

Laporan Tugas Besar 2

IF2123 Aljabar Linier dan Geometri
Sistem Persamaan Linier, Determinan, dan Penerapannya
Semester I Tahun 2025/2026



Dibuat Oleh:

Jonathan Levi 13523132

Nathanael Shane Bennet 13524131

Ahmad Fauzan Putra 13524141

PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2025

DAFTAR ISI

DAFTAR ISI.....	2
Bagian 1	
Pendahuluan.....	4
1.1 Kisah.....	4
1.2 Tujuan.....	5
Bagian 2	
Dasar Teori.....	6
1. Nilai dan Vektor Eigen.....	6
2. Matriks Kovarians.....	6
3. Singular Value Decomposition.....	6
4. Principal Component Analysis.....	7
Bagian 3	
Desain dan Implementasi.....	8
1. Arsitektur Aplikasi.....	8
2. Teknologi yang digunakan.....	8
2.1. Frontend.....	8
2.2. Backend.....	8
3. Struktur Program dan Kelas-Kelas Penting.....	9
3.1. Struktur Backend.....	9
3.2. Struktur Frontend.....	11
4. Cara Kerja dan Alur Program.....	12
4.1. Backend.....	12
4.2. Text Preprocessing.....	12
4.3. Term-Document Matrix.....	12
4.4. TF-IDF.....	13
4.5. LSA.....	13
4.7. Caching.....	14
4.8. Rekomendasi Buku.....	14
4.9. Search Buku dengan Title.....	15
5. UI/UX.....	16
5.1. Design UI.....	16
5.2. Prinsip UI/UX.....	16
Bagian 4	
Eksperimen.....	19
1. Pencarian Berdasarkan Judul.....	19
2. Pencarian Berdasarkan Sampul (Test Case Asisten).....	21
Test Case 1.....	21

3. Pencarian Berdasarkan Sampul (Test Case Sendiri).....	23
4. Hasil Rekomendasi Buku.....	28
5. Pencarian Berdasarkan Dokumen.....	30
Penutup.....	32
Kesimpulan.....	32
Saran.....	32
Refleksi.....	32
Daftar Pustaka.....	33
Lampiran.....	34

Bagian 1

Pendahuluan

1.1 Kisah

Di Desa Konohagakure, hiduplah dua orang mahasiswa Teknik Informatika bernama Ayu dan Asep Kasep. Akhir-akhir ini, Ayu merasa kewalahan memahami materi dekomposisi matriks yang diajarkan dalam mata kuliah IF2123 Aljabar Linier dan Geometri karena ketimbang memperhatikan penjelasan dosen di depan, ia malah asyik doomscrolling di media sosial. Supaya bisa bertanggung jawab kepada kedua orang tuanya yang telah membayai kuliahnya, Ayu pun bertekad untuk memperbaiki situasi dengan belajar ke perpustakaan. Sesampainya di sana, ia takjub akan banyaknya koleksi buku yang tersedia. Sejak saat itu, Ayu pun gemar mengunjungi perpustakaan untuk mengeksplorasi berbagai macam buku yang menarik minatnya.

Di sisi lain, Asep adalah seorang mahasiswa yang bekerja paruh waktu sebagai pustakawan di perpustakaan kampus. Sejak Ayu mulai rajin berkunjung ke perpustakaan, Asep sering membantunya mencari buku-buku yang diinginkannya. Awalnya, Ayu mengira Asep adalah seorang performative male yang hanya ingin pamer di hadapannya. Apalagi, Asep sering sekali minum matcha. Namun, seiring berjalannya waktu, Ayu menyadari bahwa Asep adalah sosok yang cerdas dan tulus dalam membantu sesama. Asep juga sadar bahwa Ayu adalah pribadi yang tekun dan pantang menyerah. Akhirnya, mereka pun jatuh cinta.



Gambar 1: Anggaplah ini Ayu dan Asep.

Sumber: Jaap Buitendijk; Copyright 2007 Warner Bros. Ent.; Harry Potter Publishing Rights
J.K.R.

Asep menyadari bahwa Ayu sering kali kesulitan menemukan buku yang diinginkannya atau melanjutkan bacaan dari buku yang pernah dipinjamnya sebelumnya. Hal ini dikarenakan sistem pencarian buku di perpustakaan yang masih konvensional dan kurang efisien. Melihat hal tersebut, Asep terinspirasi untuk mengembangkan sebuah sistem temu balik informasi berbasis singular value decomposition (SVD) dan vektor eigen yang dapat membantu Ayu dan pengunjung perpustakaan lainnya dalam menemukan buku dengan lebih mudah dan cepat.

Untuk membuktikan bahwa mahasiswa Teknik Informatika memanglah solid, mari kita bantu Asep dalam merancang dan mengimplementasikan sistem temu balik informasi tersebut dengan memanfaatkan konsep-konsep aljabar linier yang telah kita pelajari di mata kuliah IF2123 Aljabar Linier dan Geometri.

1.2 Tujuan

Pada Tugas Besar 2 ini, kita akan mengembangkan sebuah aplikasi web untuk sistem perpustakaan digital. Anda akan diberikan sebuah dataset berisi koleksi buku dengan rincian judul, konten (berupa file txt), dan sampul (berupa file jpg). Aplikasi ini harus mampu menampilkan daftar buku dalam dataset dengan paginasi, menampilkan konten buku untuk dibaca, melakukan pencarian buku berdasarkan input sampul gambar dari penggunaan atau kata kunci teks dari pengguna, serta memberikan rekomendasi buku yang relevan berdasarkan buku yang sedang dibaca.

Bagian 2

Dasar Teori

1. Nilai dan Vektor Eigen

Jika matriks A adalah matriks persegi ($n \times n$), maka vektor tak nol pada R^n disebut vektor eigen dari matriks A, jika

$$Ax = \lambda x$$

dengan λ adalah nilai eigen dari matriks A; x adalah vektor eigen yang berkoresponden dengan λ .

$Ax = \lambda x$ dikalikan matriks identitas kedua ruasnya, menjadi

$$IAx = \lambda Ix$$

$$Ax = \lambda Ix$$

$$(\lambda I - A)x = 0$$

Agar $(\lambda I - A)x = 0$ memiliki solusi tak nol, maka

$$\det(\lambda I - A) = 0$$

Vektor dan nilai eigen dapat merepresentasikan fitur utama dari sekumpulan data, karena prinsip dasar PCA adalah mengurangi dimensi, tetapi mempertahankan variansi terbesar (informasi terpenting) dalam data. Dengan memilih sejumlah vektor eigen yang memiliki nilai eigen terbesar, reduksi dimensi data dapat tetap mempertahankan karakteristik yang paling signifikan dan membuang *noise* (informasi yang kurang relevan).

2. Matriks Kovarians

Matriks kovarians dalam konteks PCA adalah

$$C = \frac{1}{N} \times XX^T,$$

yang mana matriks X adalah matriks data yang tiap kolomnya adalah sebuah vektor data yang sudah dinormalisasi.

3. Singular Value Decomposition

Singular value decomposition adalah teknik dekomposisi matriks.

$$X = U\Sigma V^T$$

Σ adalah matriks berukuran $m \times n$ yang mana elemen diagonalnya adalah akar kuadrat dari nilai-nilai eigen (semakin ke bawah/kanan, semakin kecil) dari $X^T X$. V adalah matriks berukuran $n \times n$ yang mana tiap kolomnya adalah vektor eigen dari $X^T X$. U adalah matriks berukuran $m \times m$ yang mana tiap kolom adalah m vektor eigen teratas.

Singular value decomposition digunakan sebagai pengganti matriks kovarians karena setiap gambar berukuran 200×300 piksel = 60000. Jika menggunakan eigendecomposition, perkalian matriks kovarians XX^T yang harus dicari akan berukuran 60000×60000 . Ukuran matriks tersebut sangat besar dan memakan waktu dan sumber daya. Dengan SVD, X difaktorkan menjadi $U\Sigma V^T$, di mana kolom-kolom U sudah merepresentasikan vektor-vektor eigen dari matriks kovarians tanpa perlu menghitung matriks kovarians secara langsung.

4. Principal Component Analysis

Principal Component Analysis adalah teknik pengurangan dimensi data dengan tetap mempertahankan informasi dataset aslinya. Transformasi ini seperti transformasi antar ruang vektor dengan dimensi yang berbeda. Dataset ditransformasikan menggunakan basis yang dinamakan *principal component*, yang didapatkan dari proses berikut:

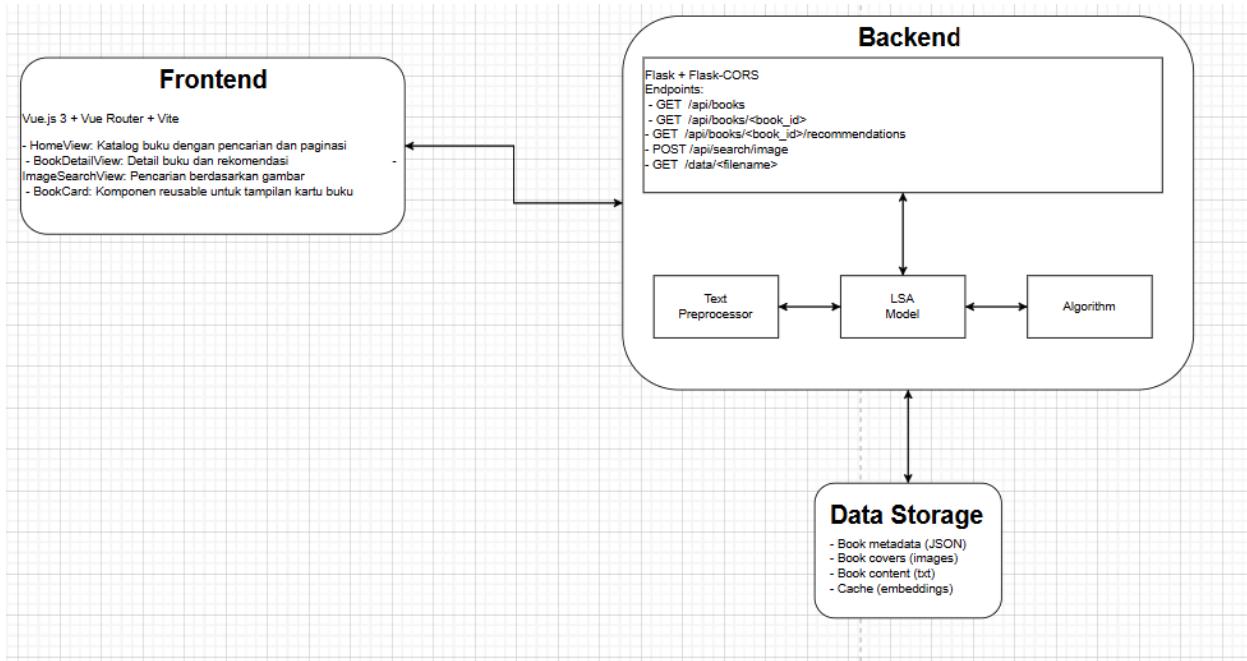
1. Standardisasi data
2. *Singular value decomposition*
3. Pemilihan sejumlah vektor eigen dari hasil SVD dengan nilai eigen terbesar dipilih menjadi *principal component*, karena akan menangkap banyak informasi dari dataset awal.

Namun, teknik PCA kurang optimal jika gambar diputar, digeser, atau di-*crop* (di-zoom), karena teknik ini bergantung pada posisi piksel. Matriks gambar akan diubah menjadi vektor, dan pergeseran posisi piksel akan mengubah representasi vektor meskipun gambarnya terlihat sama jika dilihat secara langsung. Teknik ini juga kurang optimal untuk membedakan sampul buku yang memiliki pola identik namun berbeda warna, karena gambar akan diubah ke *grayscale*.

Bagian 3

Desain dan Implementasi

1. Arsitektur Aplikasi



2. Teknologi yang digunakan

2.1. Frontend

Framework & Libraries:

- Vue.js : UI
- Vue Router : Routing
- Pinia : State management
- Vite : Build tool dan development server modern

2.2. Backend

Framework & Libraries:

- Flask : API
- Flask-CORS : Komunikasi Frontend-Backend

Computing:

- numpy : Operasi array dan komputasi numerik
- scipy : Sparse matrix (csr_matrix) untuk efisiensi memori

- nltk : Text processing

Image Processing:

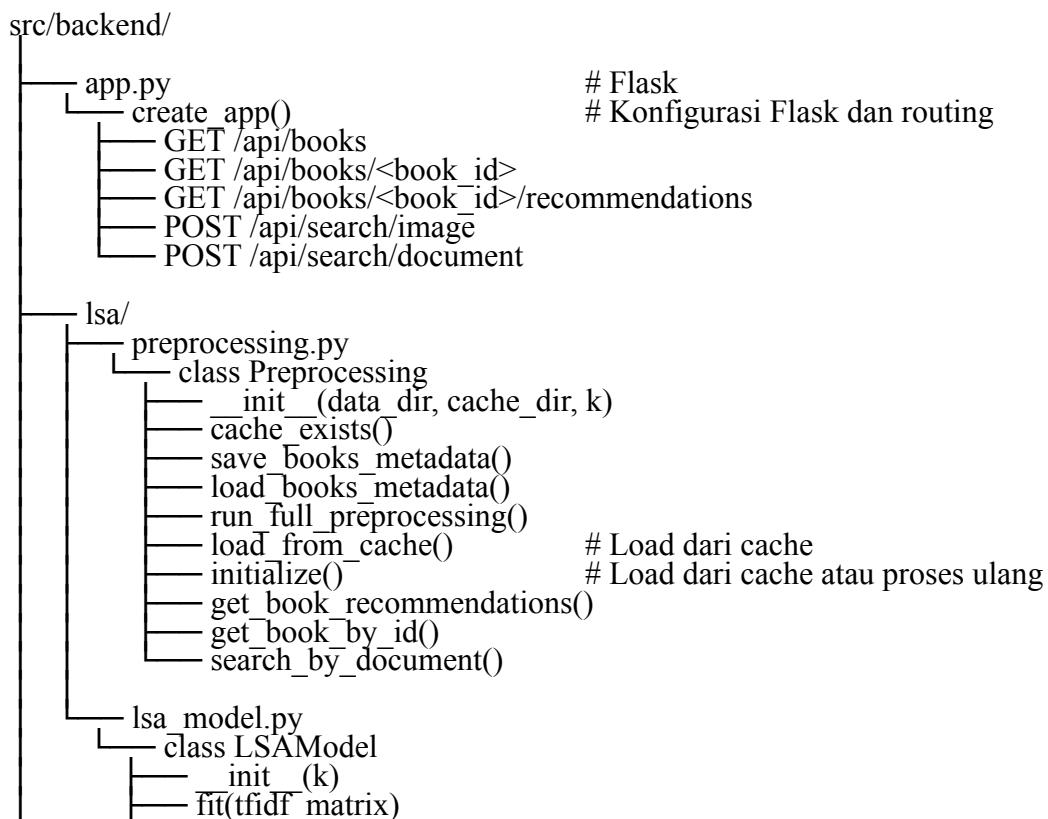
- PIL

Algoritma buatan:

- QR Decomposition : Gram-Schmidt
- QR Algorithm : Eigenvalue dan Eigenvector
- Singular Value Decomposition (SVD)
- Truncated SVD : Reduksi Dimensi
- TF-IDF (Term Frequency-Inverse Document Frequency)
- Cosine Similarity : Perhitungan similaritas
- LSA (Latent Semantic Analysis)

3. Struktur Program dan Kelas-Kelas Penting

3.1. Struktur Backend



```

    └── normalize_embeddings()
    └── get_similar_documents()
    └── find_query_embedding()
    └── find_similar_to_query()
    └── save(output_dir)           # Caching
    └── load(input_dir)          # Load dari cache

pca/
    preprocessing.py
        └── class PCAPreprocessing
            ├── init_(data_dir, cache_dir, k)
            ├── cache_exists()
            ├── save_books_metadata()
            ├── load_books_metadata()
            ├── run_full_preprocessing()
            ├── load_from_cache()          # Load dari cache
            ├── initialize()             # Load dari cache atau proses ulang
            └── get_similar_books_by_uploaded_image()

    pca_model.py
        └── class PCA
            ├── init_(k)
            ├── fit(data_dir)           # Training PCA dengan SVD
            ├── process_uploaded_image() # Preprocess gambar query
            ├── find_similar_to_uploaded()
            ├── calculate_pca()
            ├── save(output_dir)         # Caching
            └── load(input_dir)          # Load dari cache

textPreprocessor/
    data_loader.py
        └── load_dataset(data_dir)

    text_processor.py
        └── STOPWORDS
        └── class TextPreprocessor
            ├── init_()
            └── preprocess(text)
            └── preprocess_documents(documents)

    document_indexer.py          # Membuat term-document matrix
        └── class DocumentIndexer
            ├── build_vocabulary()
            ├── build_term_document_matrix()
            └── build_matrix_from_documents()

    tfidf.py
        └── class TfIdf
            ├── init_()
            ├── fit(term_doc_matrix)      # Hitung IDF vector
            ├── transform(term_doc_matrix) # Transformasi ke TF-IDF
            ├── fit_transform()
            └── transform_query()

algorithms/
    eigenvalue.py
        └── qr_decomposition(A)
        └── qr_algorithm(A)
        └── vector_length(v)
        └── create_diagonal_matrix(values)

```

```

    └── extract_diagonal(A)
    └── svd.py
        └── svd(A)
            └── truncated_svd(A, k)      # Truncated SVD untuk k komponen
    └── similarity.py
        └── find_top_k_similar(query_idx, embeddings, k)

```

3.2. Struktur Frontend

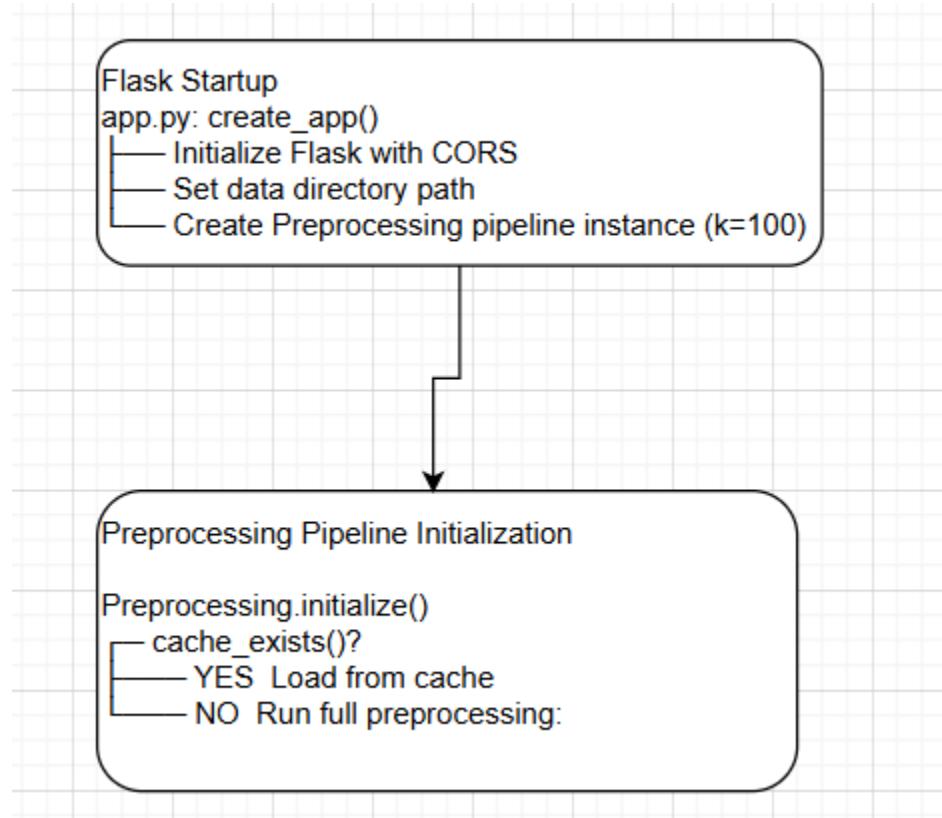
```

src/frontend/src/
    └── App.vue
    └── router/
        └── index.js
            └── /
                └── /book/:id          → HomeView
                └── /image-search       → BookDetailView
                └── /image-search       → ImageSearchView
    └── views/
        └── HomeView.vue
            └── fetchBooks()      # GET /api/books
            └── handleSearch()
            └── changePage()
        └── BookDetailView.vue
            └── fetchBook()        # GET /api/books/:id
            └── fetchRecommendations() # GET /api/books/:id/recommendations
            └── watch(route.params.id)
        └── ImageSearchView.vue
            └── searchByImage()    # POST /api/search/image
    └── components/
        └── BookCard.vue
            └── navigateToDetail()

```

4. Cara Kerja dan Alur Program

4.1. Backend



Inisialisasi untuk PCA juga sama, hanya saja menggunakan PCAPreprocessing

4.2. Text Preprocessing

Input : Raw Text dari pipeline(data)

1. **Tokenization:** Menggunakan NLTK word_tokenize untuk memecah teks menjadi list of words
2. **Normalisasi:** lowercasing dan alphabetical only
3. **Stemming:** Porter Stemmer untuk mengubah kata ke bentuk dasar
4. **Stopwords:** Menghapus kata-kata umum yang tidak membawa makna penting seperti "a", "the", "is", "while", "and", "or"

Output: List of preprocessed tokens

4.3. Term-Document Matrix

Input: List[List[str]] (preprocessed documents)

VOCABULARY:

- Kumpulkan semua term unik dari semua dokumen
- Hitung document frequency untuk setiap term
- Filter term dengan $\text{min_df} = 2$ (term harus muncul di minimal 2 dokumen)
- Sort terms
- Buat mapping: term → index

MATRIX:

- Untuk setiap dokumen, hitung term frequency (Counter)
- Bangun sparse matrix menggunakan COO format
- Konversi ke CSR (Compressed Sparse Row) format untuk efisiensi
- Shape: $(\text{num_terms} \times \text{num_documents})$

Output: CSR sparse matrix, vocabulary dict, term_list

4.4. TF-IDF

Input: Term-Document Matrix (sparse CSR)

1. **Document Frequency** : $\text{df}[i] = \text{jumlah dokumen dengan term } i$
2. **IDF** : $\text{IDF}[i] = \log_{10}(n / (1 + \text{df}[i]))$
3. **Term Frequency** : $\text{TF} = \text{term_doc_matrix} / \text{document_lengths}$
4. **TF-IDF** : $\text{TF-IDF}[i,j] = \text{TF}[i,j] * \text{IDF}[i]$

Output: TF-IDF Matrix (sparse CSR)

4.5. LSA

Input: TF-IDF Matrix ($m \times n$), $m = \text{terms}$, $n = \text{documents}$

1. **$A^T \times A$** : $n \times n$ symmetric matrix
2. **QR Algorithm**: Eigenvalue decomposition dengan iterasi QR decomposition menggunakan Gram-Schmidt Hingga konvergen
3. **Singular Values**: $\text{singular_values}[i] = \sqrt{\text{eigenvalues}[i]}$ (sorted menurun)
4. **Left Singular Vector**: $U = A * V * (\Sigma)^{(-1)}$
5. **Truncated SVD**: hanya menyimpan svd sampai komponen ke K
6. **Document Embeddings** : $V_k \times \Sigma_k$
7. **Normalization**: Normalisasi setiap embedding

Output: Normalized Document Embeddings ($n \times k$) untuk recommendation system

4.6. PCA

Input: Cover images dari dataset

1. **Load Images:** Load dari mapper.json, convert ke RGB, resize ke 200×300 pixels
2. **Grayscale Conversion:** $\text{grayscale} = 0.2126 * R + 0.7152 * G + 0.0722 * B$.
3. **Dataset Matrix:** $(60000 \times N_ENTRIES)$
4. **Mean Centering:** $u = \text{mean}$, $\text{datasetMatrix}[x] -= u$
5. **Truncated SVD:** Compute U_k dari datasetMatrix ,
6. **Coefficient Matrix:** $\text{coeffMatrix}[i] = U_k^T \times (\text{datasetMatrix})$,

Output: u , U_k , coeffMatrix

4.7. Caching

Cache Files (`./cache/`):

- `document_embeddings_normalized.npy`
- `books_metadata.json`
- `idf_vector.npy`
- `sigma_k.npy`
- `term_list.json`
- `U_k.npy`

Cache Files (`./cache_pca/`):

- `books_metadata.json`
- `coeffMatrix.npy`
- `u.npy`
- `uMatrix.npy`

Startup: (sesuai dengan backend startup)

- IF cache exists: Load dari .npy dan JSON
- IF cache NOT exists: Full preprocessing + SVD + save cache

4.8. Rekomendasi Buku

USER clicks buku di homepage

FRONTEND calls:

- GET `/api/books/{id}` untuk detail buku
- GET `/api/books/{id}/recommendations` untuk rekomendasi

BACKEND mencari similarity:

- Get query embedding dari document index
- cosine similarity: `all_embeddings @ query_embedding`
- set similarity = -1 untuk index dari dokumen yang di cari

- Sort dan return top-5 most similar books
- FRONTEND displays recommendations

4.9. Search Buku dengan Title

USER ketik query di search bar

FRONTEND (HomeView.vue): GET
`/api/books?page=1&per_page=20&search=query`

BACKEND filter books dengan substring matching:

- filtered = [b for b in books if query in b['title'].lower()]

BACKEND pagination dan return JSON

FRONTEND render dengan hasil search

4.10. Search Buku dengan Gambar

- USER upload gambar cover buku
- FRONTEND: POST /api/search/image dengan image file (multipart/form-data)
- BACKEND:
- Load image (PIL), resize ke 200×300 , convert RGB ke grayscale ($0.2126R + 0.7152G + 0.0722B$)
- Flatten ke vector (60000 dimensi), mean centering: img_vector - u
- Project ke PCA space: img_coeffs = $U_k^T \times \text{img_vector}$
- Compute Euclidean distances: distance[i] = $\|\text{img_coeffs} - \text{coeffMatrix}[i]\|$
- Sort ascending, return top-k books, convert ke similarity: max(0, $1 - \text{distance}/100000$)
- FRONTEND: Display top-k buku dengan similarity scores, covers, titles

4.11. Search Buku dengan Dokumen

USER upload file dokumen (.txt)

FRONTEND: POST /api/search/document dengan form data (file, top_k)

BACKEND:

Load File dan Preprocess

Build query vector: count term frequencies untuk terms di vocabulary

TF-IDF transform: query_tfidf = query_vector / sum(query_vector) * idf_vector

Embeddings : query_embedding = $\sum_k (-1)^k \times U_k^T \times \text{query_tfidf}$

Normalize:

Compute similarities: similarities = document_embeddings_normalized @ query_normalized
Sort descending dan return top-k results
FRONTEND: Display top-k buku

4.12.

5. UI/UX

5.1. *Design UI*

Design UI dan warna-warna yang dipilih adalah berdasarkan Material 3 Expressive.

5.2. *Prinsip UI/UX*

Responsive Design

Letak dan kerangka UI akan beradaptasi sesuai dengan lebar/ukuran layar. Misalnya, pada layar *smartphone*, buku yang ditampilkan setiap barisnya adalah 2 buku, sedangkan pada layar komputer, buku yang ditampilkan setiap barisnya bisa mencapai 5 buku.

Visibility of System Status

Saat menekan tombol "Search" pada halaman Image Search, pengguna akan melihat teks pada tombol berubah menjadi "Searching...".

Navigation bar juga menampilkan halaman pengguna sekarang dengan indikasi warna di belakang ikon yang berbeda.

Match Between the System and the Real World

Aplikasi tidak menampilkan kata-kata yang cukup teknis, seperti "LSA", "SVD", meskipun *backend* dari aplikasi ada menggunakan algoritma tersebut. Istilah tersebut diabstraksi dengan tombol "Search", "Rekomendasi", dan kata-kata lain yang mudah dimengerti orang awam.

User Control and Freedom

Pencarian buku dapat dilakukan berdasarkan *similarity threshold* yang dipilih oleh pengguna.

Consistency and Standards

Penggunaan warna dari *color palette* yang diberikan oleh dokumentasi Material Design 3, sehingga pemilihan warna tidak dilakukan secara sembarang.

Navigation bar tetap berada di atas halaman meskipun pengguna pergi ke halaman mana pun.

Komponen *card* dari buku yang digunakan pada *list* buku di halaman Home, hasil pencarian di halaman Image Search, serta pada rekomendasi di halaman setiap buku terlihat sama (**BookCard**).

Error Preventions

Tombol-tombol pada *website* tidak bisa diklik pada kondisi tertentu untuk mencegah terjadinya *error*. Misalnya, tombol "Previous" pada kumpulan buku tidak bisa diklik jika berada di halaman pertama. Tombol "Next" tidak bisa diklik jika berada di halaman terakhir. Tombol "Search" pada halaman Image Search tidak bisa diklik jika belum memilih gambar atau program sedang *loading* (sedang melakukan pencarian berdasarkan gambar).

Pada halaman Image Search, *file* yang bisa di-*upload* hanyalah *file* gambar, sedangkan *file* lainnya tidak bisa di-*upload*.

Recognition Rather than Recall

Rekomendasi buku membantu pengguna menemukan buku lain yang relevan tanpa harus mencari secara manual.

Penggunaan ikon dan warna yang berbeda sebagai *background* dari ikon pada *navigation bar* membantu pengguna untuk langsung mengenali fungsi menu dan halaman sekarang.

Flexibility and Efficiency of Use

Pengguna bisa mencari buku berdasarkan judul atau gambar sampul buku. Pada halaman pencarian berdasarkan gambar, pengguna juga dapat melakukan drag-and-drop untuk mempercepat meng-*upload* gambar.

Aesthetic and Minimalist Design

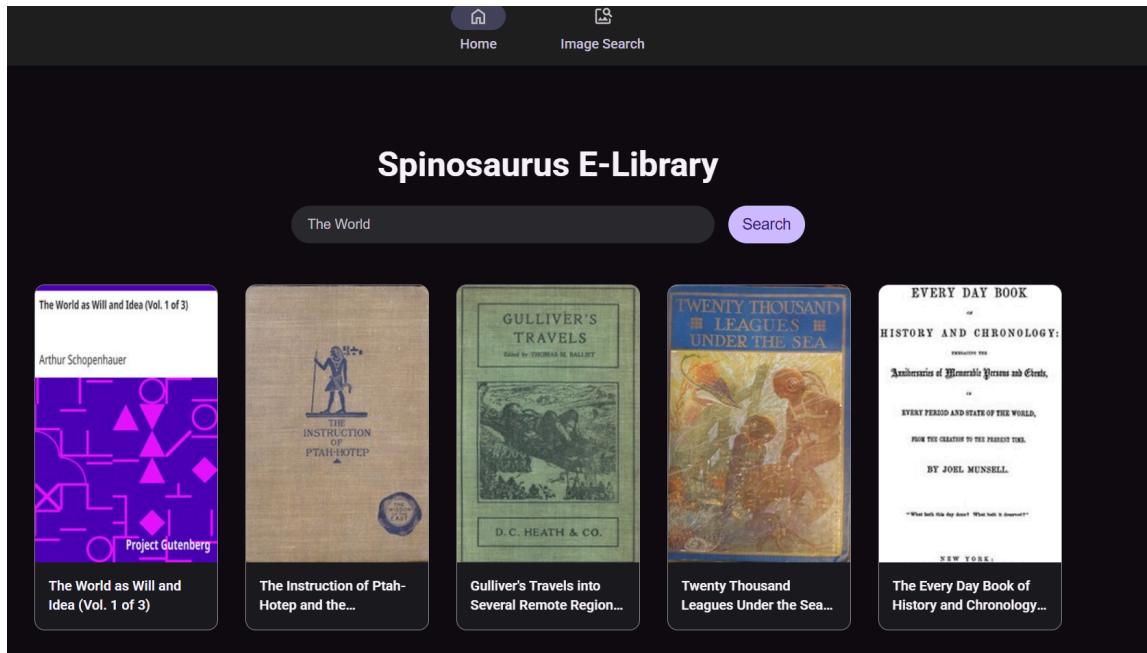
Aplikasi menggunakan *dark mode* untuk meringankan mata pada ruang yang lebih gelap. Selain itu, *design* dari aplikasi juga berdasarkan Material Design 3 yang termasuk minimalis.

Judul dari sebuah buku dibuat lebih tebal dan besar, dan informasi sekunder dibuat lebih kecil, sehingga membantu pengguna untuk fokus ke informasi penting terlebih dahulu.

Bagian 4

Eksperimen

1. Pencarian Berdasarkan Judul



Spinosaurus E-Library

Meditations

Search

Meditations

Emperor of Rome Marcus Aurelius

Meditations

Project Gutenberg

The Meditations of the Emperor Marcus Aurelius Antoninus

Emperor of Rome Marcus Aurelius

Project Gutenberg

The Meditations of the Emperor Marcus...

Spinosaurus E-Library

Will

Search

The World as Will and Idea (Vol. 1 of 3)

Arthur Schopenhauer

Project Gutenberg

The World as Will and Idea (Vol. 1 of 3)

The Will to Believe, and Other Essays in Popular Philosophy

William James

Project Gutenberg

The Will to Believe, and Other Essays in Popular...

The Works of William Shakespeare

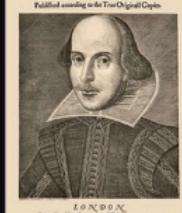
Volume I



Edited by
William George Clark
and
William Aldis Wright

The Tempest The Works of William Shakespeare...

Mr. WILLIAM SHAKESPEARE'S COMEDIES, HISTORIES, & TRAGEDIES.



The Complete Works of William Shakespeare

The Wind in the Willows

Project Gutenberg



The Wind in the Willows

2. Pencarian Berdasarkan Sampul (Test Case Asisten)

Test Case 1

Search by Image

Upload Cover Image
tc1.png

Similarity Threshold: 70%

Search

The search results show five book covers with their titles and similarity scores:

- The Wonderful Wizard of Oz** (Match: 98.3%)
- The Time Machine** (Match: 94.1%)
- Thus Spake Zarathustra: A Book for All and None** (Match: 93.9%)
- Simple Sabotage Field Manual** (Match: 93.8%)
- David Copperfield** (Match: 93.7%)

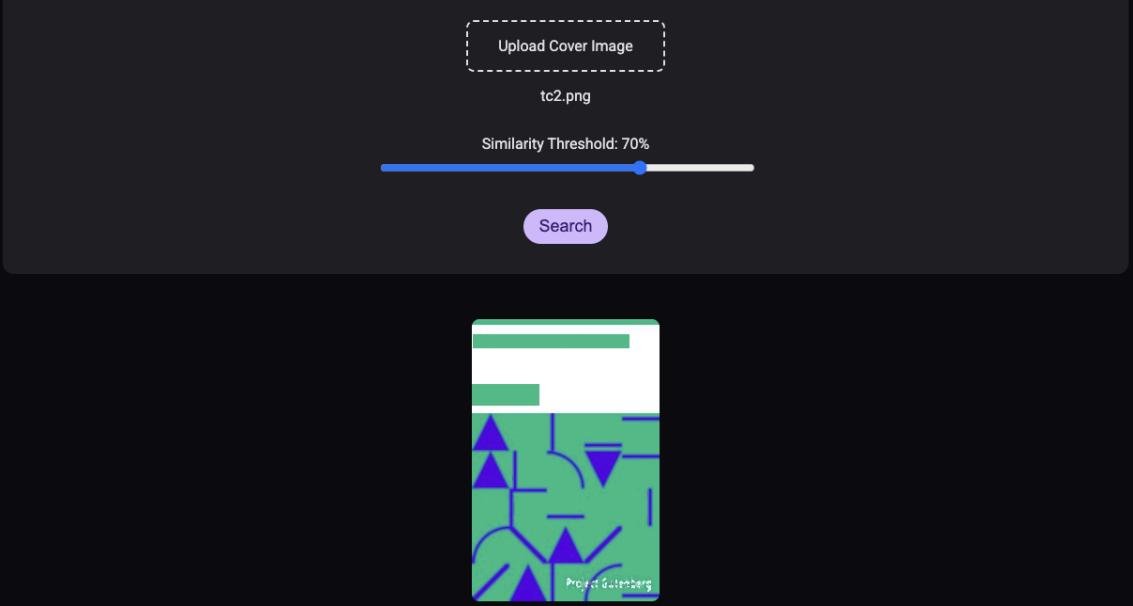
Test Case 2

Search by Image

Upload Cover Image
tc2.png

Similarity Threshold: 70%

Search



A Treatise of Human Nature

David Hume



Project Gutenberg

A Treatise of Human Nature

Match: 96.7%

A Child's History of England

Charles Dickens



Project Gutenberg

A Child's History of England

Match: 94.2%

Notes from the Underground

Fyodor Dostoyevsky



Project Gutenberg

Notes from the Underground

Match: 93.7%

How to Live on 24 Hours a Day

Arnold Bennett



Project Gutenberg

How to Live on 24 Hours a Day

Match: 93.6%

The Wonderful Wizard of Oz

L. Frank Baum



Project Gutenberg

The Wonderful Wizard of Oz

Match: 93.5%

Hasil Pencarian

Test Case 3

Search by Image

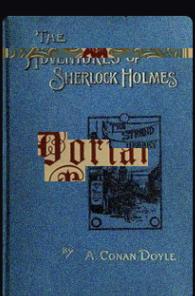
Upload Cover Image



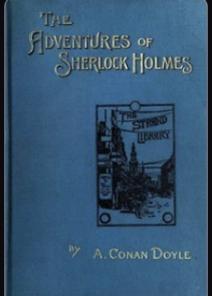
tc3 (1).png

Similarity Threshold: 70%

Search



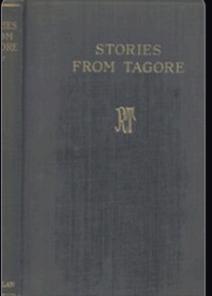
Hasil Pencarian



The Adventures of
Sherlock Holmes
by A. CONAN DOYLE

The Adventures of
Sherlock Holmes

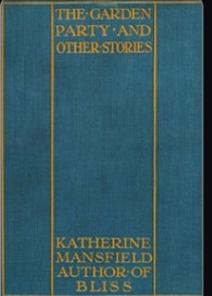
Match: 97.5%



STORIES
FROM TAGORE

Tagore

Match: 96.4%

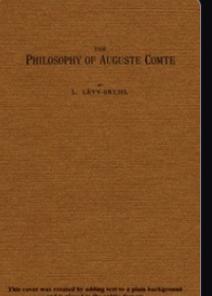


THE GARDEN
PARTY AND
OTHER STORIES

KATHERINE
MANSFIELD
AUTHOR OF
BLISS

The Garden Party, and
Other Stories

Match: 96.3%



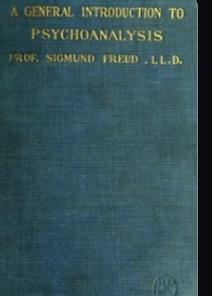
PHILOSOPHY OF AUGUSTE COMTE

L. LEVY-ROHL

This cover was created by adding text to a plain background
and is placed in the public domain.

The Philosophy of
Auguste Comte

Match: 96.0%



A GENERAL INTRODUCTION TO
PSYCHOANALYSIS

PROF. SIGMUND FREUD, LL.D.

A General Introduction to
Psychoanalysis

Match: 95.9%

3. Pencarian Berdasarkan Sampul (Test Case Sendiri)

- Hasil pencarian sampul langsung dengan *threshold* 70%:

Similarity Threshold: 70%

Search

The main image at the top is a blue book cover titled "THE ADVENTURES OF SHERLOCK HOLMES" by A. CONAN DOYLE. Below it, the heading "Hasil Pencarian" is displayed. Five book covers are shown in cards:

- The Adventures of Sherlock Holmes** (Match: 100.0%)
- Stories from Tagore** (Match: 96.5%)
- The Garden Party, and Other Stories** (Match: 96.4%)
- A General Introduction to Psychoanalysis** (Match: 96.1%)
- The Philosophy of Auguste Comte** (Match: 95.7%)

This cover was created for adding cover on a photo background and is placed in the public domain.

- Hasil pencarian sampul langsung dengan *threshold* 96%:

Similarity Threshold: 96%

Search

THE ADVENTURES OF SHERLOCK HOLMES
by A. CONAN DOYLE

Hasil Pencarian

<p>The Adventures of Sherlock Holmes Match: 100.0%</p>	<p>STORIES FROM TAGORE Match: 96.5%</p>	<p>THE GARDEN PARTY AND OTHER STORIES KATHERINE MANSFIELD AUTHOR OF BLISS Match: 96.4%</p>	<p>A GENERAL INTRODUCTION TO PSYCHOANALYSIS PROF. SIGMUND FREUD, LL.D. Match: 96.1%</p>
--	---	--	---

- Hasil pencarian sampul buku yang dicoret-coret:

Similarity Threshold: 50%

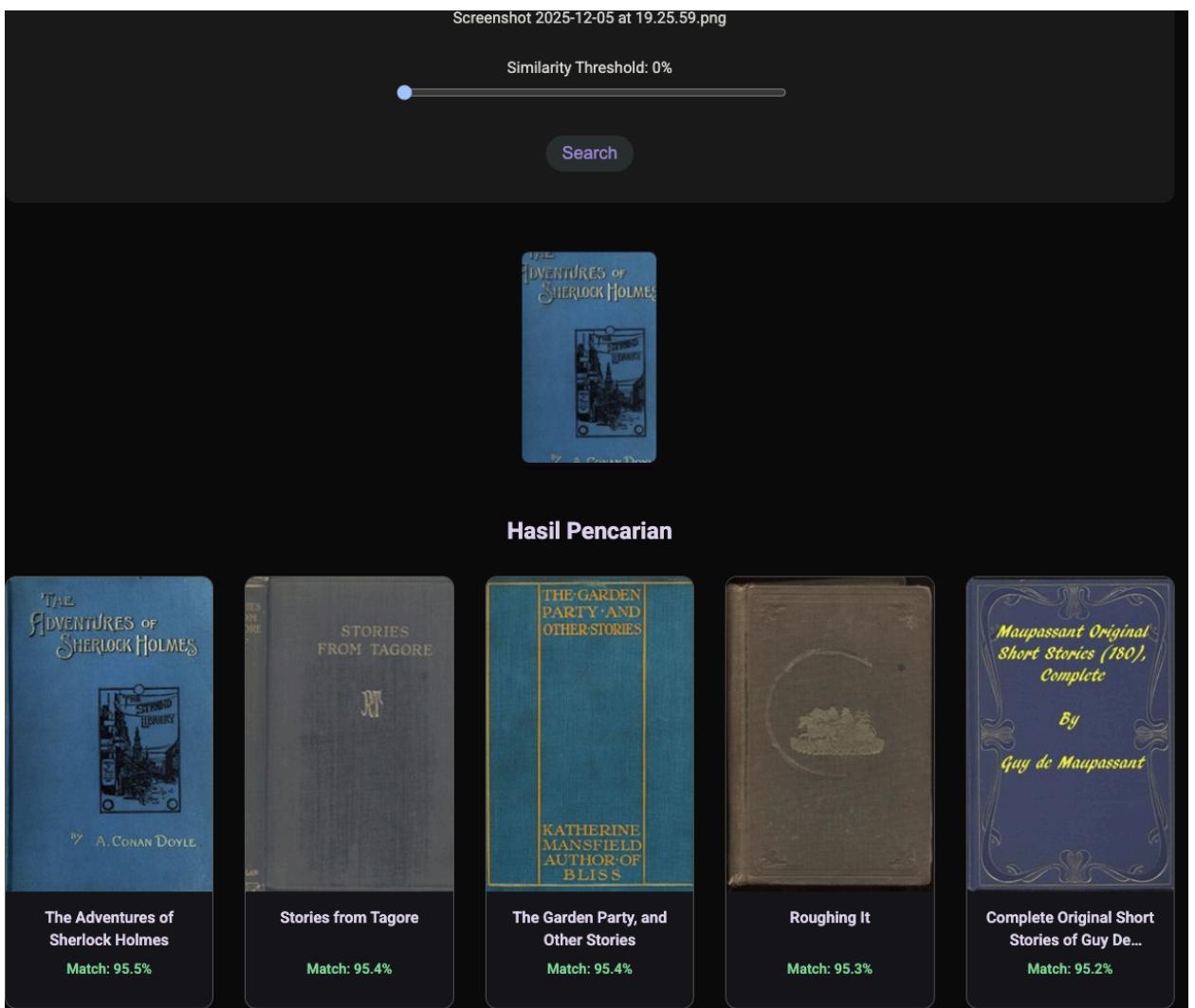
Search

The screenshot shows a search interface with a similarity threshold set at 50%. A search bar contains the text "Tak parani". Below the search bar, a book cover for "The Adventures of Sherlock Holmes" by A. Conan Doyle is displayed. The title "Tak parani" is overlaid on the book cover in red, and several red arrows point to the author's name "A. CONAN DOYLE" at the bottom of the cover. The background is dark.

Hasil Pencarian

Book Cover	Title	Author	Match (%)
	The Adventures of Sherlock Holmes	A. CONAN DOYLE	Match: 98.3%
	Stories from Tagore	R. K. NARAYAN	Match: 96.8%
	The Garden Party, and Other Stories	KATHERINE MANSFIELD	Match: 96.5%
	A General Introduction to Psychoanalysis	SIGMUND FREUD	Match: 96.2%
	The Jungle Book	RUDYARD KIPLING	Match: 95.8%

- Sampul buku setelah di-*crop*:



4. Hasil Rekomendasi Buku

The World as Will and Idea (Vol. 1 of 3)

Arthur Schopenhauer



Project Gutenberg

The World as Will and Idea (Vol. 1 of 3)

Content Preview

THE WORLD AS WILL AND IDEA (VOL. 1 OF 3) ***

The World As Will And Idea
By
Arthur Schopenhauer
Translated From The German By
R. B. Haldane, M.A.
And
J. Kemp, M.A.
Vol. I.
Containing Four Books.
"Ob nicht Natur zuletzt sich doch ergünde?"—GOETHE
Seventh Edition
London
Kegan Paul, Trench, Trübner & Co.
1909

Related Books



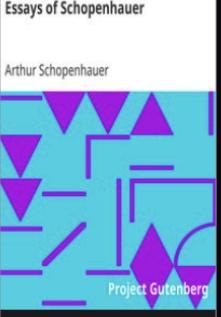
An Enquiry Concerning Human Understanding
David Hume and Sir L. A. Selby-Bigge
Project Gutenberg

An Enquiry Concerning Human Understanding



Fundamental Principles of the Metaphysic of Morals
Immanuel Kant
Project Gutenberg

Fundamental Principles of the Metaphysic of...



Essays of Schopenhauer
Arthur Schopenhauer
Project Gutenberg

Essays of Schopenhauer



A Treatise of Human Nature
David Hume
Project Gutenberg

A Treatise of Human Nature



The Ethics of Aristotle
Aristotle
Project Gutenberg

The Ethics of Aristotle

Meditations

Emperor of Rome Marcus Aurelius



Project Gutenberg

Meditations

Content Preview

MEDITATIONS ***

MEDITATIONS
By Marcus Aurelius

CONTENTS

NOTES

INTRODUCTION

FIRST BOOK

SECOND BOOK

THIRD BOOK

FOURTH BOOK

FIFTH BOOK

SIXTH BOOK

SEVENTH BOOK

EIGHTH BOOK

Related Books

<p>The Confessions of St. Augustine</p> <p>Bishop of Hippo Saint Augustine</p>  <p>Project Gutenberg</p> <p>The Confessions of St. Augustine</p>	<p>Divine Comedy, Longfellow's Translation, Hell</p> <p>Dante Alighieri</p>  <p>Project Gutenberg</p> <p>Divine Comedy, Longfellow's Translatio...</p>	<p>Devotions Upon Emergent Occasions; Together with Death's Duel</p> <p>John Donne</p>  <p>Project Gutenberg</p> <p>Devotions Upon Emergent Occasions;...</p>	<p>The Pilgrim's Progress from this world to that which is to come</p> <p>John Bunyan</p>  <p>Project Gutenberg</p> <p>The Pilgrim's Progress from this world to that...</p>	<p>Paradise Lost</p> <p>John Milton</p>  <p>Project Gutenberg</p> <p>Paradise Lost</p>
---	---	---	---	---

5. Pencarian Berdasarkan Dokumen

doc_1.txt
Top K Results: 5
Search

File Content Preview:
THE DECLARATION OF INDEPENDENCE OF THE UNITED STATES OF AMERICA ***

The United States Declaration of Independence was the first E-text released by Project Gutenberg, early in 1971. The title was stored in an emailed instruction set which required a tape or diskpack be hand mounted for retrieval. The disk pack was the size of a large cake in a cake carrier, cost \$1500, and contained 5 megabytes, of

Search Results

Result	Title	Author	Match (%)
1	The Declaration of Independence of the United States of America	In CONGRESS, July 4, 1776.	97.6%
2	A Modest Proposal For preventing the children of poor people from being a Burden to their Parents or the Country, &c.	(Price Threepence.)	77.0%
3	The Federalist Papers	Alexander Hamilton et al.	75.7%
4	The Jewish State	Theodor Herzl	75.3%
5	An Inquiry into the Nature and Causes of the Wealth of Nations	by Adam Smith	73.8%

doc_23.txt

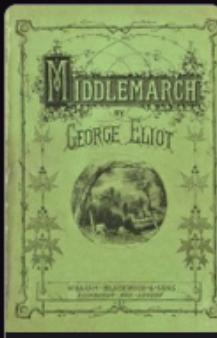
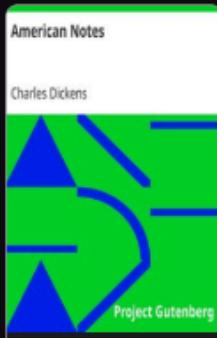
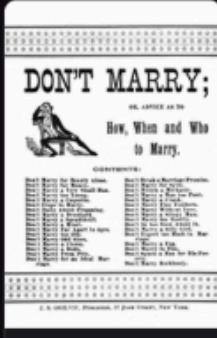
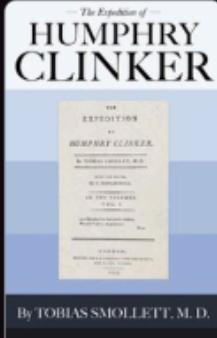
Top K Results: 9

Search

Nantucket, at which it was my happiness to become acquainted with _Frederick Douglass_, the writer of the following Narrative. He was a stranger to nearly every member of that body; but, having recently made his escape from the southern prison-house of bondage, and feeling his curiosity excited to ascertain the principles and measures of the abolitionists,—of whom he had heard a somewhat vague description while he was a slave,—he was induced to give his attendance, on the occasion alluded to, though at that time a resident in New Bedford.

Fortunate, most fortunate occurrence!—fortunate for the millions of his manacled brethren, yet panting for deliverance from their awful

Search Results

				
Narrative of the Life of Frederick Douglass, an American Slave Frederick Douglass Project Gutenberg	UP FROM SLAVERY AN AUTOBIOGRAPHY BOOKER T. WASHINGTON Up from Slavery: An Autobiography Match: 88.8%	MIDDLEMARCH BY GEORGE ELIOT Middlemarch Match: 86.9%	American Notes Charles Dickens Project Gutenberg	Bartleby, the Scrivener: A Story of Wall-Street Herman Melville Project Gutenberg
Narrative of the Life of Frederick Douglass, a... Match: 96.9%			American Notes Match: 86.9%	Bartleby, the Scrivener: A Story of Wall-Street Match: 86.5%
				
Ruggles of Red Gap Harry Leon Wilson Project Gutenberg	DON'T MARRY; OR, ADVICE ON HOW, WHEN AND WHO TO MARRY. Don't Marry; or, Advice on How, When and Wh... Match: 84.3%	—The Expedition of— HUMPHRY CLINKER The Expedition of Humphry Clinker By TOBIAS SMOLLETT, M. D. The Strange Case of Dr. Jekyll and Mr. Hyde Match: 84.2%		
Ruggles of Red Gap Match: 86.0%				

Penutup

Kesimpulan

Dalam Tugas Besar 2 ini, kami berhasil mengimplementasikan konsep Principal Component Analysis (PCA) untuk pencarian gambar. Program mampu mereduksi dimensi gambar sampul buku (teknik eigenfaces) dan menghitung tingkat kemiripan dengan gambar yang *di-upload* pengguna. Kami berhasil mengimplementasikan fitur rekomendasi otomatis. Ketika pengguna membuka *detail* sebuah buku, program akan menghitung dan menampilkan buku-buku lain yang memiliki konten (semantik) yang paling mirip menggunakan model LSA yang sama. Ini sangat mempermudah pengguna dalam menemukan buku baru. Kami juga berhasil mengimplementasikan Latent Semantic Analysis untuk pencarian dokumen teks.

Saran

Saat ini, pencarian dilakukan dengan membandingkan query terhadap seluruh dataset secara linear ($O(N)$). Jika jumlah buku mencapai jutaan, ini akan lambat. Seharusnya hal ini dipertimbangkan oleh asisten agar aplikasi tetap dapat digunakan dengan lancar di masa depan dengan asumsi *database* akan *di-update*.

Saat ini metadata buku disimpan dalam file JSON dan dimuat ke memori. Sebaiknya, disimpan ke database relasional seperti PostgreSQL agar manajemen data lebih terstruktur dan aman.

Refleksi

Melalui penggerjaan tugas besar ini, kami menjadi paham dengan lebih mendalam tentang konsep aljabar linier seperti nilai eigen dan vektor eigen, dan bagaimana cara menerapkannya pada permasalahan nyata. Kami juga menyadari pentingnya kolaborasi tim dan manajemen waktu dalam menyelesaikan tugas besar dengan kompleksitas ini.

Daftar Pustaka

- Munir, R. (2025). Vektor di ruang Euclidean (Bagian 1). Homepage Rinaldi Munir. Retrieved November November 30, 2025, from
<https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2025-2026/Algeo-11-Vektor-di-Ruang-Euclidean-Bag1-2025.pdf>
- Munir, R. (2025). Nilai Eigen dan Vektor Eigen (Bagian 1). Homepage Rinaldi Munir. Retrieved November November 30, 2025, from
<https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2025-2026/Algeo-19-Nilai-Eigen-dan-Vektor-Eigen-Bagian1-2025.pdf>
- Munir, R. (2025). Singular Value Decomposition (Bagian 1). Homepage Rinaldi Munir. Retrieved November November 30, 2025, from
<https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2025-2026/Algeo-21-Singular-value-decomposition-Bagian1-2025.pdf>
- Google. (2025, May). *Material Design 3*. Material Design. Retrieved November 30, 2025, from
<https://m3.material.io/>
- Nielsen, J. (2024, February 20). *10 Usability heuristics for user interface design*. Nielsen Norman Group. Retrieved December 5, 2025, from
<https://www.nngroup.com/articles/ten-usability-heuristics/>

Lampiran

Repository GitHub: <https://github.com/IRK-23/algeo2-spinosaurus>