

TUGAS BESAR 2
IF2123 ALJABAR LINIER DAN GEOMETRI
Image Retrieval dan Music Information Retrieval Menggunakan
PCA dan Vektor



Kelompok 35
ikan dan pisang

Disusun Oleh:

Bertha Soliany Frandi	13523026
Rafen Max Alessandro	13523031
Grace Evelyn Simon	13523087

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

2024

DAFTAR ISI

BAB 1: DESKRIPSI MASALAH	4
BAB 2: TEORI SINGKAT	6
2.1. Image Retrieval	6
2.1.1. Image Processing and Loading	6
2.1.2. Data Centering (Standardization)	7
2.1.3. PCA Computation Using Singular Value Decomposition (SVD)	7
2.1.4. Similarity Computation	8
2.2. Music Information Retrieval (MIR)	9
2.2.1. Preprocessing File MIDI	10
2.2.2. Pemrosesan Audio MIDI	10
2.2.3. Ekstraksi Fitur	10
2.2.4. Perhitungan Similaritas	11
BAB 3: ARSITEKTUR WEBSITE	13
3.1. Implementasi Program Berbasis Website	13
3.1.1. Python	13
3.1.2. Next.JS	13
3.1.3. Flask API	14
3.2. Struktur Front End Website	14
3.2.1. Halaman Utama (Home)	14
3.2.2. Halaman Finder	14
3.2.3. Halaman About Us	15
3.3. Struktur Back End Website	15
3.4. Prosedur Penggunaan Website	15
BAB 4: EKSPERIMEN	17
BAB 5: PENUTUP	23
5.1. Kesimpulan	23
5.2. Saran	23
5.3. Komentar	24
5.4. Refleksi	24
LAMPIRAN	25
1. Referensi	25
2. Tautan Repository	25
3. Tautan Video	25

DAFTAR GAMBAR

Gambar 1. Aplikasi Shazam yang memanfaatkan konsep sistem temu balik	4
Gambar 2. Perhitungan konversi gambar berwarna menjadi grayscale	6
Gambar 3. Hasil vektor untuk suatu gambar berdimensi M x N piksel	7
Gambar 4. Rumus perhitungan rata-rata keseluruhan data vektor hasil pengolahan dataset	7
Gambar 5. Proses standarisasi data vektor hasil pengolahan dataset	7
Gambar 6. Dekomposisi SVD	8
Gambar 7. Persamaan untuk menentukan matriks vektor dataset setelah proyeksi	8
Gambar 8. Perhitungan proyeksi vektor query	8
Gambar 9. Perhitungan jarak Euclidean	9
Gambar 10. Formula Umum Normalisasi	11
Gambar 11. Formula Cosine Similarity	12
Gambar 12. Tampilan Deskripsi Singkat	17
Gambar 13. Cara Penggunaan Website	17
Gambar 14. Tampilan Tentang Proyek Ini	17
Gambar 15. Tampilan Awal Halaman Finder	18
Gambar 16. Tampilan Error Message	18
Gambar 17. Tampilan Unggah Album	18
Gambar 18. Tampilan Unggah Lagu	18
Gambar 19. Tampilan Unggah Dataset dan Mapper	19
Gambar 20. Tampilan Pemberitahuan Generate Auto Mapper	19
Gambar 21. Hasil Random Generate Mapper	19
Gambar 22. Tampilan Proses Searching	19
Gambar 23. Hasil Pencarian Album	20
Gambar 24. Hasil Pencarian Album Menunjukkan Waktu Eksekusi	20
Gambar 25. Hasil Pencarian Lagu	20
Gambar 26. Hasil Pencarian Lagu Menunjukkan Waktu Eksekusi	21
Gambar 27. Tampilan Penyimpanan Otomatis	21
Gambar 28. Tampilan Halaman About Us Anggota 1	21
Gambar 29. Tampilan Halaman About Us Anggota 2	22
Gambar 30. Tampilan Halaman About Us Anggota 3	22

DAFTAR TABEL

Tabel 5.1 Tabel Hasil Eksperimen

17

BAB 1

DESKRIPSI MASALAH

Suara menjadi salah satu elemen yang paling penting dalam kehidupan manusia. Melalui suara, manusia berkomunikasi, berbicara, mendengarkan, dan memproses informasi yang diterima oleh otak. Tidak hanya sebagai alat komunikasi, suara juga menjadi medium yang tak ternilai untuk menciptakan karya seni. Salah satu contoh pemanfaatan suara dalam seni dan teknologi adalah alat pendekripsi lagu. Manusia menggunakan indera pendengar untuk mengenali suara. Melalui proses pada otak, manusia mampu memberikan kesimpulan mengenai jenis suara tersebut. Menariknya, teknologi kini mampu meniru, bahkan melampaui kemampuan manusia dalam mendekripsi dan menginterpretasikan suara melalui algoritma canggih. Dengan menggunakan algoritma apapun, konsep dari pendekripsi dan interpretasi suara itu 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. Aplikasi Shazam yang memanfaatkan konsep sistem temu balik

Selain suara, manusia memiliki indera penglihatan yang memungkinkan kita untuk melihat berbagai warna dan gambar. Dengan begitu pesatnya perkembangan teknologi, teknologi komputasi kini mampu memiliki kapabilitas serupa. Teknologi tidak hanya mampu melihat gambar sebagaimana manusia, tetapi juga dapat merepresentasikan gambar tersebut dalam bentuk angka-angka. Algoritma seperti *Eigenvalue*, *Cosine Similarity*, dan *Euclidean Distance* sering digunakan untuk mendukung analisis dan pengolahan data gambar. Matriks,

sebagai komponen penting dalam aplikasi aljabar vektor, kembali menjadi inti dari tugas ini. Tim pengembang diminta untuk menciptakan sebuah aplikasi yang mampu menerima input berupa lagu dan mendeteksi nama lagu beserta beberapa detail lainnya. Tidak hanya itu, digunakan pula konsep Principal Component Analysis (PCA) untuk mencari kumpulan audio melalui deteksi wajah berbagai orang.

Pada Tugas Besar 2 ini, diterapkan dua jenis *Information Retrieval*, yakni *Image Retrieval* dan *Music Information Retrieval*. *Image Retrieval* adalah proses memasukkan input berupa gambar dan mendapatkan gambar serupa yang ada di dalam basis data berdasarkan informasi dan perhitungan tertentu. PCA digunakan untuk mengurangi dimensi data sehingga pencarian menjadi lebih efisien. *Music Information Retrieval* (MIR) adalah proses memasukkan input berupa audio dan mendapatkan audio yang sesuai di dalam basis data berdasarkan fitur dan analisis tertentu. Dalam tugas ini, pengimplementasian MIR dilakukan dengan pendekatan berbasis humming—yaitu pengenalan suara berdasarkan melodi yang dinyanyikan atau direkam.

BAB 2

TEORI SINGKAT

2.1. Image Retrieval

Principal Component Analysis (PCA) adalah teknik reduksi dimensi yang digunakan untuk mengurangi kompleksitas data visual dengan tetap mempertahankan informasi yang paling penting. Dalam *Image Retrieval*, PCA digunakan untuk merepresentasikan gambar dengan jumlah fitur yang lebih sedikit, tetapi cukup untuk membedakan gambar satu dengan yang lain. PCA mengubah data berdimensi tinggi menjadi beberapa dimensi yang lebih kecil disebut dengan *principal components*. Penggunaan *principal components* memastikan bahwa pengolahan data tidak menghilangkan esensi atau pola utama dalam data tersebut. Hasil data yang didapatkan dari PCA ini akan berupa *eigenvector* dan proyeksi data. Langkah-langkah dalam melakukan *Image Retrieval* melalui metode PCA adalah sebagai berikut.

2.1.1. Image Processing and Loading

Image Processing and Loading adalah tahapan pemrosesan seluruh gambar yang ada pada *dataset*. Gambar terlebih dahulu diubah menjadi grayscale untuk mengurangi kompleksitas gambar sehingga fokus komparasi hanya pada komponen terang dan gelap gambar. Setiap gambar direpresentasikan dalam intensitas pixel saja tanpa informasi warna, dengan rumus yang digunakan untuk *grayscale* sebagai berikut

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

Gambar 2. Perhitungan konversi gambar berwarna menjadi *grayscale*

Gambar lalu diubah besarnya sehingga ukurannya sama untuk seluruh gambar. Proses pengubahan ukuran ini dilakukan menggunakan algoritma *nearest neighbor* yang mengambil rasio gambar dengan ukuran yang diinginkan sebagai perhitungan pelebaran atau penyempitan piksel. Ukuran seluruh gambar dibuat konsisten untuk membentuk perhitungan yang semakin akurat.

Kemudian, vektor *grayscale* pada gambar diubah menjadi 1 dimensi menggunakan fungsi bawaan Python berupa *flatten*. Proses menjadikan vektor sebagai 1 dimensi dikhususkan untuk membentuk dimensi vektor yang dapat dilakukan pemrosesan data. Jika

gambar memiliki dimensi $M \times N$, maka hasilnya adalah vektor dengan panjang $M \cdot N$ sebagai berikut.

$$\mathbf{I} = [I_1, I_2, \dots, I_{M \cdot N}]$$

Gambar 3. Hasil vektor untuk suatu gambar berdimensi $M \times N$ piksel

2.1.2. Data Centering (Standardization)

Setelah image diproses, dilakukan standarisasi data di sekitar nilai 0. Standarisasi dilakukan dengan cara menghitung rata rata dari setiap gambar untuk suatu piksel. Digunakan rumus:

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

Gambar 4. Rumus perhitungan rata-rata keseluruhan data vektor hasil pengolahan dataset

di mana:

- x_{ij} : nilai piksel ke- j pada gambar ke- i
- N : jumlah total gambar dalam *dataset*

Lalu, dilakukan standarisasi dengan mengurangkan piksel tersebut dengan rata-rata yang sudah dihitung melalui rumus:

$$x'_{ij} = x_{ij} - \mu_j$$

Gambar 5. Proses standarisasi data vektor hasil pengolahan dataset

2.1.3. PCA Computation Using Singular Value Decomposition (SVD)

Kemudian, dilakukan dekomposisi nilai singular untuk mendapatkan komponen utama. Matriks yang didekomposisi merupakan matriks vektor *dataset* gambar secara keseluruhan, tanpa dilakukan perhitungan matriks kovarian terlebih dahulu. Hal ini menyesuaikan dengan library *NumPy* yang dapat melakukan dekomposisi dan mencari matriks eigenvector langsung dari matriks vektor *dataset* sehingga perhitungan dapat dilakukan dengan lebih cepat. Untuk mempertahankan skala, data pada matriks vektor *dataset* dibagi dengan akar dari jumlah vektor terlebih dahulu sebelum didekomposisikan.

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

Gambar 6. Dekomposisi SVD

dengan:

- C: matriks vektor *dataset* yang telah dibagi dengan akar dari banyaknya vektor
- U: matriks eigenvector (komponen utama)
- Σ: matriks eigenvalue (menunjukkan varian data di sepanjang komponen utama)

Lalu digunakan k komponen utama untuk diproyeksikan ke setiap vektor dalam matriks vektor *dataset* semula. Penentuan nilai k didasarkan terhadap informasi variansi data yang ditentukan berdasarkan matriks sigma dengan batasan kontribusi data berada pada batas 90%. k -komponen utama kemudian diproyeksikan terhadap matriks vektor *dataset*

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

Gambar 7. Persamaan untuk menentukan matriks vektor *dataset* setelah proyeksi

Dengan:

- Z: matriks hasil proyeksi
- X': matriks vektor *dataset*
- \mathbf{U}_k : matriks *eigenvector* dengan k -dimensi

2.1.4. Similarity Computation

Pencarian kesamaan didasarkan terhadap jarak Euclidean antara setiap vektor hasil pengolahan *dataset* dengan vektor hasil pengolahan *query*. Gambar *query* sebelumnya direpresentasikan terlebih dahulu dengan proyeksi yang sama dengan matriks vektor *dataset*.

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

Gambar 8. Perhitungan proyeksi vektor *query*

Dengan:

- q: Vektor proyeksi *query* terhadap ruang komponen utama
- q': Hasil pengolahan gambar *query* menjadi vektor, sesuai dengan pengolahan *dataset* sebelumnya

- μ : Rata-rata piksel dari *dataset*
- U_k : Matriks *eigenvector* dengan k -dimensi utama, berperan sebagai ruang komponen utama

Setelah keseluruhan gambar telah dipastikan berupa vektor hasil proyeksi, perhitungan tingkat kedekatan antara setiap gambar dihitung menggunakan jarak Euclidean untuk kemudian diurutkan dari jarak terkecil.

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

Gambar 9. Perhitungan jarak Euclidean

Dengan:

- $d(\mathbf{q}, \mathbf{z}_i)$: jarak antara gambar *query* \mathbf{q} dan gambar ke- i dalam ruang komponen utama
- \mathbf{z}_i : Vektor proyeksi gambar ke- i dalam ruang komponen utama
- q_j : Elemen ke- j pada vektor proyeksi *query* \mathbf{q}
- z_{ij} : Elemen ke- j dari \mathbf{z}_i
- k : jumlah dimensi dalam ruang komponen utama

2.2. Music Information Retrieval (MIR)

Music Information Retrieval (MIR) merupakan teknologi yang memungkinkan sistem untuk menganalisis, memahami, dan mengambil informasi musik dari basis data digital. Salah satu metode MIR yang populer untuk digunakan adalah *query by humming* yang memungkinkan pengguna mencari lagu hanya dengan senandung (*humming*) melodi.

Terdapat beberapa langkah utama yang dilakukan dalam tahapan *query by humming*. Pertama, dilakukan pemrosesan audio terhadap suara yang diinput direkam atau diterima dan dipersiapkan untuk tahap berikutnya. Selanjutnya, data audio tersebut melalui tahap ekstraksi fitur vektor, yaitu proses konversi data audio menjadi representasi numerik berupa vektor fitur yang dapat dianalisis. Terakhir, pada tahap pencarian similaritas vektor, vektor fitur yang dihasilkan dibandingkan dengan dataset untuk menemukan hasil yang paling sesuai atau

diatas nilai kemiripan/similaritas yang telah ditentukan. Berikut penjelasan langkah demi langkah yang dilakukan tim pengembang untuk menciptakan sistem MIR:

2.2.1. Preprocessing File MIDI

Preprocessing file MIDI Dimulai dengan membaca *file* MIDI dan memastikan *file* tidak memiliki data byte yang tidak valid. *File* MIDI yang sudah diperbaiki akan disimpan ulang di lokasi folder yang sama. Data byte yang berada pada rentang di bawah 0 akan diubah menjadi 0, sedangkan data byte yang berada pada rentang di atas 127 akan diubah menjadi 127.

2.2.2. Pemrosesan Audio MIDI

Langkah berikutnya adalah mengekstrak not melodi utama dari *file* MIDI. Pemrosesan audio dalam sistem *query by humming* menggunakan *file* MIDI dengan fokus pada *track* melodi utama. *Track* melodi utama umumnya berada pada *channel* 1, sehingga apabila ditemukan *channel* 1, maka otomatis *channel* tersebut menjadi *channel* utamanya. Apabila tidak ditemukan *channel* 1, maka dicari *channel* dengan jumlah *notes* di atas 50 dan *pitch* berkisar di antara 60-80. Kedua faktor tersebut memberikan kemungkinan tinggi bahwa *channel* berperan sebagai *channel* utama. Apabila tidak terdapat *channel* dengan karakteristik tersebut, dilakukan pencarian nilai maksimum *notes* sebagai *channel* utama.

File MIDI kemudian dinormalisasi terhadap tempo dan *pitch*-nya. Setelah dinormalisasi, *file* diproses menggunakan metode *windowing* yang membagi melodi menjadi segmen 20 *beat* dengan *sliding window* 4 *beat*. Teknik ini memungkinkan pencocokan fleksibel dari berbagai bagian lagu yang mungkin diingat oleh pengguna.

2.2.3. Ekstraksi Fitur

Fitur melodi kemudian diekstrak dalam bentuk histogram. Terdapat tiga jenis histogram distribusi *tone*, dengan yang pertama merupakan Fitur Absolute Tone Based (ATB) yang menghitung frekuensi kemunculan setiap nada berdasarkan skala MIDI (0-127). Histogram yang dihasilkan memberikan gambaran distribusi absolut nada dalam data. Hal ini penting untuk menangkap karakteristik statis melodi dalam sinyal audio. Fitur ini didapatkan dengan membuat histogram dengan 128 bin, sesuai dengan rentang nada MIDI dari 0 hingga 127. Frekuensi kemunculan masing-masing nada MIDI dalam data kemudian dihitung dan dinormalisasi untuk mendapatkan distribusi yang terstandarisasi.

Histogram distribusi *tone* yang kedua adalah Fitur Relative Tone Based (RTB) untuk menganalisis perubahan antara nada yang berurutan, menghasilkan histogram dengan nilai

dari -127 hingga +127. RTB berguna untuk memahami pola interval melodi, yang lebih relevan dalam mencocokkan *humming* dengan dataset yang tidak bergantung pada *pitch* absolut. Fitur ini didapatkan dengan membangun histogram yang memiliki 255 bin dengan rentang nilai dari -127 hingga +127. Selanjutnya, selisih antara nada-nada yang berurutan dalam data dihitung dan dilakukan normalisasi terhadap histogram yang telah dibuat.

Terakhir, histogram distribusi *tone* yang dimanfaatkan adalah Fitur First Tone Based (FTB) yang berfokus pada perbedaan antara setiap nada dengan nada pertama, menciptakan histogram yang mencerminkan hubungan relatif terhadap titik referensi awal. Pendekatan ini membantu menangkap struktur relatif nada yang lebih stabil terhadap variasi pitch pengguna. Histogram dibentuk dengan 255 bin mencakup rentang nilai dari -127 hingga +127. Kemudian, selisih antara setiap nada dalam data dengan nada pertama dihitung. Histogram yang dihasilkan lalu dinormalisasi untuk menghasilkan distribusi yang seimbang dan memastikan bahwa semua nilai dalam histogram berada dalam skala probabilitas.

Ketiga histogram distribusi *tone* yang dibentuk melalui tahapan normalisasi sebagai standar yang memastikan bahwa ketiga histogram minim dari redundansi dan inkonsistensi data. Berikut adalah formula umum dari normalisasi yang digunakan:

$$H_{norm} = \frac{H[d]}{\sum_{d=1}^{127} H[d]}$$

Gambar 10. Formula Umum Normalisasi

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

2.2.4. Perhitungan Similaritas

Similaritas dihitung dengan melakukan perbandingan antara file query dengan file dalam basis data. Setiap histogram menjadi sebuah vektor dan dilakukan perhitungan kemiripannya menggunakan *cosine similarity*. *Cosine Similarity* adalah ukuran untuk menentukan seberapa mirip dua vektor dalam ruang berdimensi tinggi, dengan menghitung sudut cosinus di antara keduanya. Semakin kecil sudutnya (semakin dekat ke 1 hasilnya), semakin mirip kedua vektor tersebut.

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

Gambar 11. Formula Cosine Similarity

Setelah melakukan pengujian, kelompok setuju untuk memberikan bobot 40% terhadap pencocokan histogram hasil analisis Absolute Tone Based, 30% untuk pencocokan histogram hasil analisis Relative Tone Based, dan 30% untuk pencocokan histogram hasil analisis First Tone Based. Sebagai representasi terhadap data MIDI yang memiliki banyak variabel terkait, seperti tempo, notasi, durasi, dan lain sebagainya, ATB memiliki gambaran yang lebih kuat dan stabil dalam menyoroti persamaan *query* dengan *dataset*, karena lebih tidak berpengaruh terhadap perubahan relatif atau posisi dalam data MIDI. Walaupun demikian, ketiga fitur tetap memberikan kontribusi penting, sehingga pemberian perbedaan bobot yang tidak begitu besar untuk masing-masing komponen tetap mempertimbangkan ketiga fitur dengan baik tanpa mengandalkan salah satu sumber informasi.

BAB 3

ARSITEKTUR WEBSITE

3.1. Implementasi Program Berbasis Website

Tugas besar 2 mata kuliah Aljabar Linier dan Geometri mengharuskan keseluruhan program untuk dapat dijalankan dalam sebuah *website*. Oleh karena itu, kelompok membentuk *website* dengan *back-end* dan *front-end* dengan bahasanya masing-masing. *Back-end* diimplementasikan menggunakan bahasa pemrograman Python, sedangkan *framework* Next.JS dimanfaatkan dalam implementasi *front-end*. Lalu, digunakan Flask API sebagai penghubung dari keduanya.

3.1.1. Python

Python adalah bahasa pemrograman yang umum digunakan dalam pembentukan dan pengembangan suatu situs *web*. Bahasa pemrograman ini termasuk ke dalam bahasa pemrograman yang ditujukan dalam pembentukan secara umum, yaitu tidak terdapat batasan tertentu dalam pembentukan berbagai jenis program. Karena karakteristiknya sebagai bahasa pemrograman umum dan kemudahan penggunaannya, Python menjadi salah satu bahasa pemrograman yang populer untuk digunakan.

Dalam konteks penyelesaian tugas besar 2 mata kuliah Aljabar Linier dan Geometri, pemilihan penggunaan bahasa pemrograman Python adalah variasi *library* / *dependencies* yang diberikan oleh Python. Python memberikan banyak *library* yang dapat dimanfaatkan dengan baik dalam pembentukan program, meliputi NumPy, PIL, os, scipy, dan lain sebagainya.

3.1.2. Next.JS

Next.js adalah *framework* yang siap digunakan dalam pembangunan projek dengan mudah dan cepat secara produksi. Ketika Next.js diatur untuk pertama kalinya, projek dapat dijalankan secara langsung tanpa perlu mengubah mode pengembangan menjadi mode produksi terlebih dahulu. Selain itu, Next.js memiliki keunggulan-keunggulan yang membantu kinerja *website* berjalan dengan lebih efisien, seperti pembentukan HTML saat *browser* memuat halaman, pengaturan projek yang mudah dan sederhana menggunakan Node.js tanpa kebutuhan *plug-ins* atau *libraries* tambahan, serta penerapan *code splitting* yang mampu mengurangi waktu loading secara signifikan. Next.JS menjadi *framework* yang sesuai untuk

digunakan dalam implementasi tugas besar 2 mata kuliah Aljabar Linier dan Geometri mengingat *performance benchmark* dari *website* menjadi aspek yang diperhatikan.

3.1.3. Flask API

Flask berperan sebagai alat bantu dalam proses pengembangan *web development*. Menggunakan Flask, pembangunan *web* dapat dibangun dengan lebih efisien memanfaatkan kemampuan Flask untuk mengintegrasikan Python dengan lebih ringan dan mudah. Flask mengurangi ketergantungan terhadap *libraries* dan *external extensions*, sehingga memungkinkan untuk membentuk prosedur inti *web* dengan penggunaan sintaks sederhana.

Flask API digunakan sebagai perantara antara *front-end* dan *back-end* karena karakteristiknya yang ringan, mudah untuk dipahami, bersifat fleksibel, mudah diimplementasikan, dan bekerja dengan baik untuk pengembangan program dalam tahapan *prototype*. Selain itu, Flask API juga memungkinkan pengiriman data dalam bentuk *file JSON*, sesuai dengan kebutuhan program yang menerima *file mapper* dalam bentuk *file JSON*.

3.2. Struktur Front End Website

Pembagian halaman pada *website* yang telah kami bentuk bertujuan untuk memudahkan pengguna dalam mengakses program *Image Retrieval* dan *Music Information Retrieval* serta informasi tambahan seperti cara penggunaan dan data anggota kelompok. Setiap halaman menggunakan sistem *scrolling* dan untuk menampilkan data digunakan *pagination* agar tidak terjadi *infinite scrolling*. Terdapat pula bar navigasi yang memungkinkan pengguna untuk berganti halaman.

3.2.1. Halaman Utama (Home)

Halaman ini berisi deskripsi singkat program, cara penggunaan, dan penjelasan singkat mengenai Music Information Retrieval dan Album Information Retrieval. Deskripsi program berisi mengenai kegunaan dari program. Cara penggunaan terbagi menjadi dua, untuk pencarian album dan pencarian lagu.

3.2.2. Halaman Finder

Halaman ini berisi program utama yang memungkinkan pengguna untuk mencari album dan lagu yang diinginkan. Pada halaman ini pengguna dapat mengunggah query, dataset, dan mapper. Pengguna juga dapat memilih ingin melihat hasil pencarian album atau lagu.

3.2.3. Halaman About Us

Halaman ini berisi informasi singkat mengenai anggota kelompok. Informasi yang diberikan adalah foto diri, nama lengkap, dan NIM. Terdapat pula sedikit tambahan berupa kata-kata untuk tugas besar ini.

3.3. Struktur Back End Website

Pembagian fungsi-fungsi pada *back-end* dilakukan berdasarkan tujuan pembentukan fungsi. Kelompok membentuk tiga *file* Python utama, yaitu MIR.py sebagai *file* yang bertanggung jawab atas *music information retrieval*, AIR.py yang bertanggung jawab atas *image retrieval*, dan mapper.py yang bertanggung jawab atas *file* mapper.json. Pemisahan / modularisasi *file* didasarkan terhadap perbedaan kegunaan fungsi yang dibentuk serta pembagian tugas dalam kelompok. Kemudian, setelah setiap *file* dikatakan mampu menjalankan fungsinya dengan baik, dilakukan integrasi secara menyeluruh ke dalam *file* app.py yang digunakan untuk membangun API *routes* dalam Next.JS. *Back-end* Python, menggunakan Flask API, diintegrasikan ke dalam *framework front-end* Next.JS melalui app.py yang berperan dalam menangani permintaan API dan logika lainnya.

3.4. Prosedur Penggunaan Website

Berikut adalah tata cara penggunaan *website*. Sebelum melakukan tahapan berikut, pengguna diminta untuk telah melakukan langkah-langkah menjalankan program yang tertera pada bagian README dalam *repository* GitHub tugas besar kelompok.

1. Pengguna dapat melakukan navigasi dengan menggunakan tombol pada tengah atas halaman. Tersedia navigasi untuk pergi ke ‘Home’ (tombol berupa logo), ‘Finder’, dan ‘About Us’.
2. Tekan tombol *Finder* pada bar navigasi untuk mencari album atau lagu yang diinginkan.
3. Pada bagian kiri halaman, pengguna dapat mengunggah gambar album atau lagu yang ingin dicari, dataset, dan mapper. Pengguna diimbau untuk mengunggah album atau lagu terlebih dahulu untuk bisa menjalankan program.
4. Unggah dataset audio dan dataset gambar yang diinginkan. Tipe file yang diterima adalah .zip dengan dataset audio adalah .mid dan dataset gambar adalah jpg/png/jpeg.

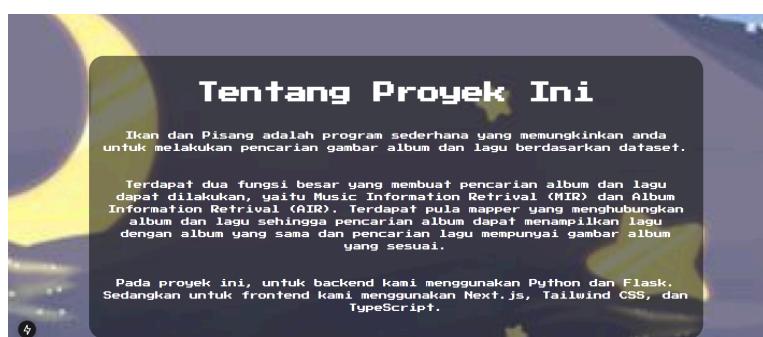
5. Unggah mapper yang menghubungkan antar album dan gambar. Apabila pengguna tidak mengunggah mapper, program secara otomatis melakukan pengacakan dalam memasangkan audio dan gambar dari dataset.
6. Tekan tombol ‘Process’ setelah melakukan semua pengunggahan yang diperlukan dan tunggu beberapa saat.
7. Pada bagian kanan halaman, pengguna dapat memilih untuk menekan ‘Album’ dan ‘Music’. Kedua tombol ini berguna untuk melihat hasil pencarian. Tombol ‘Album’ untuk melihat hasil pencarian album dan tombol ‘Music’ untuk melihat hasil pencarian lagu.
8. Hasil yang ditampilkan sudah terurut dari album atau lagu yang memiliki tingkat kemiripan tertinggi hingga terendah dengan batas 55% adalah persentase terendah. Hasil ini juga disimpan pada file json.

BAB 4

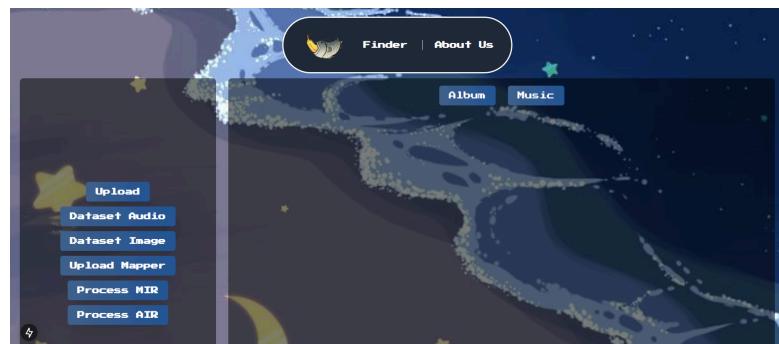
EKSPERIMENT

Pada bab ini ditampilkan hasil dari eksperimen yang dilakukan pada *website* dalam bentuk tabel.

Tabel 5.1 Tabel Hasil Eksperimen

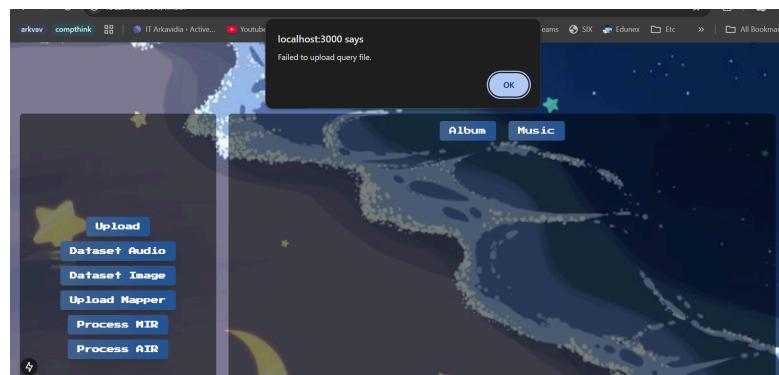
Eksperimen	Hasil
Halaman utama <i>website</i> yang berisi deskripsi singkat, cara penggunaan, dan penjelasan tambahan	
	Gambar 12. Tampilan Deskripsi Singkat
	
	Gambar 13. Cara Penggunaan <i>Website</i>
	
	Gambar 14. Tampilan Tentang Proyek Ini

Halaman *Finder* sebelum dilakukan pengunggahan album/lagu, dataset, serta mapper



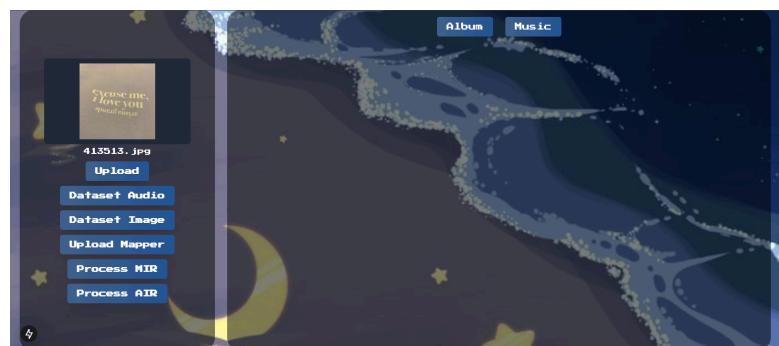
Gambar 15. Tampilan Awal Halaman Finder

Pengujian pengunggahan *file* dengan tipe *file* yang tidak sesuai (bukan merupakan mid, jpg, jpeg, png, atau zip khusus untuk dataset)



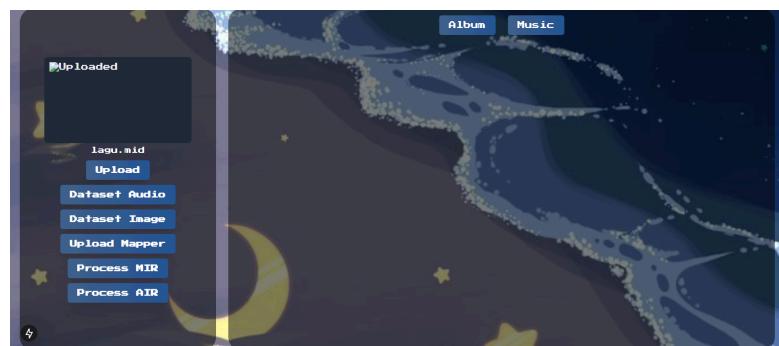
Gambar 16. Tampilan Error Message

Pengujian pengunggahan album



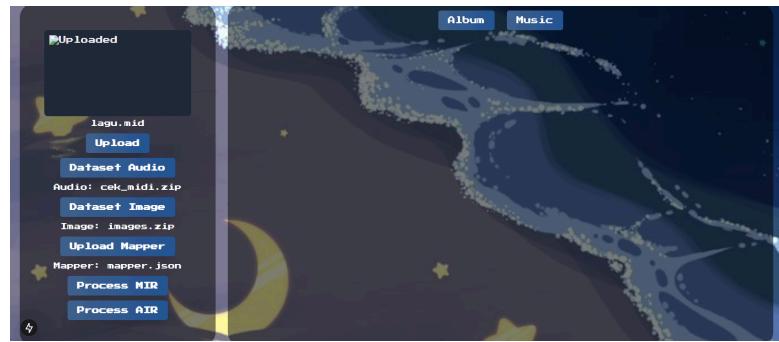
Gambar 17. Tampilan Unggah Album

Pengujian pengunggahan lagu



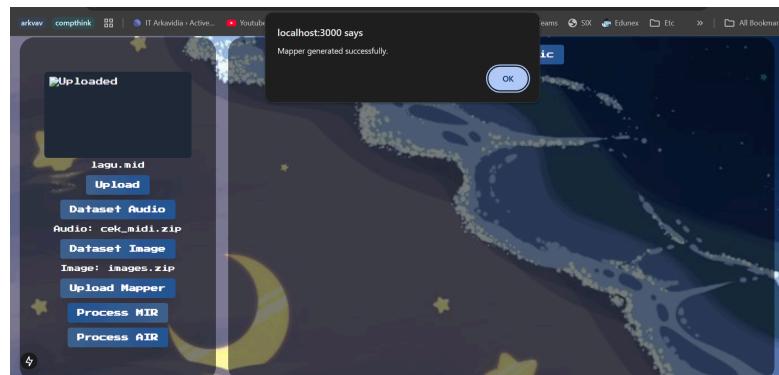
Gambar 18. Tampilan Unggah Lagu

Pengujian pengunggahan dataset dan mapper



Gambar 19. Tampilan Unggah Dataset dan Mapper

Hasil mapper ketika pengguna tidak melakukan pengunggahan mapper



Gambar 20. Tampilan Pemberitahuan Generate Auto Mapper

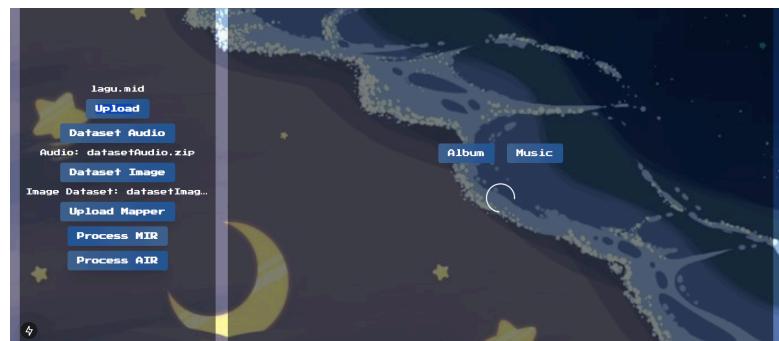
```

{
  "image": "1672897683542.jpg",
  "audioSrc": "...\\audios\\All-4-One\\I_Can_Love_You_Like_That.1.mid"
},
{
  "image": "1698839973994.jpg",
  "audioSrc": "...\\audios\\All-4-One\\I_Can_Love_You_Like_That.mid"
},
{
  "image": "20211206_185158.jpg",
  "audioSrc": "...\\audios\\All-4-One\\I_Swear.1.mid"
},
{
  "image": "20211206_185206.jpg",
  "audioSrc": "...\\audios\\All-4-One\\I_Swear.2.mid"
},
{
  "image": "306834.jpg",
  "audioSrc": "...\\audios\\All-4-One\\I_Swear.3.mid"
},
{
  "image": "311149.jpg",
  "audioSrc": "...\\audios\\All-4-One\\I_Swear.4.mid"
},
{
  "image": "311298 copy.jpg",
  "audioSrc": "...\\audios\\All-4-One\\I_Swear.mid"
}
}

```

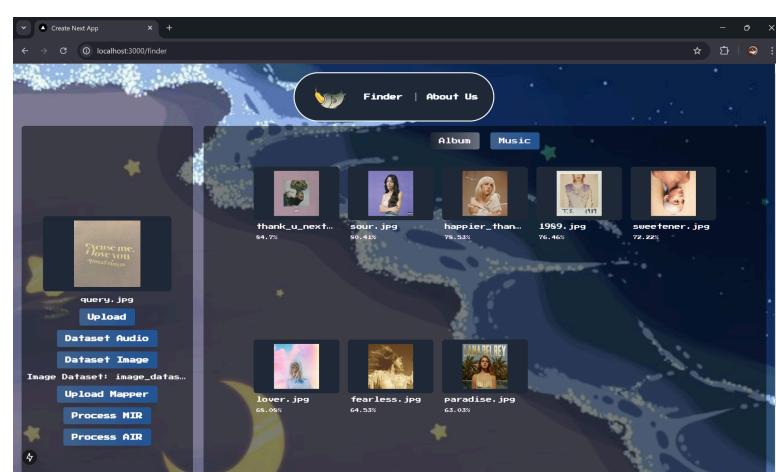
Gambar 21. Hasil Random Generate Mapper

Pengujian tombol 'Process' untuk memulai proses pencarian album dan lagu

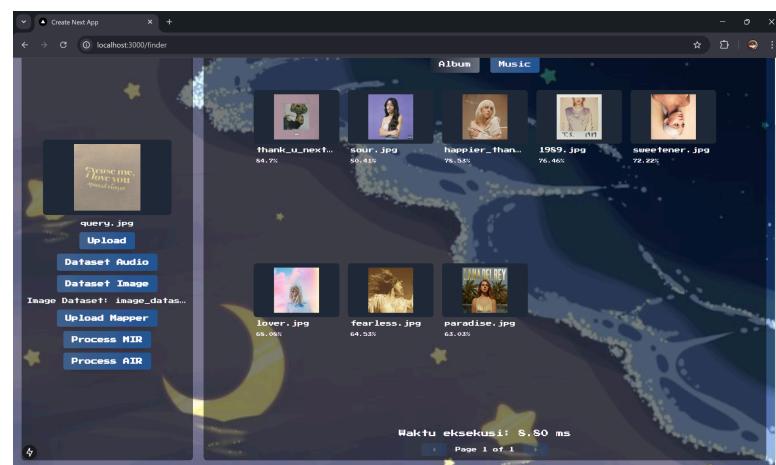


Gambar 22. Tampilan Proses Searching

Hasil dari pencarian album

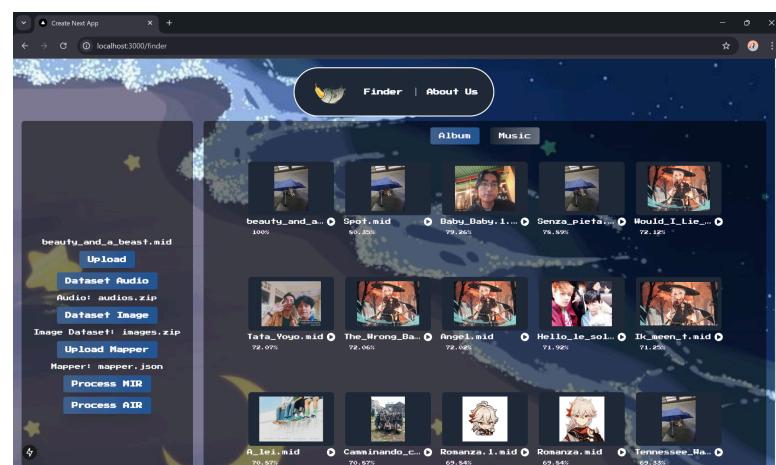


Gambar 23. Hasil Pencarian Album

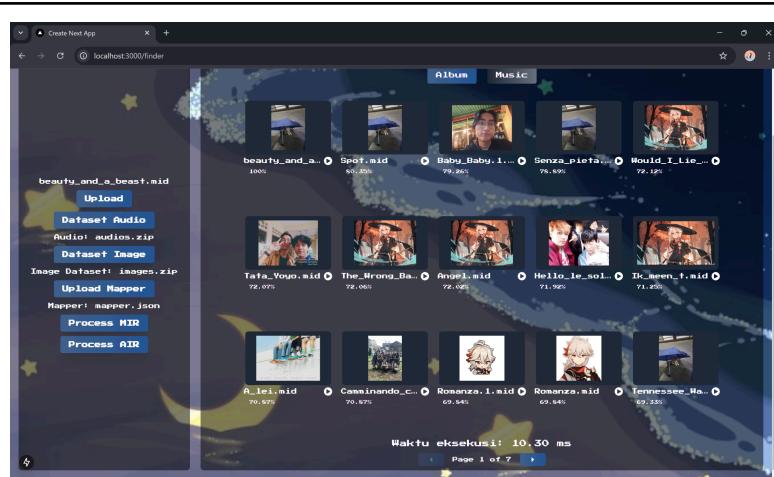


Gambar 24. Hasil Pencarian Album Menunjukkan Waktu Eksekusi

Hasil dari pencarian lagu



Gambar 25. Hasil Pencarian Lagu



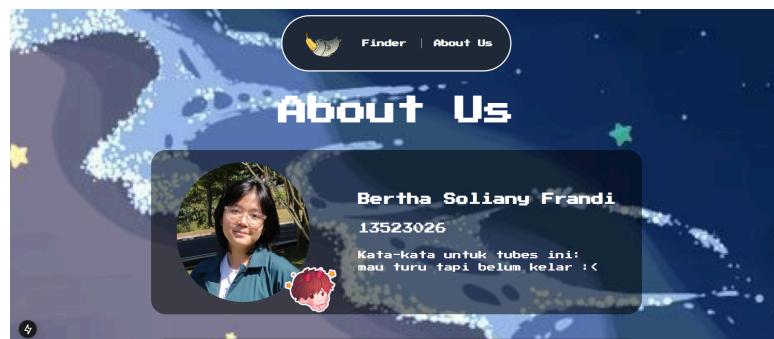
Gambar 26. Hasil Pencarian Lagu Menunjukkan Waktu Eksekusi

Hasil dari kemiripan secara otomatis tersimpan pada file

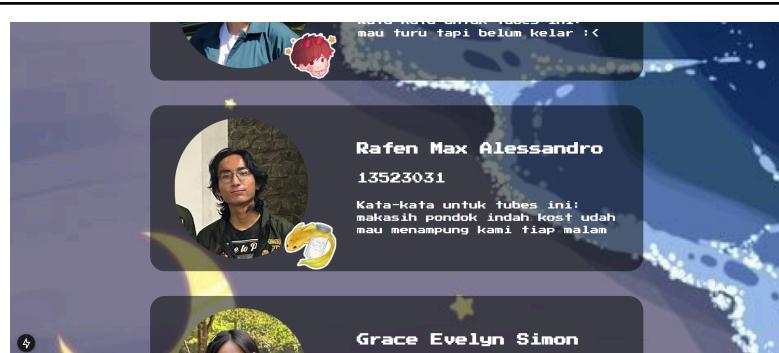
```
{} mapper.json {} output_music.json ...
test > {} output_music.json ...
1 [
2   {
3     "name": "Theme-From-'Beauty-And-The-Beast'-(Walt-Disney).mid",
4     "similarity": 100,
5     "image": null,
6     "audioSrc": null
7   },
8   {
9     "name": "I_Want_It_That_Way.mid",
10    "similarity": 57.91,
11    "image": "I_Want_It_That_Way.jpg",
12    "audioSrc": "../\\audios\\\\I_Want_It_That_Way.mid"
13 }
14 ]
```

Gambar 27. Tampilan Penyimpanan Otomatis

Halaman *About Us* berisikan identitas anggota kelompok



Gambar 28. Tampilan Halaman About Us Anggota 1



Gambar 29. Tampilan Halaman About Us Anggota 2



Gambar 30. Tampilan Halaman About Us Anggota 3

BAB 5

PENUTUP

5.1. Kesimpulan

Melalui tugas besar 2 mata kuliah Aljabar Linier dan Geometri ini, kelompok mendapatkan kesempatan besar untuk melakukan eksplorasi terhadap penggunaan aljabar vektor untuk menentukan perbandingan suatu *query* (berupa gambar atau audio) dengan *dataset* untuk menentukan kemiripan di antara keduanya. Dalam implementasinya, kelompok juga telah menggunakan konsep *Principal Component Analysis* (PCA) sebagai pendekatan dalam *Image Retrieval*, dan konsep *humming*, meliputi ekstraksi distribusi *tones* berdasarkan tiga buah *viewpoints* (*absolute tone*, *relative tone*, dan *first tone*) dan normalisasi histogram sebagai pendekatan dalam *Music Information Retrieval*.

Lebih lanjut, kelompok juga telah membentuk *website* yang menghubungkan *Image Retrieval* dan *Music Information Retrieval* ke dalam satu wadah untuk digunakan secara bersamaan. Pembentukan *website* dipastikan mengimplementasikan seluruh komponen yang dibuat dalam program. Pembentukan *website* memanfaatkan Python sebagai bahasa pemrograman dalam pembentukan *back-end*, Next.JS sebagai *framework* dalam pembentukan *front-end*, dan Flask API sebagai API yang bertanggung jawab dalam pengintegrasian *front-end* dan *back-end* sebagai *website* yang terintegrasi secara utuh. Seluruh pemilihan komponen memiliki alasan yang telah kelompok jelaskan pada *BAB 3: Arsitektur Website*.

Segala macam bentuk penggerjaan, seperti dasar teori, hasil pembentukan *source code*, alasan penggunaan algoritma tertentu, pengimplementasian fungsi, serta tampilan layar *website* hasil kerja kelompok telah kelompok dokumentasikan secara terstruktur melalui laporan ini. Meskipun masih jauh dari kata sempurna, dengan berbangga hati kelompok menyatakan bahwa kelompok telah berhasil membentuk *website* yang mampu memberikan gambar dan audio terdekat dengan *query* terhadap suatu *dataset* sebagai hasil akhir dari tugas besar kedua mata kuliah Aljabar Linier dan Geometri.

5.2. Saran

Kelompok menyadari bahwa hasil akhir yang dibentuk oleh kelompok masih jauh dari kata maksimal. Terdapat beberapa perbaikan-perbaikan yang dapat diusahakan lebih lanjut, seperti dalam pembentukan algoritma yang lebih efektif, implementasi *website*, dan *benchmark performance* dari kinerja *website* keseluruhan. Namun, mengingat waktu yang

minimal akibat kesibukan masing-masing anggotanya, pengembangan lebih lanjut diserahkan sebagai implementasi lanjutan untuk dikerjakan di waktu luang sebagai projek pribadi.

5.3. Komentar

1. 13523026: “Beneran gak suka *backend* :/”
2. 13523031: “Suka deh tubes algeo materi algeonya cuma satu baris kode 😊”
3. 13523087: “Agak stress tapi gapapa, gapapa tapi agak stress”

5.4. Refleksi

Kelompok mengucapkan terima kasih yang sebesar-besarnya kepada Pak Rila dan Pak Rinaldi selaku dosen pengampu mata kuliah Aljabar Linier dan Geometri pada kampus Ganesa, seluruh asisten yang telah memberikan perannya dalam pelaksanaan tugas besar pertama, serta kak Naufal Ramadan selaku asisten bagi kelompok kami atas kesempatan yang telah diberikan untuk mengerjakan tugas besar kedua mata kuliah Aljabar Linier dan Geometri. Melalui tugas ini, kelompok telah mendapatkan beragam bentuk ilmu-ilmu dalam lingkup keinformatikaan, pengalaman dalam membentuk proyek informatika berbasis *website*, serta latihan keterampilan-keterampilan yang relevan bagi seorang mahasiswa informatika, seperti kemampuan bekerja sama dan berkomunikasi dalam kelompok, kemampuan mengoperasikan *git* sebagai instrumen kolaborasi, kemampuan *debugging* dan melakukan *handle* terhadap *error* dalam kode pemrograman, dan keterampilan-keterampilan relevan lainnya.

Tidak hanya sebagai sarana penyelesaian capaian mata kuliah, kelompok turut memanfaatkan tugas besar ini selaku media adaptasi bagi anggota kelompok dalam lingkup jurusan yang jauh berbeda dengan lingkungan TPB dalam konteks akademis. Walaupun dihadapkan dengan berbagai macam tugas, kuis, dan tanggung jawab akademis lainnya dalam waktu yang singkat, kelompok telah mengusahakan usaha terbaik dalam mengelola dan mengatur segala bentuk tanggung jawab untuk menyelesaikan tugas secara maksimal sesuai kapasitas kemampuan masing-masing anggota kelompok. Kiranya apa yang didapatkan melalui penggeraan tugas besar ini menjadi bekal yang baik bagi anggota untuk melangsungkan keseharian perkuliahan di jurusan Teknik Informatika kedepannya.

LAMPIRAN

1. Referensi

Informatika.stei.itb.ac.id. (2024). Nilai Eigen dan Vektor Eigen (Bagian 1). Diakses pada 23 November 2024, dari <https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-19-Nilai-Eigen-dan-Vektor-Eigen-Bagian1-2023.pdf>

Informatika.stei.itb.ac.id. (2024). Nilai Eigen dan Vektor Eigen (Bagian 2). Diakses pada 23 November 2024, dari <https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-20-Nilai-Eigen-dan-Vektor-Eigen-Bagian2-2023.pdf>

Informatika.stei.itb.ac.id. (2024) Singular Value Decomposition (SVD) (Bagian 1). Diakses pada 23 November 2024, dari <https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-21-Singular-value-decomposition-Bagian1-2023.pdf>

Informatika.stei.itb.ac.id. (2024). Singular Value Decomposition (SVD) (Bagian 2). Diakses pada 23 November 2024, dari <https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-22-Singular-value-decomposition-Bagian2-2023.pdf>

Liberty, E. (2015). Singular Value Decomposition (SVD) and Principal Component Analysis (PCA). *vectors*, 1, 0.

2. Tautan Repository

<https://github.com/BerthaSoliany/Algeo02-23026>

3. Tautan Video

<https://youtu.be/Zx68sLtdHY4>