis, ilustrasi ilmiah, dan gambar komersial. Untuk mengatasi tantangan ini, sistem temu balik gambar menjadi sangat penting. Sistem ini memungkinkan pengguna untuk mencari, mengakses, dan mengelola koleksi gambar mereka dengan mudah. Contoh penerapan yang terkenal adalah Google Lens, yang memungkinkan pengguna melakukan pencarian gambar pribadi, analisis gambar medis, pencarian ilustrasi ilmiah, dan pencarian produk berdasarkan gambar komersial. Pada tugas ini, ingin dibuat sistem temu balik dan komparasi gambar, terutatama dengan memanfaatkan CBIR yang akan dijelaskan kemudian.

I.2 Tujuan

Mengimplementasikan sistem temu balik gambar dengan memanfaatkan Aljabar Vektor dalam bentuk sebuah *website*, dimana hal ini merupakan pendekatan yang penting dalam dunia pemrosesan data dan pencarian informasi. Dalam konteks ini, aljabar vektor digunakan untuk menggambarkan dan menganalisis data menggunakan pendekatan klasifikasi berbasis konten (*Content-Based Image Retrieval* atau CBIR), di mana sistem temu balik gambar bekerja dengan mengidentifikasi gambar berdasarkan konten visualnya, seperti warna dan tekstur.

I.3 Spesifikasi

Website dapat menerima masukan berupa:

1. *Image query*, berisi gambar yang akan digunakan untuk melakukan pencarian gambar.
2. Kumpulan gambar (dataset), dilakukan dengan cara mengunggah *multiple image* dalam bentuk *folder* ke dalam *web browser*. Setelah memasukkan *image query*, kumpulan gambar inilah yang akan diseleksi menjadi *result*.

Website memiliki komponen-komponen berikut:

1. Judul *Website*
2. Tombol *Insert* Gambar, beserta *Display* Gambar yang ingin dicari

3. *Toggle Button* untuk memilih *searching* berdasarkan Warna atau Tekstur
4. Tombol *Search* untuk memulai pencarian
5. Kumpulan Gambar (*result*) yang didapat dari hasil pencarian
6. Informasi mengenai Jumlah *Result* yang didapat dan Waktu Eksekusi
7. Tingkat Kemiripan Setiap Gambar (*result*) dengan gambar yang ingin dicari, dalam persentase (%)
8. Tombol *Upload* Dataset untuk mengunggah kumpulan gambar (dataset) dalam suatu folder
9. Page-page tambahan: Konsep singkat search engine yang dibuat, *How to Use*, dan *About us*

Secara umum, berikut adalah cara umum penggunaan program:

1. Pengguna terlebih dahulu memasukkan dataset gambar dalam bentuk *folder* yang berisi kumpulan gambar. Dataset gambar ini diperlukan sebelum proses *searching* agar ada perbandingan untuk gambar yang ingin dicari.
2. Setelah dataset sudah dimasukkan, pengguna memasukkan sebuah gambar yang ingin di-*search* dari dataset.
3. Pilih opsi pencarian, ingin melakukan pencarian berdasarkan warna atau tekstur.
4. Tekan tombol *search*, program kemudian akan memproses, mencari gambar-gambar dari dataset yang memiliki kemiripan dengan gambar yang dimasukkan tadi.
5. Program akan menampilkan kumpulan gambar yang mirip, diurutkan dari yang memiliki kemiripan paling tinggi ke yang paling rendah. Setiap gambar yang muncul diberi persentase kemiripannya.
6. Terdapat informasi terkait jumlah gambar yang muncul, dan waktu eksekusi programnya.

BAB II
LANDASAN

II.1 Content-Based Image Retrieval (CBIR)

Suatu CBIR pada umumnya membutuhkan suatu image descriptor. Image descriptor terdiri atas algoritma untuk men-ekstraksi dari suatu gambar menjadi suatu vektor yang berisi fitur dan mengukur tingkat kesamaan dua gambar. Kemudian, CBIR menggunakan algoritma pencocokan untuk membandingkan vektor-fitur dari gambar yang dicari dengan vektor-fitur gambar dalam dataset. Tingkat kesamaan dari sepasang gambar direpresentasikan dengan vektor deskripsi masing-masing gambar. Hasil dari pencocokan ini digunakan untuk mengurutkan gambar-gambar dalam dataset dan menampilkan gambar yang paling mirip dengan gambar yang dicari.

II.2 CBIR Berdasarkan Warna

CBIR berdasarkan perbandingan warna adalah cara yang dasar dan paling penting dalam CBIR. Warna adalah suatu unsur yang penting dan paling mudah terlihat dari suatu gambar. Jika dibandingkan dengan tekstur dan bentuk, warna adalah suatu unsur dari gambar yang stabil, tidak dipengaruhi oleh rotasi, translasi, perbesaran dan transformasi lainnya. Kalkulasi warna juga bukanlah sesuatu hal yang sulit dilakukan. Histogram warna biasa digunakan dalam men-ekstraksi fitur atau ciri warna dari suatu gambar. Histogram warna biasa digunakan karena dapat merepresentasikan distribusi warna dari gambar secara universal, hal ini menguntungkan jika gambar sulit untuk dipisahkan menjadi beberapa segmen dan tidak memperdulikan posisi dari warna tersebut. Namun kelemahannya dari histogram warna adalah tidak dapat mendeskripsikan objek yang spesifik dari gambar.

II.2.1 Histogram Warna

Histogram warna adalah suatu representasi dari distribusi warna yang ada pada sebuag gambar. Pada gambar digital, histogram warna akan merepresentasikan jumlah pixel yang dibedakan berdasarkan warnanya. Histogram warna dapat dibuat dari berbagai jenis representasi warna, seperti RGB atau HSV. Pada gambar yang tidak berwarna atau monokromatik, biasanya intensity histrogram lebih umum digunakan. Suatu histogram warna dari suatu gambar dapat dibuat dengan mendiskiritisasi warna dalam gambar ke dalam sejumlah bin dan jumlah piksel gambar dihitung dalam setiap bin. Misalnya histogram merah-biru dapat dibentuk dari nilai-nilai warna normalisasi piksel yang

membagi nilai-nilai RGB. Dalam membuat histogram warna, kita dapat menghitung jumlah piksel dalam masing-masing 256 skala dalam tiga channel RGB atau ruang warna lainnya seperti HSV dan memisahkannya menjadi tiga buah histogram, yaitu untuk warna merah, hijau dan biru.

Histogram warna adalah statistik frekuensi untuk warna-warna berbeda dalam suatu ruang warna tertentu. Keuntungannya adalah bahwa ini menggambarkan distribusi warna global untuk gambar. Ini sangat cocok untuk gambar yang sulit di-segmentasi dan mengabaikan lokasi spasial. Namun, kelemahannya adalah bahwa ini tidak dapat menggambarkan distribusi lokal gambar dalam ruang warna dan posisi spasial setiap warna. Ini berarti bahwa histogram warna tidak dapat menggambarkan objek atau hal-hal tertentu dalam gambar.

II.2.2 Konversi RGB ke HSV

Untuk gambar yang dipilih dari basis data gambar, karena ruang warna RGB tidak memenuhi persyaratan visual orang, untuk pengambilan gambar, gambar biasanya dikonversi dari ruang warna RGB ke ruang warna lain. Ruang warna HSV digunakan dalam makalah ini karena merupakan ruang warna yang lebih umum. Histogram warna global dapat dihitung dengan terlebih dahulu mengkonversi nilai RGB ke HSV dengan langkah sebagai berikut sebagai berikut:

a. Nilai dari RGB harus dinormalisasi dengan mengubah nilai *range* [0, 255] menjadi [0, 1]

$$R' = \frac{R}{255} \qquad G' = \frac{G}{255} \qquad B' = \frac{B}{255}$$

b. Mencari *Cmax, Cmin*, dan Δ

$$C_{max} = \max(R', G', B')$$
$$C_{min} = \min(R', G', B')$$
$$\Delta = C_{max} - C_{min}$$

c. Kemudian mendapatkan nilai HSV dengan

$$H = \begin{cases} 0° & \Delta = 0 \\ 60° \times \left(\frac{G' - B'}{\Delta} mod 6\right) & , C' \max = R' \\ 60° \times \left(\frac{B' - R'}{\Delta} + 2\right) & , C' \max = G' \\ 60° \times \left(\frac{R' - G'}{\Delta} + 4\right) & , C' \max = B' \end{cases}$$

$$S = \begin{cases} 0 & , C_{max} = 0 \\ \frac{\Delta}{C_{max}} & , C_{max} \neq 0 \end{cases}$$

$$V = Cmaks$$

II.2.3 Histogram Blok

Untuk histogram warna blok, sebuah gambar dibagi menjadi blok-blok nxn. Setiap blok akan memiliki arti yang lebih sedikit jika blok terlalu besar, sementara perhitungan proses pengambilan akan meningkat jika blok terlalu kecil. Melalui analisis perbandingan penulis, ruang dua dimensi yang dibagi menjadi 3x3 akan lebih efektif.Untuk setiap blok, penulis melakukan perhitungan konversi ruang warna dan kuantisasi warna. Fitur warna yang dinormalisasi untuk setiap blok dapat dihitung. Untuk tugas ini, menggunakan 4x4 blok. Histogram warna tersebutlah yang mengandung nilai-nilai vektor HSV.

II.3 CBIR dengan Parameter Tekstur

CBIR dengan perbandingan tekstur menggunakan suatu matriks yang dinamakan co-occurrence matrix. Matriks ini digunakan karena membutuhkan tingkat pemrosesan yang mudah dan cepat.

II.3.1 Konversi nilai RGB ke Grayscale

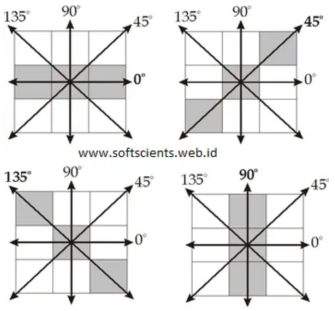
Konversi warna gambar menjadi greyscale, ini dilakukan karena warna tidaklah penting dalam penentuan tekstur. Sehingga dari warna RGB dapat diubah menjadi suatu warna greyscale Y dengan rumus:

$$Y = 0.29 \times R + 0.587 \times G + 0.114 \times B$$

Kemudian, lakukan kuantifikasi dari nilai *grayscale*. Karena citra *grayscale* berukuran 256 piksel, maka matriks yang berkoresponden akan berukuran 256 × 256. Matriks tersebut mungkin tidak seluruhnya terisi menyebabkan nilai nilai kosong dan menggunakan memori yang besar, sehingga *Grayscale* semula dari suatu gambar akan dikompresi untuk mengurangi operasi perhitungan sebelum dibentuknya *co-occurrence matrix*. Misalnya dengan mengkuantifikasinya menjadi 16 × 16.

II.3.2 *Gray-Level CoOccurence Matrix*

Gray-Level Co-occurrence matrix (GLCM) merupakan teknik analisis tekstur pada citra. GLCM merepresentasikan hubungan antara 2 pixel yang bertetanggaan (neighboring pixels) yang memiliki intensitas keabuan (grayscale intensity), jarak dan sudut. Terdapat 8 sudut yang dapat digunakan pada GLCM, diantaranya sudut 0°, 45°, 90°, 135°, 180°, 225°, 270°, atau 315°.



Langkah pembuatan matrix GLCM :

1. Pembuatan framework matrix
2. Pembuatan co-occurrence matrix (mengisi framework matrix)

Pembuatan co occurrence matrix dengan langkah berikut. Misalkan terdapat suatu gambar *I* dengan $n \times m$ piksel dan suatu parameter offset $(\Delta x, \Delta y)$, maka dengan menggunakan nilai *i* dan *j* sebagai nilai intensitas dari gambar dan *p* serta *q* sebagai posisi dari gambar, maka *offset* Δx dan Δy bergantung pada arah θ dan jarak yang digunakan melalui persamaan di bawah ini.

$$C_{\Delta x,\Delta y}(i,j) = \sum_{p=1}^{n} \sum_{q=1}^{m} \begin{cases} 1, & \text{jika } I(p,q) = i \text{ dan } I(p+\Delta x, q+\Delta y) = j \\ 0, & \text{jika lainnya} \end{cases}$$

3. Pembuatan symmetric matrix (penjumlahan co-occurrence matrix dengan transpose matrix)
4. Matrix normalization yang akan menghasilkan nilai matrix antara 0–1

$$\text{MatrixNorm} = \frac{\text{MatrixOcc}}{\sum \text{MatrixOcc}}$$

II.3.3 Ekstraksi Fitur Tekstur GLCM

Dari *co-occurrence matrix* bisa diperoleh berbagai komponen ekstraksi tekstur, di antaranya *contrast*, *entropy, homogeneity, dissimilarity, ASM, energy, correlation*. Berikut adalah rumus-rumus untuk fitur-fitur tersebut.

- 'contrast': $\sum_{i,j=0}^{levels-1} P_{i,j}(i-j)^2$
- 'dissimilarity': $\sum_{i,j=0}^{levels-1} P_{i,j}|i-j|$
- 'homogeneity': $\sum_{i,j=0}^{levels-1} \frac{P_{i,j}}{1+(i-j)^2}$
- 'ASM': $\sum_{i,j=0}^{levels-1} P_{i,j}^2$
- 'energy': $\sqrt{ASM}$
- **'correlation':**

$$\sum_{i,j=0}^{levels-1} P_{i,j} \left[\frac{(i-\mu_i)\,(j-\mu_j)}{\sqrt{(\sigma_i^2)(\sigma_j^2)}} \right]$$

Untuk i,j merupakan koordinat pixel pada matrix GLCM, levels merupakan rentang gray tone, dan Pi,j merupakan nilai pixel pada koordinat i,j GLCM matrix. Contoh penghitungan fitur:

$$\begin{aligned} contrast &= \sum_{i,j}^{levels-1} P_{i,j}(i-j)^2 \\ &= \sum_{i,j}^{3} P_{i,j}(i-j)^2 \\ &= (0.16\times[0-0]^2 + 0.16\times[0-1]^2 ... 0\times[0-2]^2) \end{aligned}$$

II.4 Vektor dan Pemanfaatan Penghitungan Vektor untuk Komparasi

Vektor adalah suatu objek geometri di dalam bidang matematika maupun fisika yang mempunyai besaran dan arah. Terdapat beberapa operasi dalam vektor, seperti penjumlahan kedua vektor, pengurangan kedua vektor dan lainnya. Untuk dari fitur atau ciri-ciri gambar yang diambil dari kumpulan gambar-gambar. Berikut adalah contoh penjumlahan dua vektor yaitu A dan B. Dalam tugas ini, digunakan Teorema *Cosine Similarity* untuk mengukur kemiripan dua gambar, yaitu dengan rumus di bawah ini di mana A dan B adalah vektor dari dua gambar.

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}}$$

II.5 Pengembangan Website Sederhana

Pengembangan web secara umum merujuk pada tugas-tugas yang terkait dengan pengembangan situs web untuk dihoskan melalui intranet atau internet. Proses pengembangan web melibatkan desain web, pengembangan konten web, scripting sisi klien/sisi server, dan konfigurasi keamanan jaringan, di antara tugas-tugas lainnya. Secara sederhana, pengembangan website sederhana seperti tugas ini mencakup sisi antarmuka pengguna *front-end* dan hal yang tidak dikontrol pengguna di *back-end*.

Frontend adalah bagian di dalam web development yang menangani user side dari suatu website seperti pembuatan GUI (Graphical User Interface) yang user friendly. Agar user dapat berinteraksi dengan website, diperlukan tampilan website yang tertata, rapih, dan yang terpenting mudah dimengerti oleh user. Data yang dikirimkan user melalui antarmuka akan dihandle oleh API.

Backend adalah bagian di dalam web development yang menangani server side dari suatu website seperti response and request handling, URL building, file handling, dan data processing.

Dalam pemrograman secara umum, istilah API, singkatan dari *Application Programming Interfac*e, merujuk pada bagian dari program komputer yang dirancang untuk digunakan atau dimanipulasi oleh program lain, berbeda dengan antarmuka yang dirancang untuk digunakan atau dimanipulasi oleh manusia. Program komputer seringkali perlu berkomunikasi di antara mereka atau dengan sistem operasi dasar, dan API adalah salah satu cara yang digunakan untuk melakukannya.

BAB III
ANALISIS PEMECAHAN MASALAH

III.1 Langkah-langkah Pemecahan Masalah

Untuk membuat tugas projek ini, langkah-langkah pemecahan masalah yang penulis lakukan adalah:

1. Membuat program untuk mengambil nilai vektor yang akan dikomparasi, baik berdasarkan warna maupun tekstur
2. Membuat program untuk secara umum membandingkan nilai fitur vektor
3. Merancang antarmuka *website*
4. Menghubungkan antarmuka *website* dengan program pemrosesan

III.2 Proses Pemetaan Masalah

Dengan langkah-langkah pemecahan masalah yang telah disebutkan, dapat digunakan penerapan aljabar vektor untuk melakukan Content-Based Image Retrieval.

Untuk mendapatkan nilai-nilai vektor dari CBIR berdasarkan parameter warna, dilakukan konversi nilai RGB ke HSV berdasarkan cara yang telah dijelaskan pada Bab II. Karena CBIR berdasarkan warna tidak menyimpan informasi spasial, digunakan pembagian gambar menjadi 4 x 4 blok, kemudian dibentuk histogram warna H, S, dan V dari masing masing-blok. Seluruh nilai histogram HSV berbentuk matriks tersebut dikonkatenasi menjadi satu array panjang yang selanjutnya kita sebut vektor dengan 16 elemen.

Untuk mendapatkan nilai-nilai vektor dari CBIR berdasarkan parameter tekstur, terlebih dahulu dilakukan konversi nilai RGB ke *grayscale* yang kemudian dikuantifikasi dari 256 menjadi 32 untuk membentuk *Gray-Level Co Occurrence Matrix* yang lebih kecil dan kompak. Kemudian dilakukan pembuatan GLCM berdasarkan cara yang telah dijelaskan pada Bab II dengan mengambil derajat 0 dan *distance* 1. Dari GLCM yang sudah terbentuk, diekstraksi nilai-nilai *contrast*, *entropy, homogeneity, dissimilarity, ASM, energy, correlation* berdasarkan cara yang telah dijelaskan pada Bab II. Nilai-nilai tersebut kemudian dijadikan satu array yang selanjutnya kita sebut vektor dengan 7 elemen.

Setelah mendapatkan nilai-nilai vektor baik 16 elemen (parameter warna) maupun 7 elemen (parameter tekstur) dapat diperoleh nilai cosinus sudut antara dua vektor menggunakan rumus yang telah dijelaskan pada Bab II memanfaatkan *dot product* dan panjang vektor. Nilai cosinus sudut berkisar antara 0 hingga 1 sehingga cocok dinilai

sebagai tingkat kemiripan berdasarkan persentase. Kemudian, komparasi dilakukan dengan seluruh gambar pada dataset.

III.3 Ilustrasi Kasus dan Penyelesaian

Misalnya, dengan menggunakan parameter tekstur, nilai kesamaan dua gambar dibawah ini adalah 92.73%. Sementara kemiripan dengan parameter warna bernilai 79.56%


BAB IV
IMPLEMENTASI DAN UJI COBA

IV.1 Implementasi Program Utama

Program dibuat dengan bahasa pemrograman python, menggunakan framework Flask untuk *backend* dan framework React untuk *frontend* dan aplikasi webnya. Library numpy untuk operasi matriks, dan library OpenCV untuk pemrosesan gambar. Program utama nya mungkin dapat dilihat pada file compare.py. Program mengambil vektor nilai pada *query image* lalu melakukan *loop through* datasets file dan mengambil nilai vektor fiturnya. Nilai hasil komparasi vektor fitur kemudian disimpan dalam suatu array jika nilainya lebih dari 60%. Array tersebut kemudian di*sort* berdasarkan nilai kemiripan secara terurut mengecil.

IV.2 Penjelasan Struktur Program

Secara garis besar, program memiliki struktur seperti gambar di bawah ini.

A. Bagian pemrosesan nilai vektor

Memuat modul-modul untuk pemrosesan gambar. Modul-modul tersebut antara lain :

1. byColor.py

Fungsi/Prosedur	Keterangan	Penjelasan Algoritma
def CBIRbyColor(path) { mengembalikan array dari konkatenasi histogram HSV }	Mengubah gambar menjadi matriks dengan bantuan module opencv2	Mengembalikan nilai array vektor fitur warna

2. byTexture.py

Fungsi/Prosedur	Keterangan	Penjelasan Algoritma
def rgbToGrayscale(vektorRGB) {fungsi yang mbalikan vektor grayscale masukan vektorRGB hasil OPENCV2}	Melakukan konversi dari vektor RGB ke vektor grayscale	Menerima nilai matrik RGB yang dibaca oleh opencv2

def quantifyGrayscale(vektorGrayscale) { mengembalikan vektor telah dikuantisasi }	Melakukan kuantifikasi terhadap vektor grayscale	Nilai yang besar (256 x 256) dapat dikuantifikasi menjadi 32 x 32
def buildCoOccuranceMatrix(vetorQuantized)	Membangun matriks co-occurence matriks vektor grayscale dengan jarak 1 skala 0 derajat.	
def transpose(matrix)	Melakukan operasi transpos pada matriks.	
def getStmmetryMatrix(matrix1, matrix2)	Melakukan operasi penjumlahan pada matriks.	
def normalizeMatrix(matrix)	Melakukan operasi normalisasi pada matriks.	
def getContrast(GLCM)	Menghitung nilai contrast dari matriks GLCM.	
def getDissimilarity(GLCM)	Menghitung nilai dissimilarity dari matriks GLCM.	
def getHomogeneity(GLCM)	Menghitung nilai homogeneity dari matriks GLCM.	
def getASM(GLCM)	Menghitung nilai ASM dari matriks GLCM.	
def getEnergy(GLCM)	Menghitung nilai energy dari matriks GLCM.	
def getEntropy(GLCM)	Menghitung nilai entropy dari matriks GLCM.	
def getCollereation(GLCM)	Menghitung nilai correlation dari atriks GLCM.	

def getTextureFeatures(ImageMatrix)	Menyimpan nilai contrast, dissimilarity, homogeneity, ASM, energy, dan correlation dari matriks GLCM dalam suatu vektor.	Mengembalikan nilai array vektor 7 elemen
def CosineSimilarity(Texture1, Texture2)	Melakukan perhitungan cosine similarity.	
def decimalToPercentage(decimal)	Mengubah suatu angka dari desimal ke persentase.	

3. compare.py

Fungsi/Prosedur	Keterangan	Penjelasan Algoritma
def compareImage(str reference_image_path , str folder_path) {Mengembalikan array of tuple yang menyimpan (nilai_cossin_similarity, folder_path)}	Metode yang digunakan untuk meng *compare* image yang baru saja dimasukkan dengan dataset yang baru saja dihapus dengan langkah-langkah. Menggunakan libary numpy, opencv2, concurrent.features	Langkah-langkah : ● Mengubah gambar menjadi matriks dengan menggunakan opencv ● Meresize gambar ● Mendapatkan Texture Fitur dari funsi getTextureFitur yang ada di byTexture.py ● Melakukan looping untuk mencompare image dengan dataset.

B. Bagian Website

Website program ini menggunakan framework React untuk bagian antarmuka nya. Framework tersebut memadukan HTML untuk kemudahan keterbacaan dengan memanfaatkan komponen-komponen. Selain itu penulis menggunakan Tailwind CSS yang diintegrasikan dengan React untuk pemercantik elemen. Pada bagian *backend* juga menggunakan Werkzeug sebagai penampung response serta request handling yang dikirim dari user ke server dan Flask sebagai framework yang menggabungkan semua module tersebut.

Pada bagian antarmuka, banyak sekali elemen yang dipakai di program ini, seperti input dan form untuk menerima masukkan dari user berupa file gambar, button untuk menerima state masukan dari user (tekstur atau gambar), serta img untuk menampilkan file image masukan user. Tailwind CSS digunakan untuk mempercantik komponen-komponen baik dengan menata ukuran elemen, posisi elemen, serta font text.

Framework Flask pada program ini digunakan untuk menghubungkan semua module Python yang dibutuhkan untuk membangun server side dari suatu website. Dengan menggunakan Flask, penulis dapat membuat website yang responsive. User dapat mengirimkan data lewat website, kemudian Flask akan membaca dan menyimpannya dengan bantuan Werkzeug.

IV.3 Penjelasan Penggunaan Program

Secara umum, berikut adalah cara umum penggunaan program:

1. Pengguna memasukkan gambar yang ingin dicari dengan menekan tombol ‘Upload File’, setelah memilih, pengguna menekan tombol ‘Insert Image’.
2. Kemudian, pengguna memasukkan multifiles gambar-gambar dataset yang ingin dicari dengan menekan tombol ‘Upload File’, setelah memilih, pengguna menekan tombol ‘Upload Dataset’.
3. Pilih opsi pencarian, ini dapat dilakukan dengan menge-*switch* ingin melakukan pencarian berdasarkan warna atau tekstur.
4. Tekan tombol *search*, program kemudian akan memproses, mencari gambar-gambar dari dataset yang memiliki kemiripan dengan gambar yang dimasukkan tadi.
5. Program akan menampilkan kumpulan gambar yang mirip, diurutkan dari yang memiliki kemiripan paling tinggi ke yang paling rendah dan juga memunculkan persentase kemiripannya.
6. Pengguna juga dapat melihat informasi terkait jumlah gambar yang muncul, dan waktu eksekusi programnya.

IV.4 Hasil Pengujian

Data Uji	Hasil Uji

IV.5 Analisis Desain Solusi Algoritma

Pada hasil uji, terlihat bahwa nilai-nilai kemiripan berdasarkan tekstur jauh lebih tinggi dan seringkali mencapai nilai 100%, sementara nilai kemiripan berdasarkan parameter warna terlihat lebih rasional. Hal tersebut terjadi pada hampir semua kasus. Hal ini karena beberapa metode ekstraksi fitur tekstur memiliki sifat yang dapat menyebabkan nilai kemiripan yang tinggi, terutama jika gambar memiliki pola tekstur yang sangat mirip. Hal ini juga mungkin karena dataset yang digunakan untuk uji dapat memiliki sejumlah besar gambar yang memiliki perbedaan warna yang relatif kecil tetapi memiliki tekstur yang sangat mirip. Ini dapat menyebabkan metode pengambilan tekstur memberikan bobot yang tinggi pada kesamaan.

BAB V
PENUTUP

V.1 Kesimpulan

Sistem temu balik gambar dapat dilakukan dengan memanfaatkan Aljabar Vektor, melakukan komparasi dengan *cosine similarity* (memanfaatkan *dot product* dan panjang vektor) nilai-nilai vektor fitur gambar baik tekstur maupun warna. Aljabar vektor digunakan untuk menggambarkan dan menganalisis data menggunakan pendekatan klasifikasi berbasis konten (*Content-Based Image Retrieval* atau CBIR). Website dibuat untuk kemudahan pengguna dalam melakukan komparasi atau temu balik gambar.

V.2 Saran

Pengerjaan Tugas Besar 2 Aljabar Linier dan Geometri tentunya tidak luput dari hambatan dan juga kendala. Demi pelaksanaan yang lancar, penulis telah mencatat beberapa saran serta poin yang dapat diperhatikan bagi pihak yang berminat untuk membuat program temu balik gambar memanfaatkan CBIR dan aljabar vektor:

1. Terutama jika belum pernah melakukan project website, penulis menyarankan untuk mengalokasi waktu demi mendalami serta memahami masalah yang akan diselesaikan serta penyelesaian yang dirasa sesuai sebelum memulai tahap pengerjaan. Seperti memahami hal apa saja yang dibutuhkan dalam proses pengerjaan serta mengetahui keunggulan dan kekurangan dari setiap framework dan modul yang mungkin dapat digunakan
2. Penulis menyarankan untuk melakukan eksplorasi terhadap skema *multiprocessing* yang baik dalam proses ekstraksi fitur dan penghitungan nilai kemiripan gambar agar pelaksanaan program lebih efisien.
3. Apabila pengerjaan dilakukan dalam bentuk kelompok, sangat direkomendasikan agar pembagian tugas dilakukan setelah setiap anggota sudah melakukan eksplorasi terhadap permasalahan yang akan dipecahkan dan gambaran umum program sudah ada agar tugas yang dibagi sudah jelas dan pengerjaannya dilakukan secara terarah walaupun terpisah.

V.3 Komentar

Tanggapan terhadap tugas besar ini adalah tugas ini sangat mendukung eksplorasi mandiri mahasiswa di bidang keinformatikaan serta tidak terbatas pada materi Aljabar Vektor maupun mata kuliah Aljabar Linear dan Geometri saja.

V.4 Refleksi

Dalam pengerjaan tugas ini, penulis banyak belajar hal baru dan sangat berharga. Meski tidak banyak diimplementasikan, penulis jadi mengetahui adanya berbagai *library* di Python yang dapat digunakan untuk mempermudah pembuatan ataupun pengoptimalan program. Akibatnya, para penulis juga semakin menghargai keberadaan dan juga usaha yang dituangkan dalam penciptaan modul dan framework seperti NumPy, OpenCV, Flask, dan lain sebagainya. Penulis juga belajar mengenai pengembangan website sederhana dan hal-hal yang dibutuhkan dalam pengembangan website sederhana.

Tentunya kelancaran proses pengerjaan tugas besar ini jauh dari ideal. Beragam kendala dan juga hambatan ditemui oleh para penulis selama proses penyelesaian tugas besar. Misalnya dalam pengintegrasian website *front-end, back-end,* dan program. Namun, karena keterbatasan waktu dan terutama kemampuan, website SangkuLens yang telah penulis buat tersebut adalah usaha penulis sebaik-baiknya.

V.5 Ruang Pengembangan

Tentu banyak hal yang bisa dikembangkan untuk tugas penulis, misalnya dalam hal kecepatan pemrosesan gambar. Tugas juga bisa lebih baik jika dilengkapi kemampuan menangkap gambar dari kamera, mengambil gambar dengan *scraping,* ataupun mengunduh hasil pemrosesan. Hal tersebut tentu dapat dikembangkan lebih jauh lagi dan semoga dapat penulis pelajari dan eksplorasi di kesempatan lain.

V.6 Link Release Terakhir

Link rilis terakhir kelompok penulis untuk Tugas Besar 2 IF2123 Aljabar Linier dan Geometri adalah sebagai berikut:

<insert link>

DAFTAR PUSTAKA

Smyth, Patrick. (2018). "Creating APIs with Python and Flask." Programming Historian. https://programminghistorian.org/en/lessons/creating-apis-with-python-and-flask#what-users-want-in-an-api

Haralick, R. M., Shanmugam, K., & Dinstein, I. (1973). "Textural Features for Image Classification." In IEEE Transactions on Systems, Man, and Cybernetics (Vol. SMC-3, No. 6, pp. 610-621). https://www.sciencedirect.com/science/article/pii/S0895717710005352#s000030

Logianto, Albert. (2015). "Aplikasi Aljabar Vektor dalam Sistem Temu-balik Informasi Visual Menggunakan Tekstur dan Warna". https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2015-2016/Makalah-2015/Makalah-IF2123-2015-061.pdf

Nasution, M. A., & Sinaga, P. (2020). "Implementation of Gray Level Co-Occurrence Matrix (GLCM) Algorithm for Object Recognition in Digital Images." In Journal of Information Technology and Electrical Engineering, 2(2), 102-107. https://ojs.uma.ac.id/index.php/jite/article/view/3885/2785

Yunus, Muhammad. (2021). "Feature Extraction: Gray Level Co-occurrence Matrix (GLCM)." https://yunusmuhammad007.medium.com/feature-extraction-gray-level-co-occurrence-matrix-glcm-10c45b6d46a1