LAPORAN TUGAS BESAR II

IF2123 ALJABAR LINEAR DAN GEOMETRI

"Aplikasi Aljabar Vektor dalam Sistem Temu Balik Gambar"

Dosen:

Ir. Rila Mandala, M. Eng, Ph. D.

Arrival Dwi Sentosa, S. Kom, M. T.

Kelompok 45:

13522125 Satriadhikara Panji Yudhistira

13522140 Yasmin Farisah Salma

13522145 Farrel Natha Saskoro

PROGRAM STUDI TEKNIK INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

SEMESTER I TAHUN 2023/2024

KATA PENGANTAR

Puji syukur kepada Tuhan Yang Maha Esa penulis ucapkan atas keberhasilan dalam menyelesaikan Tugas Besar II IF2123 Aljabar Linear dan Geometri ini.

Laporan ini merangkum seluruh fase pengembangan algoritma program dan *website*, mulai dari konsepsi ide hingga realisasi teknis. Kami menguraikan bagaimana kami mengimplementasikan ekstraksi fitur penting seperti warna dan tekstur, dan bagaimana fitur-fitur tersebut dikonversi menjadi vektor numerik yang siap dibandingkan dengan koleksi gambar yang luas menggunakan algoritma pencocokan yang efisien.

Lebih dari sekadar tugas akademis, proyek ini adalah sebuah perjalanan pembelajaran tentang bagaimana konsep aljabar linear dapat diaplikasikan dalam memecahkan masalah dunia nyata—dalam hal ini, menciptakan metode pencarian gambar yang canggih dan *user-friendly*. Kami berkomitmen untuk menyajikan sebuah *platform* yang tidak hanya akurat dalam mengidentifikasi kesamaan visual, tetapi juga menyediakan pengalaman pengguna yang intuitif dan memuaskan. Pengembangan *website* ini juga merupakan hasil atas kolaborasi dan kerja tim kami yang di dalamnya terjalin sinergi antara pengetahuan teoretis dan keterampilan teknis.

Dengan dukungan dari dosen, asisten kelompok, input dari rekan mahasiswa, dan berbagai sumber daya yang tersedia, kami menghadirkan sebuah *website reverse image search* yang kami harap akan bekerja dengan baik dan efisien. Akhir kata, kami ucapkan terima kasih kepada semua yang telah berpartisipasi dan mendukung. Kami berharap agar tugas besar ini dapat memenuhi mata kuliah Aljabar Linear dan Geometri dan bermanfaat untuk pengembangan yang lebih maju.

Sumedang, 16 November 2023,

Kelompok 45.

DAFTAR ISI

BAB I
DESKRIPSI MASALAH

1.1. Deskripsi Masalah

Dalam era digital yang terus berkembang ini, kita menyaksikan peningkatan eksponensial dalam produksi dan penyimpanan gambar. Fenomena ini menyebar luas di berbagai sektor, termasuk kehidupan kita secara pribadi yang mengabadikan kenangan melalui foto, sektor kesehatan yang bergantung pada gambar medis untuk diagnosis, dunia akademik yang berinteraksi dengan ilustrasi ilmiah untuk penelitian, dan industri komersial yang menggunakan gambar untuk mempromosikan produk dan layanan. Pada intinya, setiap gambar ini membawa data visual yang apabila dikelola dan diakses dengan tepat, dapat memberikan nilai yang luar biasa.

Dalam menghadapi kompleksitas dan volume gambar yang terus meningkat, sistem temu balik gambar merupakan solusi yang tepat. Sistem ini tidak hanya memfasilitasi proses pencarian dan akses cepat ke koleksi gambar yang luas, tetapi juga mengoptimalisasi pengelolaan aset digital tersebut. Fitur ini menjadi sangat *powerful* dalam konteks di mana kecepatan dan efisiensi dalam menemukan informasi visual menjadi kunci.

Maka dari itu, ntuk menjawab tantangan dalam Tugas Besar ini, kami telah mengembangkan sebuah sistem temu balik gambar yang inovatif melalui penerapan Aljabar Vektor yang terintegrasi dalam suatu *website*. Pendekatan ini adalah inti dari pemrosesan data dan pencarian informasi yang canggih. Dengan menggunakan teknik Content-Based Image Retrieval (CBIR), sistem kami mengidentifikasi dan mengklasifikasikan gambar berdasarkan karakteristik kontennya, yakni warna atau tekstur, sehingga memungkinkan pengguna untuk menggali dan menavigasi melalui kumpulan data visual mereka dengan cara yang intuitif dan efisien.

BAB II
LANDASAN TEORI

2.1 Dasar Teori

Content-Based Image Retrieval (CBIR) merupakan teknologi yang memungkinkan pencarian dan pengambilan gambar berdasarkan analisis konten visualnya, tanpa bergantung pada anotasi teks. CBIR bekerja melalui ekstraksi fitur penting dari gambar—seperti warna, tekstur, dan bentuk—yang selanjutnya diwakili dalam bentuk vektor atau deskripsi numerik. Fitur-fitur ini kemudian dibandingkan dengan dataset gambar yang ada menggunakan algoritma pencocokan khusus. Hasil dari proses pencocokan ini digunakan untuk mengurutkan dan menampilkan gambar-gambar yang paling mirip dengan query yang diajukan pengguna. Proses CBIR membantu pengguna dalam mengakses dan mengeksplorasi koleksi gambar dengan cara yang lebih efisien, karena tidak memerlukan pencarian berdasarkan teks atau kata kunci, melainkan berdasarkan kesamaan nilai citra visual antara gambar-gambar tersebut.

Dalam pencarian berbasis konten visual pada sistem Content-Based Image Retrieval (CBIR), analisis warna merupakan faktor kunci. Metode ini memulai dengan konversi data gambar dari format RGB ke ruang warna HSV. Alasan utama menggunakan HSV adalah kemampuannya untuk mengakomodir gambar dengan latar belakang putih yang sering terjadi pada dataset umum. Setelah konversi, frekuensi kemunculan berbagai warna dalam gambar dihitung dan direpresentasikan dalam bentuk histogram warna. Histogram ini bisa bersifat global, yang mempertimbangkan keseluruhan gambar, atau berbasis blok, di mana gambar dibagi menjadi segmen-segmen lebih kecil dan histogram dihitung untuk masing-masing segmen tersebut.

Perbandingan antara histogram dari gambar yang dicari dengan histogram dari gambar dalam dataset dilakukan melalui penghitungan kesamaan kosinus, yang memberikan suatu nilai kemiripan antara dua gambar berdasarkan distribusi warna mereka.

Sementara itu, analisis tekstur dalam CBIR dilakukan melalui penggunaan matriks kejadian bersama, atau co-occurrence matrix. Ini dimulai dengan konversi gambar ke skala abu-abu, karena warna dianggap tidak memberikan kontribusi signifikan dalam analisis tekstur. Matriks kejadian bersama kemudian dibangun untuk menangkap hubungan tekstural antar piksel, yang memungkinkan identifikasi fitur tekstur seperti kontras, entropi, dan homogenitas. Fitur-fitur ini diwakili dalam bentuk vektor yang digunakan untuk mengukur kemiripan tekstur antara dua gambar. Seperti halnya dengan warna, pengukuran kemiripan tekstur juga menggunakan konsep kesamaan kosinus. Proses ini menghasilkan perbandingan yang efektif antara tekstur gambar yang dicari dengan yang ada dalam database, memungkinkan sistem CBIR untuk menemukan gambar dengan tekstur yang serupa.

2.2 Pengembangan Website

Pengembangan sebuah website secara umum merupakan proses yang kompleks dan detail, dimulai dengan fase perencanaan yang melibatkan pemahaman mendalam tentang kebutuhan dan tujuan. Dalam fase ini, tim pengembang menentukan fungsi-fungsi utama yang harus diintegrasikan, dan merencanakan arsitektur informasi yang akan menuntun pengalaman pengguna. Setelah perencanaan, tahap desain dijalankan dengan menciptakan prototipe visual yang merefleksikan identitas merek dan memastikan usability yang baik. Desainer UX/UI berkolaborasi erat untuk menghasilkan desain antarmuka yang intuitif dan menarik. Dengan desain sebagai panduan,

pengembangan *website* dibagi menjadi dua area utama: pengembangan *frontend* dan *backend*.

Frontend berkaitan dengan bagaimana website ditampilkan kepada pengguna, memanfaatkan teknologi seperti HTML, CSS, dan JavaScript, serta *framework* seperti React atau Angular untuk membuat interaksi yang dinamis. Sementara itu, *backend* fokus pada server, aplikasi, dan *database*, memastikan bahwa logika bisnis dijalankan dengan benar dan data diproses secara aman. Bahasa pemrograman seperti Python, Ruby, atau PHP sering digunakan bersama dengan sistem manajemen database seperti MySQL atau PostgreSQL.

Setelah kedua bagian ini dikembangkan, proses integrasi dimulai, memastikan bahwa *frontend* dan *backend* bekerja dengan baik dan harmonis. Pengujian menyeluruh dilakukan untuk mengidentifikasi bug atau masalah kegunaan, meliputi pengujian fungsional, pengujian beban, dan pengujian keamanan. Ketika semua komponen telah terverifikasi dan diuji, website siap untuk diluncurkan. Namun, pengembangan website tidak berakhir di sini; pemeliharaan berkelanjutan diperlukan untuk memperbarui konten, mengatasi masalah keamanan, dan menambahkan fitur baru sesuai dengan feedback pengguna dan perkembangan teknologi. Proses ini menjamin bahwa website tetap relevan, aman, dan efektif dalam memenuhi tujuan bisnis.

Dalam konteks pembuatan *website* untuk Content-Based Image Retrieval (CBIR), kami memanfaatkan Next.js dan Python. Dengan React.js yang digunakan untuk membangun *user interface frontend*, pengguna dapat berinteraksi dengan website secara dinamis. React.js memudahkan pengelolaan *state* pada aplikasi dan memastikan tampilan yang cepat serta interaktif. Untuk bagian *backend*, Python digunakan karena kemampuan integrasinya yang luas dengan algoritma yang esensial untuk fungsi CBIR. Dalam hal ini, *backend* akan menangani logika CBIR, termasuk ekstraksi fitur

gambar, pencocokan, dan pengambilan hasil pencarian dari *database*. Berikut adalah tahapan pengembangan *website* CBIR:

1) Perencanaan
   Menentukan fitur-fitur inti seperti upload gambar, ekstraksi fitur, pencocokan gambar, dan tampilan hasil.
2) Desain
   Menggunakan alat desain untuk membuat wireframe dan desain *interface* yang akan menentukan bagaimana komponen-komponen dari Next.js dan React.js akan diintegrasikan.
3) Pengembangan Frontend
   Menggunakan React.js untuk membangun komponen-komponen interaktif dan halaman *website*.
4) Pengembangan Backend
   Menulis *script* Python untuk proses CBIR, membuat API yang dipanggil oleh frontend.
5) Integrasi
   Menghubungkan *frontend* dengan backend melalui API, memastikan bahwa data yang dikirim dan diterima berjalan dengan lancar.
6) Pengujian
   Melakukan pengujian secara menyeluruh untuk memastikan bahwa semua fitur berfungsi dengan benar dan website responsif di berbagai perangkat.
7) Deployment
   Setelah pengujian selesai dan bug diperbaiki, *website* kemudian di-*deploy* ke server.

BAB III
ANALISIS PEMECAHAN MASALAH

3.1 Langkah-Langkah Pemecahan Masalah

Pada bagian ini, kami akan menguraikan langkah-langkah pemecahan masalah yang kami terapkan dalam pengembangan website yang memungkinkan pembandingan gambar dengan penerapan GLCM untuk parameter warna dan tekstur. Masalah yang dihadapi adalah pengembangan sebuah website yang mampu membandingkan gambar dengan menggunakan GLCM untuk menganalisis parameter warna dan tekstur dari gambar-gambar yang diunggah.

Identifikasi dan penyelesaian langkah-langkah yang diperlukan dalam pengembangan website, termasuk:

- Perancangan antarmuka pengguna yang memungkinkan pengguna untuk mengunggah dan membandingkan gambar.
- Implementasi algoritma GLCM untuk mengekstrak parameter warna dan tekstur.
- Pengembangan perangkat lunak untuk membandingkan hasil GLCM dari gambar yang diunggah.
- Pengujian dan evaluasi kinerja sistem.

3.2 Proses Pemetaan Masalah

Kami akan memetakan masalah pengembangan website ke dalam elemen-elemen aljabar geometri untuk memahami hubungan antara komponen-komponen yang terlibat dalam penerapan GLCM sebagai berikut:

- Pemetaan konsep warna dalam gambar ke dalam dimensi aljabar geometri.
- Representasi tekstur gambar sebagai vektor dalam ruang aljabar.

- Penggunaan GLCM sebagai matriks untuk merepresentasikan hubungan antara intensitas piksel.

3.3 Ilustrasi Kasus dan Penyelesaian

Ilustrasi Kasus:

- Seorang pengguna mengunggah dua gambar dengan konten yang serupa.
- Pengguna memilih dengan parameter apa kedua gambar tersebut ingin dibandingkan apakah warna atau tekstur.
- Sistem menggunakan GLCM untuk mengekstrak parameter warna atau tekstur dari kedua gambar tergantung parameter apa yang dipilih.
- Perangkat lunak membandingkan hasil GLCM dan menampilkan berapa persen kesamaan kedua gambar dan time execution-nya

Penyelesaian:

Implementasi GLCM untuk ekstraksi parameter warna dan tekstur dan perbandingan hasil GLCM untuk menampilkan perbedaan antara gambar-gambar yang diunggah yang akan dijelaskan di bab empat.

BAB IV

IMPLEMENTASI DAN UJI COBA

4.1 Implementasi Program Utama

Color:

```
using base64
using cv2
using numpy
using async

function bgr_to_hsv(bgr: matrix) -> matrix
    bgr <- bgr.astype("float32") / 255
    blue, green, red <- bgr[..., 0], bgr[..., 1], bgr[..., 2]
    max_color, min_color <- np.max(bgr, axis=-1), np.min(bgr, axis=-1)
    diff <- max_color - min_color + 1e-10
    value <- max_color
    saturation <- np.zeros_like(max_color)
    saturation[max_color != 0] <- diff[max_color != 0] /
max_color[max_color != 0]
    hue <- np.zeros_like(max_color)
    mask <- max_color == red
    hue[mask] <- (green[mask] - blue[mask]) / diff[mask]
    mask <- max_color == green
    hue[mask] <- 2.0 + (blue[mask] - red[mask]) / diff[mask]
    mask <- max_color == blue
    hue[mask] <- 4.0 + (red[mask] - green[mask]) / diff[mask]
    hue <- (hue / 6) % 1
    hsv <- np.stack([hue, saturation, value], axis=-1)
    -> (hsv * np.array([180, 255, 255])).astype("uint8")

function calculate_similarity(hist1: matrix, hist2: matrix) -> float
    hist1 <- hist1 / (sum(hist1) + 1e-10)
    hist2 <- hist2 / (sum(hist2) + 1e-10)
    dot_product <- np.dot(hist1.flatten(), hist2.flatten())
    norm_hist1 <- np.linalg.norm(hist1)
    norm_hist2 <- np.linalg.norm(hist2)
    similarity <- dot_product / (norm_hist1 * norm_hist2)
```

```
   -> similarity * 100

async function color(dataset: list of matrix, image: matrix) -> list of dict
   image_contents <- await image.read()
   root_image <- cv2.imdecode(np.fromstring(image_contents, uint8),
cv2.IMREAD_COLOR)
   root_image_hsv <- bgr_to_hsv(root_image)
   root_histogram <- cv2.calcHist(
      [root_image_hsv], [0, 1], None, [180, 256], [0, 180, 0, 256]
   )

   similar_images <- []
   for dataset_image in dataset:
      dataset_contents <- await dataset_image.read()
      dataset_image <- cv2.imdecode(
         np.fromstring(dataset_contents, uint8), cv2.IMREAD_COLOR
      )
      dataset_image_hsv <- bgr_to_hsv(dataset_image)
      dataset_histogram <- cv2.calcHist(
         [dataset_image_hsv], [0, 1], None, [180, 256], [0, 180, 0, 256]
      )

      similarity <- calculate_similarity(root_histogram, dataset_histogram)

      if similarity >= 60 then
         _, buffer <- cv2.imencode(".jpg", dataset_image)
         dataset_image_base64 <- base64.b64encode(buffer).decode("utf-8")
         similar_images.append(
            {
               "base64imagedata": dataset_image_base64,
               "similaritypercentage": similarity,
            }
         )
   similar_images <- sorted(
      similar_images, key=lambda x: x["similaritypercentage"],
reverse=True
   )
   -> similar_images
```

Texture:

```
using base64
using cv2
using numpy
using async

function bgr_to_gray(bgr: matrix) -> matrix
    blue, green, red <- bgr[..., 0], bgr[..., 1], bgr[..., 2]
    gray <- (0.29 * red + 0.587 * green + 0.114 * blue).astype(uint8)
    -> gray

function glcm_func(image: matrix) -> matrix
    _, width <- image.shape
    frameworkMatrix <- zeros((256, 256), uint64)
    idxI <- image[:, : width - 1]
    idxJ <- image[:, 1:]
    np.add.at(frameworkMatrix, (idxI, idxJ), 1)
    transposeFramework <- frameworkMatrix.T
    frameworkMatrix += transposeFramework
    glcm <- normaliseSymmetricMatrix(frameworkMatrix)
    -> glcm

function metric(image: matrix) -> list of float
    contrast <- 0
    homogeneity <- 0
    dissimilarity <- 0
    asm <- 0
    glcm <- glcm_func(image)
    i, j <- np.indices((256, 256))
    contrast <- np.sum(glcm * (i - j) ** 2)
    homogeneity <- np.sum(glcm / (1 + (i - j) ** 2))
    dissimilarity <- np.sum(glcm * np.abs(i - j))
    asm <- np.sum(glcm**2)
    energy <- np.sqrt(asm)
```

```
    vektor <- [contrast, homogeneity, dissimilarity, asm, energy]
    -> vektor

function cosineSimilarityTexture(image1: matrix, image2: matrix) -> float
    v1 <- metric(image1)
    v2 <- metric(image2)
    dotProduct <- np.dot(v1, v2)
    magnitude1 <- np.sqrt(np.dot(v1, v1))
    magnitude2 <- np.sqrt(np.dot(v2, v2))
    result <- dotProduct / (magnitude1 * magnitude2) * 100
    -> result

function normaliseSymmetricMatrix(matrix: matrix) -> matrix
    total_sum <- np.sum(matrix)
    -> matrix / total_sum

async function texture(dataset: list of matrix, image: matrix) -> list of dict
    image_contents <- await image.read()
    root_image <- cv2.imdecode(np.fromstring(image_contents, uint8),
cv2.IMREAD_COLOR)
    root_image_g <- bgr_to_gray(root_image)
    similar_images <- []
    for dataset_image in dataset:
        dataset_contents <- await dataset_image.read()
        dataset_image <- cv2.imdecode(
            np.fromstring(dataset_contents, uint8), cv2.IMREAD_COLOR
        )
        dataset_image_g <- cv2.cvtColor(dataset_image,
cv2.COLOR_BGR2GRAY)
        similarity <- cosineSimilarityTexture(root_image_g, dataset_image_g)
        if similarity >= 60 then
            _, buffer <- cv2.imencode(".jpg", dataset_image)
            dataset_image_base64 <- base64.b64encode(buffer).decode("utf-8")
            similar_images.append(
                {
                    "base64imagedata": dataset_image_base64,
                    "similaritypercentage": similarity,
                }
            )
    similar_images <- sorted(
        similar_images, key=lambda x: x["similaritypercentage"],
reverse=True
```

```
)
-> similar_images
```

4.2 Penjelasan Struktur Program Berdasarkan Spesifikasi

Struktur program ini dibagi menjadi dua bagian yaitu frontend *directory* dan backend *directory*.

1. Frontend

   Frontend *directory* ini berisi pemrograman dokumen bahasa javascript yang menggunakan *framework* next.js.

   a. src

      Directory ini berisi *source file* untuk menghasilkan komponen-komponen javascript ke dalam layar user.

   b. public

      Directory ini berisi aset-aset (seperti *vector* atau foto) yang bakal dipakai di dalam tampilan web tersebut.

2. Backend

   Backend *directory* ini berisi pemrograman dokumen bahasa python yang menggunakan *framework* fastapi.

   a. app.py

      Dokumen ini berisi kode untuk mengatur *route endpoint* untuk menghasilkan *api* dari dokumen color.py dan texture.py

   b. color.py

      Dokumen ini berisi kode untuk menghitung perbandingan dengan perhitungan warna

   c. texture.py

Dokumen ini berisi kode untuk menghitung perbandingan dengan perhitungan tekstur

3. doc
   Directory ini berisi dokumen laporan tugas besar ini
4. img
   Directory ini berisi foto hasil screenshot hasil dari perhitungan
5. test
   Directory ini berisi dataset-dataset yang bakal digunakan

4.3 Penjelasan Tata Cara Penggunaan Program

Sebelum memulai program, user perlu memenuhi beberapa persyaratan berikut:

1. Sudah menginstall python (direkomendasikan versi 3.8-3.11) dan pip
2. Sudah menginstall npm (atau yarn)

Cara menggunakan program, user harus memulai kedua program backend dan frontend:

1. Buka dua terminal dengan *directory* nya *repository* program ini, kemudian jalankan *script* ini
   cd frontend (terminal 1)
   cd backend (terminal 2)
2. Pada terminal 1 (frontend), install semua komponen yang diperlukan dengan *script* ini
   npm install
   Pada terminal 2 (backend), atur *virtual environment* dan install semua

komponen yang diperlukan

pip install virtualenv

virtualenv venv

source venv/Scripts/activate (windows gitbash)

source venv/bin/activate (unix)

pip install -r requirements.txt

3. Mulai kedua program tersebut

   npm run dev (terminal 1)

   python app.py (terminal 2)

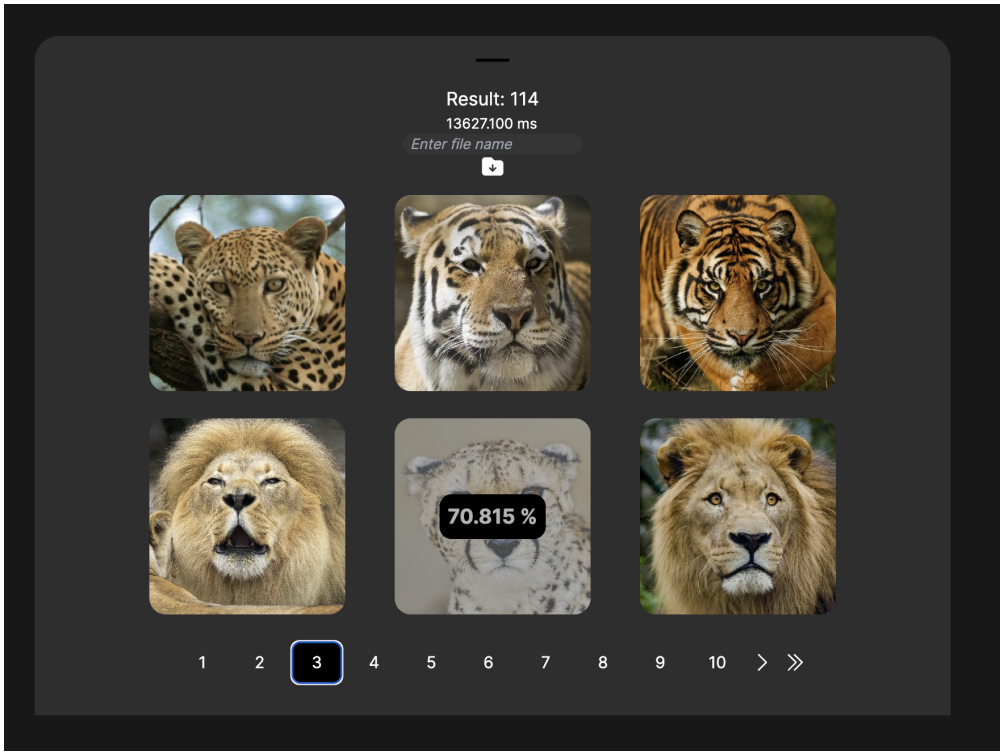
4. Dan wala! Silahkan memulai program kita di:

   http://localhost:3000/

4.4 Hasil Pengujian

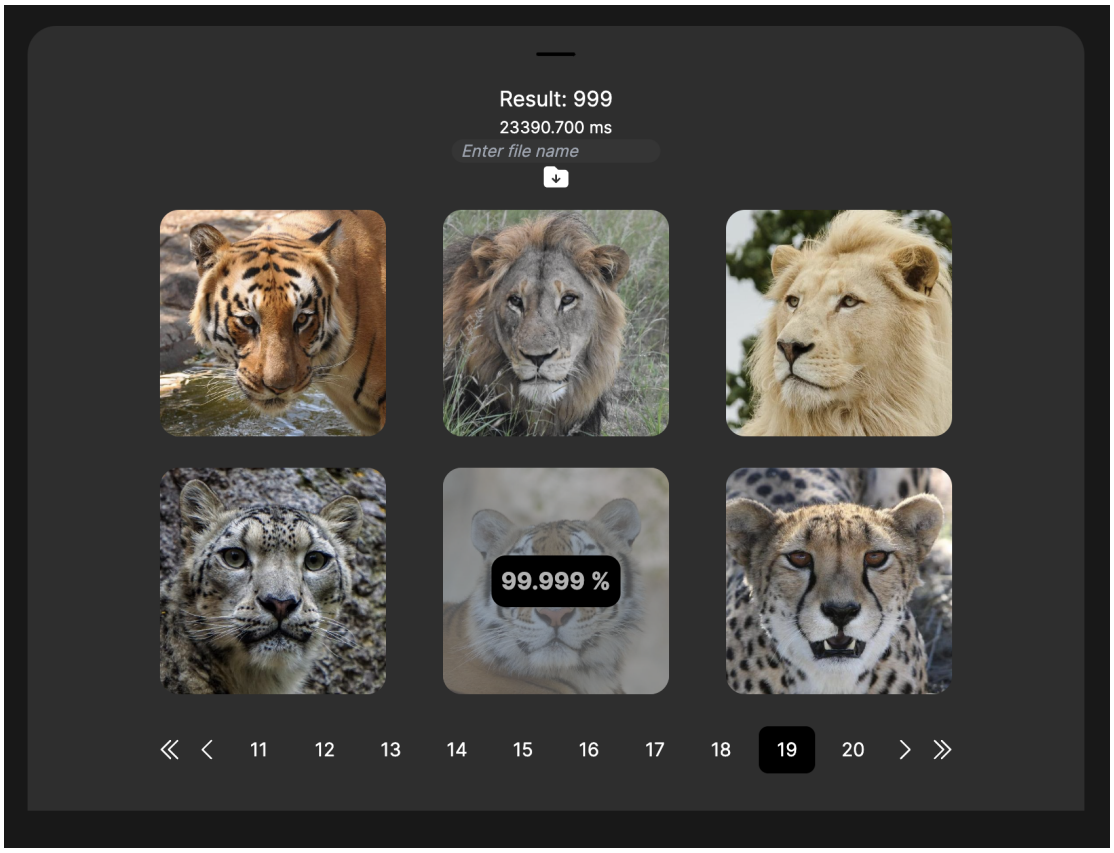
1. Perhitungan color

Dalam pengujian ini, kita menguji 999 dataset yang disediakan di web yang disediakan dalam spesifikasi. Hasil perhitungan ini menghasilkan 114 banyak foto dalam dataset yang tingkat kemiripan dengan diatas 60% dengan foto image.



2. Perhitungan texture

Dalam pengujian ini, kita menguji 999 dataset yang disediakan di web yang disediakan dalam spesifikasi. Hasil perhitungan ini menghasilkan 999 banyak foto dalam dataset yang tingkat kemiripan dengan diatas 60% dengan foto image. Bisa dilihat rata-rata persen kemiripannya yaitu > 98%



4.5 Analisis Desain Solusi Algoritma Pencarian

Untuk membuat perbandingan image semakin optimal lebih baik untuk menggabungkan kedua parameter glcm, namun jika harus memilih diantara keduanya, dari hasil kode kami terlihat bahwa perbandingan dengan parameter color lebih optimal dibanding parameter texture.

BAB V
PENUTUP

5.1 Kesimpulan

Pada tugas besar ini, kami telah mengembangkan dan menguji coba sebuah sistem temu balik gambar berbasis aljabar vektor. Sistem ini menggunakan metode Content-Based Image Retrieval (CBIR) yang memanfaatkan fitur-fitur visual seperti warna dan tekstur untuk mengidentifikasi dan mengambil gambar dari sebuah database. Kesuksesan sistem ini dalam mengidentifikasi gambar serupa berdasarkan konten visual menunjukkan aplikasi praktis aljabar linear dalam teknologi pencarian. Implementasi ini memberikan pandangan baru tentang bagaimana aljabar linear bisa dimanfaatkan dalam pengembangan aplikasi nyata, terutama dalam sistem temu balik informasi.

5.2 Saran

Untuk pengembangan lebih lanjut, kami menyarankan untuk memperluas kapasitas *database* dan meningkatkan algoritma pencocokan untuk mengakomodasi variasi gambar yang lebih luas, termasuk gambar dengan resolusi yang berbeda dan kondisi pencahayaan yang beragam. Selain itu, integrasi pembelajaran mesin dan kecerdasan buatan bisa dijajaki untuk meningkatkan akurasi sistem dalam mengenali pola kompleks dalam gambar. Penambahan fitur-fitur seperti pencarian berdasarkan sketsa atau integrasi dengan teknologi *augmented reality* juga dapat dipertimbangkan untuk menciptakan *user experience* yang lebih baik lagi.

5.3 Komentar atau Tanggapan

Kami dengan rendah hati membuka diri terhadap segala bentuk masukan yang dapat membantu dalam meningkatkan sistem yang kami kembangkan. Kami menyadari pentingnya pandangan objektif dari *user* dan para pakar dalam bidang IT yang dapat memberikan perspektif dalam hal usabilitas dan fungsi. Pendekatan ini esensial untuk mengasah dan meningkatkan kualitas program kami secara keseluruhan agar lebih intuitif dan efisien.

5.4 Refleksi terhadap Tugas Besar

Melalui tugas besar ini, kami telah mempelajari pentingnya kerja sama tim, pengelolaan waktu, dan aplikasi teori ke dalam praktik. Tantangan yang dihadapi selama pengembangan sistem telah mengajarkan kami untuk lebih inovatif dan persisten dalam mengatasi masalah. Kami belajar untuk menavigasi melalui tantangan yang tidak terduga dan mengatasi hambatan dengan solusi kreatif, yang pada akhirnya membentuk kami menjadi problem solver yang lebih baik.

5.5 Ruang Perbaikan atau Pengembangan

Kami mengakui dan menyadari bahwa setiap sistem dapat diperbaiki. Dalam konteks sistem kami, perbaikan dapat diarahkan pada peningkatan efisiensi algoritma pencarian yang saat ini masih memiliki potensi untuk dioptimalkan. Desain *interface* pengguna juga dapat dirombak untuk menciptakan pengalaman pengguna yang lebih mulus dan intuitif. Kami merencanakan untuk melakukan serangkaian pengujian pengguna yang intensif untuk mendapatkan umpan balik yang relevan dan mengimplementasikan perubahan yang akan meningkatkan kualitas dan efektivitas sistem.

DAFTAR PUSTAKA

Jun Yue, Z. L. (2011, August). *Content-based image retrieval using color and texture fused features*. Retrieved from ScienceDirect: https://www.sciencedirect.com/science/article/pii/S0895717710005352

MakersOfTailwindCSS. (n.d.). *Beautiful hand-crafted SVG icons, by the makers of Tailwind CSS.* Retrieved from heroicons: https://heroicons.com/

Munir, R. (2023). *Aljabar Linier dan Geometri.* Retrieved from Homepage Rinaldi Munir: https://informatika.stei.itb.ac.id/~rinaldi.munir/

RGB to HSV Color Conversion Algorithm. (2021). Retrieved from StackExchange: https://math.stackexchange.com/questions/556341/rgb-to-hsv-color-conversion-algorithm

Siagian, P., & Fernando, E. (2012). *KLASIFIKASI CITRA CONTENT-BASED IMAGE RETRIEVAL DENGAN METODE SHAPE BASE*

THRESHOLD. Retrieved from SNTIKI Seminar Nasional Teknologi Informasi Komunikasi dan Industri: https://ejournal.uin-suska.ac.id/index.php/SNTIKI/article/view/2846

Smyth, P. (2018, April 2). *Creating Web APIs with Python and Flask*. Retrieved from Programming Historian: https://programminghistorian.org/en/lessons/creating-apis-with-python-and-flask

Warongan, T. S. (2018). *Penerapan Metode Content-Based Image.* Retrieved from E-Journal UNSRAT: https://ejournal.unsrat.ac.id/index.php/informatika/article/download/28070/27542

Yunus, M. (2020, July 16). *Feature Extraction : Gray Level Co-occurrence Matrix (GLCM)*. Retrieved from Medium: https://yunusmuhammad007.medium.com/feature-extraction-gray-level-co-occurrence-matrix-glcm-10c45b6d46a1

LAMPIRAN

6.1 GitHub Repository (Latest Release)

https://github.com/caernations/Algeo02-22125

6.2 Video

https://youtu.be/L5K14amSUsA