LAPORAN TUGAS BESAR 2
IF2123 - ALJABAR LINIER DAN GEOMETRI

Oleh

Kelompok Uzumaki Bayu :

Rayhan Alghifari Fauzta	**(13519039)**	**K03**
M. Rafli Zamzami	**(13519067)**	**K03**
Raihan Astrada Fathurrahman	**(13519113)**	**K01**

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2020

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 vektor 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.

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:

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

1.2. Tujuan Tugas

Membuat sebuah search engine sederhana dengan model ruang vektor dan memanfaatkan cosine similarity.

1.3. Spesifikasi Tugas

1. Program mampu menerima search query. Search query dapat berupa kata dasar maupun berimbuhan.

2. Dokumen yang akan menjadi kandidat dibebaskan formatnya dan disiapkan secara manual. Minimal terdapat 15 dokumen berbeda sebagai kandidat dokumen. Bonus: Gunakan web scraping untuk mengekstraksi dokumen dari website.

3. Hasil pencarian yang terurut berdasarkan similaritas tertinggi dari hasil teratas hingga hasil terbawah berupa judul dokumen dan kalimat pertama dari dokumen tersebut. Sertakan juga nilai similaritas tiap dokumen.
4. Program disarankan untuk melakukan pembersihan dokumen terlebih dahulu sebelum diproses dalam perhitungan cosine similarity. Pembersihan dokumen bisa meliputi hal-hal berikut ini. a. Stemming dan Penghapusan stopwords dari isi dokumen. b. Penghapusan karakter-karakter yang tidak perlu.
5. Program dibuat dalam sebuah website lokal sederhana. Dibebaskan untuk menggunakan framework pemrograman website apapun. Salah satu framework website yang bisa dimanfaatkan adalah Flask (Python), ReactJS, dan PHP.
6. Kalian dapat menambahkan fitur fungsional lain yang menunjang program yang anda buat (unsur kreativitas diperbolehkan/dianjurkan).
7. Program harus modular dan mengandung komentar yang jelas.
8. Dilarang menggunakan library cosine similarity yang sudah jadi.

BAB II

LANDASAN TEORI

2.1. Cosine Similarity

Cosine similarity atau kesamaan kosinus adalah ukuran kemiripan antara dua vektor bukan nol dalam *inner product space*. Cosine similarity didapat dari kosinus sudut di antara dua vektor yang menandakan keduanya ada di arah yang sama atau tidak. Aplikasi utama dari kesamaan kosinus antara lain mengukur kesamaan antara dokumen dengan *query* pada sistem temu-balik informasi.

Kosinus dari dua vektor bukan nol dapat diturunkan dengan rumus perkalian Euclidean:

$$\mathbf{A} \cdot \mathbf{B} = \|\mathbf{A}\| \, \|\mathbf{B}\| \cos\theta$$

Jika terdapat dua vektor atribut ***A*** dan ***B***, kesamaan kosinus cos(θ) direpresentasikan dengan perkalian titik dan besaran sebagai berikut:

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

Dimana *Ai* dan *Bi* masing-masing adalah komponen vektor *A* dan *B*.

Hasil kemiripan yang didapat berkisar dari -1 atau sangat berlawanan sampai 1 atau sama persis. Nilai 0 menunjukkan ortogonalitas atau dekorasi, sedangkan nilai di antaranya menunjukkan kesamaan atau ketidaksamaan menengah.

Dalam kasus pencocokan teks, vektor atribut *A* dan *B* biasanya adalah vektor *term frequency* dari dokumen. Vektor *term frequency* ini tidak boleh negatif sehingga nilai cosine similarity yang dihasilkan pasti berkisar antara 0 dan 1. Sudut antara dua *term frequency* juga tidak boleh lebih besar dari 90°.

2.2. Term Frequency - Inverse Document Frequency

Term frequency - inverse document frequency (disingkat tf-idf) adalah statistik numerik yang mencerminkan tingkat kepentingan sebuah kata dalam dokumen atau kumpulan dokumen (korpus) pada sistem temu-balik informasi. Nilai tf-idf berbanding lurus dengan jumlah kemunculan sebuah kata dalam dokumen dan berbanding terbalik dengan jumlah dokumen dalam korpus yang mengandung kata tersebut. Tf-idf banyak digunakan dalam temu-balik informasi, penambangan teks, dan pemodelan pengguna.

Untuk menentukan term frequency *tf*, langkah paling mudah adalah menghitung jumlah kemunculan istilah *i* dibagi dengan total istilah dalam dokumen *j* sesuai dengan formula:

$$tf_{ij} = \frac{f_d(i)}{\max_{j \in d} f_d(j)}$$

Inverse document frequency mengukur seberapa banyak informasi yang diberikan oleh istilah tadi dari keseluruhan dokumen yang mengindikasikan istilah tersebut umum atau jarang dijumpai dalam korpus. Inverse document frequency diperoleh secara logaritmik dengan rumus:

$$\mathrm{idf}(t, D) = \log \frac{N}{|\{d \in D : t \in d\}|}$$

Dengan:
N : jumlah dokumen dalam korpus (*N* = |*D*|)
$|\{d \in D : t \in d\}|$: jumlah dokumen dengan kemunculan term *t*.
Jika tidak ada *t* dalam korpus, hal ini menimbulkan pembagian dengan nol. Untuk mengatasinya, penyebut biasanya ditambah dengan 1 untuk mencegah kemunculan nol.

Nilai keseluruhan tf-idf didapat dengan mengalikan tf dengan idf:

$$\mathrm{tfidf}(t, d, D) = \mathrm{tf}(t, d) \cdot \mathrm{idf}(t, D)$$

Nilai tf-idf yang tinggi menandakan term tersebut sangat relevan dalam kumpulan dokumen yang digunakan.

2.3. Preprocessing Dokumen

Pada *natural language processing* (NLP), informasi yang akan digali berisi data-data yang strukturnya "sembarang" atau tidak terstruktur. Oleh karena itu, diperlukan proses pengubahan bentuk menjadi data yang terstruktur untuk kebutuhan lebih lanjut (*sentiment analysis*, *topic modelling*, dll). Preprocessing ini umumnya dilakukan menggunakan *library* Python seperti NLTK untuk bahasa Inggris dan Sastrawi untuk bahasa Indonesia.
Terdapat empat tahapan utama dalam preprocessing dokumen yaitu:

1. Case folding
2. Tokenizing
3. Filtering / penghapusan stopwords
4. Stemming

Preprocessing dimulai dengan case folding atau mengubah semua huruf dokumen menjadi huruf kecil. Karakter yang diterima hanya huruf a sampai z sehingga karakter lain seperti angka, tanda baca, dan *whitespace* juga dihilangkan. Proses ini umumnya tidak memerlukan *library* eksternal dan dapat menggunakan modul bawaan Python seperti regex.

Tahap selanjutnya adalah tokenizing. *Tokenizing* adalah proses pemisahan teks menjadi potongan-potongan yang disebut sebagai token untuk kemudian dianalisis. Kata, angka, simbol, tanda baca dan entitas penting lainnya dapat dianggap sebagai token. Dalam NLP, token diartikan sebagai "kata" meskipun *tokenize* juga dapat dilakukan pada paragraf maupun kalimat.

Tahap ketiga adalah filtering atau penghapusan stopwords. *Stopword* adalah kata umum yang biasanya muncul dalam jumlah besar dan dianggap tidak memiliki makna. Contoh *stopword* dalam bahasa Indonesia adalah "yang", "dan", "di", "dari", dll. Makna di balik

penggunaan *stopword* yaitu dengan menghapus kata-kata yang memiliki informasi rendah dari sebuah teks, kita dapat fokus pada kata-kata penting sebagai gantinya.

Tahap terakhir preprocessing adalah stemming. *Stemming* adalah proses menghilangkan infleksi kata ke bentuk dasarnya, namun bentuk dasar tersebut tidak berarti sama dengan akar kata (*root word*). Misalnya kata “mendengarkan”, “dengarkan”, “didengarkan” akan ditransformasi menjadi kata “dengar”.

BAB III

IMPLEMENTASI

File-file yang digunakan, fungsi, dan kegunaannya:

1. docs_processing.py
   -> File yang berisi modul-modul yang digunakan untuk mengolah dokumen

   function is_link_double(input list: List, input filename: String, input idx: Int) -> boolean
   { I.S. : Parameter terdefinisi }
   { F.S. : Mengembalikan True jika filename sudah terdapat dalam list }

   function retrieve_docs() -> List
   { I.S. : Terdapat link eksternal berita yang telah ditetapkan }
   { F.S : Mengembalikan list dokumen yang berisi isi berita }

   function get_title() -> List
   { I.S. : Terdapat link eksternal berita yang telah ditetapkan }
   { F.S : Mengembalikan list dokumen yang berisi judul berita }

   function get_txt() -> List
   { I.S : Terdapat folder 'txt' yang telah berisi file-file dengan format (*.txt*) }
   { F.S : Mengembalikan list yang berisi isi konten dari file-file dengan format (*.txt*) }

   function title_txt() -> List
   { I.S : Terdapat folder 'txt' yang telah berisi file-file dengan format (*.txt*) }
   { F.S : Mengembalikan list yang berisi judul konten dari file-file dengan format (*.txt*) }

   function clean_docs(input documents: List) -> List
   { I.S. : documents terdefinisi }
   { F.S : Mengembalikan documents yang telah dibersihkan (menghilangkan angka, tanda petik, spacing yang lebih dari satu, serta me-*lowercase* dokumen) }

   function remove_stop_words(input documents: List) -> List
   { I.S. : documents terdefinisi }
   { F.S : Mengembalikan documents yang telah dihilangkan *stop words*-nya }

   function stemming(input documents: List) -> List
   { I.S. : documents terdefinisi }
   { F.S : Mengembalikan documents yang telah di-*stem* (menghilangkan imbuhan) }

function preprocessing_docs(input database: List) -> List
{ I.S. : Parameter terdefinisi }
{ F.S : Mengembalikan file yang telah di clean, dihilangkan stopwords, dan di stemm }

Function get_DataFrame(input database: List) -> DataFrame
{ I.S. : Parameter terdefinisi }
{ F.S : Mengembalikan database yang telah diubah menjadi tabel vektor }

function get_similiar(input query: Strings, df: DataFrame, database: List) -> List
{ I.S. : Seluruh parameter terdefinisi }
{ F.S : Mengembalikan list kemiripan query dengan data frame }

Function get_Tab(input query: Strings, documents: List) -> DataFrame
{ I.S : Parameter terdefinisi dan tidak kosong }
{ F.S : Mengembalikan tabel frekuensi dari dokumen-dokumen dengan kata pada query
}

2. main.py
   -> Backend dari *website*, menghubungkan dengan modul-modul yang ada sehingga website dapat berfungsi dengan tujuan awal

3. tfidf.py
   -> File yang berisi modul-modul yang digunakan dalam mengubah dokumen-dokumen menjadi vektor yang dapat diolah, (scratch vectorizer)

   Function tf(input wordDict: Dict, input bow: List) -> Dict
   { I.S. : Parameter terdefinisi, wordDict adalah kamus yang berisi kata-kata dan bow kata-kata yang muncul dalam dalam dokumen }
   { F.S : Mengembalikan terms frequency }

   Function idf(input listDocument: List) -> Dict
   { I.S : Parameter terdefinisi, listDocument merupakan list yang berisi kamus tiap dokumen }
   { F.S : Mengembalikan inverse document frequency }

   Function tfidf(input documents: List) -> List
   { I.S : Parameter terdefinisi, documents adalah file yang sudah melalui preprocessing }
   { F.S : Mengembalikan tf-idf dari documents }

   Function tfidf2(input query: String, input documents: List) -> List
   { I.S : Parameter terdefinisi }
   { F.S : Mengembalikan tf-idf dari query }

4. vectors.py
   -> File yang berisi modul-modul yang digunakan untuk melakukan perhitungan vector yang diperlukan untuk mencari similarity query dengan dokumen

   function dot_product(input vector1,vector2 : List) -> Integer
   { I.S : Parameter fungsi terdefinisi }
   { F.S : Mengembalikan hasil perkalian dot **(.)** dari vektor 1 dan vektor 2 }

   function norm(input vector: List) -> Integer
   { I.S : Parameter fungsi terdefinisi }
   { F.S : Mengembalikan norm atau panjang dari vektor }

5. Folder Static, berisi file-file css
   style.css -> Styling css dari website

6. Folder Templates, berisi file-file html

   about.html -> Struktur website halaman yang berisi informasi tentang website

   index.html -> form untuk mengupload dokumen atau URL dan menginput query

   layout.html -> Berisi navbar yang menampung link ke halaman about dan logo website

   result.html -> halaman untuk menampilkan hasil query pengguna

   Garis Besar Program

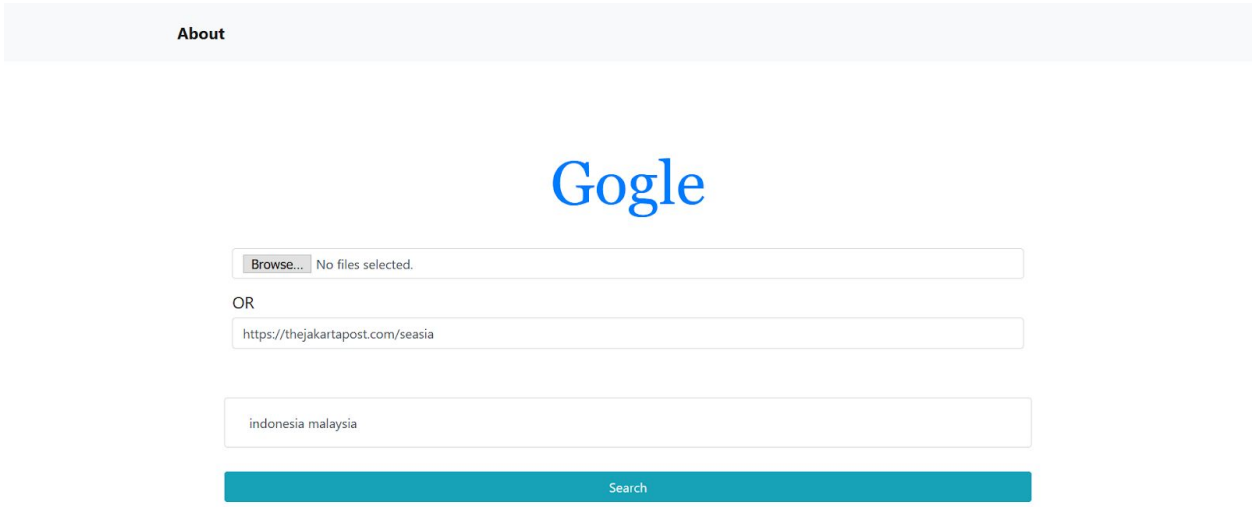
   Program memiliki cara kerja seperti *Search Engine* pada umumnya. Pada tampilan awal terdapat kata ‘About’, logo ‘Gogle’, kolom untuk mengupload file serta tombol ‘Choose Files’, kolom untuk memasukkan query (kalimat yang ingin di-*search*), kolom untuk memasukkan URL website dan juga tombol ‘Search’. Jika pengguna menekan kata ‘About’ maka pengguna akan diarahkan ke halaman ‘About’ yang berisi beberapa informasi tentang website. Pada halaman ‘About’ pengguna dapat menekan logo logo ‘Gogle’ untuk kembali ke tampilan awal. Jika pengguna menekan logo ‘Gogle’ maka pengguna akan tetap berada di tampilan yang sama, yaitu tampilan awal. Jika pengguna menekan tombol ‘Choose Files’ maka pengguna dapat mengupload file dengan format (*.txt*) yang kemudian dapat digunakan sebagai database untuk pencarian query. Jika file yang diunggah hanya satu maka kolom files akan menampilkan nama file tersebut, jika terdapat lebih dari 1 maka akan menampilkan banyak file, dan jika tidak terdapat file maka akan menampilkan tulisan ‘No file chosen’. Kolom untuk memasukkan URL website merupakan kolom yang bertujuan untuk mengambil dokumen-dokumen atau berita dari suatu website, tetapi kolom tersebut belum fungsional.

Kolom query adalah tempat bagi pengguna untuk memasukkan kata yang akan di-*search*. Setelah memasukkan kalimat ke dalam query, pengguna dapat menekan tombol ‘Search’ untuk menjalankan Searching. Jika input query kosong tetapi program menekan tombol ‘Search’ maka program akan tetap menampilkan halaman utama. Jika query tidak kosong dan terdapat file yang diupload maka program akan menggunakan file-file tersebut sebagai database. Jika query tidak kosong dan tidak terdapat file yang diupload maka program akan menggunakan website eksternal yang telah ditetapkan pada *source code* untuk mengakses berita-berita sebagai database. Tombol search mencari dokumen-dokumen pada database yang memiliki tingkat kemiripan yang tinggi antara query dan dokumen dengan menggunakan perhitungan cosine similarity. Program kemudian akan mengarahkan pengguna ke halaman hasil pencarian yang menampilkan logo ‘Gogle’ (jika ditekan akan kembali ke tampilan awal), query yang dimasukkan, kumpulan dokumen terurut berdasarkan similaritas tertinggi dan tiap dokumennya menampilkan judul, jumlah kata, persentase kemiripan, dan juga kalimat pertama dari dokumen. Selain itu, terdapat tabel frekuensi dari kata-kata pada query di bawah hasil search.

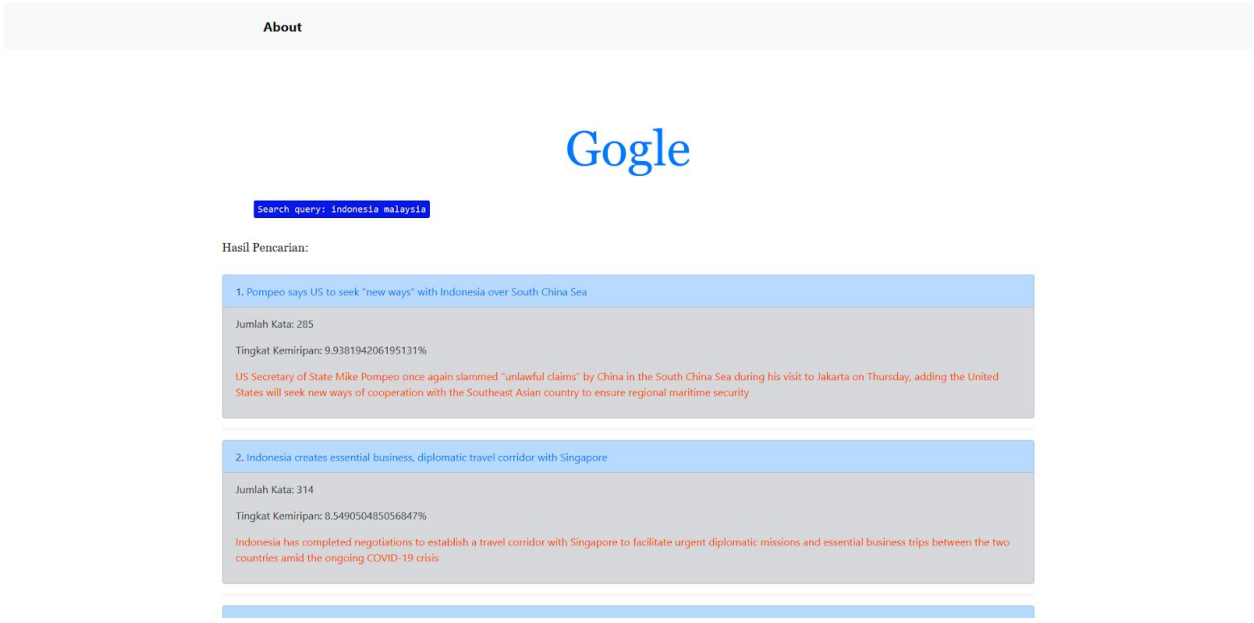
BAB IV

EKSPERIMEN

Menginput query dan dokumen dengan URL:



Hasil pencarian dengan URL:



4. ASEAN leaders begin summit amid 'major power rivalries'

Jumlah Kata: 337

Tingkat Kemiripan: 7.978067763437 96%

Southeast Asian leaders kicked off a multilateral summit on Thursday expected to address tensions in the South China Sea and tackle plans for a post-pandemic economic recovery in a region where US-China rivalry has been rising

5. Anwar Ibrahim meets king in bid to topple government

Jumlah Kata: 323

Tingkat Kemiripan: 5.4677208923260 13%

Malaysian opposition leader Anwar Ibrahim had a long-awaited meeting with the king Tuesday, seeking to prove he has support to take power and fulfil a decades-old ambition of becoming premier

6. Embassy urges Singapore to act upon violence against Indonesian worker

Jumlah Kata: 363

Tingkat Kemiripan: 5.3359398110917 84%

The Indonesian Embassy in Singapore has urged the Singaporean government to act upon a report of alleged violence committed against an Indonesian migrant worker in the city-state

7. Singapore reports single digit COVID-19 cases for first time since March

Jumlah Kata: 159

Tingkat Kemiripan: 4.2366655555668 12%

Singapore reported on Saturday new coronavirus cases in the single digits for the first time since March

Jumlah Kata: 289

Tingkat Kemiripan: 1.2910718795717 107%

The biggest party in Malaysia's ruling alliance has called for an election to be held once the coronavirus pandemic is over, as Prime Minister Muhyiddin Yassin grapples with a resurgence in infections and a leadership challenge

16. Malaysia police to summon Anwar over list of backers of PM bid

Jumlah Kata: 271

Tingkat Kemiripan: 1.2908773595333 367%

Malaysia's police said on Monday they have asked opposition leader Anwar Ibrahim to give a statement after they received complaints about names of lawmakers appearing in news media reportedly backing his claim to the premiership

17. Malaysian king rejects Muhyiddin's proposal for emergency declaration

Jumlah Kata: 477

Tingkat Kemiripan: 0.9855066260945 535%

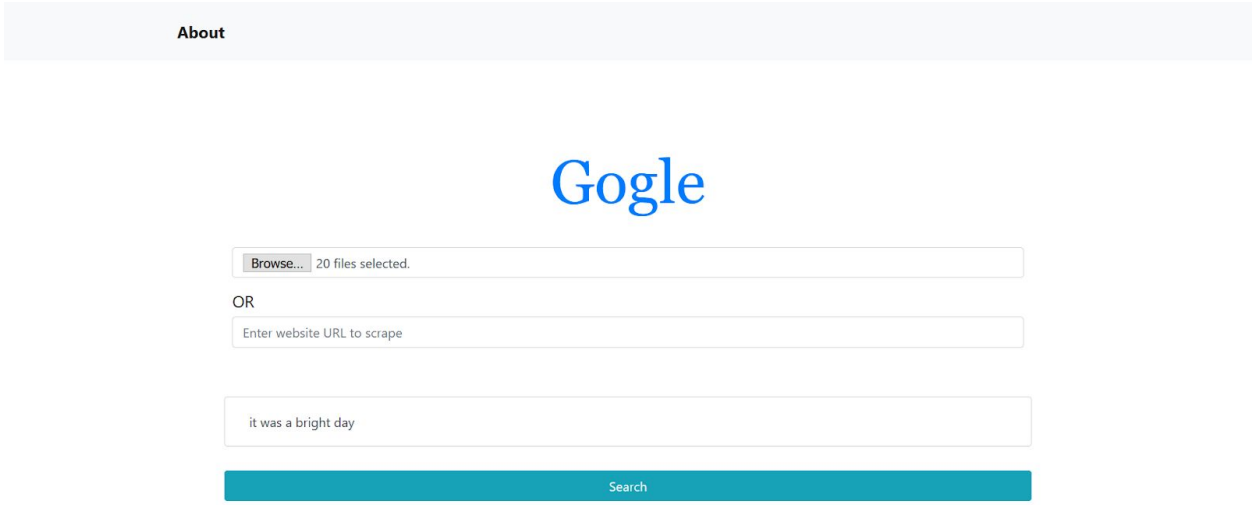
Malaysian King Abdullah Ri'ayatuddin has rejected Prime Minister Muhyiddin Yassin's proposal to declare a state of emergency to deal with a surge in coronavirus cases, the palace said Sunday

Tabel Frekuensi

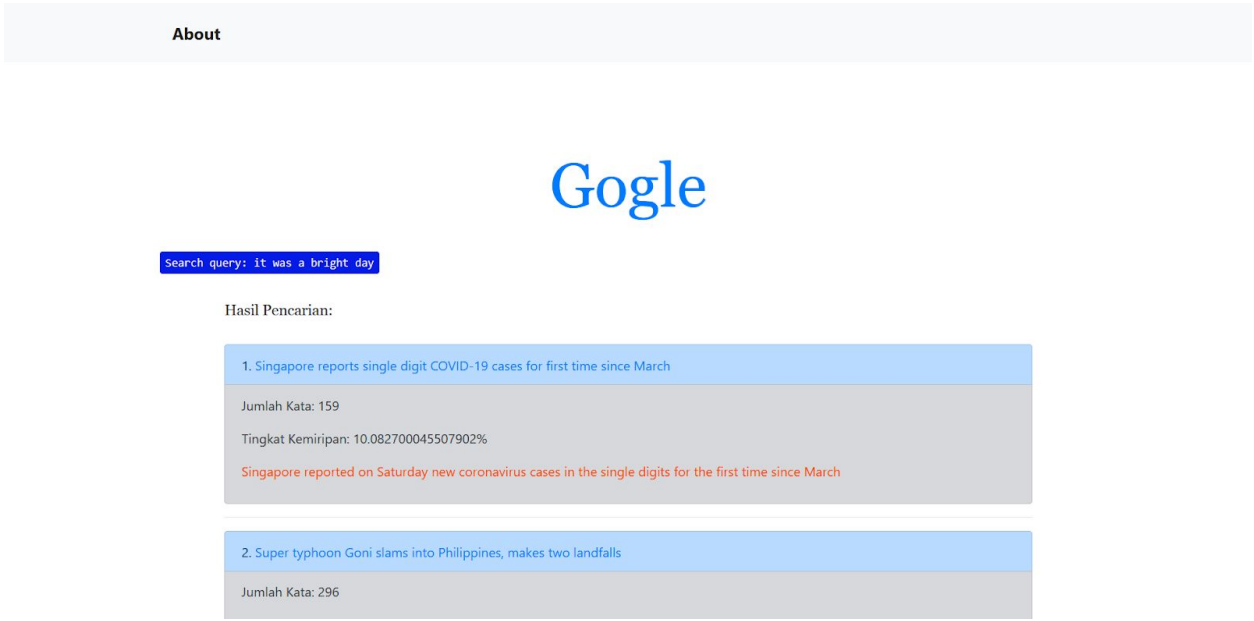
	Q	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16	D17	D18	D19	D20	D21	D22	D23	D24	D25
indonesia	1	2	0	2	0	0	3	0	0	0	4	0	1	0	0	0	0	0	0	0	0	0	0	4	0	1
malaysia	1	2	0	0	2	0	0	0	0	1	0	1	0	0	0	2	0	2	2	4	2	2	1	0	0	0

Tubes 2 Algeo 2020

Menginput query dan dokumen dengan upload dari local files:



Hasil pencarian dengan local files:



3. Ten dead, three missing as 2020's strongest typhoon slams Philippines

Jumlah Kata: 397

Tingkat Kemiripan: 5.696049786740747%

At least 10 people died and three others were missing after Typhoon Goni, the world's strongest typhoon this year, barrelled through the south of the Philippines' main island of Luzon on Sunday, an initial government report showed

4. Indonesian worker mauled to death by crocodile in Malaysia

Jumlah Kata: 266

Tingkat Kemiripan: 5.2778910546726125%

An Indonesian migrant worker from Medan, North Sumatra, was found dead by the Sarawak River in Malaysia, the result of a suspected crocodile attack

5. Philippines cancels 'Black Nazarene' parade as pandemic lingers

Jumlah Kata: 255

Tingkat Kemiripan: 3.419322634407544%

The Philippine capital Manila on Friday cancelled an annual procession of a centuries-old black wooden statue of Jesus Christ that draws millions of Roman Catholic devotees as the coronavirus pandemic continues to afflict the country

6. Suga, Vietnam's Phuc to focus on economic, defense cooperation

Jumlah Kata: 273

Tingkat Kemiripan: 3.267126385029583%

Japanese Prime Minister Yoshihide Suga started a meeting Monday with his Vietnamese counterpart Nguyen Xuan Phuc in Hanoi, with the two set to discuss economic and defense cooperation

7. Indonesia creates essential business, diplomatic travel corridor with Singapore

Jumlah Kata: 314

Tingkat Kemiripan: 2.543200265167021%

Indonesia has completed negotiations to establish a travel corridor with Singapore to facilitate urgent diplomatic missions and essential business trips between the two countries amid the ongoing COVID-19 crisis

Tabel Frekuensi

	Q	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16	D17	D18	D19	D20	D21	D22	D23	D24	D25
bright	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
day	1	0	0	0	2	0	0	2	2	0	0	0	1	1	0	0	0	0	0	0	0	0	0	1	0	2

Tubes 2 Algeo 2020

BAB V

SIMPULAN, SARAN, DAN REFLEKSI

5.1. Simpulan

Dari Tugas Besar 2 IF 2123 Aljabar Linier dan Geometri Semester 1 Tahun 2020/2021 ini dapat disimpulkan bahwa salah satu implementasi dari Materi Vektor dalam dunia nyata adalah pada Sistem Temu-balik Informasi atau *Information Retrieval System*. Kita dapat merepresentasikan informasi (dalam hal ini kata-kata) menjadi bentuk vektor-vektor yang berisi seberapa banyak kemunculan suatu kata. Setelah itu, kita dapat menggunakan perkalian dot (*dot product*) pada vektor dan juga norma atau panjang dari vektor, untuk mencari *cosine similarity* atau kemiripan antara masing-masing vektor yang merupakan representasi dari kata-kata. Cosine similarity inilah yang kemudian digunakan sebagai dasar dalam mesin-mesin pencari seperti *Google*, *Yahoo! Search*, dan juga *Bing*.

5.2. Saran

Dalam pengerjaan Tugas Besar 2 IF 2123 Aljabar Linier dan Geometri Semester I Tahun 2020/2021 ini, tentu saja pembuat program memiliki keterbatasan baik itu waktu dan maupun wawasan. Maka dari itu pembuat program mengakui bahwa program ini masih jauh dari kata sempurna dan terdapat berbagai macam hal yang dapat lebih ditingkatkan lagi. Beberapa hal yang dapat ditingkatkan seperti, menambah pembacaan file dari berbagai tautan (*link*) eksternal, merubah interface website sehingga lebih menarik & lebih sesuai dengan pengguna, mempercepat algoritma program sehingga searching dapat berjalan lebih cepat, dan bahkan menambah fitur predictive search layaknya mesin-mesin pencari yang kita kenal.

5.3. Refleksi

Refleksi yang penulis dapatkan dalam pengerjaan Tugas Besar 2 IF 2123 Aljabar Linier dan Geometri Semester I Tahun 2020/2021 ini adalah dalam proyek membuat suatu website akan sangat sulit jika tidak ada orang yang pernah membuatnya, terutama dalam bagian backend.

DAFTAR PUSTAKA

https://en.wikipedia.org/wiki/Cosine_similarity, diakses 14 November 2020

https://en.wikipedia.org/wiki/Tf%E2%80%93idf, diakses 14 November 2020

https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2020-2021/Algeo-12-Aplikasi-dot-product-pada-IR.pdf, diakses 14 November 2020

https://medium.com/@ksnugroho/dasar-text-preprocessing-dengan-python-a4fa52608ffe, diakses 14 November 2020