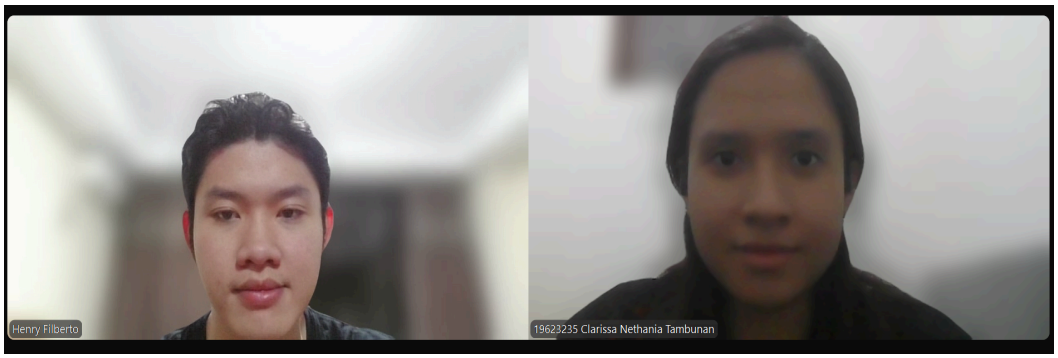
TUGAS BESAR 2
IF2123 ALJABAR LINIER DAN GEOMETRI



Dipersiapkan oleh:

Kelompok 54

Clarissa Nethania Tambunan 13523016
Henry Filberto Shenelo 13523108

Dosen Pengampu:

Ir. Rila Mandala, M.Eng., Ph.D.
Dr. Ir. Rinaldi Munir, M.T.
Arrival Dwi Sentosa, S. Kom., M.T.

PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
JL. GANESA 10, BANDUNG 40132

2024

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
DESKRIPSI MASALAH

Suara selalu menjadi hal yang paling penting dalam kehidupan manusia. Manusia berbicara mengeluarkan suara dan mendengarkan suatu suara untuk diserap ke otak dan mencari informasi dari suara tersebut. Suara juga bisa dijadikan orang-orang di dunia ini sebuah media untuk membuat karya seni. Contohnya adalah alat mendeteksi lagu. Manusia bisa mendeteksi suara dengan menggunakan indera pendengar dan memberikan kesimpulan akan apa jenis suara tersebut melalui respon dari otak. Sama seperti manusia, teknologi juga bisa mendeteksi suara dan memberikan jawaban mereka melalui algoritma-algoritma yang beragam bahkan bisa melebihi kapabilitas manusia. Dengan menggunakan algoritma apapun, konsep dari pendeteksi dan interpretasi suara itu bisa juga disebut dengan sistem temu balik suara atau bisa disebut juga dengan *audio retrieval system*. Banyak aplikasi yang menggunakan konsep sistem temu balik contohnya adalah Shazam.

Gambar 1 Shazam Sebagai Aplikasi Audio Retrieval System

Selain suara, manusia juga memiliki penglihatan sebagai salah satu inderanya dan bisa melihat warna dan gambar yang bermacam-macam. Teknologi komputasi juga memiliki kapabilitas yang sama dan bisa melihat gambar sama seperti kita, tetapi teknologi seperti ini juga bisa merepresentasikan gambar tersebut sebagai beragam-ragam angka yang bisa disebut juga fitur. Tahun ke tahun, *image processing* selalu menjadi fokus utama dari tugas besar 2 Algeo. Algoritma yang digunakan adalah Eigenvalue, *Cosine Similarity*, *Euclidean Distance*, dll.

Anda sudah melewati Tugas Besar 1 yaitu tentang matriks dan implementasi terhadap berbagai hal. Matriks adalah salah satu komponen yang penting dalam aplikasi aljabar vektor. Di dalam Tugas Besar 2 ini, anda diminta untuk membuat semacam aplikasi Shazam yaitu sebuah aplikasi yang meminta input lagu dan aplikasi tersebut mendeteksi apa nama dari lagu tersebut dan beberapa detail lainnya. Pada tugas besar ini, anda akan menggunakan aljabar vektor untuk mencari perbandingan antar satu audio dengan audio yang lain. Anda akan menggunakan konsep yang bernama *Music Information Retrieval* atau

MIR untuk mencari dan mengidentifikasi suara berdasarkan fitur-fitur yang dimilikinya. Tidak hanya itu, anda juga akan menggunakan konsep *Principal Component Analysis* (PCA) untuk mencari kumpulan audio melalui deteksi wajah berbagai orang (anggap saja mereka sebagai seorang penyanyi).

Information Retrieval adalah konsep meminta informasi dari sebuah data dengan memasukkan data tertentu. *Information Retrieval* ini terdiri dari 2 jenis yaitu *Image Retrieval* dan *Music Information Retrieval*. *Image Retrieval* adalah konsep untuk memasukkan sebuah input gambar dan berharap mendapatkan gambar yang ada di data sesuai dengan informasi dan perhitungan yang diinginkan. Sedangkan *Music Information Retrieval* (MIR) adalah konsep untuk memasukkan sebuah input audio dan berharap mendapatkan audio yang ada di data sesuai dengan informasi dan perhitungan yang diinginkan. Implementasikan *Image Retrieval* ini dengan menggunakan *Principal Component Analysis* dan *Music Information Retrieval* dengan menggunakan humming.

Information Retrieval ini kemudian dibuat menjadi sebuah website dengan arsitektur sebagai berikut:

1. *Audio query*, berisi file audio yang akan digunakan untuk melakukan pencocokan suara
2. *Voice recording module*, tombol untuk memproseskan suara dari microphone pengguna dan disimpan untuk diproses
3. Kumpulan suara yang disimpan dalam bentuk dataset, bisa dilakukan dengan cara mengunggah *multiple audio*, folder, .zip, .rar, dan file kompresi lainnya ke dalam website anda. Setelah memasukkan kumpulan suara tersebut, diharapkan semua audio yang ada dalam dataset dimunculkan di halaman website. Agar pengguna tidak melakukan *scrolling* yang berlebihan, silahkan gunakan *pagination*.
4. Bila mengerjakan bonus PCA, website dapat memetakan file lagu ke file gambar yang bersangkutan, gambar tersebut merepresentasikan album dari lagu tersebut. Pemetaan dapat disimpan dalam file .txt dan .json.

Komponen-komponen yang terdapat pada website tersebut adalah sebagai berikut:

1. Judul Website
2. Tombol untuk mengunggah audio ketika dalam moda *query audio by humming* dan mengunggah gambar ketika dalam moda *query album finder with PCA*
3. Tombol untuk merekam dan mengunggah rekaman audio tersebut (bonus)
4. Tombol untuk mengunggah dataset gambar wajah dan audio
5. Tombol untuk memilih antara *query audio by humming* dan *album finder by PCA*
6. Tombol untuk mengunggah *mapper*
7. Daftar lagu beserta album gambar yang bersangkutan yang ada di dataset
8. *Pagination* agar lagu yang muncul tidak banyak. Jumlah lagu dalam satu page dibebaskan asalkan tidak membuat pengguna harus scrolling terlalu lama
9. Tombol search untuk melakukan pencarian
10. Persentase kemiripan pada setiap audio atau album yang ada pada dataset
11. Waktu dari eksekusi program

Secara umum, berikut adalah langkah-langkah bagaimana sebuah website Information Retrieval bisa digunakan.

1. Pengguna terlebih dahulu mengunggah dataset pada tombol audio dan *picture* yang tertera di website.
2. Pengguna mengunggah pemetaan file lagu dengan judulnya dan gambar album yang bersangkutan (Bersifat opsional dan bonus pada bagian judul)

3. Pengguna melakukan query dengan dua cara yaitu mengunggah file audio atau gambar
4. Pengguna memilih akan melakukan pencarian dengan metode *album finder with PCA* atau *query by humming*
5. Program akan menampilkan kumpulan audio atau gambar yang mirip, diurutkan dari yang memiliki kemiripan paling tinggi ke yang paling rendah. Setiap audio atau gambar yang muncul diberi persentase kemiripannya.
6. Terdapat informasi terkait jumlah audio atau gambar yang muncul, dan waktu eksekusi programnya.

BAB II
DASAR TEORI

2.1. Sistem Temu Balik Suara (MIR)

Music Information Retrieval (MIR) adalah bidang penelitian yang menggabungkan teknik dari pemrosesan sinyal, pembelajaran mesin, dan teori informasi untuk menganalisis dan mengambil informasi dari data musik. Sistem MIR memungkinkan pengguna untuk memasukkan file audio sebagai kueri, kemudian sistem tersebut akan memproses dan mengekstrak fitur-fitur audio tersebut. Proses ekstraksi melibatkan analisis distribusi nada dan penggunaan berbagai fitur akustik untuk membentuk vektor representasi dari audio kueri. Setelah vektor representasi dari audio kueri dibuat, sistem MIR kemudian membandingkan vektor tersebut dengan vektor-vektor audio lainnya yang ada dalam dataset. Proses ini bertujuan untuk mencari audio dalam dataset yang paling mirip dengan kueri, dan mengurutkan hasilnya berdasarkan tingkat kemiripan dari yang paling mirip hingga yang paling tidak mirip. Dengan demikian, MIR membantu pengguna untuk menemukan kecocokan audio kueri mereka dengan dataset yang tersedia, memfasilitasi pencarian musik dan analisis audio yang lebih efisien.

Dalam mengaplikasi MIR ini yaitu sistem *query* dengan humming langkah paling utama yang dilakukan adalah melakukan pemrosesan audio dimana audio ini menggunakan file MIDI dan berfokus pada *track* melodi utama yaitu *Channel 1* karena ini adalah *default channel* yang digunakan untuk instrumen utama atau lead instrument dalam banyak komposisi musik. Proses file MIDI ini melibatkan metode *windowing* yang membagi melodi menjadi segmen-segmen 20-40 beat dengan *sliding window* 4-8 beat digunakan untuk menangkap bagian-bagian spesifik dari lagu yang mungkin lebih mudah diingat oleh pengguna. Melodi dalam sebuah lagu biasanya memiliki bagian-bagian tertentu yang lebih menonjol atau lebih dikenal, seperti *chorus*, *bridge*, atau *hook*. Bagian-bagian ini sering kali lebih mudah diingat karena memiliki pola nada atau ritme yang khas dan berulang.

Metode *windowing* dilakukan dengan cara normalisasi tempo dan *pitch* yang bertujuan untuk mengurangi variasi *humming*. Normalisasi dilakukan untuk membuat perbandingan antara melodi kueri dan melodi dalam database menjadi lebih akurat dan konsisten. Cara untuk melakukan normalisasi tempo dan *pitch* ini dengan rumus,

$$NP(note) = \frac{(note - \mu)}{\sigma}$$

Gambar 2 Rumus Normalisasi Tempo dan Pitch

dengan μ adalah rata-rata dari *pitch* dan σ adalah standar deviasi dari *pitch*.

2.2. Metode Ekstraksi Fitur dengan Humming

Pada sistem *query* dengan *humming*, proses ekstraksi fitur sangat penting untuk mengkonversi input audio menjadi data yang dapat dibandingkan dengan database musik. Proses ini melibatkan beberapa langkah utama untuk mendapatkan fitur yang relevan dari audio *query* yang dimasukkan

pengguna. Ekstraksi fitur ini melibatkan distribusi nada dan normalisasi dimana distribusi nada diukur dengan tiga fitur yaitu,

1. **Absolute Tone Based (ATB)**

   Absolute Tone Based (ATB) menghitung frekuensi kemunculan setiap nada berdasarkan skala MIDI (0-127). Hasilnya adalah histogram yang memberikan gambaran distribusi absolut nada dalam data. Langkah pertama adalah membuat histogram dengan 128 bin, sesuai dengan rentang nada MIDI dari 0 hingga 127. Kemudian, dihitung frekuensi kemunculan masing-masing nada MIDI dalam data. Setelah itu, histogram dinormalisasi untuk mendapatkan distribusi yang terstandarisasi. Ini penting untuk menangkap karakteristik statis melodi dalam sinyal audio. ATB ini untuk menunjukkan karakteristik melodi atau seberapa sering nada-nada tertentu muncul.

2. **Relative Tone Based (RTB)**

   Relative Tone Based (RTB) mengukur perubahan antara nada yang berurutan, menghasilkan histogram dengan nilai dari -127 hingga +127. RTB membantu memahami pola interval melodi, yang lebih relevan untuk mencocokkan *humming* dengan dataset tanpa bergantung pada *pitch* absolut. Proses ini dimulai dengan membangun histogram dengan 255 bin yang mencakup rentang nilai dari -127 hingga +127. Setelah itu, dihitung selisih antara nada-nada berurutan dalam data. Histogram yang dihasilkan kemudian dinormalisasi. Teknik ini penting untuk menangkap dinamika melodi dan memastikan bahwa sistem dapat mengenali pola perubahan nada dengan lebih baik.

3. **First Tone Based (FTB)**

   First Tone Based (FTB) berfokus pada selisih antara setiap nada dengan nada pertama, menghasilkan histogram yang mencerminkan hubungan relatif terhadap titik referensi awal. Pendekatan ini membantu menangkap struktur relatif nada yang lebih stabil terhadap variasi *pitch* dari audio query. Histogram ini dibuat dengan 255 bin yang mencakup rentang nilai dari -127 hingga +127. Setelah itu, dihitung perbedaan antara setiap nada dalam data dengan nada pertama. Histogram yang dihasilkan kemudian dinormalisasi untuk menciptakan distribusi yang seimbang. Teknik ini berguna untuk mempertahankan konsistensi dalam mengenali pola nada meskipun terdapat variasi dalam pitch dari audio *query*.

Normalisasi memastikan bahwa semua nilai dalam histogram berada dalam skala probabilitas, artinya nilai-nilai tersebut diubah sehingga total seluruh nilai dalam histogram sama dengan 1. Ini membuat distribusi lebih mudah dibandingkan, karena setiap nilai dalam histogram sekarang mewakili proporsi dari keseluruhan yaitu dengan rumus sebagai berikut.

$$Hnorm = \frac{H[d]}{\sum_{d}^{127} H[d]}$$

Gambar 3 Rumus Normalisasi Histogram

dengan H adalah Histogram dan d adalah bin dari histogram tersebut.

Untuk menghitung kemiripan antara kedua audio, setiap histogram diubah menjadi sebuah vektor dan dihitung kemiripannya menggunakan *cosine similarity*. *Cosine similarity* digunakan untuk mengukur

kemiripan antara audio *query* dan audio dalam dataset dengan membandingkan vektor yang dihasilkan dari fitur ATB, RTB, dan FTB yang telah dinormalisasi. Dengan demikian, semakin kecil sudut yang diperoleh dari hasil cosine similarity, semakin mirip kedua vektor tersebut, yang berarti audio query memiliki kemiripan yang tinggi dengan audio dalam dataset. Kemiripan ini kemudian diubah menjadi persentase untuk memudahkan interpretasi hasil. Rumus dari *cosine similarity* yaitu,

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

Gambar 4 Rumus Cosine Similarity

dimana A dan B adalah dua vektor.

2.3. Image Retrieval dengan PCA

Principal Component Analysis (PCA) adalah metode statistik yang menggunakan transformasi ortogonal untuk mengubah sekumpulan variabel yang saling berkorelasi menjadi sekumpulan variabel yang tidak berkorelasi. PCA merupakan alat yang paling umum digunakan dalam analisis data eksploratif dan *machine learning* untuk model prediktif.

Tujuan utama dari PCA adalah mengurangi dimensi dari suatu kumpulan data sambil mempertahankan pola atau hubungan yang paling penting antar variabel tanpa memerlukan pengetahuan sebelumnya tentang variabel target. PCA digunakan untuk mengurangi dimensi data dengan menemukan himpunan variabel baru yang lebih kecil dari himpunan variabel awal, tetapi tetap mempertahankan sebagian besar informasi dalam sampel. Hasil data yang didapatkan dari PCA ini akan berupa eigenvector dan proyeksi data. Teknik ini berguna juga dalam melakukan Image Retrieval.

Langkah-langkah melakukan Image Retrieval dengan PCA adalah

1. **Image Processing**

   Ubah gambar menjadi grayscale untuk mengurangi kompleksitas gambar dan membuat fokus menjadi bagian terang dan gelap gambar.

   $$I(x,y) = 0.2989 \cdot R(x,y) + 0.5870 \cdot G(x,y) + 0.1140 \cdot B(x,y)$$

   Selanjutnya, ukuran gambar akan disesuaikan agar seragam untuk semua gambar agar lebih akurat. Kemudian, ubah vektor grayscale dari gambar menjadi bentuk satu dimensi (1D) sehingga dapat diproses dalam analisis data.

2. **Standarisasi Data**

   Dilakukan standarisasi data di sekitar nilai 0 dengan menghitung rata rata dari setiap gambar untuk suatu piksel.

$$\mu_j = \frac{1}{N}\sum_{i=1}^{N} x_{ij}$$

Gambar 5 Rumus Rata-Rata Piksel Gambar

Lalu kurangi piksel tersebut dengan rata-rata yang sudah dihitung untuk melakukan standarisasi.

$$xij' = xij - \mu j$$

3. PCA dengan menggunakan SVD

Buat matriks kovarians dari data yang sudah distandarisasi

$$C = \frac{1}{N} X'^{T} X'$$

dan lakukan Singular Value Decomposition sehingga didapatkan

$$\mathbf{C} = \mathbf{U}\mathbf{\Sigma}\mathbf{U}^{T}$$

dengan

- U: matriks eigenvector (komponen utama),
- Σ: matriks eigenvalue.

Kemudian diambil n jumlah komponen utama teratas dari hasil SVD dan dilakukan proyeksi gambar ke komponen utama.
Pilih k-komponen utama teratas ($k \ll M \cdot N$) dan proyeksikan data:

$$\mathbf{Z} = \mathbf{X}'\mathbf{U}_k$$

dengan

- Uk: matriks eigenvector dengan n-dimensi.

4. Similarity Computation

Representasikan gambar query dalam ruang komponen utama dengan proyeksi yang sama:

$$\mathbf{q} = (\mathbf{q}' - \mu)\mathbf{U}_k$$

dengan

- q = Vektor proyeksi dari gambar query ke ruang komponen utama (PCA).
- q′: Gambar query dalam format vektor (setelah grayscale, resize, dan flattening).

Kemudian, hitung jarak Euclidean antara gambar query dengan semua gambar dalam dataset:

$$d(\mathbf{q}, \mathbf{z}_i) = \sqrt{\sum_{j=1}^{k} (q_j - z_{ij})^2}$$

Gambar 6 Rumus Jarak Euclidean Antara Gambar Query dengan Semua Gambar

Hasil yang didapat diurutkan dari yang terkecil. Hasil ini akan digunakan untuk menemukan gambar yang sesuai dan dapat diolah lebih lanjut untuk dilakukan *mapping* ke audio-audio tertentu membentuk sebuah “Album Finder”.

BAB III
ARSITEKTUR WEBSITE DAN PROGRAM INFORMATION RETRIEVAL

Pada Tugas Besar ini, kami mengembangkan suatu website dalam Bahasa Python dan Next.js untuk menyelesaikan masalah-masalah dalam Aljabar Linier dan Geometri dan dalam hal ini adalah *Information Retrieval* yaitu *Image Retrieval* dengan menggunakan *Principal Component Analysis* dan *Music Information Retrieval* dengan menggunakan humming. Website ini terdiri dari folder *frontend* untuk membuat tampilan dan komponen-komponen yang diperlukan serta folder *backend* yang berisi tentang program untuk *Image Retrieval* dan *Music Retrieval*. Program pada backend ini terdiri dari banyak fungsi dari memproses audio atau gambar, menstandarisasi gambar atau mengekstrak audio menjadi fitur vektor yang dapat dianalisis, lalu pada akhirnya dilakukan pencarian kemiripan audio atau gambar dengan audio atau gambar yang berada di dataset.

3.1. Folder *app*

Folder ini berisi aplikasi frontend dari website. Di dalamnya terdapat berbagai komponen dan halaman yang membentuk antarmuka pengguna. Folder ini juga berisi API

3.2. Folder *backend*

Folder backend ini berisi program *Music Information Retrieval* yang berada pada folder *audio* dan *Image Retrieval* yang berada pada folder *pca* karena menggunakan konsep *Principal Component Analysis* (PCA).

3.2.1. Folder *audio*

3.2.1.1. process.py

Di dalam file ini terdapat fungsi-fungsi yang dapat memproses audio yang bertipe file MIDI atau dalam format .mid

Tabel 3.2.1.1. Fungsi pada process.py

Fungsi	Deskripsi
read_midi(file_path)	Membaca file MIDI dan mengekstrak nada dan waktu dari track yang ada di *channel* 1.
find_fileMidi_inZip(zip_folder, selected_folder, midi_file_name)	Mencari file MIDI dengan nama tertentu di dalam folder terkompresi (ZIP) dan mengembalikan jalur file jika ditemukan.
windowing(notes, min_window_size, max_window_size, step_size)	Membagi melodi menjadi segmen-segmen kecil dengan ukuran jendela tertentu dan langkah tertentu untuk analisis lebih lanjut.
normalisasi_tempo(segments)	Melakukan normalisasi *pitch* dan waktu pada

	segmen-segmen melodi untuk mengurangi variasi dan membuat data lebih konsisten.
representasi_numerik(segments)	Mengonversi segmen melodi menjadi representasi numerik yang mempertimbangkan durasi dan urutan nada untuk analisis dan perbandingan lebih lanjut.
process_midi_file(file_path)	Memproses file MIDI dengan menggabungkan fungsi-fungsi sebelumnya untuk menghasilkan segmen melodi yang telah dinormalisasi dan direpresentasikan secara numerik.

3.2.1.2. extraction.py

Di dalam file ini terdapat fungsi-fungsi untuk melakukan ekstraksi fitur dan mengekstrak file ZIP ke dalam suatu folder kosong, ekstraksi fitur ini dilakukan untuk mendistribusi nada dengan tiga fitur dan dilakukan normalisasi.

Tabel 3.2.1.2. Fungsi pada extraction.py

Fungsi	Deskripsi
extract_zip(zip_path, extract_to)	Mengekstrak file ZIP ke dalam folder tertentu, membuat folder jika belum ada.
fitur_atb(notes)	Menghitung histogram distribusi absolut nada berdasarkan skala MIDI dan menormalisasinya.
fitur_rtb(notes)	Menghitung histogram perubahan nada yang berurutan (interval nada) dan menormalisasinya.
fitur_ftb(notes)	Menghitung histogram selisih nada dari nada pertama dan menormalisasinya.
cosine_similarity(vec1, vec2)	Menghitung kemiripan antara dua vektor menggunakan cosine similarity.
compare_filemidi(file1, file2)	Membandingkan dua file MIDI dengan menghitung kemiripan berdasarkan fitur ATB, RTB, dan FTB, lalu menghitung rata-rata cosine similarity untuk menentukan persentase kemiripan.

3.2.1.3. mainAudio.py

Di dalam file ini terdapat fungsi-fungsi untuk melakukan program utama dari sistem *query* dengan *humming* dengan memanfaatkan fungsi-fungsi *process.py* dan *extraction.py*.

Tabel 3.2.1.3. Fungsi pada mainAudio.py

Fungsi	Deskripsi
all_midi_files(folder_path)	Mencari semua file MIDI dalam folder yang telah dilakukan ekstrak file ZIP tadi yang diberikan dan mengembalikannya sebagai *list*.
sort_similarity(target_file, zip_folder)	Mengurutkan file MIDI berdasarkan kemiripannya dengan file target menggunakan rumus *cosine similarity*.
audio_query(zip_path, extract_to, query)	Mengekstrak ZIP, menghitung kemiripan file MIDI dengan file target, dan mengembalikan urutan kemiripan.
compare_midi_files(args)	Membandingkan dua file MIDI dan menghitung kemiripannya.
speed_program(zip_path, extract_to, query)	Melakukan optimasi program dengan menggunakan multiprocessing untuk mempercepat perhitungan kemiripan file MIDI.
waktu_program_audio(zip_path, extract_to, query, use_speed_program)	Mengukur waktu eksekusi program untuk menghitung urutan kemiripan file MIDI, dengan atau tanpa optimisasi.
copy_results_to_result_folder(results)	Menyalin file MIDI hasil ke dalam folder hasil (resultaudio) dan menambahkan persentase kemiripan pada nama file.

3.2.2. Folder *pca*

3.2.2.1. albumFinder.py

Tabel 3.2.2.1. Fungsi pada albumFinder.py

Fungsi	Deskripsi
find_album()	Fungsi ini dimulai dengan mencetak pesan dan mencatat waktu mulai. Kemudian, menetapkan direktori skrip dan memuat file JSON mapper yang berisi pemetaan artis dan folder audio. Selanjutnya, fungsi ini mengekstrak file ZIP yang berisi gambar dan audio ke dalam direktori yang sesuai. Fungsi ini mencetak hasil akhir, termasuk jalur gambar terdekat, nama artis, dan folder audio, serta waktu eksekusi total

unzip_files(zip_folder, extract_to)	Fungsi dalam find_album() yang mengekstrak semua file ZIP dalam folder tertentu ke dalam folder tujuan yang ditentukan.
face_recognition(image_folder_unzip, query_image_path)	Melakukan pengenalan wajah pada gambar dalam folder yang diekstrak dan membandingkannya dengan gambar kueri untuk menemukan gambar terdekat.

3.2.2.2. mapper.py

Tabel 3.2.2.2. Fungsi pada mapper.py

Fungsi	Deskripsi
create_artist_mapping(pictures_folder, audio_folder, output_json="mapper.json")	Membuat pemetaan antara folder gambar dan folder audio untuk setiap artis yang ada, dan menyimpannya dalam sebuah file JSON. Fungsi ini memeriksa apakah folder gambar dan audio untuk setiap artis ada, lalu menyimpan informasi tersebut dalam struktur data artist_mapping. Setelah itu, hasil pemetaan disimpan sebagai file JSON dengan nama yang ditentukan.

3.2.2.3. pca.py

Tabel 3.2.2.3. Fungsi pada pca.py

Fungsi	Deskripsi
process_dataset(image_folder, width, height)	Membaca dan mengubah ukuran gambar menjadi *grayscale*, lalu mengubahnya menjadi vektor yang dapat dianalisis.
data_centering(images)	Menghitung wajah rata-rata dan mengurangkan rata-rata dari setiap gambar untuk mendapatkan gambar yang telah di-centering.
pca_svd(standardized_images)	Melakukan dekomposisi SVD pada matriks kovarians dari gambar yang telah di-standardisasi untuk mendapatkan komponen utama.
determine_k(S, threshold)	Menentukan jumlah komponen utama kk yang mencakup persentase varians tertentu.
project_k_components(U, k)	Memproyeksikan data ke kk komponen utama teratas.
project_data(data, Uk)	Memproyeksikan data pelatihan ke ruang PCA.

process_query(query_image_path, width, height, mean_face)	Memproses dan memproyeksikan gambar kueri ke ruang PCA yang sama.
euclidean_dist(q, z)	Menghitung jarak Euclidean antara dua vektor.
closest_image(query_projection, dataset_projections, image_paths)	Mencari gambar terdekat di dataset dengan menghitung jarak Euclidean antara proyeksi gambar kueri dan proyeksi dataset.
face_recognition(image_folder, query_image_path, width=64, height=64)	Menggabungkan semua fungsi dalam file untuk mengenali gambar wajah terdekat dalam dataset berdasarkan gambar kueri yang diberikan. Hasil akhirnya adalah jalur gambar terdekat dan jarak Euclidean antara gambar kueri dan gambar yang ditemukan.

3.2.3. Folder *main.py*

File main.py menginisialisasi aplikasi FastAPI, mengonfigurasi CORS untuk mengizinkan permintaan dari http://localhost:3000. File ini mendefinisikan endpoint (/upload/{file_type}) untuk menangani unggahan file, menyimpan file yang diunggah ke direktori yang sesuai. Jika file yang diunggah adalah file ZIP, file tersebut akan diekstrak ke dalam direktori yang sama dan file ZIP asli akan dihapus. Setup ini memungkinkan aplikasi untuk mengelola unggahan file dan menangani ekstraksi file ZIP dengan lancar.

3.3. Folder *components*

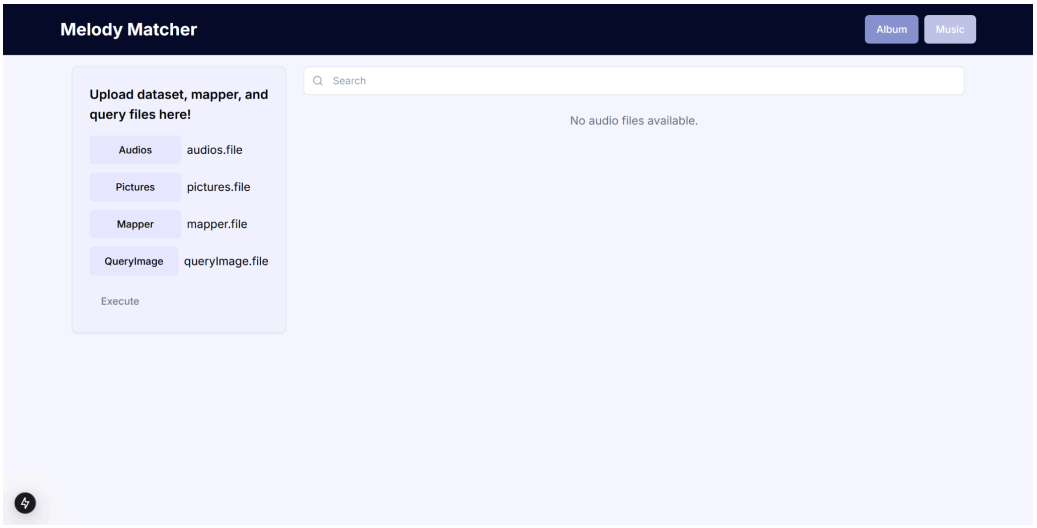
Folder ini berisi komponen-komponen dari tampilan dan interaksi di dalam website seperti uploader, audio-grid, search bar, navigation, serta untuk keperluan UI seperti button, pagination, card, input

3.4. Folder *public*

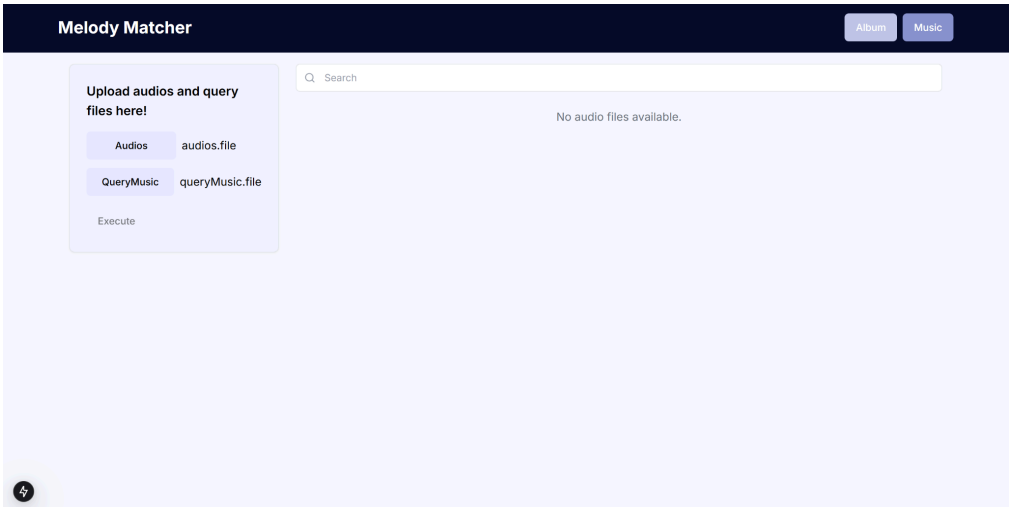
Folder ini digunakan untuk menyimpan dan mengambil aset serta gambar album yang diperoleh dari proses album finder.

BAB IV
EKSPERIMEN

Pada awal memulai aplikasi pertama kali, akan terlihat tampilan sebagai berikut



Gambar 7 Tampilan Awal Album



Gambar 8 Tampilan Awal Music

Kami melakukan total 4 eksperimen, dua untuk masing-masing fitur, yaitu album finder dan *music retrieval.* Pada eksperimen album finder, digunakan dataset dengan jumlah data sebanyak 665 file gambar

(jpg) yang terbagi ke dalam 5 folder (orang). Selain itu, terdapat satu mapper (json) dan juga 50 audio files (midi) yang juga terbagi ke dalam 5 folder. Pada eksperimen *music retrieval,* digunakan 17 file audio yang terbagi ke dalam 9 folder (orang).

1. **Eksperimen Album Finder dengan Menggunakan Input di dalam Dataset**

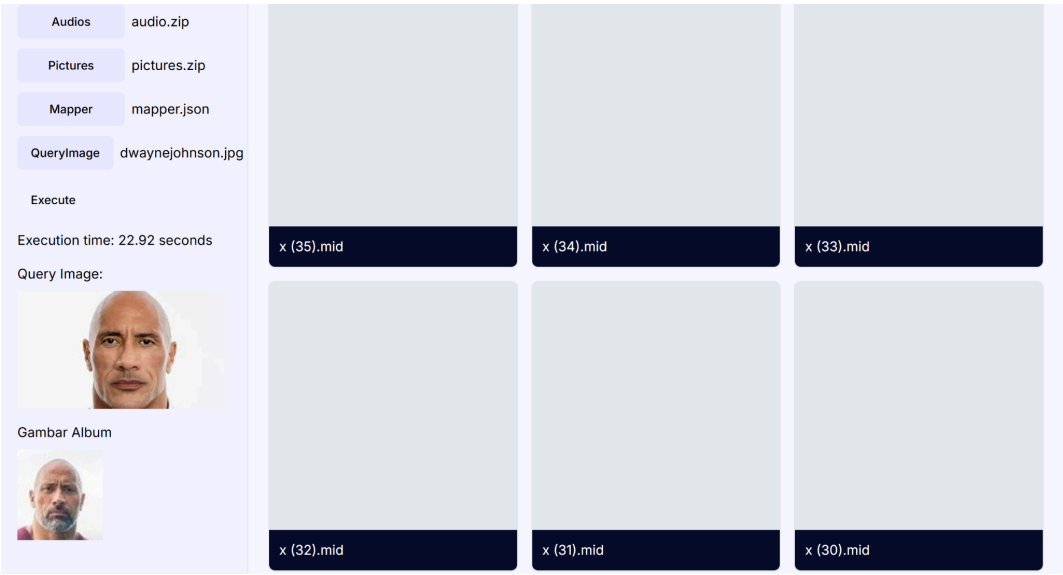
   Pada eksperimen ini, digunakan data gambar yang berada di dalam dataset itu sendiri

Gambar 9 Hasil Eksperimen Album Finder dengan Data di Dalam Dataset

Dengan menggunakan data yang berada di dalam dataset, diperoleh hasil gambar album yang tepat beserta dengan audio-audionya. Waktu eksekusinya yaitu 23,54 detik

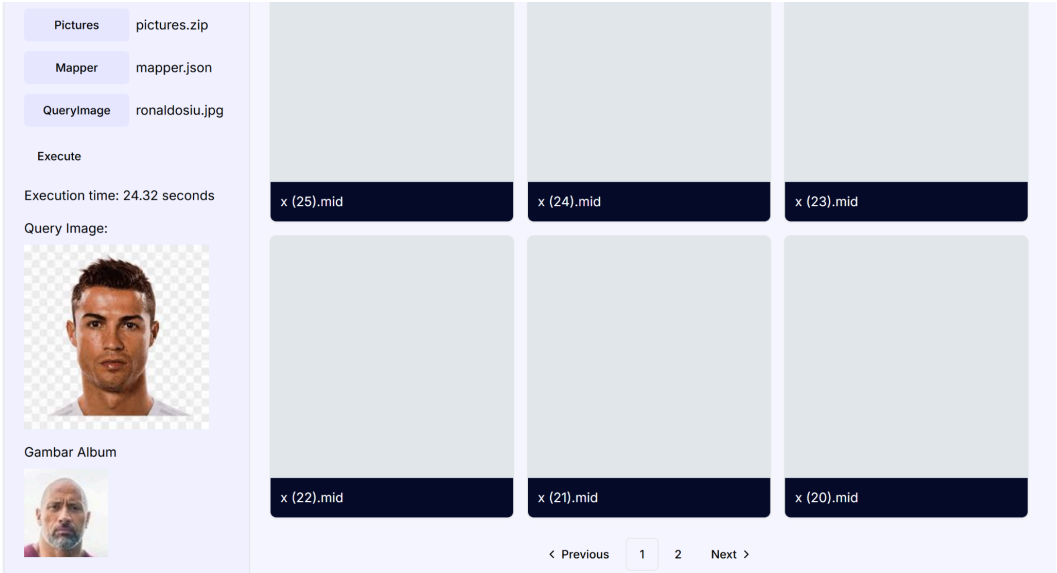
2. **Eksperimen Album Finder dengan Menggunakan Input di luar Dataset**

   Pada eksperimen ini digunakan data yang berada di luar dataset



Gambar 10 Hasil Eksperimen Album Finder dengan Data di Luar Dataset

Dengan data yang berada di luar dataset, hasil yang didapatkan masih benar dengan waktu eksekusi 22,92 detik. Walaupun begitu, kemungkinan salah dari pencarian ini dapat terjadi seperti ini



Gambar 11 Hasil Eksperimen Album Finder dengan Data di Luar Dataset (Tidak Akurat)

Hal ini dapat terjadi karena dataset yang tidak memadai, perbedaan pose, pencahayaan, terdapat *noise* pada gambar serta ketidaksempurnaan metode *face recognition* dengan menggunakan PCA dan SVD.

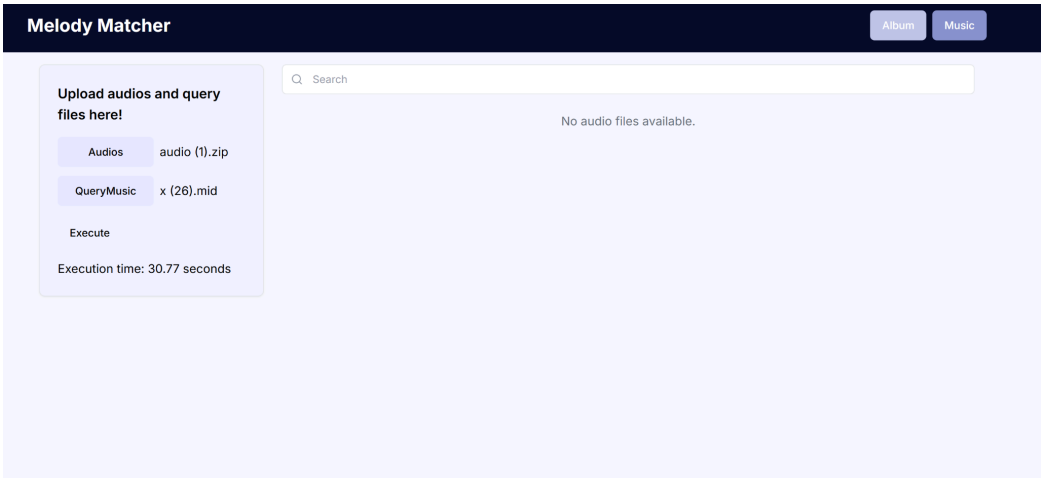
3. Eksperimen Music Retrieval dengan Menggunakan Input di dalam Dataset

Gambar 12 Hasil Eksperimen Music Retrieval dengan Data di Dalam Dataset

Dari eksperimen ini, didapat bahwa hasil cukup akurat karena query dengan hasil paling mirip (100%) adalah audio yang sama. Waktu eksekusinya yaitu 35,14 detik.

4. Eksperimen Music Retrieval dengan Menggunakan Input di Luar Dataset

Pada eksperimen ini, digunakan data di luar dataset. Hasil pada eksperimen ini adalah tidak ditemukan audio yang mendekati kemiripannya karena audio yang digunakan sangat berbeda.



Gambar 13 Hasil Eksperimen Music Retrieval dengan Data di Luar Dataset

BAB V
KESIMPULAN

5.1. Kesimpulan

Pada Tugas Besar ini, kami berhasil membuat suatu website dalam bahasa Python dan Next.js untuk menyelesaikan masalah di Aljabar Linier dan Geometri dimana mampu menangani seperti :

1. Menemukan data audio pada dataset yang memiliki kemiripan dengan audio yang dimasukkan oleh pengguna menggunakan konsep *Music Information Retrieval* dan menunjukkan persentase kemiripannya.
2. Menemukan data gambar pada dataset yang sesuai dengan file gambar yang dimasukkan oleh pengguna menggunakan konsep *Principal Component Analysis*

Website yang dibangun dapat terlebih dahulu mengunggah dataset pada tombol audio dan *picture* ,lalu pengguna dapat melakukan *query* dengan dua cara yaitu mengunggah file audio atau gambar, dimana audio ini dilakukan pencarian dengan metode *query by humming*, sedangkan file gambar dengan metode *album finder with PCA*, Setelah itu, website akan menampilkan kumpulan audio atau gambar yang mirip dengan diturunkannya dari kemiripan paling tinggi ke yang paling rendah dengan ditunjukkannya persentase kemiripannya. Lalu website juga akan menampilkan informasi terkait jumlah audio atau gambar yang muncul dan waktu eksekusi programnya

5.2. Saran

Saran untuk peningkatan website ini adalah file audio tersebut dapat di-*play* dan juga agar waktu eksekusi programnya dapat berkurang agar mengurangi ketidaknyamanan pengguna untuk menunggu dalam waktu yang lama untuk menjalankan program tersebut baik itu untuk file audio atau gambar *query*

5.3. Komentar

Tugas ini memiliki beban yang cukup berat dan diharuskan untuk banyak mengeksplor serta mampu membuat suatu website sederhana. Walaupun begitu, tugas ini dapat selesai dan mungkin masih terdapat kekurangan yang dapat ditingkatkan ke depannya.

5.4. Refleksi

Selamat mengerjakan Tugas Besar ini, kami memiliki beberapa kendala dalam mengerjakan tugas ini karena ada dari kami yang belum banyak berpengalaman dalam membuat suatu website

BAB VI
LAMPIRAN

6.1. Referensi

6.1.1. Sistem Temu Balik Suara (MIR)

Kasaka, P., Jarina, R., & Chmulik, M. (2021, Maret 22). *Music information retrieval for educational purposes - an overview*. IEEEXplore.https://ieeexplore.ieee.org/document/9379216?form=MG0AV3.

Ghias, A., Logan, J., & Chamberlin, D. (n.d.). *Query By Humming Musical Information Retrieval in An Audio Database*. Cornell.https://www.cs.cornell.edu/zeno/papers/humming/humming.pdf?form=MG0AV3&form=MG0AV3.

6.1.2. Metode Ekstraksi Fitur dengan Humming

Kosugi, N., Nishihara, Y., & Kon'ya, S. (2002, Agustus 6). *Music retrieval by humming-using similarity retrieval over high dimensional feature vector space*. IEEEXplore.https://ieeexplore.ieee.org/document/799561.

6.1.3. Image Retrieval dengan konsep PCA

Riswanto, U. (2023, Maret 2). *Image Compression Techniques: A Closer Look at Principal Component Analysis*. https://ujangriswanto08.medium.com/image-compression-techniques-a-closer-look-at-principal-component-analysis-67cf7a29fdb9.

6.2. Tautan Repository

https://github.com/henry204xx/Algeo02-23016