

**LAPORAN TUGAS BESAR II**  
**IF2123 - ALJABAR LINEAR DAN GEOMETRI**  
**SEMESTER I 2023/2024**

Kelompok 24 / TemuKangenUnch

Ibrahim Ihsan Rasyid	13522018
Marzuli Suhada M	13522070
Zahira Dina Amalia	13522085



**PROGRAM STUDI TEKNIK INFORMATIKA**  
**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA**  
**INSTITUT TEKNOLOGI BANDUNG 2023**

## DAFTAR ISI

<b>BAB I.....</b>	<b>1</b>
<b>BAB II.....</b>	<b>2</b>
1. Dasar Teori.....	2
a. CBIR Berbasis Warna.....	2
b. CBIR Berbasis Tekstur.....	4
2. Pengembangan Website.....	6
<b>BAB III.....</b>	<b>8</b>
1. Pemecahan Masalah dengan CBIR Berbasis Warna.....	8
2. Pemecahan Masalah dengan CBIR Berbasis Tekstur.....	9
3. Website sebagai Wadah.....	10
<b>BAB IV.....</b>	<b>12</b>
1. Implementasi Program Utama.....	12
a. CBIR Berbasis Warna (CBIR_Colour.py).....	12
b. CBIR Berbasis Tekstur (CBIR_Texture.py).....	14
c. Program Utama.....	15
d. Implementasi lain.....	15
2. Penjelasan Struktur Program berdasarkan Spesifikasi.....	15
3. Penjelasan Tata Cara Penggunaan Program.....	19
4. Hasil Pengujian.....	19
5. Analisis dari Desain Solusi Algoritma Pencarian yang Diimplementasikan pada Setiap Pengujian yang Dilakukan.....	20
<b>BAB V.....</b>	<b>22</b>
1. Kesimpulan.....	22
2. Saran.....	22
3. Komentar atau Tanggapan.....	23
4. Refleksi.....	23
5. Ruang Perbaikan atau Pengembangan.....	24
<b>DAFTAR PUSTAKA.....</b>	<b>26</b>
<b>LAMPIRAN.....</b>	<b>27</b>

## **BAB I**

### **DESKRIPSI MASALAH**

Dalam era digital, jumlah gambar yang dihasilkan dan disimpan semakin meningkat dengan pesat, baik dalam konteks pribadi maupun profesional. Peningkatan ini mencakup berbagai jenis gambar, mulai dari foto pribadi, gambar medis, ilustrasi ilmiah, hingga gambar komersial. Terlepas dari keragaman sumber dan jenis gambar ini, sistem temu balik gambar (*image retrieval system*) menjadi sangat relevan dan penting dalam menghadapi tantangan ini. Dengan bantuan sistem temu balik gambar, pengguna dapat dengan mudah mencari, mengakses, dan mengelola koleksi gambar mereka. Sistem ini memungkinkan pengguna untuk menjelajahi informasi visual yang tersimpan di berbagai platform, baik itu dalam bentuk pencarian gambar pribadi, analisis gambar medis untuk diagnosis, pencarian ilustrasi ilmiah, hingga pencarian produk berdasarkan gambar komersial. Salah satu contoh penerapan sistem temu balik gambar yang mungkin kalian tahu adalah Google Lens.



**Gambar 1.** Contoh penerapan *information retrieval system* (Google Lens)

Di dalam Tugas Besar 2 ini, Anda diminta untuk mengimplementasikan sistem temu balik gambar yang sudah dijelaskan sebelumnya 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.

## **BAB II**

### **LANDASAN TEORI**

#### 1. Dasar Teori

Teori yang mendasari tugas besar ini ialah *Content-Based Information Retrieval* yang biasa disebut dengan CBIR. Sistem CBIR, atau Pencarian Citra Berbasis Konten, adalah suatu metode yang digunakan untuk mencari dan mengambil gambar berdasarkan karakteristik visual atau isi mereka. Tahap awal dalam proses ini melibatkan ekstraksi fitur-fitur kunci dari gambar, seperti warna, tekstur, dan bentuk. Setelah fitur-fitur ini diperoleh, mereka diwakili dalam bentuk vektor atau deskripsi numerik yang dapat dibandingkan dengan gambar-gambar lain dalam suatu dataset. Selanjutnya, CBIR menggunakan algoritma pencocokan untuk membandingkan vektor-fitur dari gambar yang dicari dengan vektor-fitur gambar dalam dataset. Hasil dari pencocokan ini digunakan untuk menyusun rangking gambar dalam dataset dan menampilkan gambar yang memiliki kemiripan visual tertinggi dengan gambar yang dicari. Proses CBIR memberikan bantuan kepada pengguna dengan memberikan akses dan kemampuan untuk menjelajahi koleksi gambar secara efisien. Ini dilakukan tanpa perlu melakukan pencarian berdasarkan teks atau kata kunci, melainkan berdasarkan kesamaan nilai citra visual di antara gambar-gambar tersebut. Dalam konteks ini, terdapat dua parameter CBIR yang umumnya digunakan, yaitu parameter warna dan tekstur.

##### a. CBIR Berbasis Warna

Dalam CBIR berbasis warna, perbandingan dilakukan antara gambar masukan dengan gambar yang terdapat dalam dataset. Proses ini dimulai dengan mengubah gambar berformat RGB ke dalam metode histogram warna yang umum digunakan. Histogram warna merepresentasikan frekuensi kemunculan berbagai warna dalam suatu ruang warna tertentu. Tujuan dari proses ini adalah untuk mendistribusikan warna-warna dalam gambar. Namun, penting untuk dicatat bahwa histogram warna tidak dapat mengidentifikasi objek spesifik dalam gambar atau memberikan deskripsi tentang posisi warna yang terdistribusi. Pembentukan ruang warna dilakukan dengan membagi nilai-nilai citra ke dalam beberapa rentang nilai yang lebih kecil. Hal ini dilakukan untuk membuat histogram warna, di mana setiap interval atau rentang

dianggap sebagai "bin". Perhitungan histogram warna melibatkan penghitungan jumlah piksel yang memiliki nilai warna dalam setiap interval. Dalam konteks perhitungan histogram, digunakan warna global HSV karena warna tersebut dapat bekerja lebih baik pada latar belakang kertas berwarna putih, yang umumnya digunakan. Oleh karena itu, perlu dilakukan konversi warna dari RGB ke HSV. Proses normalisasi nilai RGB dilakukan dengan mengubah rentang nilai dari [0, 255] menjadi [0, 1] dengan cara berikut,

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

Selanjutnya, dilakukan pencarian nilai Cmax, Cmin, dan  $\Delta$  untuk digunakan dalam perhitungan nilai HSV.

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

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

$$\Delta = C_{max} - C_{min}$$

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

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

$$V = C_{max}$$

Setelah mendapatkan nilai HSV, dilakukan perbandingan antara gambar masukan dan gambar dalam dataset menggunakan metode cosine similarity.

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

Cosine similarity mengukur tingkat kemiripan berdasarkan sejauh mana hasil dari perhitungan cosine similarity pada vektor A dan B. Pencarian histogram juga melibatkan pembagian gambar menjadi blok-blok dengan ukuran  $n \times n$ , pada tugas ini digunakan blok  $4 \times 4$  agar proses pencarian lebih efektif.

#### b. CBIR Berbasis Tekstur

CBIR dengan perbandingan tekstur dilakukan menggunakan suatu matriks yang dinamakan co-occurrence matrix. Matriks ini digunakan karena dapat melakukan pemrosesan yang lebih mudah dan cepat. Vektor yang dihasilkan juga mempunyai ukuran yang lebih kecil. Misalkan terdapat suatu gambar I dengan  $n \times m$  piksel dan suatu parameter offset  $(\Delta x, \Delta y)$ , Maka dapat dirumuskan matriksnya sebagai berikut:

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

Dengan menggunakan nilai i dan j sebagai nilai intensitas dari gambar dan p serta q sebagai posisi dari gambar, maka offset  $\Delta x$  dan  $\Delta y$  bergantung pada arah  $\theta$  dan jarak yang digunakan melalui persamaan di bawah ini. Melalui persamaan tersebut, digunakan nilai  $\theta$  adalah  $0^\circ, 45^\circ, 90^\circ$ , dan  $135^\circ$ . Sebagai gambaran, cara pembuatan co-occurrence matrix diilustrasikan pada gambar berikut,

	1	2	3	4	5	6	7	8
1	1	2	0	0	1	0	0	0
2	0	0	1	0	1	0	0	0
3	0	0	0	0	1	0	0	0
4	0	0	0	0	1	0	0	0
5	1	0	0	0	0	1	2	0
6	0	0	0	0	0	0	0	1
7	2	0	0	0	0	0	0	0
8	0	0	0	0	1	0	0	0

**Gambar 2.** Cara Pembuatan Matrix Occurrence

Selanjutnya, dibuat symmetric matrix dengan menjumlahkan co-occurrence matrix dengan hasil transpose-nya. Lalu dicari matrix normalization dengan persamaan.

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

Pada CBIR berbasis tekstur warna gambar dikonversi menjadi grayscale dikarenakan warna tidak penting dalam penentuan tekstur. Oleh karena itu, warna RGB dapat diubah menjadi suatu warna grayscale Y dengan rumus:

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

Nilai grayscale kemudian dikuantifikasi. Karena citra grayscale berukuran 256 piksel, maka matriks yang berkoresponden akan berukuran 256 x 256. Berdasarkan penglihatan manusia, tingkat kemiripan dari gambar dinilai berdasarkan kekasaran tekstur dari gambar tersebut. Dari co-occurrence matrix bisa diperoleh 6 komponen ekstraksi tekstur, yaitu *contrast*, *homogeneity*, *entropy*, *dissimilarity*, *energy*, dan *correlation*. Persamaan yang dapat digunakan untuk mendapatkan nilai 3 komponen tersebut antara lain:

Contrast:

$$\sum_{i,j=0}^{\text{dimensi}-1} P_{i,j} (i - j)^2$$

Homogeneity:

$$\sum_{i,j=0}^{\text{dimensi}-1} \frac{P_{i,j}}{1 + (i - j)^2}$$

Entropy:

$$-\left( \sum_{i,j=0}^{\text{dimensi}-1} P_{i,j} \times \log P_{i,j} \right)$$

Dissimilarity:

$$\sum_{i,j=0}^{levels-1} P_{i,j}|i - j|$$

Energy:

$$\sqrt{ASM}, \text{ 'ASM': } \sum_{i,j=0}^{levels-1} P_{i,j}^2$$

Correlation:

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

Keterangan : P merupakan matriks co-occurrence Dari ketiga komponen tersebut, dibuatlah sebuah vektor yang akan digunakan dalam proses pengukuran tingkat kemiripan.

Kemiripan kemudian diukur dari kedua gambar dengan menggunakan Teorema Cosine Similarity, yaitu:

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

Disini A dan B adalah dua vektor dari dua gambar. Semakin besar hasil Cosine Similarity kedua vektor maka tingkat kemiripannya semakin tinggi.

## 2. Pengembangan Website

Pengembangan website Reverse Image Search dengan menggunakan Content-Based Image Retrieval (CBIR) merupakan langkah inovatif yang diambil untuk menggabungkan konsep Aljabar Linear dan Geometri ke dalam dunia pencarian gambar. Dalam kerangka tugas besar ini, tim kami menekankan pada rancangan dan implementasi sistem yang tidak hanya memungkinkan pengguna untuk mencari gambar, tetapi juga memberikan kemampuan untuk menggunakan gambar sebagai kueri pencarian. Algoritma CBIR menjadi unsur kunci dalam proses ini, di mana fitur-fitur penting dari setiap gambar diekstrak dan diubah ke dalam representasi vektor di dalam ruang fitur.

Penerapan konsep Aljabar Linear menjadi krusial dalam pengolahan matriks representasi fitur, di mana operasi-aljabar seperti perkalian matriks akan digunakan untuk menghasilkan representasi yang lebih abstrak dan ekspresif dari setiap gambar. Lebih jauh, pemahaman geometri menjadi landasan dalam memahami hubungan antar gambar yang diwakili dalam ruang fitur tersebut. Ini memungkinkan sistem untuk mengenali pola-pola dan karakteristik visual yang mungkin sulit diinterpretasikan oleh metode pencarian gambar konvensional.

Selain itu, kami memastikan pengoptimalan pencarian gambar dengan memanfaatkan konsep transformasi linear dan proyeksi dalam konteks ruang fitur. Transformasi linear digunakan untuk mengubah representasi fitur ke dalam bentuk yang lebih efisien dan memadai, sedangkan proyeksi membantu mengidentifikasi kemiripan dan perbedaan antara gambar-gambar dalam ruang fitur tersebut.

Dengan proyek ini, kami memiliki tujuan untuk mendemonstrasikan bahwa teori Aljabar Linear dan Geometri bukan hanya konsep akademis, tetapi dapat diaplikasikan secara langsung dalam pengembangan teknologi pencarian gambar yang lebih canggih dan efisien. Dengan merangkum prinsip-prinsip ini ke dalam desain dan implementasi website Reverse Image Search, kami berharap dapat menciptakan alat yang tidak hanya kuat dari segi teori, tetapi juga memberikan nilai tambah nyata dalam pemrosesan dan temu balik gambar berbasis konten.

## **BAB III**

### **ANALISIS PEMECAHAN MASALAH**

#### 1. Pemecahan Masalah dengan CBIR Berbasis Warna

Pertama-tama, dalam pemecahan masalah dengan warna dari gambar, perlu dilakukan pemilihan fitur-fitur kunci yang sesuai untuk ekstraksi informasi dari gambar. Fitur utama dalam Content-Based Image Retrieval (CBIR) berbasis warna adalah fitur warna dari suatu gambar. Dalam cara ini, fitur warna direpresentasikan sebagai vektor warna dalam ruang warna RGB dan HSV. Suatu gambar dapat dimodelkan sebagai kumpulan vektor warna, di mana setiap vektor merepresentasikan warna dari satu piksel. Operasi vektor dalam pencarian kemiripan melibatkan konsep aljabar geometri, seperti penjumlahan vektor, perkalian skalar, dan perhitungan jarak antara vektor warna. Misalnya, untuk mencari kemiripan antara dua gambar, kita dapat menggunakan formula jarak Euclidean dan/atau cosine similarity antara vektor warna. Transformasi geometri, seperti rotasi atau skalasi, dapat diterapkan pada vektor warna untuk mengoreksi perbedaan orientasi atau ukuran gambar, meningkatkan akurasi pencarian. Pemilihan ruang warna alternatif, seperti HSV atau Lab, juga dapat diperhitungkan untuk meningkatkan kualitas pemetaan warna dan akurasi pencarian.

Tabel 3.1.1 Pemetaan Masalah pada CBIR Berbasis Warna

Variabel	Pemodelan Masalah	Penerapan Aljabar Geometri
Fitur Warna	Model warna direpresentasikan sebagai vektor RGB dalam ruang warna	<ul style="list-style-type: none"><li>- Operasi vektor dalam pencarian kemiripan dengan menggunakan formula jarak euclidean dan/atau cosine similarity antara vektor warna</li><li>- Transformasi geometri untuk koreksi warna untuk meningkatkan akurasi pencarian.</li><li>- Penggunaan ruang warna alternatif seperti HSV</li></ul>

Salah satu contoh kasus yang diselesaikan dengan CBIR berbasis warna, ialah ketika hendak mencari mobil dengan warna idaman. Ketika hendak mencari sebuah mobil dengan warna tertentu, pengguna bisa memasukkan salah satu gambar mobil dengan warna yang dia inginkan. Lalu, sistem akan memberikan bervariasi macam mobil dengan warna yang mirip dengan warna mobil tersebut. Hal ini dapat memudahkan pengguna dalam mencari mobil dengan warna idamannya sehingga ia tidak perlu mengecek satu-satu untuk tiap jenis mobil untuk warna yang ia inginkan.

## 2. Pemecahan Masalah dengan CBIR Berbasis Tekstur

Dalam pemecahan masalah dengan tekstur gambar, fokus utama adalah pada variabel matriks co-occurrence yang merepresentasikan hubungan spasial antara nilai piksel dalam gambar. Dalam CBIR berbasis tekstur, setiap elemen dalam matriks co-occurrence dapat dianggap sebagai nilai dari suatu dimensi dalam ruang vektor. Suatu gambar dapat direpresentasikan sebagai vektor dalam ruang vektor berdimensi tinggi. Enam komponen ekstraksi tekstur dapat diekstraksi dari matriks co-occurrence yang sudah dinormalisasi, dan kemiripannya dapat dihitung menggunakan cosine similarity pada vektor komponen ekstraksi fitur. Operasi vektor dalam pencarian kemiripan tekstur melibatkan konsep aljabar geometri, seperti penjumlahan vektor dan perhitungan jarak antara vektor tekstur. Transformasi geometri, seperti perubahan skala, dapat diterapkan pada vektor tekstur untuk mengoreksi perbedaan skala antar gambar, menciptakan pemetaan yang lebih akurat. Penggunaan matriks simetris dalam matriks co-occurrence dapat dimodelkan sebagai sifat-sifat geometris tertentu, yang memudahkan dalam operasi dan analisis. Pemetaan ke ruang vektor tertentu berdasarkan fitur tekstur dapat membantu menciptakan pemetaan yang lebih terstruktur.

Tabel 3.2.1 Pemetaan Masalah pada CBIR Berbasis Tekstur

Variabel	Pemodelan Masalah	Penerapan Aljabar Geometri
Matriks co-occurrence Tekstur	Model matriks co-occurrence tekstur sebagai vektor dalam ruang vektor berdimensi	- Penggunaan matriks simetris untuk memudahkan dalam operasi dan analisis.

	<p>tinggi dan model vektor dengan enam komponen ekstraksi fitur dari matriks co-occurrence (<i>contrast, homogeneity, energy, dissimilarity, entropy, correlation</i>)</p>	<ul style="list-style-type: none"> <li>- Pemetaan gambar ke ruang vektor tertentu berdasarkan fitur tekstur. Ini membantu dalam menciptakan pemetaan yang lebih terstruktur.</li> </ul>
--	--	---

Salah satu contoh kasus yang diselesaikan dengan CBIR berbasis tekstur, ialah ketika hendak mendeteksi kecacatan atau menginspeksi kualitas suatu produk atau benda. Apabila dimasukkan gambar suatu benda dengan tekstur yang tidak biasa, maka CBIR dapat mendeteksinya. CBIR berbasis tekstur dapat sangat membantu sebuah perusahaan dalam meningkatkan kualitas kontrolnya dalam menghadapi perubahan tekstur dari produknya.

### 3. *Website* sebagai Wadah

*Website* merupakan salah satu platform termudah yang dapat diakses secara umum, dan memiliki potensi besar sebagai wadah untuk sistem Content-Based Image Retrieval (CBIR) berbasis tekstur dan/atau warna. Dengan memanfaatkan website sebagai antarmuka pengguna, pengguna dapat dengan mudah mengakses dan berinteraksi dengan sistem temu balik gambar ini. Fasilitas website memungkinkan pengguna untuk melakukan perbandingan gambar dengan cepat dan efisien.

Dalam konteks CBIR berbasis warna, website dapat menyajikan antarmuka yang intuitif untuk memilih fitur-fitur warna yang relevan, misalnya, memilih skala warna atau mode ruang warna. Pengguna dapat dengan mudah memasukkan gambar yang ingin dibandingkan dan melihat hasil perbandingan secara langsung. Operasi vektor dalam pencarian kemiripan, seperti penjumlahan vektor atau perhitungan jarak Euclidean, dapat diimplementasikan di belakang layar untuk memberikan hasil perbandingan yang akurat.

Sebagai tambahan, dalam CBIR berbasis tekstur, website dapat menyediakan opsi untuk memilih metode ekstraksi fitur tekstur dan mengatur parameter-parameter terkait. Pengguna dapat langsung melihat perbandingan antar gambar berdasarkan tekstur, dan

website dapat menggambarkan visualisasi vektor komponen ekstraksi fitur untuk memberikan pemahaman yang lebih baik tentang kesamaan antar gambar.

Transformasi geometri, seperti rotasi atau perubahan skala, dapat diterapkan secara dinamis melalui antarmuka website, memungkinkan pengguna untuk mengoreksi perbedaan orientasi atau ukuran gambar dengan mudah. Penggunaan matriks simetris dalam matriks co-occurrence tekstur dapat disajikan secara transparan kepada pengguna melalui visualisasi yang dapat diakses di website.

Dengan demikian, melalui website, pengguna dapat dengan cepat, mudah, dan interaktif melakukan perbandingan gambar menggunakan sistem temu balik gambar berbasis CBIR. Website sebagai wadah tidak hanya memberikan aksesibilitas yang luas, tetapi juga memfasilitasi pengalaman pengguna yang efisien dan intuitif dalam mengelola dan menganalisis koleksi gambar mereka.

Salah satu contoh kasus yang diselesaikan dengan keberadaan website adalah sebuah platform galeri gambar online, "VisualShare," memanfaatkan teknologi Sistem Temu Kembali Citra Berbasis Warna dan Tekstur (CBIR) untuk memberikan pengalaman pencarian dan pengelolaan gambar yang lebih efektif. Pengguna *VisualShare* dapat dengan mudah menemukan gambar yang mereka cari berdasarkan fitur warna dan tekstur tertentu. Misalnya, seorang desainer mode yang mencari inspirasi dapat menggunakan fitur pencarian berbasis warna untuk menemukan gambar dengan palet warna yang serupa. Selain itu, CBIR memungkinkan pengguna mengeksplorasi galeri gambar dengan cara yang lebih kontekstual, karena sistem dapat mencocokkan vektor fitur warna dan tekstur dari citra query dengan gambar dalam basis data. Pengguna juga dapat menambahkan tag atau metadata pada gambar-gambar mereka sendiri, memfasilitasi pengelolaan koleksi dan pencarian lebih lanjut. *VisualShare* tidak hanya menyajikan galeri gambar interaktif tetapi juga memungkinkan pengguna berbagi gambar melalui tautan atau media sosial, meningkatkan kolaborasi dan interaksi di dalam komunitas tersebut. Dengan adanya CBIR, *VisualShare* memberikan solusi yang lebih cerdas dan kontekstual dalam mengelola dan berbagi koleksi gambar online.

## BAB IV

### IMPLEMENTASI DAN UJI COBA

#### 1. Implementasi Program Utama

Kami menggunakan beberapa library untuk membantu dan mendukung program dalam melakukan perhitungan. Beberapa library yang kami gunakan di antaranya numpy, cv, os, dan berbagai library lainnya.

##### a. CBIR Berbasis Warna (CBIR\_Colour.py)

```
function max(a, b):
    return np.maximum(a, b)

function min(a, b):
    return np.minimum(a, b)

function rgb_to_hsv(img):
    img = img / 255.0
    r, g, b = img[:, :, 0], img[:, :, 1], img[:, :, 2]

    cmax = max(max(r, g), b)
    cmin = min(min(r, g), b)
    delta = cmax - cmin

    h = zeros_like(r)
    mask = (delta != 0)
    h[mask & (cmax == r)] = ((g[mask & (cmax == r)] - b[mask & (cmax == r)]) / delta[mask & (cmax == r)] % 6) / 6
    h[mask & (cmax == g)] = ((b[mask & (cmax == g)] - r[mask & (cmax == g)]) / delta[mask & (cmax == g)] + 2) / 6
    h[mask & (cmax == b)] = ((r[mask & (cmax == b)] - g[mask & (cmax == b)]) / delta[mask & (cmax == b)] + 4) / 6
    h[~mask] = 0

    s = zeros_like(r)
    s[cmax == 0] = 0
    s[cmax != 0] = delta[cmax != 0] / cmax[cmax != 0]

    v = cmax

    hsv_matrix = []
    for i in range(h.shape[0]):
        row = []
        for j in range(h.shape[1]):
            row.append({'h': h[i, j], 's': s[i, j], 'v': v[i, j]})
        hsv_matrix.append(row)

    return hsv_matrix
```

```

function display_matrix(mat):
    for i in range(len(mat)):
        for j in range(len(mat[0])):
            print("{:.2f}, {:.2f}, {:.2f}], ".format(mat[i][j]['h'],
mat[i][j]['s'], mat[i][j]['v']), end="")
    print()

function compare(A, B):
    dot_product = np.dot(A, B)
    norm_A = np.linalg.norm(A)
    norm_B = np.linalg.norm(B)

    if norm_A == 0 or norm_B == 0:
        return 0
    else:
        return dot_product / (norm_A * norm_B)

function average_hsv_block(mat, i, j, size):
    avg_h = 0.0
    avg_s = 0.0
    avg_v = 0.0

    count = 0
    for k in range(i, min(i + size, len(mat))):
        for l in range(j, min(j + size, len(mat[0]))):
            if k < len(mat) and l < len(mat[0]):
                avg_h += mat[k][l]['h']
                avg_s += mat[k][l]['s']
                avg_v += mat[k][l]['v']
                count += 1

    if count > 0:
        avg_h /= count
        avg_s /= count
        avg_v /= count

    return {'h': avg_h, 's': avg_s, 'v': avg_v}

function compute_histogram(block, histogram, bins):
    histogram[int(block['h'] / (360.0 / bins))] += 1
    histogram[min(int(block['s']) * 100 / (100.0 / bins)), bins * 3 - 1] += 1
    histogram[min(int(block['v']) * 100 / (100.0 / bins)), bins * 3 - 1] += 1

function cosine_similarity_block(mat1, mat2, bins, size):
    sum_similarity = 0.0
    step = 0

    histogram1 = zeros(bins * 3, dtype=int)
    histogram2 = zeros(bins * 3, dtype=int)

    for i in range(0, len(mat1), size):

```

```

for j in range(0, len(mat1[0]), size):
    histogram1.fill(0)
    histogram2.fill(0)

    avg_block1 = average_hsv_block(mat1, i, j, size)
    avg_block2 = average_hsv_block(mat2, i, j, size)

    compute_histogram(avg_block1, histogram1, bins)
    compute_histogram(avg_block2, histogram2, bins)

    step += 1
    similarity = compare(histogram1, histogram2)
    sum_similarity += similarity

    if i + size >= len(mat1) and j + size >= len(mat1[0]):
        break

average_similarity = sum_similarity / step
return average_similarity

```

### b. CBIR Berbasis Tekstur (CBIR\_Texture.py)

```

function to_gray(img):
    r, g, b = img[:, :, 0], img[:, :, 1], img[:, :, 2]
    return 0.299 * r + 0.587 * g + 0.114 * b

function process_chedec(image):
    gray_img = to_gray(image)

    c = create_zeros_matrix(256, 256, dtype=uint32)

    i_values = gray_img[:, :-1].astype(uint8)
    j_values = gray_img[:, 1: ].astype(uint8)

    add_to_matrix_at_indices(c, (i_values, j_values), 1)
    c_transpose = transpose_matrix(c)
    c_symmetric = add_matrices(c, c_transpose)

    val = sum_of_elements(c)
    normalized = divide_matrix_elements(c_symmetric, val)

    i, j = create_indices_matrix(256, 256)
    contrast = sum_of_elements(normalized * (i - j) ** 2)
    dissimilarity = sum_of_elements(normalized * absolute_value(i - j))
    homogeneity = sum_of_elements(normalized / (1 + (i - j) ** 2))
    entropy_mask = normalized > 0
    entropy = -sum_of_elements(normalized[entropy_mask] * log10(normalized[entropy_mask]))
    asm = sum_of_elements(normalized ** 2)
    energy = square_root(asm)

    miui = sum_of_elements(normalized * i)
    miuj = sum_of_elements(normalized * j)

```

```

mean_i = mean_value(i)
mean_j = mean_value(j)
sigma_i = sum_of_elements(normalized * (i - mean_i) ** 2)
sigma_j = sum_of_elements(normalized * (j - mean_j) ** 2)
correlation = sum_of_elements(normalized * (i - miui) * (j - miuj) / square_root(sigma_i ** 2 * sigma_j ** 2))

diff = [contrast, homogeneity, entropy, dissimilarity, energy,
correlation]
return diff

function compare(chedec1, chedec2):
    numerator = dot_product(chedec1, chedec2)
    denominator = norm(chedec1) * norm(chedec2)
    similarity = (numerator / denominator) * 100
    return similarity

function cbir_texture(img1, img2):
    chedec1 = process_chedec(img1)
    chedec2 = process_chedec(img2)
    similarity = compare(chedec1, chedec2)
    return similarity

```

## 2. Penjelasan Struktur Program berdasarkan Spesifikasi

Secara keseluruhan program ini terbagi atas dua bagian. Bagian yang pertama ialah program yang mengimplementasikan konsep CBIR. Bagian ini terbagi lagi menjadi dua bagian sesuai dengan basisnya, yaitu warna dan tekstur. Adapun bagian yang kedua ialah program untuk pengembangan website *Image Retrieval System*.

Pada bagian implementasi konsep CBIR berbasis warna, terdapat beberapa fungsi yang dibuat untuk mendukung perhitungannya. Berikut beberapa fungsi tersebut dan penjelasannya,

### a. max(a, b)

Fungsi ini mengembalikan nilai maksimum antara dua bilangan, a dan b.

Fungsi ini menggunakan fungsi np.maximum dari NumPy untuk melakukan perbandingan elemen-wise dan menghasilkan elemen yang lebih besar dari kedua input.

### b. min(a, b)

Sama seperti fungsi max, fungsi ini mengembalikan nilai minimum antara dua bilangan, a dan b. Menggunakan fungsi np.minimum untuk melakukan perbandingan elemen-wise dan menghasilkan elemen yang lebih kecil dari kedua input.

c. `rgb_to_hsv(img)`

Fungsi ini melakukan konversi warna dari mode RGB ke mode HSV. Proses konversi diimplementasikan dengan membagi setiap nilai RGB dengan 255, menghitung nilai Hue (H), Saturation (S), dan Value (V) sesuai rumus konversi standar HSV.

d. `compare(A, B)`

Menghitung kemiripan cosinus antara dua vektor, A dan B. Digunakan dot product dan norma vektor dari NumPy untuk perhitungan ini. Jika norma salah satu vektor adalah nol, fungsi mengembalikan 0 untuk menghindari pembagian dengan nol.

e. `average_hsv_block(mat, i, j, size)`

Menghitung nilai rata-rata dari blok matriks HSV. Fungsi ini digunakan untuk mengekstrak fitur warna dari blok gambar dengan menghitung rata-rata H, S, dan V.

f. `compute_histogram(block, histogram, bins)`

Menghitung histogram dari suatu blok gambar berwarna. Histogram merepresentasikan distribusi nilai warna dalam blok tersebut. Blok direpresentasikan dalam tiga channel (H, S, V) dan histogram di update sesuai dengan nilai masing-masing channel.

g. `cosine_similarity_block(mat1, mat2, bins, size)`

Menghitung kemiripan cosinus antara dua blok matriks HSV. Fungsi ini menggunakan fungsi average hsv block untuk mendapatkan nilai rata-rata blok dari dua matriks, lalu menghitung histogram untuk masing-masing blok dan mengukur kesamaan antara histogram-histogram tersebut menggunakan fungsi compare. Hasil akhir adalah rata-rata kemiripan dari seluruh blok.

Ada pula pada bagian implementasi konsep CBIR berbasis tekstur, beberapa fungsi yang melakukan perhitungan *co-occurrence matrix* untuk melakukan perbandingan gambar berbasis tekstur tersebut ialah,

a. `to_gray(img)`

Fungsi ini digunakan untuk mengubah gambar yang berwarna menjadi dalam *grayscale*-nya. Hal ini dilakukan karena pada perbandingan tekstur tidak

dibutuhkan warna sehingga diperlukan parameter gambar untuk diambil komponen rgbnnya dan mengembalikan hasil perkalian dot komponen *red* dengan 0.299, *green* dengan 0.587, dan *blue* dengan 0.114.

### b. process\_chedec(image)

Pada awal fungsi ini, diimplementasikan fungsi *to\_gray* dengan memasukkan parameter *image*. Kemudian, dibuat matrix dengan dimensi 256x256 sesuai dengan skala pada gambar citra *grayscale*. Lalu, *co-occurrence matrix* dibuat untuk nantinya diambil komponen *contrast*, *homogeneity*, *entropy*, *dissimilarity*, *energy*, dan *correlation* dengan mengaplikasikan rumus dan perhitungan yang terdapat pada teori dasar. Fungsi ini akan mengembalikan sebuah vektor yang berisikan komponen-komponen tersebut. Pada fungsi ini, *co-occurrence matrix* dibuat dengan menggunakan metode  $0^\circ$ . Hal ini dilakukan karena terbukti bahwa pembuatan matriks dengan  $0^\circ$  lebih efektif dari segi waktu melalui data yang dicoba saat melakukan testing.

First Image		Gambar ke		Derajat			
Jenis	Kode	Jenis	Kode	Nilai	Waktu	Nilai	Waktu
			12	99.99946		99.99531	99.99766
			15	99.96393		99.94388	99.98862
		harimau ▾	20	99.99841		99.99036	99.99581
			22	99.96098		99.96653	99.98293
			27	99.88787		99.89014	99.93788
			10	99.97593		99.97710	99.99609
		macan ▾	18	99.94948		99.88274	99.94805
			23	99.99490		99.96391	99.98997
			24	99.87228		99.89529	99.95321
			32	99.96229		99.95559	99.98858
			25	99.88347		99.90245	99.95724
			26	99.97239		99.96487	99.98233
		hari mau ▾	41	99.99591	0.6596934795	99.99772	0.6752729416
			79	99.98319		99.93515	99.98146
			91	99.99321		99.99362	99.99904
			13	99.99698		99.99744	99.99925
			16	97.07556		95.86793	98.81629
		singa ▾	19	99.93137		99.95341	99.98345
			21	99.95803		99.98283	99.99448
			33	99.89003		99.90999	99.95585
			150	99.93359		99.94442	99.97588
			151	99.70744		99.91243	99.94443
		rubah ▾	478	99.98243		99.99247	99.99496
			551	99.99562		99.99812	99.99924
			637	99.99113		99.98290	99.99846

Gambar 4.2.1 Perbandingan Penggunaan Variasi Derajat pada CBIR Tekstur

### c. compare(chedec1, chedec2)

Fungsi ini menerima dua parameter dalam bentuk vektor, yakni vektor yang telah dihasilkan pada `process_chedec(image)`. Fungsi ini membandingkan vektor `chedec1` dan `chedec2` untuk mendapatkan persentase kemiripan dengan metode cosine similarity dan mengembalikan besar persentase kemiripan.

d. `cbir_texture(img1, img2)`

Fungsi ini mengimplementasikan semua fungsi yang telah dibuat. Fungsi ini menerima dua parameter gambar. Kedua gambar kemudian diambil vektornya dengan menggunakan `process_chedec(image)`, lalu membandingkannya dengan `compare(chedec1, chedec2)`.

Pada bagian kedua yakni website, pengembangan dilakukan dengan menggunakan salah satu *framework* yang ada pada bahasa Python, yaitu Django. Pemilihan Django dilakukan untuk memastikan struktur yang kokoh, kemudahan pengembangan, dan efisiensi kerja. Dalam framework ini, kami membuat satu app yang kami beri nama 'cbir', yang berperan sebagai inti dari sistem temu balik gambar berbasis konten (CBIR). App ini dirancang dengan beberapa view yang dibangun sesuai kebutuhan fungsional, memberikan kerangka kerja yang terstruktur untuk website ini.

Salah satu view utama adalah 'Home', yang menjadi tampilan utama dari website dan memberikan pengguna gambaran komprehensif tentang fitur utama dan informasi penting lainnya. 'Upload\_Search' bertindak sebagai penangan untuk unggahan gambar yang ingin dicari, memungkinkan pengguna untuk dengan mudah memasukkan gambar ke dalam sistem. View 'Search\_CBIR' merupakan inti dari metode CBIR, menangani proses pencarian gambar berbasis konten visual dan perhitungan similarity.

Untuk memperluas fungsionalitas, kami juga merancang view 'Upload\_Dataset', yang memungkinkan pengguna untuk mengunggah folder dataset, memberikan lebih banyak opsi dalam melakukan pencarian. View 'Refresh' memberikan kemudahan bagi pengguna untuk me-refresh halaman dengan menghapus dataset yang ada, memberikan kontrol penuh terhadap konten yang digunakan.

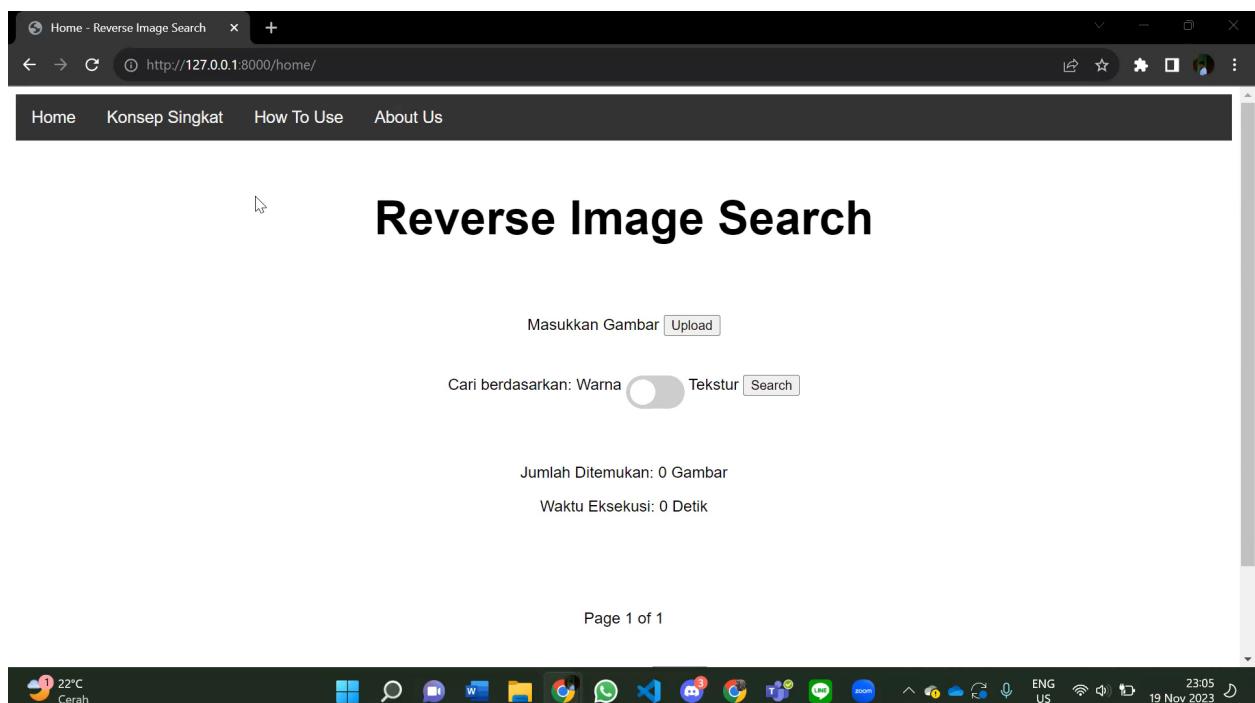
Selain itu, kami menyajikan halaman-halaman tambahan seperti 'Konsep\_Singkat', 'How\_to\_Use', dan 'About\_Us' melalui view yang terkait. Ini

memberikan informasi tambahan kepada pengguna tentang konsep mesin pencarian, panduan penggunaan, dan penjelasan tentang tim kami.

Dalam struktur database, dua model utama dibuat: 'ImageSearch', yang menyimpan gambar yang menjadi objek pencarian, dan 'ImageDataSet', yang menangani gambar-gambar pada dataset. Model ini dirancang dengan atribut ImageField untuk menyimpan gambar dan FloatField untuk nilai cosine similarity, memastikan penyimpanan dan pengambilan data yang efisien.

Proses utama CBIR, seperti unggahan gambar, perhitungan cosine similarity, dan manajemen dataset, diimplementasikan dalam app 'cbir' dan di-import ke dalam views. Dengan demikian, struktur pengembangan website ini tidak hanya memastikan antarmuka yang menarik tetapi juga memberikan fondasi yang solid untuk fungsionalitas proyek. Melalui pendekatan ini, kami bertujuan untuk menciptakan sistem temu balik gambar yang efisien dan mudah digunakan.

### 3. Penjelasan Tata Cara Penggunaan Program



Website temu balik gambar yang kami buat menawarkan pengalaman pengguna yang sangat interaktif dan komprehensif. Melalui desain antarmuka Front-End mudah

dinavigasi, pengguna dapat dengan mudah mengakses berbagai fitur yang disediakan untuk melakukan pencarian gambar berbasis Aljabar Vektor. Pada halaman utama website, pengguna dapat melihat fitur utama dari temu balik gambar ini. Dengan tata letak yang bersih dan judul yang jelas, pengguna dapat dengan cepat mengidentifikasi tujuan utama dari sistem ini.

Tombol "Insert Image" dirancang untuk mempermudah pengguna dalam memasukkan gambar yang ingin dicari. Setelah gambar dipilih, tampilan gambar terpilih akan ditampilkan dengan jelas, memberikan gambaran visual tentang data yang akan dijadikan acuan pencarian. Keleluasaan pengguna dalam memilih kriteria pencarian menjadi salah satu fitur utama. Melalui tombol "Toggle," pengguna dapat beralih antara pencarian berbasis warna atau tekstur, sesuai dengan kebutuhan analisis visual yang diinginkan.

Proses pencarian diinisiasi dengan menekan tombol "Search." Hasil pencarian akan muncul dalam bentuk kumpulan gambar yang informatif dan relevan dengan gambar query. Pengguna dapat langsung melihat jumlah hasil yang ditemukan dan waktu eksekusi, memberikan pemahaman instan tentang kinerja sistem. Setiap gambar hasil pencarian dilengkapi dengan informasi lebih lanjut, termasuk persentase kesamaan terhadap gambar query. Hal ini membantu pengguna untuk lebih memahami tingkat relevansi setiap hasil, memungkinkan pengambilan keputusan yang lebih akurat. Program ini juga memfasilitasi pengguna dengan tombol "Upload Dataset" untuk mengunggah folder yang berisi dataset gambar tambahan. Ini memperluas cakupan pencarian dan meningkatkan kemungkinan menemukan hasil yang lebih bervariasi.

Menu halaman menyediakan akses cepat ke informasi yang mungkin diperlukan pengguna. Ringkasan konsep mesin pencarian, petunjuk pengguna ("How to Use"), dan informasi tentang kelompok kami ("About Us") dapat diakses dengan mudah melalui menu tersebut.

Dengan antarmuka pengguna yang intuitif dan fitur-fitur yang dirancang dengan cermat, program temu balik gambar kami memberikan pengalaman pengguna yang

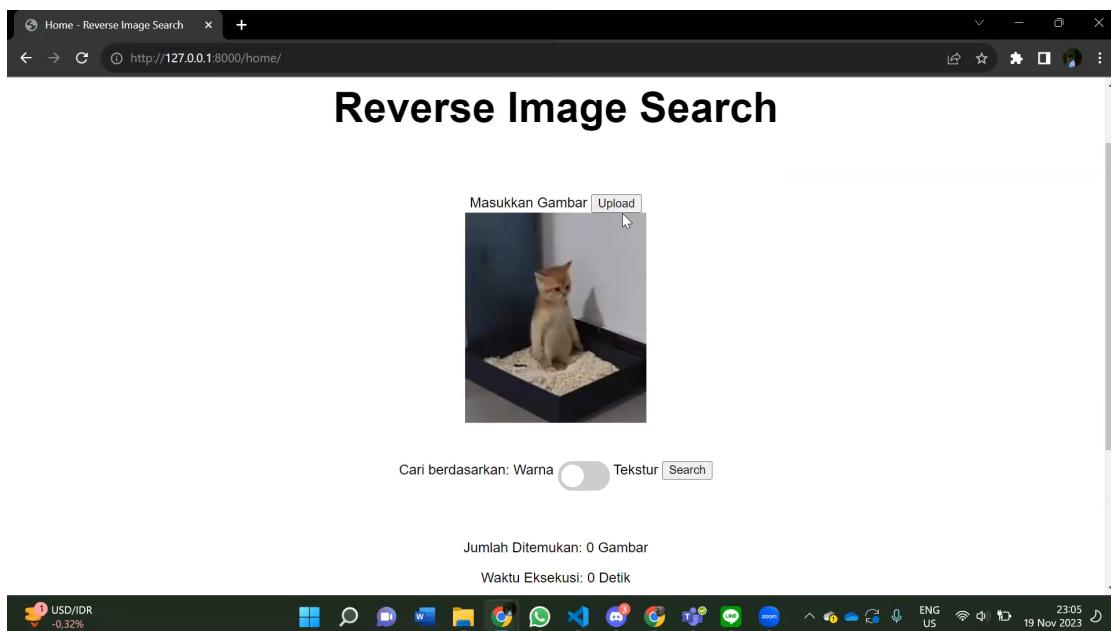
memuaskan dan informatif. Melalui langkah-langkah ini, pengguna dapat menjelajahi dan memanfaatkan kemampuan canggih sistem ini dengan mudah dan efektif.

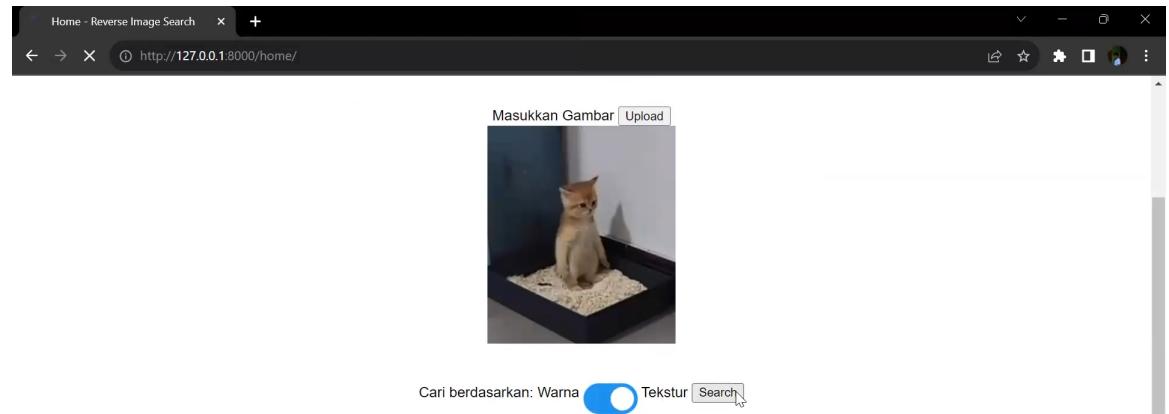
#### 4. Hasil Pengujian

Hasil pengujian yang didapatkan dengan menggunakan sebuah dataset berisikan 480 gambar, website yang telah kami buat dapat melakukan pencarian pada dataset dengan CBIR berbasis warna dengan waktu 208 detik, sedangkan untuk CBIR berbasis tekstur dengan waktu sekitar 22 detik.

Ada pula hasil yang lebih bervariasi dihasilkan oleh CBIR berbasis warna. Pada CBIR berbasis tekstur, hampir semua hasil kemiripan dengan dataset menyentuh angka di atas 90-an%.

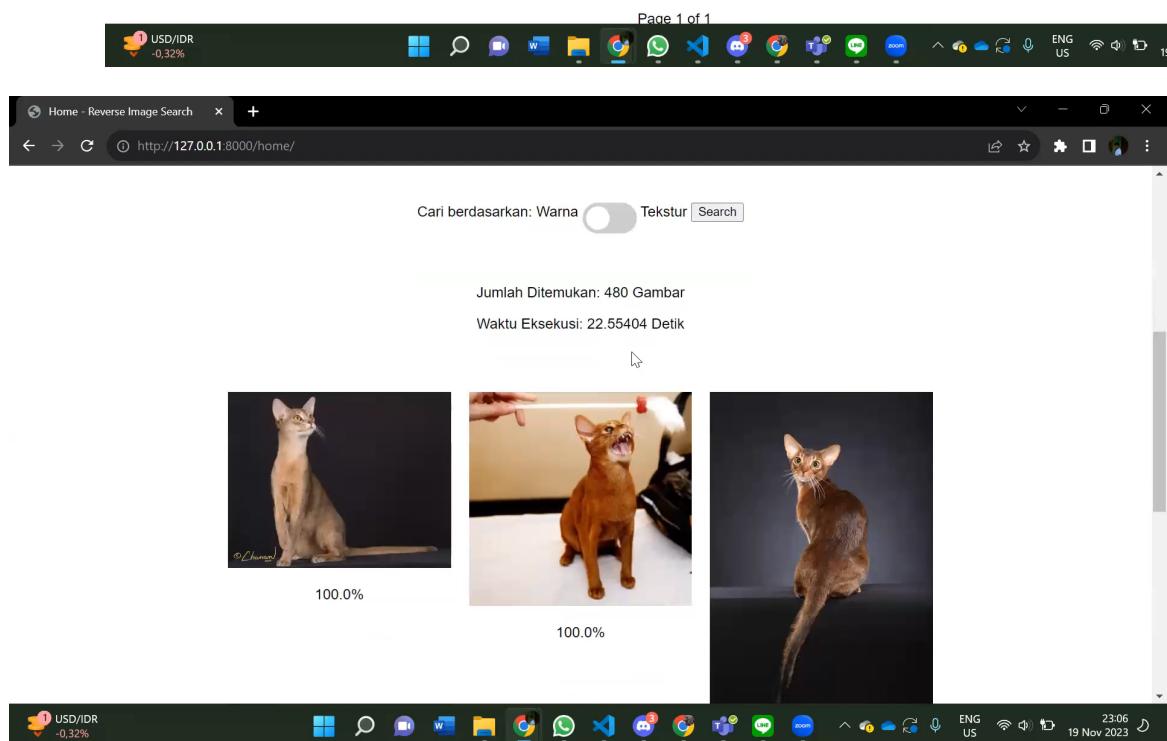
#### Pengujian pertama : Inputan berupa kucing yang sedang berdiri

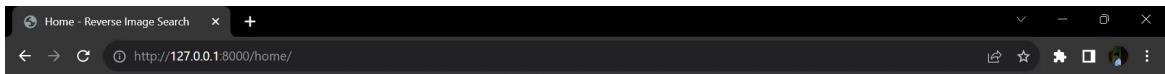




Jumlah Ditemukan: 0 Gambar

Waktu Eksekusi: 0 Detik





Jumlah Ditemukan: 480 Gambar

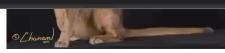
Waktu Eksekusi: 22.55404 Detik



100.0%



USD/IDR  
-0.32%



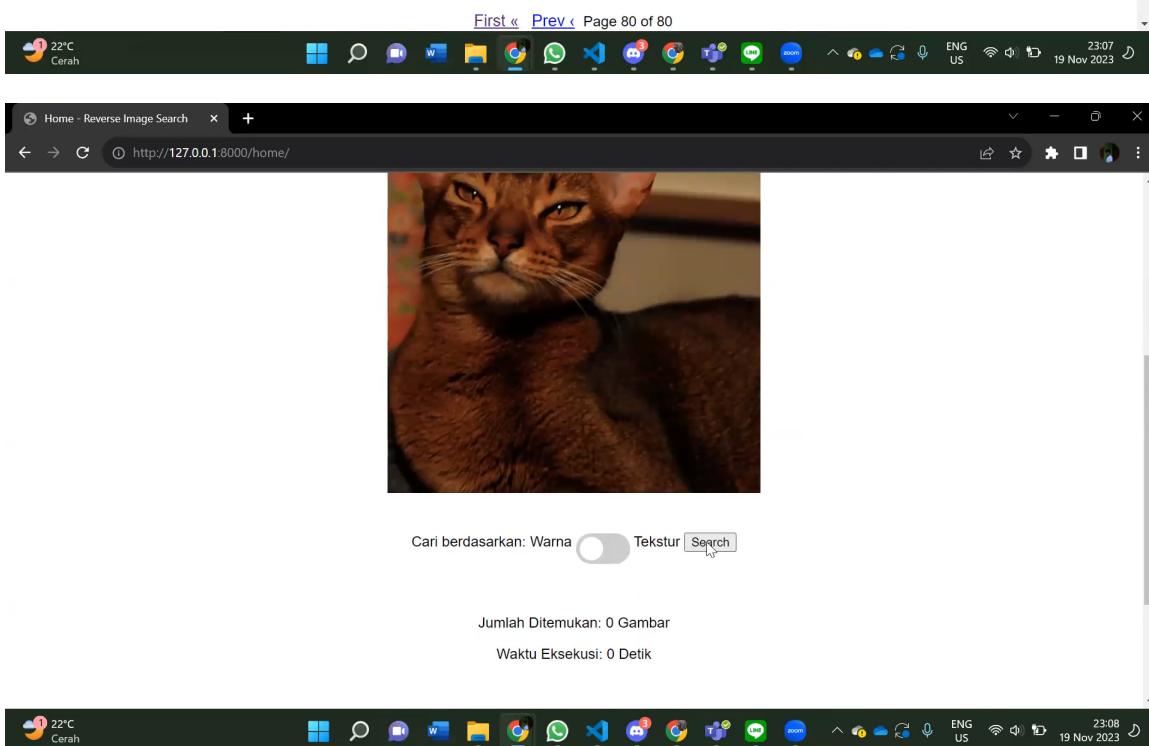
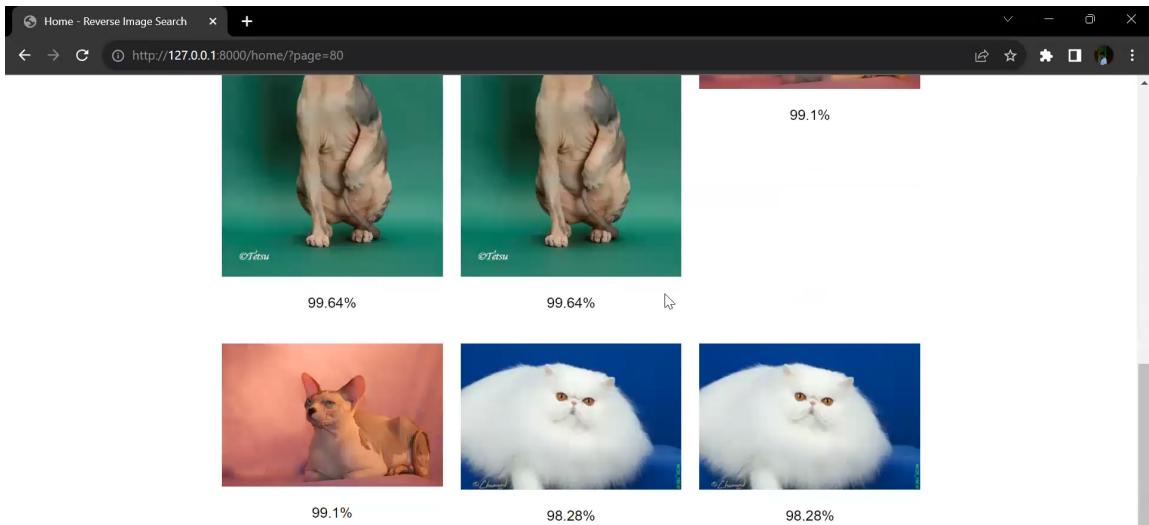
100.0%

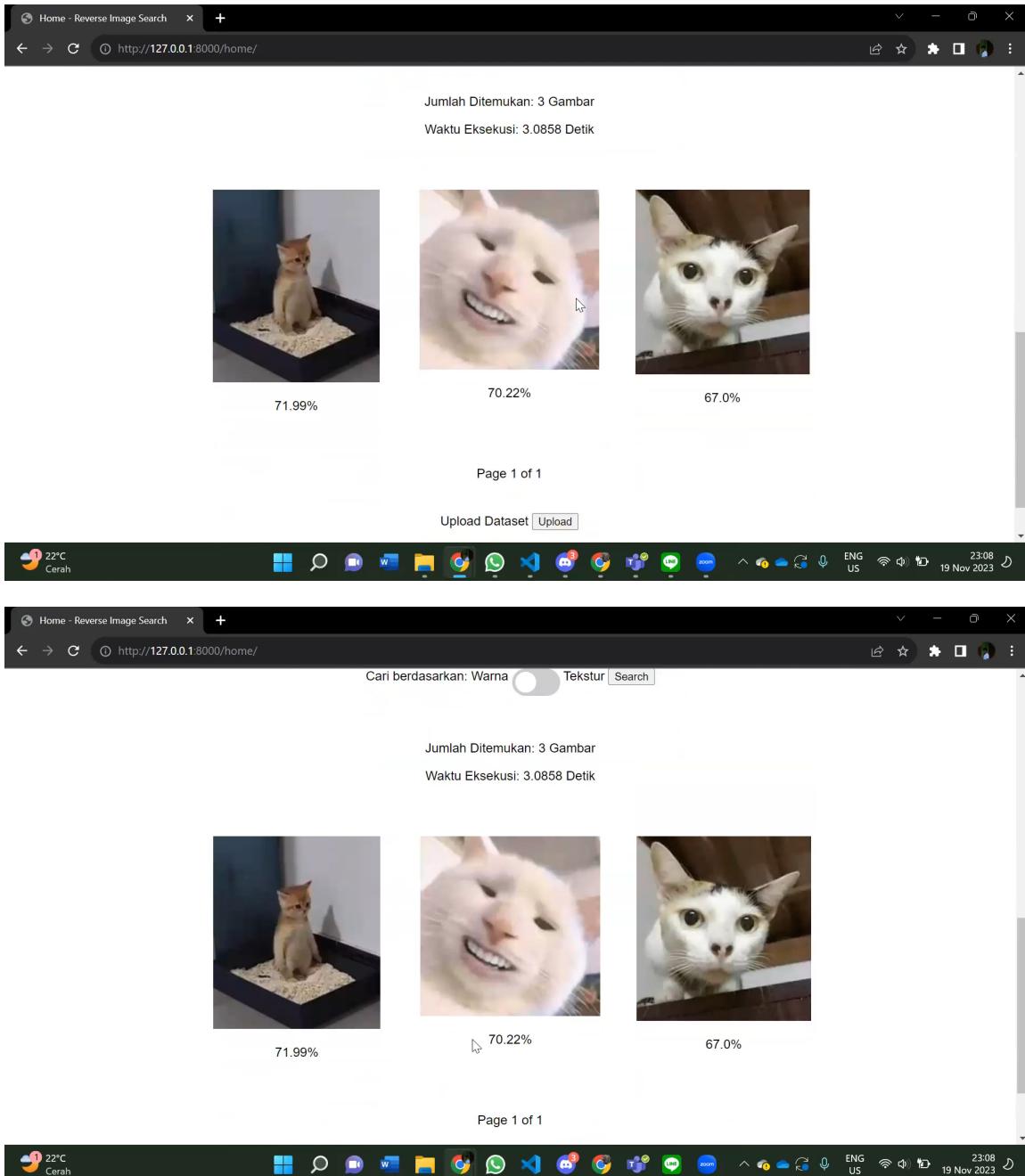


100.0%

100.0%







## 5. Analisis dari Desain Solusi Algoritma Pencarian yang Diimplementasikan pada Setiap Pengujian yang Dilakukan.

Dari segi waktu, fitur tekstur lebih baik dibandingkan fitur warna. Proses ekstraksi fitur warna dalam Sistem Temu Kembali Citra Berbasis Warna (CBIR) seringkali memerlukan waktu lebih lama dibandingkan dengan ekstraksi fitur tekstur. Hal

ini disebabkan oleh sifat kompleksitas dan dimensi yang tinggi pada representasi warna. Pada umumnya, fitur warna melibatkan informasi dari setiap kanal warna, seperti merah, hijau, dan biru (RGB), atau komponen warna lainnya seperti nilai, saturasi, dan hue (HSV). Proses ekstraksi fitur warna melibatkan transformasi matriks dan perhitungan statistik yang lebih rumit untuk menghasilkan vektor fitur warna yang mencerminkan karakteristik citra. Seiring dengan itu, citra berwarna dapat memiliki dimensi warna yang tinggi, yang memerlukan pengolahan data yang lebih intensif. Di samping itu, sejumlah besar warna yang mungkin ada dalam citra dapat memperlambat proses pencarian dan perbandingan citra dalam basis data. Sebaliknya, ekstraksi fitur tekstur cenderung lebih cepat karena sering melibatkan analisis pola lokal dan operasi filter yang dapat dihitung secara efisien. Oleh karena itu, meskipun fitur warna memberikan informasi visual yang kaya, pengolahan dan analisis fitur warna dalam CBIR memerlukan waktu lebih lama dibandingkan dengan fitur tekstur.

Dalam segi hasil kemiripan, Sistem Temu Kembali Citra Berbasis Warna (CBIR) dengan fitur warna cenderung memberikan hasil yang lebih akurat dalam hal kemiripan dibandingkan dengan fitur tekstur. Hal ini dapat dijelaskan oleh kemampuan fitur warna dalam menangkap informasi visual yang langsung terkait dengan persepsi manusia terhadap kesamaan citra. Fitur warna mencakup aspek visual utama yang mudah diidentifikasi oleh mata manusia, seperti perbedaan warna dan distribusi warna dalam citra. Oleh karena itu, sistem berbasis warna dapat secara efektif membedakan citra yang memiliki karakteristik warna yang serupa, sehingga memberikan hasil pencarian yang lebih relevan. Sebaliknya, fitur tekstur dapat lebih sulit untuk menangkap kesamaan citra karena interpretasinya seringkali bergantung pada representasi pola atau struktur dalam citra. Kekurangan ini dapat menyebabkan hasil pencarian yang kurang akurat, terutama ketika citra memiliki tekstur yang mirip namun dengan perbedaan pola atau orientasi tertentu. Fitur warna, dengan fokus pada perbedaan warna yang dapat dikenali dengan mudah, cenderung memberikan hasil yang lebih tajam dalam menangkap kemiripan visual antar citra. Namun demikian, keakuratan CBIR tidak hanya bergantung pada jenis fitur yang digunakan, tetapi juga pada kualitas ekstraksi fitur, metode pencocokan, dan representasi data yang diimplementasikan. Oleh karena itu, pemilihan fitur harus

disesuaikan dengan karakteristik khusus dari set data dan tujuan aplikasi tertentu untuk mencapai hasil yang optimal.

Misalnya, apakah pembangunan CBIR berdasarkan fitur warna lebih baik dari tekstur pada kasus-kasus tertentu beserta analisis mengenai mengapa hal itu bisa terjadi jika benar.

## **BAB V**

### **KESIMPULAN**

#### 1. Kesimpulan

Dalam Tugas Besar 2 ini, kami berhasil mengimplementasikan sebuah website sistem temu balik gambar berbasis Aljabar Vektor. Proyek ini memanfaatkan konsep Aljabar Vektor yang kami pelajari selama semester 3 dalam mata kuliah Aljabar Linier dan Geometri, serta pengetahuan tambahan yang kami peroleh secara mandiri. Beberapa konsep yang diterapkan melibatkan pengolahan vektor untuk mewakili dan menganalisis konten visual gambar, seperti warna dan tekstur.

Proses utama dalam proyek ini mencakup pengelolaan dataset gambar, pemilihan gambar uji, dan penerapan algoritma pencarian berbasis konten (Content-Based Image Retrieval). Kami memastikan kemudahan interaksi dengan pengguna melalui desain antarmuka depan (Front-End) yang menarik, dengan berbagai fitur seperti tombol untuk memasukkan gambar, opsi pemilihan kriteria pencarian, tampilan hasil pencarian yang informatif, dan waktu eksekusi yang ditampilkan secara real-time.

Proyek ini dibangun dengan menggunakan bahasa pemrograman yang populer, dan struktur penyimpanan yang terorganisir dalam tiga folder: src (source code), doc (dokumentasi), dan test (dataset). Metode-metode yang diterapkan termasuk konsep Aljabar Vektor, pengolahan gambar dengan OpenCV, dan penggunaan numpy untuk manipulasi matriks. Dengan demikian, proyek ini tidak hanya mencakup aspek implementasi Aljabar Vektor, tetapi juga menggabungkan berbagai alat dan konsep pemrograman untuk menciptakan sebuah sistem yang efisien dan mudah digunakan dalam temu balik gambar berbasis konten.

#### 2. Saran

Dari proses penggerjaan tugas besar ini, kami memiliki banyak kendala selama mengerjakan seperti waktu eksekusi yang cukup lama, dan terutama di pembuatan website. Oleh karena itu, kami memiliki beberapa saran agar tugas ini bisa menjadi lebih baik, yaitu

- a. Optimasi algoritma dengan bahasa pemrograman lain menjadi lebih cepat.
- b. Pertimbangkan penggunaan framework Front-End yang ringan dan efisien

### 3. Komentar atau Tanggapan

Kami mengapresiasi penilaian positif terhadap implementasi Aljabar Vektor dalam proyek temu balik gambar kami. Dengan segala kerendahan hati, kami juga mengakui bahwa kendala waktu eksekusi dan penampilan website adalah suatu tantangan yang perlu kami selesaikan.

Kami sepenuhnya menyadari bahwa waktu eksekusi yang cukup lama merupakan aspek kritis yang perlu diperbaiki. Kami setuju bahwa optimasi algoritma, termasuk pertimbangan untuk menggunakan bahasa pemrograman yang lebih efisien, adalah langkah yang tepat. Kami akan melakukan evaluasi mendalam terhadap algoritma yang digunakan dan mencari solusi terbaik untuk meningkatkan kinerja secara keseluruhan.

Saran untuk mempertimbangkan penggunaan framework Front-End yang lebih ringan dan efisien juga sangat relevan. Kami akan melakukan penelitian lebih lanjut untuk memilih framework yang lebih cocok dengan kebutuhan proyek kami. Responsivitas antarmuka pengguna adalah prioritas kami, dan kami berkomitmen untuk meningkatkannya agar pengguna dapat merasakan kenyamanan dan efisiensi dalam penggunaan website kami.

Kami sangat menghargai masukan konstruktif yang diberikan, dan kami yakin bahwa dengan implementasi saran-saran tersebut, proyek kami dapat mencapai tingkat kualitas yang lebih tinggi lagi nanti. Semua tanggapan ini akan menjadi pedoman berharga bagi kami dalam mengembangkan dan memperbaiki sistem temu balik gambar berbasis Aljabar Vektor ini.

### 4. Refleksi

Tugas Besar 2 dalam mata kuliah IF2123 Aljabar Linier dan Geometri Semester I Tahun 2023/2024 merupakan sebuah perjalanan menarik dan penuh pembelajaran. Melalui tugas ini, kami mengimplementasikan sebuah sistem temu balik gambar berbasis Aljabar Vektor dalam bentuk website. Pendekatan ini membuktikan relevansi Aljabar Vektor dalam pemrosesan data dan pencarian informasi di era digital. Aljabar vektor menjadi landasan untuk menggambarkan dan menganalisis data dengan fokus pada klasifikasi berbasis konten (Content-Based Image Retrieval atau CBIR).

Tantangan utama dalam proyek ini adalah memahami dan mengoptimalkan algoritma pencarian fitur visual dan berlangsung dengan waktu eksekusi yang cepat, serta

merancang antarmuka pengguna yang efisien. Kendala ini diatasi melalui eksplorasi mendalam mengenai karakteristik visual gambar, serta kolaborasi tim yang sinergis dalam menjalankan tugas masing-masing.

Selesainya Tugas Besar 2 ini bukan hanya mencerminkan implementasi sistem temu balik gambar, tetapi juga menunjukkan kemampuan kami dalam menghadapi tantangan pemrograman dan menerapkan konsep Aljabar Vektor dalam situasi praktis. Kami yakin proyek ini akan memberikan dampak positif terhadap pemahaman kami terkait Aljabar Vektor dan aplikasinya dalam memecahkan permasalahan dunia nyata.

## 5. Ruang Perbaikan atau Pengembangan

### a. Optimasi Algoritma

Sebagai respons terhadap kendala waktu eksekusi, kami mengidentifikasi optimasi algoritma sebagai fokus utama pengembangan. Kami akan melakukan evaluasi mendalam terhadap setiap langkah algoritma, mencari peluang untuk meningkatkan efisiensi tanpa mengorbankan kualitas temu balik gambar. Penerapan algoritma dengan kompleksitas waktu yang lebih rendah atau pemilihan strategi komputasi yang lebih efisien menjadi langkah kunci dalam ruang perbaikan ini.

### b. Pemilihan Bahasa Pemrograman

Kami akan mempertimbangkan saran untuk memilih bahasa pemrograman yang lebih cepat dan efisien. Pengujian lebih lanjut akan dilakukan untuk mengevaluasi performa berbagai bahasa dan memilih yang paling sesuai dengan karakteristik proyek kami. Pemilihan bahasa yang tepat dapat memberikan dampak positif terhadap waktu eksekusi dan kinerja keseluruhan.

### c. Framework Front-End

Dalam hal antarmuka pengguna, kami akan melakukan penelitian lebih lanjut terkait pemilihan framework Front-End. Kami akan mencari solusi yang ringan dan dioptimalkan untuk kebutuhan proyek, meningkatkan responsivitas dan pengalaman pengguna. Penyesuaian antarmuka dengan desain yang efisien akan menjadi langkah penting dalam ruang perbaikan ini.

### d. Pengukuran dan Evaluasi Kinerja

e. Kami akan meningkatkan metode pengukuran kinerja sistem, termasuk waktu eksekusi dan responsivitas antarmuka. Melalui pemantauan yang lebih ketat, kami dapat mengidentifikasi area-area yang memerlukan perbaikan lebih lanjut dan mengukur dampak perubahan yang kami terapkan.

f. Pengembangan Fitur Tambahan

Selain perbaikan inti terkait kinerja, kami akan mempertimbangkan pengembangan fitur tambahan yang dapat meningkatkan nilai tambah sistem temu balik gambar. Ini dapat mencakup integrasi teknologi terbaru, pemrosesan gambar yang lebih canggih, atau fitur pengguna yang lebih lanjut untuk meningkatkan kebergunaan sistem.

Ruang perbaikan ini akan menjadi panduan untuk pengembangan selanjutnya, dan kami berkomitmen untuk terus meningkatkan proyek ini agar dapat memenuhi standar tinggi dalam temu balik gambar berbasis konten. Dengan adanya ruang perbaikan ini, kami percaya bahwa proyek ini akan menjadi lebih inovatif dan efisien.

## DAFTAR PUSTAKA

Munir, Rinaldi. (2023). *IF2123 Aljabar Geometri - Semester I Tahun 2023/2024*.

Institut Teknologi Bandung. Diakses pada 6 Oktober 2022, dari  
<https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/algeo23-24.htm>

Stack Exchange. (2013). *RGB to HSV color conversion algorithm*.

Retrieved from  
<https://www.sciencedirect.com/science/article/pii/S0895717710005352>

ScienceDirect. (2010). *Content-based image retrieval using color and texture fused features*.

Retrieved from  
<https://math.stackexchange.com/questions/556341/rgb-to-hsv-color-conversion-algorithm>

Muhammad, Yunus. (2020). *Feature Extraction: Gray Level Co-occurrence Matrix (GLCM)*.

Retrieved from  
<https://yunusmuhammad007.medium.com/feature-extraction-gray-level-co-occurrence-matrix-glcm-10c45b6d46a1>

Kurt, Gorkem. (2020). *Content Based Image Retrieval With Django*.

Retrieved from  
<https://medium.com/@gorkemkurt/content-based-image-retrieval-with-django-3906aead2801>

## **LAMPIRAN**

### **A. Link Repository**

<https://github.com/ibrahim-rasyid/Algeo02-22018.git>

### **B. Link Video Bonus (Youtube)**