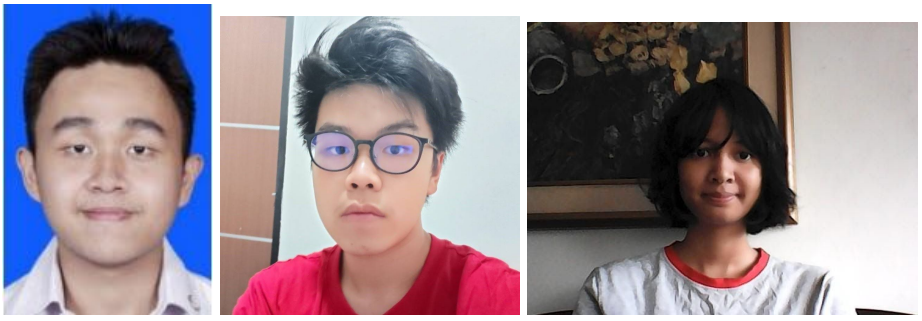
# LAPORAN TUGAS BESAR 2

Mata Kuliah Aljabar Linear dan Geometri IF 2123

Dosen Pengampu : Rinaldi Munir, Nugraha Priya Utama, Judhi Santoso, Rila Mandala




Disusun Oleh :

| | |
|---|---|
| Jesson Gossal Yo | 13519079 |
| Marcello Faria | 13519086 |
| Hera Shafira | 13519131 |

**PROGRAM STUDI TEKNIK INFORMATIKA**

**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA**

**INSTITUT TEKNOLOGI BANDUNG**


**2020**

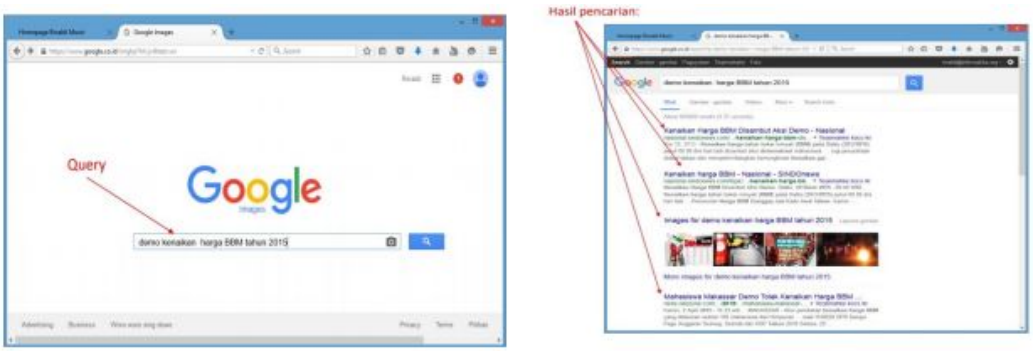
# DAFTAR ISI

# BAB I
# DESKRIPSI MASALAH

## 1.1 Abstraksi

Hampir semua dari kita pernah menggunakan search engine, seperti google, bing dan yahoo! search. Setiap hari, bahkan untuk sesuatu yang sederhana kita menggunakan mesin pencarian Tapi, pernahkah kalian membayangkan bagaimana cara search engine tersebut mendapatkan semua dokumen kita berdasarkan apa yang ingin kita cari?

Sebagaimana yang telah diajarkan di dalam kuliah pada materi vector di ruang Euclidean, temu-balik informasi (information retrieval) merupakan proses menemukan kembali (retrieval) informasi yang relevan terhadap kebutuhan pengguna dari suatu kumpulan informasi secara otomatis. Biasanya, sistem temu balik informasi ini digunakan untuk mencari informasi pada informasi yang tidak terstruktur, seperti laman web atau dokumen.




*Gambar 1. Contoh penerapan Sistem Temu-Balik pada mesin pencarian*
sumber: Aplikasi Dot Product pada Sistem Temu-balik Informasi by Rinaldi Munir

Ide utama dari sistem temu balik informasi adalah mengubah search query menjadi ruang vektor Setiap dokumen maupun query dinyatakan sebagai vektor w = (w1, w2,..., wn) di dalam Rn , dimana nilai wi dapat menyatakan jumlah kemunculan kata tersebut dalam dokumen (term frequency). Penentuan dokumen mana yang relevan dengan search query dipandang sebagai pengukuran kesamaan (similarity measure) antara query dengan dokumen. Semakin sama suatu vektor dokumen dengan vektor query, semakin relevan dokumen tersebut dengan query. Kesamaan tersebut dapat diukur dengan cosine similarity dengan rumus:

$$sim(\mathbf{Q}, \mathbf{D}) = \cos\theta = \frac{\mathbf{Q} \cdot \mathbf{D}}{\|\mathbf{Q}\|\|\mathbf{D}\|}$$

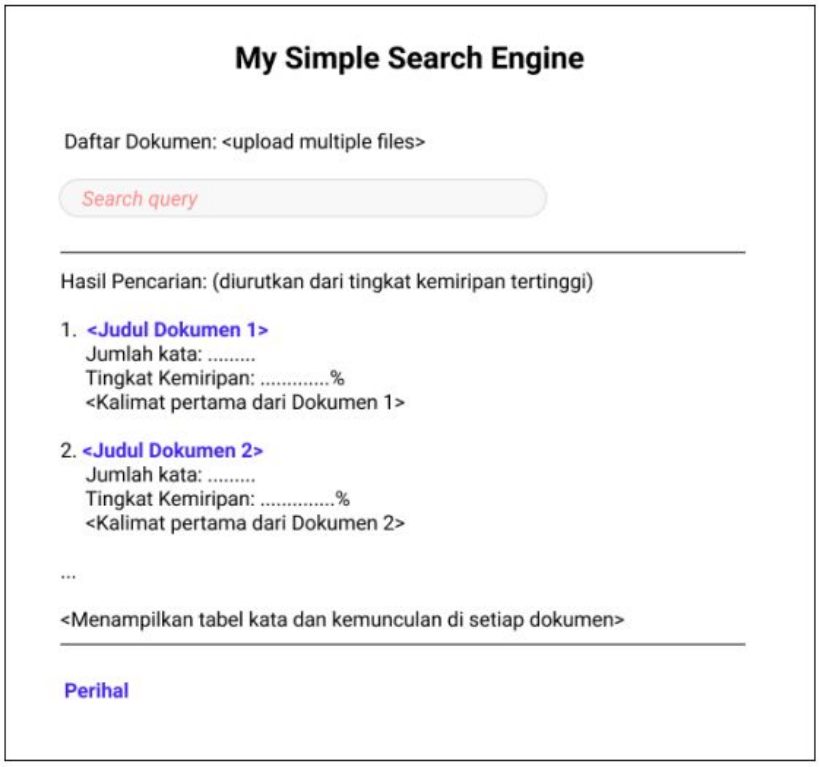
Pada kesempatan ini, kami ditantang untuk membuat sebuah search engine sederhana dengan model ruang vector dan memanfaatkan cosine similarity.

### 1.2 Penggunaan Program

Berikut ini adalah input yang akan dimasukkan pengguna untuk eksekusi program.

1. Search query, berisi kumpulan kata yang akan digunakan untuk melakukan pencarian
2. Kumpulan dokumen, dilakukan dengan cara mengunggah multiple file ke dalam web browser.

Tampilan layout dari aplikasi web yang akan dibangun adalah sebagai berikut.




*Gambar 2. Contoh Layout Tampilan Web*

**Perihal**: link ke halaman tentang program dan pembuatnya (Konsep singkat search engine yang dibuat, How to Use, About Us). 3 Catatan: Teks yang diberikan warna biru merupakan hyperlink yang akan mengalihkan halaman ke halaman yang ingin dilihat. Apabila menekan hyperlink , maka akan diarahkan pada sebuah halaman yang berisi full-text terkait dokumen 1

tersebut (seperti Search Engine). Anda dapat menambahkan menu lainnya, gambar, logo, dan sebagainya. Tampilan Front End dari website dibuat semenarik mungkin selama mencakup seluruh informasi pada layout yang diberikan di atas. Data uji berupa dokumen-dokumen yang akan diunggah ke dalam web browser. Format dan extension dokumen dibebaskan selama bisa dibaca oleh web browser (misalnya adalah dokumen dalam bentuk file txt atau file html). Minimal terdapat 15 dokumen berbeda. Tabel term dan banyak kemunculan term dalam setiap dokumen akan ditampilkan pada web browser dengan layout sebagai berikut.

| Term | Query | D1 | D2 | ... | D3 |
|---|---|---|---|---|---|
| Term1 | | | | | |
| Term2 | | | | | |
| ... | | | | | |
| TermN | | | | | |

*Gambar 3. Contoh Tabel Terms*

Untuk menyederhanakan pembuatan search engine, terdapat hal-hal yang perlu diperhatikan dalam eksekusi program ini.

1. Silahkan lakukan stemming dan penghapusan stopwords pada setiap dokumen
2. Tidak perlu dibedakan antara huruf-huruf besar dan huruf-huruf kecil.
3. Stemming dan penghapusan stopword dilakukan saat penyusunan vektor, sehingga halaman yang berisi full-text terkait dokumen tetap seperti semula.
4. Penghapusan karakter-karakter yang tidak perlu untuk ditampilkan (jika menggunakan web scraping atau format dokumen berupa html)
5. Bahasa yang digunakan dalam dokumen adalah bahasa Inggris atau bahasa Indonesia (pilih salah satu) Petunjuk: silahkan gunakan library

# BAB II
# TEORI SINGKAT

## 2.1 Temu balik informasi

Temu balik informasi adalah proses menemukan kembali (retrieval) informasi yang relevan terhadap kebutuhan pengguna dari suatu kumpulan informasi secara otomatis. Salah satu aplikasi umum dari sistem temu kembali informasi adalah search-engine atau mesin pencarian yang terdapat pada jaringan internet. Pengguna dapat mencari halaman-halaman Web yang dibutuhkannya melalui mesin tersebut. Dalam Information Retrieval, mendapatkan dokumen yang relevan tidaklah cukup. Tujuan yang harus dipenuhi adalah bagaimana mendapatkan dokumen relevan dan tidak mendapatkan dokumen yang tidak relevan. Tujuan lainnya adalah bagaimana menyusun dokumen yang telah didapatkan tersebut ditampilkan terurut dari dokumen yang memiliki tingkat relevansi lebih tinggi ke tingkat relevansi rendah. Penyusunan dokumen terurut tersebut disebut sebagai perangkingan dokumen. Model Ruang Vektor dan Model Probabilistik adalah 2 model pendekatan untuk melakukan hal tersebut.

## 2.2 Vektor di ruang Euclidean

Vektor merupakan kuantitas fisik yang memiliki besar dan arah. Vektor dapat dilambangkan dengan huruf kecil cetak tebal **u** atau dengan menggunakan huruf kecil dengan panah di atasnya $\overline{u}$. Tempat dimana vektor didefinisikan disebut dengan ruang vektor atau disebut juga dengan ruang vektor Euclidean. Ruang vektor dinotasikan dengan $R^2$, $R^3$,..., $R^n$ dengan superscript 2,3,...,n menyatakan dimensi ruang vektor tersebut sehingga sebuah vektor di ruang $R^n$ dinotasikan dengan $\mathbf{v} = (v_1, v_2, ..., v_n)$. Operasi-operasi yang dapat dilakukan pada vektor adalah

**penjumlahan** : $\mathbf{v} + \mathbf{w} = (v_1, v_2, ..., v_n) + (w_1, w_2, ..., w_n) = ((v_1 + w_1), ..., (v_n + w_n))$
**pengurangan** : $\mathbf{v} - \mathbf{w} = (v_1, v_2, ..., v_n) - (w_1, w_2, ..., w_n) = ((v_1 - w_1), ..., (v_n - w_n))$
**perkalian vektor dengan skalar** : $k.\mathbf{v} = (k v_1, k v_2, ..., k v_n)$

Selain itu, sebuah vektor juga memiliki panjang atau disebut juga dengan norma Euclidean. Norma sebuah vektor **x** dinotasikan sebagai ||**x**|| dengan rumus :

$$|\mathbf{x}|_2 = |\mathbf{x}| = \sqrt{x_1^2 + x_2^2 + \ldots + x_n^2}\,.$$

**perkalian titik** (*dot product*) :

misal ada 2 buah vektor **u** = ( $u_1$ , $u_2$ ,..., $u_n$ ) dan **v** = ( $v_1$ , $v_2$ ,..., $v_n$ ) dengan **u** dan **v** bukan merupakan vektor nol, maka dot product 2 vektor tersebut dinotasikan sebagai **u**.**v** dengan rumus :

1. **u**.**v** = ||**u**|| . ||**v**||. *cos* θ , dengan θ adalah sudut yang dibentuk oleh vektor **u** dan **v**
2. **u**.**v** = $u_1 . v_1 + u_2 . v_2 + \ldots + u_n . v_n$

hasil perkalian titik adalah skalar.

**perkalian silang** (*cross product*) :

misal ada 2 buah vektor **u** = ( $u_1$ , $u_2$ , $u_3$ ) dan **v** = ( $v_1$ , $v_2$ , $v_3$ ), maka cross product 2 vektor tersebut dinotasikan sebagai **u** x **v** , dengan rumus :

$$\mathbf{u} \times \mathbf{v} = \left(\begin{vmatrix} u_2 & u_3 \\ v_2 & v_3 \end{vmatrix}, -\begin{vmatrix} u_1 & u_3 \\ v_1 & v_3 \end{vmatrix}, \begin{vmatrix} u_1 & u_2 \\ v_1 & v_2 \end{vmatrix}\right)$$

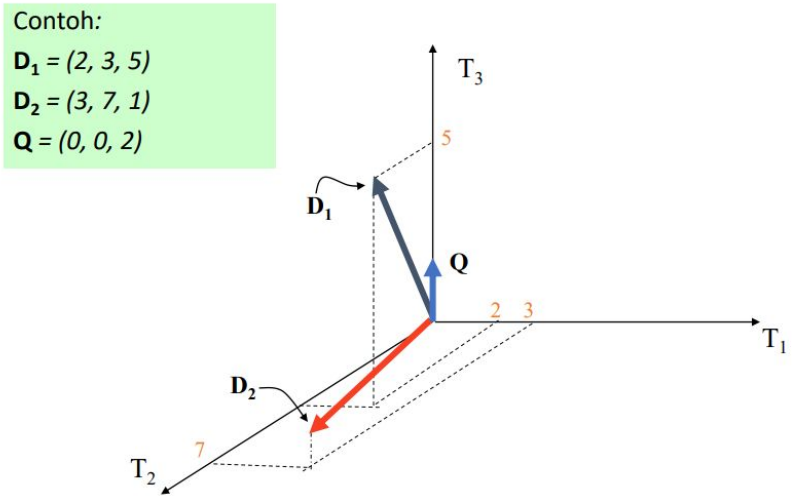
hasil perkalian silang adalah sebuah vektor yang tegak lurus dengan vektor **u** dan **v**. Panjang dari vektor **u** x **v** dinotasikan sebagai || **u** x **v** || dengan rumus :

**u**.**v** = ||**u**|| . ||**v**||. *sin* θ , dengan θ adalah sudut yang dibentuk oleh vektor **u** dan **v**

### 3. *Cosine similarity* dalam temu balik informasi

Salah satu metode yang digunakan dalam sistem temu balik informasi adalah dengan menggunakan model ruang vektor. Metode ini dilakukan dengan cara memodelkan dokumen dan query sebagai vektor **w** = ( $w_1$ , $w_2$ ,..., $w_n$ ) dalam $R^n$ , dengan $w_i$ adalah bobot/jumlah kemunculan tiap kata i dalam sebuah dokumen/query. Metode ini sering digunakan untuk menghitung kesamaan dokumen dalam analisis teks. Suatu dokumen dapat direpresentasikan dalam ribuan atau bahkan jutaan atribut, dan setiap dari atribut tersebut mencatat frekuensi dari kata tertentu dan setiap dokumen memiliki objek yang dinamakan dengan *term-frequency vector*.

Misal ada 2 buah dokumen $D_1$ dan $D_2$ dan sebuah query Q yang akan digunakan untuk mencari informasi di kedua dokumen tersebut dengan $D_1$ = (2,3,5) , $D_2$ = (3,7,1), Q = (0,0,2). Misalkan dari gabungan $D_1$ , $D_2$ , Q ada 3 buah kata yaitu $T_1$ , $T_2$ , dan $T_3$ . Maka makna dari vektor $D_1$ adalah $D_1$ mengandung 2 buah $T_1$ , 3 buah $T_2$ , dan 5 buah $T_3$ . $D_2$ mengandung 3 buah kata $T_1$ , 7 buah kata $T_2$ , dan 1 buah kata $T_3$ dan Q. Terakhir, Q hanya mengandung 2 buah kata $T_3$ . Ketiga vektor ini dapat digambarkan sebagai berikut :




*Gambar 4. Contoh Visualisasi Vektor Terms*

Semakin relevan sebuah dokumen dengan query maka vektor dari dokumen tersebut akan semakin mirip dengan vektor query. Untuk mengukur kemiripan antara 2 vektor maka digunakan *cosine similarity* dengan rumus sebagai berikut.

$$sim(\mathbf{Q}, \mathbf{D}) = \cos\theta = \frac{\mathbf{Q} \cdot \mathbf{D}}{\|\mathbf{Q}\|\|\mathbf{D}\|}$$

rumus tersebut diturunkan dari rumus perkalian titik 2 buah vektor yaitu **u**.**v** = ||**u**|| . ||**v**||. *cos* θ . Dari rumus *cosine similarity* tersebut diketahui apabila nilai *cos* θ (similarity) sebuah vektor dokumen sama dengan 1 maka sudut yang terbentuk di antara vektor dokumen dengan vektor query adalah 0° alias kedua vektor tersebut berimpit sehingga dapat dinyatakan kedua vektor tersebut adalah vektor yang sama. Dari pernyataan tersebut, maka dapat disimpulkan bahwa semakin besar atau semakin nilai *cos* θ (similarity) mendekati 1 maka dokumen tersebut semakin relevan dengan query yang digunakan.

Maka dalam proses temu balik informasi, langkah yang harus dilakukan adalah :

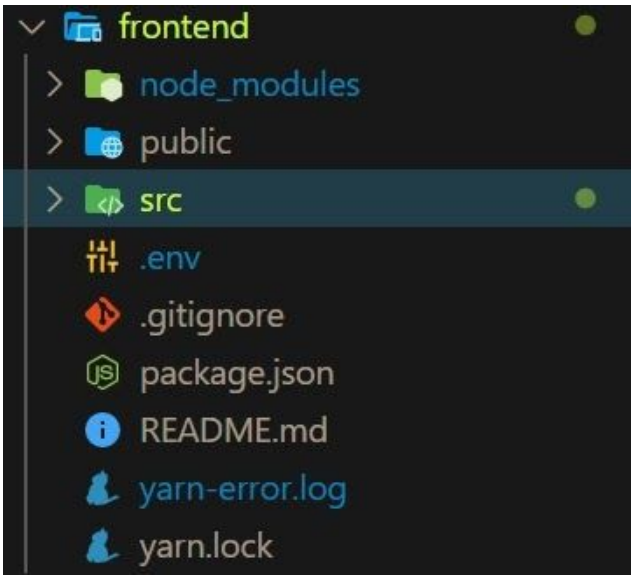
1. Ubah semua dokumen dan query ke dalam bentuk vektor
2. Hitung nilai *cosine similarity* antara vektor query dengan vektor dokumen untuk semua dokumen yang ada.
3. Urutkan hasil pencarian berdasarkan nilai *cosine similarity*, semakin besar (semakin mendekati 1) nilai *cosine similarity* dari suatu dokumen, maka urutan dokumen tersebut semakin atas.

# BAB III
# IMPLEMENTASI PROGRAM

Program ini adalah implementasi sistem temu informasi dalam bentuk *website*. Secara garis besar, implementasi web ini terbagi menjadi 2 bagian, yaitu *frontend* dan *backend*. Dalam pengembangan *frontend* dari program ini, *framework* yang digunakan adalah `React.js`. Dan yang terakhir, untuk pengembangan *backend* digunakan bahasa pemrograman `Python` dengan *framework* `Flask`. Ada dua cara dalam menggunakan web ini, cara pertama adalah pengguna akan diminta untuk mengupload beberapa file dokumen dalam format `.html` kemudian setelah file diupload, pengguna bisa memasukkan query untuk mencari dokumen yang relevan. Cara kedua adalah dengan menggunakan *web scraping*, pengguna tidak diminta untuk memberi input dokumen apapun karena dokumen sudah disediakan melalui metode *web scraping*, pengguna cukup memasukkan query saja.

## 3.1 Frontend

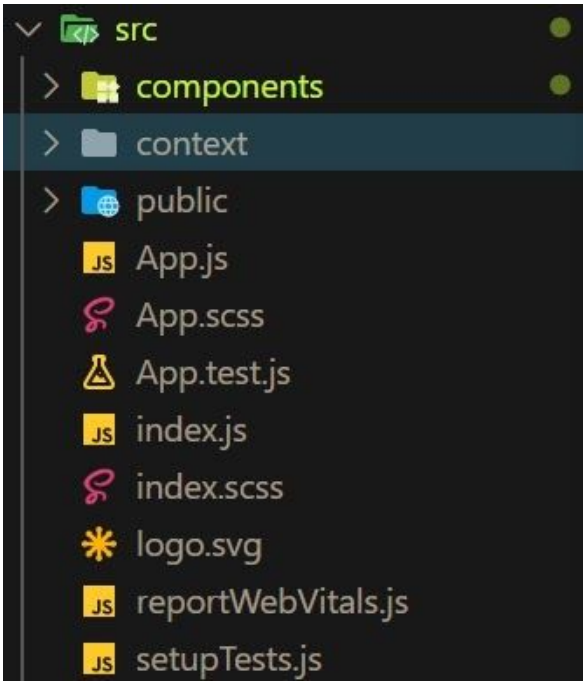
Pada folder frontend, struktur folder yang dibuat adalah sebagai berikut




*Gambar 5. Struktur Folder Frontend*

Folder frontend secara garis besar mengandung 3 folder utama. Node modules menyimpan semua modules dan dependencies yang diperlukan aplikasi public digunakan untuk menyimpan file-file statik seperti robots.txt dan index.html, serta src menyimpan semua komponen utama program. Selain 3 folder tersebut ada beberapa file lain seperti .env sebagai

penyimpan environment variables, serta gitignore dan package.json dan yarn.lock untuk menyimpan daftar dependencies.




*Gambar 6. File Pelengkap Frontend*

Di dalam src ada beberapa folder seperti components sebagai tempat untuk menyimpan komponen react seperti react functional components dan scss-nya. Context sebagai folder untuk menyimpan react context serta public untuk menyimpan gambar-gambar yang diperlukan. Aplikasi berpusat di dalam App.js dan index.js menangani DOM rendering keseluruhan aplikasi. Dibawah ini kami akan menjelaskan setiap komponen yang ada di web kami.

Berikut ini adalah components yang digunakan pada web kami :

1. **App.js** : Modul ini menangani routing url serta konsumsi context API.
2. **ThemeContext.js** : File ini menangani 2 mode pada website yaitu mode webscrap dan normal. Jika mode normal maka ketika user melakukan query, query akan dilakukan terhadap file di backend, sedangkan jika pengguna menggunakan webscrap mode maka ketika pengguna melakukan query, query dilakukan terhadap hasil webscrap dari beberapa website yang sudah ditentukan di backend.
3. **UploadPage.js** : modul ini berisi sebuah komponen fungsional react yang mengembalikan JSX untuk halaman upload. Page ini merupakan parent dari component Uploader.js yang berguna sebagai container wrapper dari komponen uploader yang telah

dibuat.

4. **Uploader.js :** File ini merupakan komponen children dari uploadPage yang berguna sebagai container untuk menampung file dalam bentuk single file maupun multiple file. Komponen ini juga dapat menampilkan file-file yang akan diupload dan me-remove file yang tidak jadi di upload. File-file tersebut akan diupload setelah menekan button Upload. Komponen ini juga mampu menghandle jika user ingin memilih untuk onclick maupun drag and drop file yang akan diupload.
5. **SearchPage.js :** Search page adalah komponen fungsional react yang sebatas mengembalikan JSX untuk halaman search,setelah pengguna memasukkan query pada search bar dan menekan submit maka pengguna akan dipindahkan ke halaman result. Saat dipindahkan, SearchPage akan mengoper props mode query dan string query.
6. **ResultPage.js :** File ini menampilkan hasil dari search query yang telah dilakukan pada SearchPage,yang berisi Hasil pencarian dari dokumen-dokumen yang telah disimpan secara statik jika props yang diberikan oleh SearchPage adalah "normal" dan dari webscraping jika props yang diberikan dari searchpage adalah "webscrap". Dengan urutan tingkat kemiripan tertinggi hingga terendah. Setelah itu, ditampilkan juga tabel term dengan kolom pertama dan kedua berisi sejumlah term dan query yang diketikkan oleh pengguna, dan sisa kolomnya berisi banyaknya kemunculan term pada dokumen yang disimpan.
7. **About.js :** modul ini berisi sebuah fungsi yang mengembalikan JSX untuk halaman About.
8. **Cerita.js :** modul ini berisi sebuah fungsi yang mengembalikan isi cerita
9. **TableData.js :** modul ini berisi sebuah fungsi yang mengembalikan JSX untuk tabel terms
10. **Header.js :** modul ini berisi sebuah fungsi yang mengembalikan JSX untuk navigation bar

### 3.2 Backend

Ada 5 file utama yaitu :

1. **App.py :** Pusat REST API backend, ada 4 route yang dipakai /api/search, /api/webscrap, /api/cerita, dan /api/upload. Search menangani GET method dan menghasilkan response json dengan beberapa properti seperti ranks,table,dan kolom. Dalam ranks ada properti-properti seperti title, similarity, header(kalimat pertama), dan words count. Table berisi representasi vektor setiap dokumen dan query. Webscrap melakukan hal yang sama dengan search tetapi sumber datanya adalah dari webscraping website yang sudah

ditentukan. Api/cerita berfungsi untuk mengembalikan document html yang disimpan dalam folder static. Api/upload berguna untuk menangani upload files dengan menerima upload file yang berasal dari front-end, lalu difilter dengan secure_filename menggunakan library werkzeug.utils, secure_filename ini bertujuan untuk menyaring kemungkinan-kemungkinan file yang di upload dari user yang mungkin merusak atau men-inject server. Setelah difilter dengan secure_filename, file yang masuk akan disimpan ke dalam folder src/static/.

2. **filtering.py** : file ini terdiri dari beberapa fungsi :
   a. **stemkata**
      fungsi ini berfungsi untuk menstemming kata menjadi word stem dengan menggunakan library *NLTK* yang merupakan library bawaan dari *Python*. Selain itu, didalam fungsi ini, dokumen-dokumen yang dimasukkan juga difilter dengan stopwords yang juga menggunakan library NLTK.
   **b. Cleaningkata**
      Fungsi ini digunakan untuk memfilter punctuation seperti titik, koma, dan tag-tag yang berasal dari html yang tidak diperlukan.
   c. **Cleaningquery**
      Fungsi ini digunakan untuk memfilter punctuation yang berasal dari Search query user.

3. **Matrixterm.py**: file ini terdiri dari beberapa fungsi :
   **a. generateTermsFromFiles**
      Berfungsi untuk menghasilkan vektor terms dari files
   **b. generateMatrixFromTerms**
      Berfungsi membangun matrix dari terms yang dihasilkan fungsi sebelumnya
   **c. generateTermsFromWebscrap**
      Menghasilkan vektor terms dari hasil webscrap
   **d. generateMatrixFromWebTerms**
      Berfungsi membangun matrix dari vektor terms
   **e. generateTermsFromWebscrap**
      Berfungsi membangun matrix dari vektor terms untuk sumber dokumen dari webscraping
   **f. generateQueryVector**
      Berfungsi membangun vektor terms dari query yang diberikan dari searching frontend
   **g. updateTerms**
      Berfungsi untuk memperbaharui terms yang ada di salah satu vektor terms

4. **vectorizer.py** : Dalam vectorizer ada 3 fungsi
   **a. sim**
      Fungsi ini memiliki 2 parameter yaitu dictionary Q dan D yang merepresentasikan vektor query dan vektor dokumen

b. **vectorLength**
   Mengembalikan panjang vektor
c. **dotProduct**
   Melakukan operasi dot product antara Q dan D dengan memperhatikan keys

5. **webscrape.py** : Dalam webscrape ada 3 fungsi yaitu
   a. **webscrape**
      fungsi ini menerima input url suatu web dan akan mengembalikan array of dictionary, dengan key dictionary berupa term dan valuenya adalah banyaknya term tersebut muncul pada konten teks web tersebut.
   b. **getFirstSentence**
      fungsi ini menerima input url suatu web dan akan mengembalikan kalimat pertama dari konten teks web tersebut.
   c. **wordsCount**
      fungsi ini menerima input url suatu web dan akan mengembalikan jumlah kata tanpa stemming pada konten teks web tersebut.

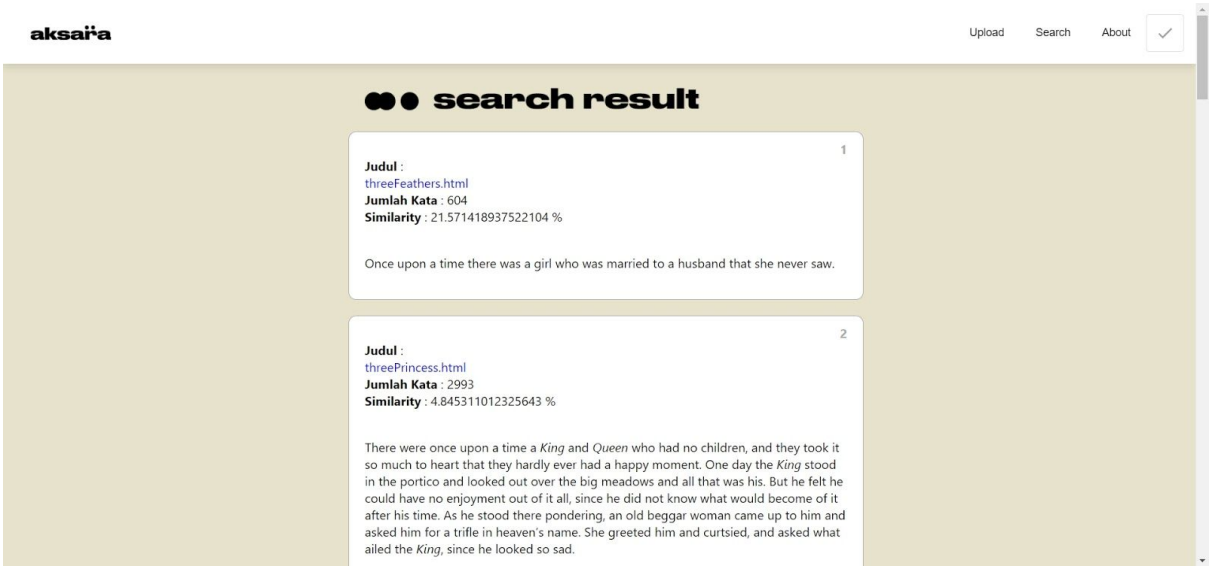
# BAB IV
# EKSPERIMEN

## 4.1. Hasil eksekusi program saat melakukan search query

Contoh saat user memasukkan query “three feather wonder”, lalu user dapat menekan tombol di sebelah search bar untuk melakukan search query dari dokumen-dokumen yang dimasukkan.

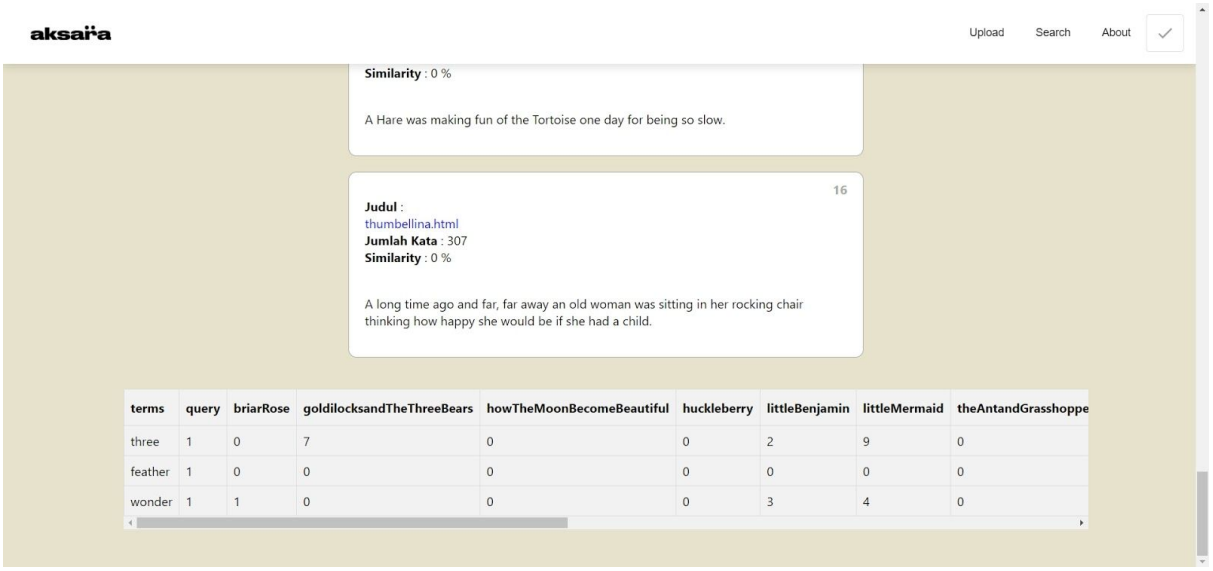

*Gambar 7. Hasil Eksekusi 1*

Berikut merupakan hasil search query yang telah dilakukan. Hasilnya berupa judul, jumlah kata, similarity serta beberapa kata dari dokumen yang dimasukkan.



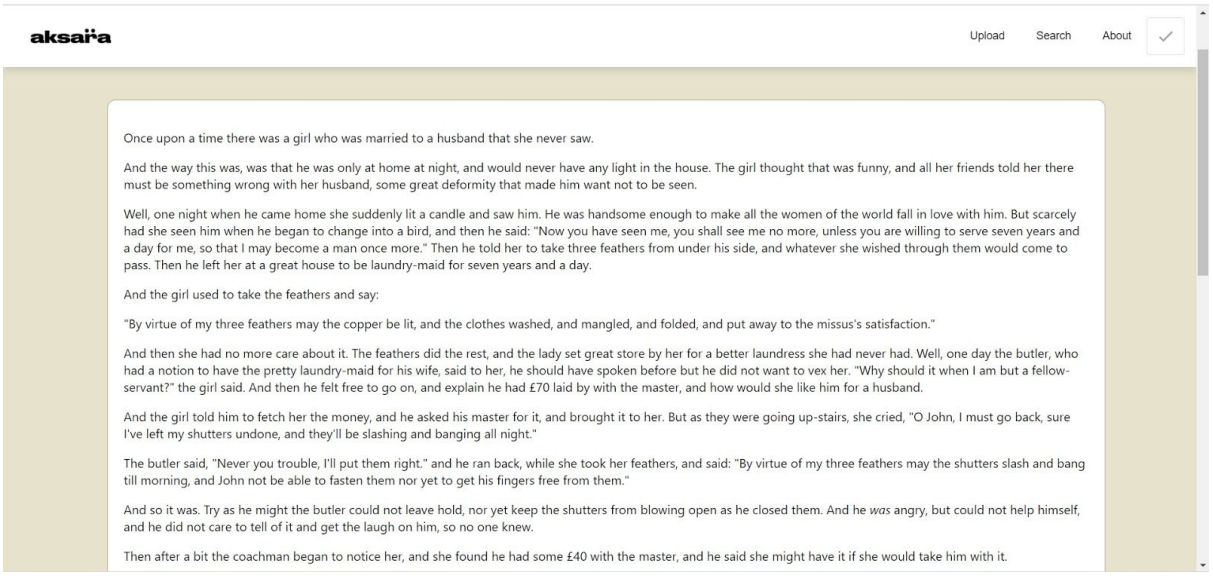

*Gambar 8. Hasil Eksekusi 2*

Setelah semua hasil query telah ditampilkan, dibawahnya terdapat tabel yang setiap kolom nya berisi judul dokumen dengan setiap baris merupakan banyak kemunculan terms dalam dokumen tersebut.




*Gambar 9. Hasil Eksekusi 3*

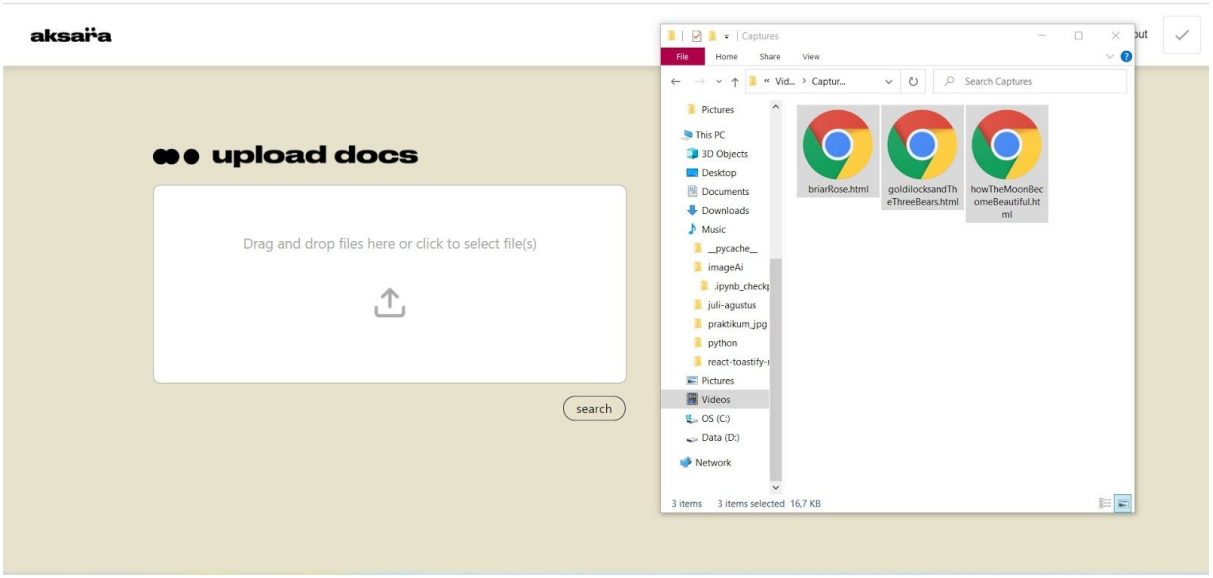
Jika user menekan judul yang ditampilkan sebelumnya pada hasil query dokumen, maka user akan di redirect menuju link page yang berisi isi dari dokumen tersebut.




*Gambar 10. Hasil Eksekusi 4*

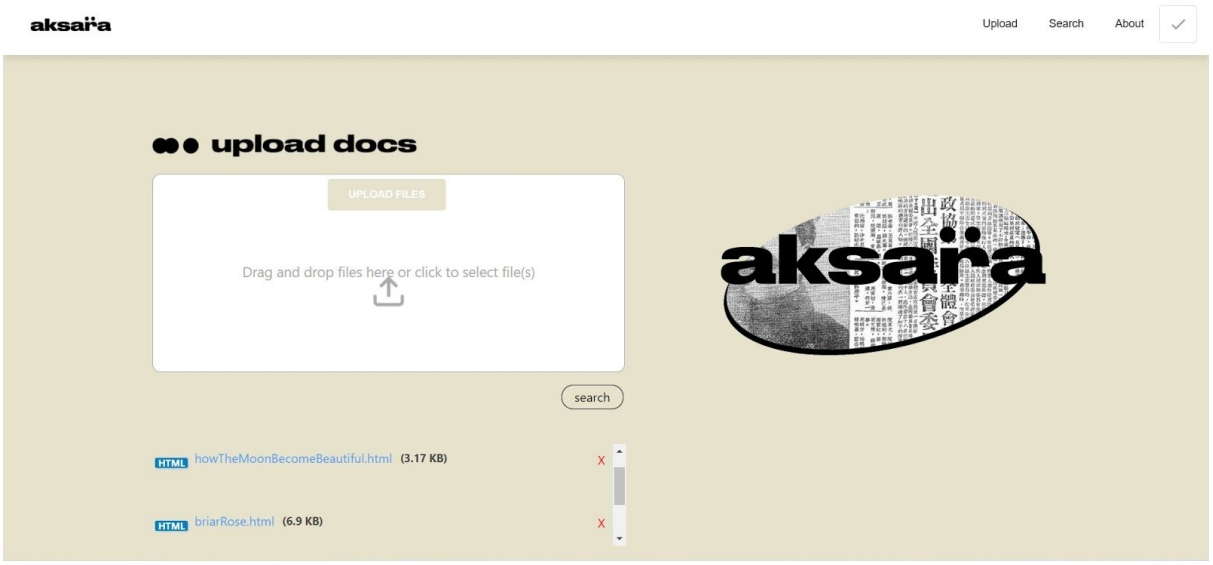
## 4.2 Hasil eksekusi program saat mengupload file

User dapat meng-klik pada container untuk memunculkan file seperti gambar dibawah ini :



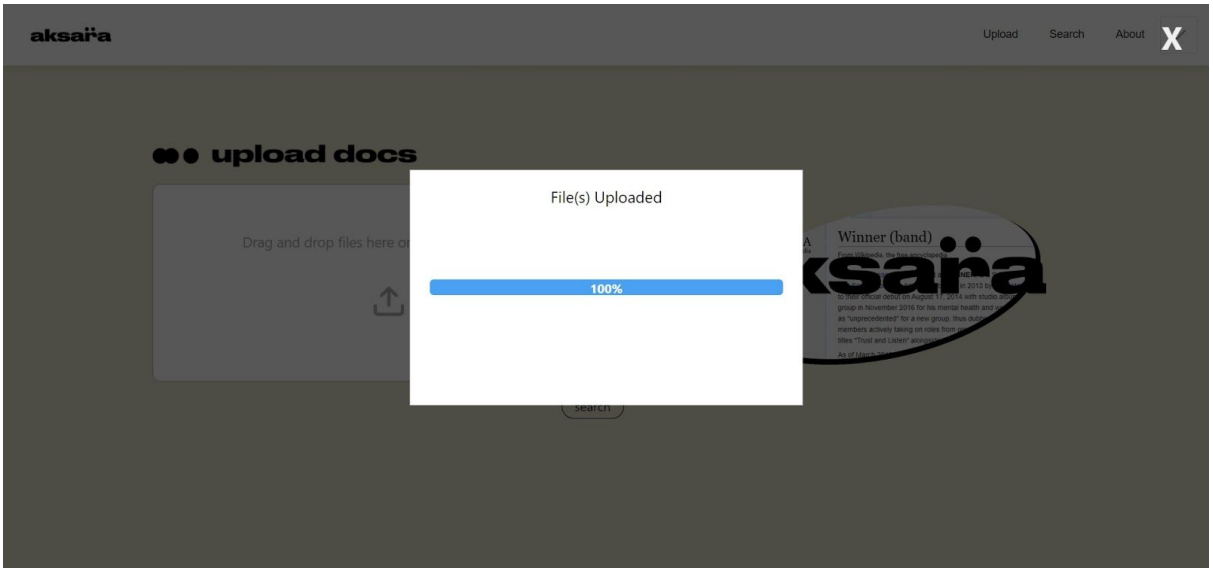

*Gambar 11. Hasil Eksekusi 5*

User dapat memilih untuk melakukan drag and drop maupun double click file yang akan dipilih, lalu file akan muncul pada web seperti dibawah ini. Jika user salah mengupload file atau ingin meremove file, maka user dapat menekan tombol silang disamping kiri scroll bar.




*Gambar 12. Hasil Eksekusi 6*

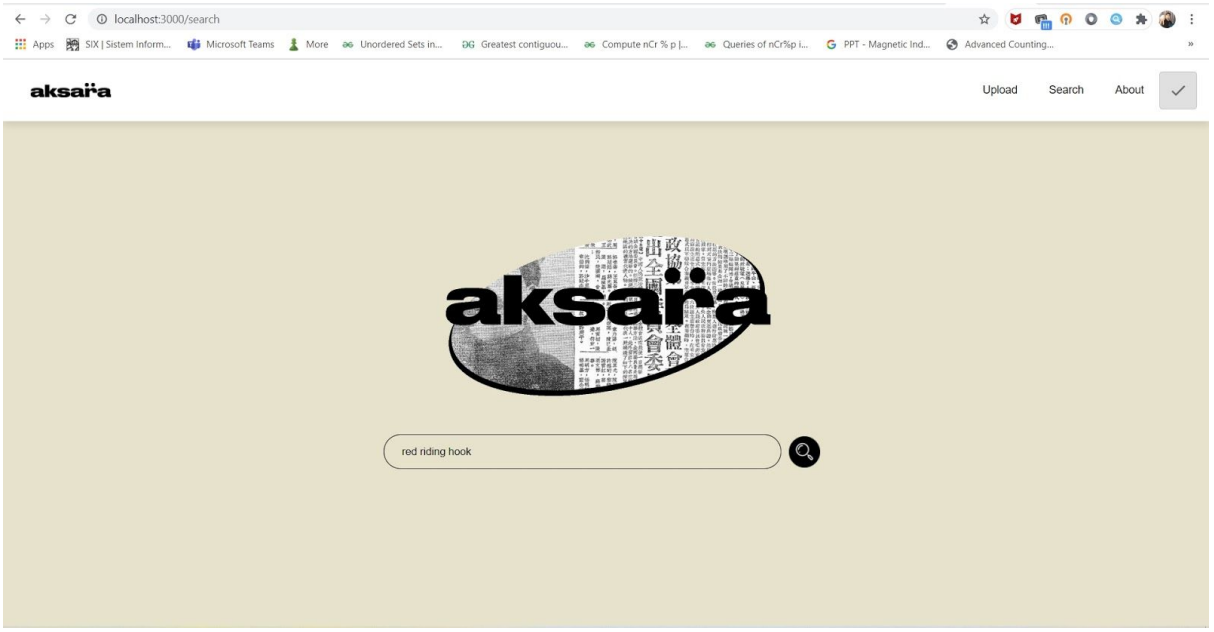
Jika file yang diupload berhasil sampai ke server tanpa kendala, maka akan muncul modal seperti berikut ini :




*Gambar 13. Hasil Eksekusi 7*

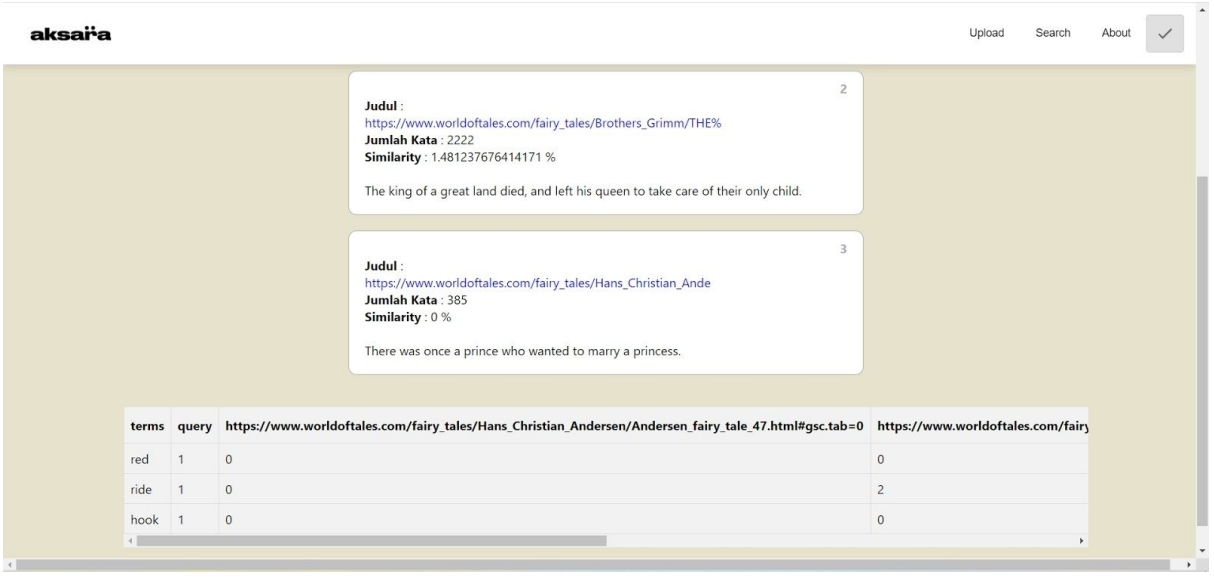
### 4.3 hasil eksekusi program saat melakukan search query dengan mode webscrap

User dapat melakukan web scraping dengan menekan tombol centang yang terdapat pada posisi kanan atas web.




*Gambar 14. Hasil Eksekusi 8*

Lalu akan menghasilkan hasil query dengan format yang sama dengan mode normal.



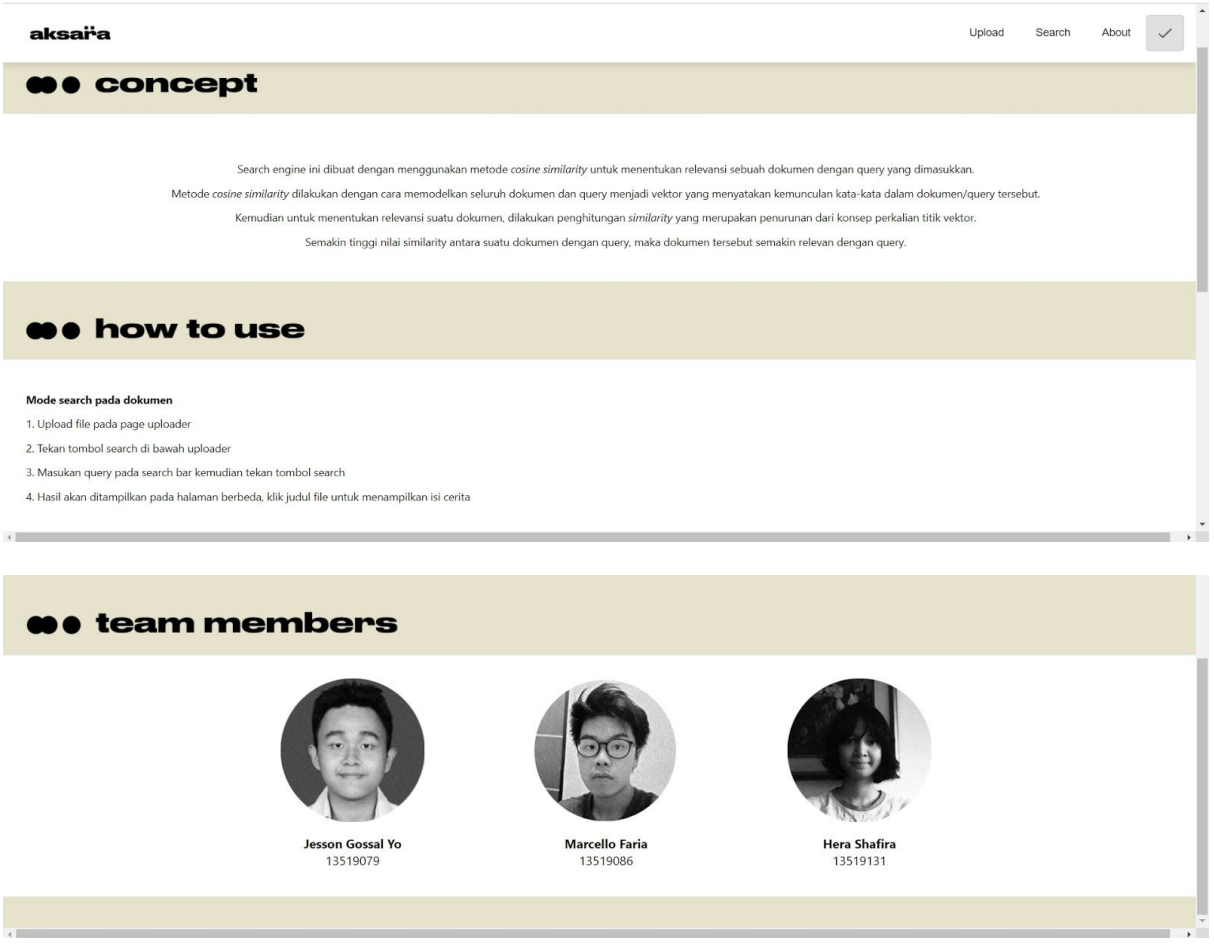

*Gambar 15. Hasil Eksekusi 9*

Jika user menekan link judul, maka user akan di-redirect menuju website yang di-webscrap di tab baru.


*Gambar 16. Hasil Eksekusi 10*

### 4.4 Hasil eksekusi halaman About

Jika user menekan tombol About pada navigasi, maka user dapat melihat konsep, cara penggunaan dan informasi mengenai developer yang membuat website ini.




*Gambar 17. Hasil Eksekusi 11*

# BAB V
# KESIMPULAN & SARAN

## 5.1. Kesimpulan

Temu balik informasi merupakan proses menemukan kembali (*retrieval*) informasi yang relevan terhadap kebutuhan pengguna dari suatu kumpulan informasi secara otomatis. Temu balik informasi dilakukan dengan cara seorang pengguna memasukkan query (hal yang ingin dicari pada dokumen-dokumen yang tersedia), kemudian sistem temu balik informasi akan menampilkan dokumen mana yang relevan dengan query tersebut. Salah satu metode yang dapat digunakan untuk melakukan temu balik informasi adalah dengan menggunakan *cosine similarity*. Metode *cosine similarity* dilakukan dengan cara mengubah seluruh dokumen dan query yang ada ke dalam bentuk vektor. Vektor ini merepresentasikan kemunculan kata-kata dalam dokumen/query tersebut. Kemudian untuk menentukan dokumen mana yang paling relevan dengan query, dilakukan proses penghitungan similarity. Penghitungan similarity dilakukan dengan menggunakan rumus

$$sim(\mathbf{Q}, \mathbf{D}) = \cos\theta = \frac{\mathbf{Q} \cdot \mathbf{D}}{\|\mathbf{Q}\|\|\mathbf{D}\|}$$

Semakin tinggi nilai similarity, maka dokumen tersebut semakin relevan dengan query yang dimasukkan.

Perhitungan similarity dapat dipengaruhi oleh floating point precision bahasa yang digunakan. Contohnya flask yang merupakan sebuah framework pada bahasa python memiliki floating point precision yang lebih baik dibanding dengan pemrosesan dengan menggunakan node js. Perhitungan similarity juga dipengaruhi oleh algoritma natural language processing, stemming, dan pembersihan stopwords yang digunakan. Kemudian, banyak kata pada sebuah dokumen jika perbedaannya terlalu jauh dapat menghasilkan bias karena normalisasi (pembagi similarity berupa panjang vektor) dapat membuat hasil similaritynya menjadi rendah walaupun *term frequencynya* relatif tinggi

## 5.2. Saran

1. Flask tidak sesuai untuk dijadikan server production karena scalabilitynya buruk
2. Selalu uji program dengan kasus ekstrim untuk menemukan bug

### 5.3 Refleksi

1. Seharusnya kami selalu memberikan komentar pada code kami agar code mudah dibaca saat proses debugging
2. Seharusnya kami melakukan riset sebelum menggunakan library-library yang belum pernah digunakan

# REFERENSI


1. Anton, H., Rorres, C. and Kaul, A., n.d. *Elementary Linear Algebra*.
2. Jansen, B. J. and Rieh, S. (2010) The Seventeen Theoretical Constructs of Information Searching and Information Retrieval Archived 2016-03-04 at the Wayback Machine. Journal of the American Society for Information Sciences and Technology. 61(8), 1517-1534
3. "Aplikasi Dot Product pada sistem temu balik aplikasi" by Rinaldi Munir https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2020-2021/Algeo12-Aplikasi-dot-product-pada-IR.pdf
4. "Vektor di ruang Euclidean" by Rinaldi Munir "https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2020-2021/Algeo12-Aplikasi-dot-product-pada-IR.pdf