**APLIKASI DOT PRODUCT PADA SISTEM TEMU BALIK INFORMASI**  
Laporan Tugas Besar 2 IF 2123 Aljabar Linier dan Geometri   
Semester I Tahun 2020/2021

Oleh:

Kelompok 8 - tasdasdadwda

Rexy Gamaliel Rumahorbo 13519010

Dionisius Darryl Hermansyah 13519058

Wilson Tandya 13519209

**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA**

**INSTITUT TEKNOLOGI BANDUNG**

**2020**

**BAB I**  
**DESKRIPSI MASALAH**

**1.1. Deskripsi masalah**

Pada tugas besar 2 IF2123 Aljabar Linier dan Geometri ini, akan dibuat sebuah sistem temu balik informasi dalam bentuk website dengan memanfaatkan teori *dot product* dari vektor.

**1.2. Spesifikasi program**

Adapun spesifikasi dari program yang akan dibuat adalah sebagai berikut:

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 bentuk sebuah website lokal yang sederhana serta 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**  
**TEORI SINGKAT**

**2.1. Vektor**

Vektor merupakan kuantitas fisik yang memiliki besar dan arah. Salah satu contoh vektor adalah vektor di R2 dan R3 yang masing-masing memiliki 2 dan 3 komponen, serta dapat direpresentasikan ke dalam ruang 2 dimensi dan ruang 3 dimensi.

Sebuah vektor di ruang Rn umumnya dapat dinyatakan dengan **v** = (v1, v2, v3, …, vn). Misalkan terdapat dua buah vektor berdimensi n, **u** = (u1, u2, u3, …, un) dan **v** = (v1, v2, v3, …, vn). Hasil penjumlahan keduanya adalah

**u** + **v** = (u1+v1, u1+v2, u1+v3, …, u1+vn)

dan hasil hasil perkalian vektor **u** dengan skalar k adalah

k**u** = k(v1, v2, v3, …, vn) = (kv1, kv2, kv3, …, kvn)

Norma/*magnitude* vektor **v** adalah

**||v|| =** sqrt(v12 + v22 + v3 + … + vn2).

Terdapat dua operasi perkalian antarvektor: perkalian titik dan perkalian silang. Perkalian titik/*dot product* dari vektor **u** dan **v** adalah

**u** • **v** = u1v1 + u2v2 + u3v3 + … + unvn = **||u**|| ||**v**|| cos a

dengan a adalah sudut yang dibentuk **u** dan **v**. Ketidaksamaan Cauchy-Schwarz menyatakan

**u** • **v ≤ ||u**|| ||**v**||

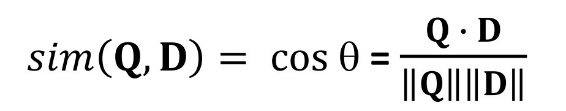
Kesamaan diperoleh saat kedua vektor searah, sementara nilai **u** • **v** = 0 diperoleh saat kedua vektor saling tegak lurus/*orthogonal*, dan nilai **u** • **v = -**||u|| **||v||** diperoleh saat kedua vektor berlawanan arah.

**2.2. *Retrieval information***

Sistem Temu-Balik Informasi atau *Information Retrieval System* merupakan kegiatan memperoleh informasi yang sesuai dengan kebutuhan dari suatu kumpulan informasi secara otomatis. Pencarian dapat didasarkan pada teks atau pengindeksan berbasis konten. *Information Retrieval System* digunakan untuk mengurangi atau mengantisipasi *information overload*. Salah satu dari pengaplikasian *Information Retrieval System* ini adalah *search engine*.

**2.3. *Cosine similarity***

*Cosine similarity* adalah besaran yang digunakan untuk mengukur seberapa mirip dokumen berdasarkan kemunculan kata terlepas dari panjang dokumen. Setiap dokumen direpresentasikan oleh vektor frekuensi yang setiap komponennya mewakili banyak kemunculan kata tertentu. Vektor frekuensi biasanya sangat panjang dan jarang (memiliki banyak nilai 0). Secara matematis, *cosine similarity*  mengukur nilai kosinus sudut antar dua vektor yang diproyeksikan dalam ruang multidimensi. Sebuah dokumen dapat diwakili oleh ribuan atribut, masing-masing merekam frekuensi kata tertentu (kata kunci). *Cosine similarity* dapat memberikan urutan (*ranking*) berdasarkan kemiripan dokumen terhadap *query*  yang diberikan. *Cosine similarity* dapat dihitung dengan persamaan berikut.

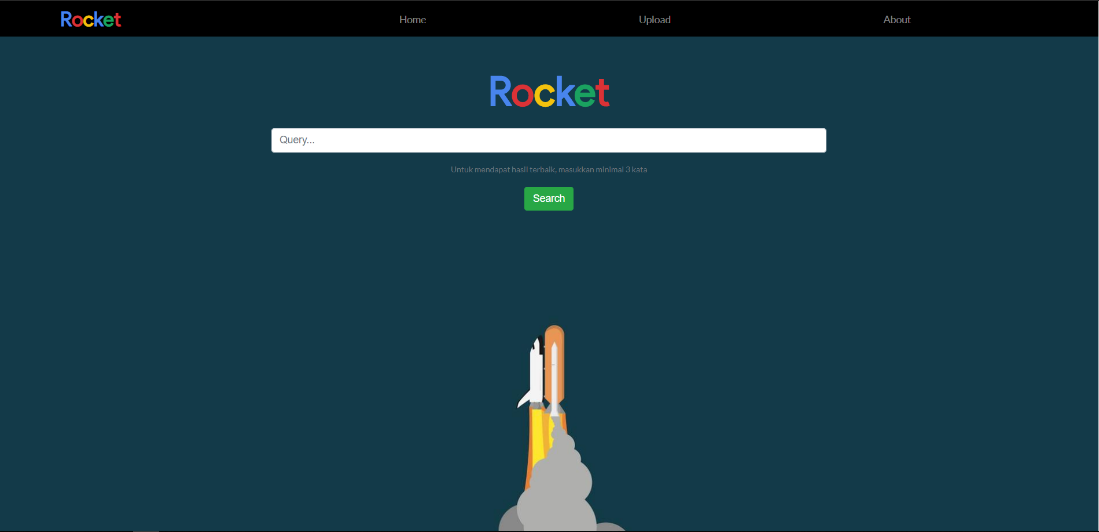


Dengan **Q** = (q1 , q2 , …, qn ) dan **D** = (d1 , d2 , …, dn ) dan **Q . D** adalah perkalian titik yang didefinisikan sebagai **Q .** **D =** (q1 d1 , q2 d2 , …, qn dn ). Nilai kosinus 0 menyatakan bahwa kedua vektor saling ortogonal dan tidak memiliki kesamaan. Semakin dekat nilai kosinus ke 1, semakin kecil sudut antar kedua vektor yang menyatakan bahwa nilai kesamaannya semakin tinggi.

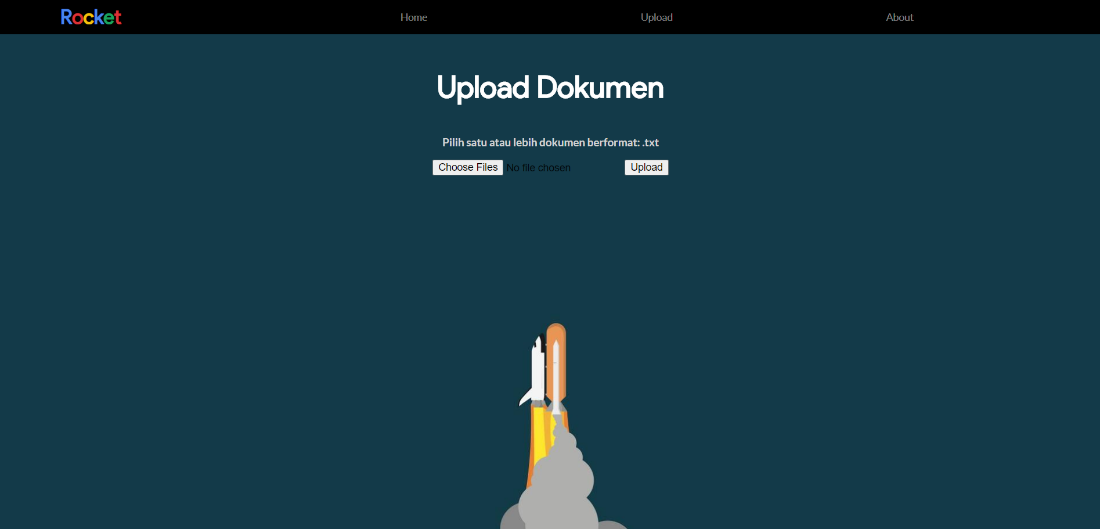
**BAB III**  
**IMPLEMENTASI PROGRAM**

Tampilan utama *website* kami ditunjukkan oleh gambar di bawah:

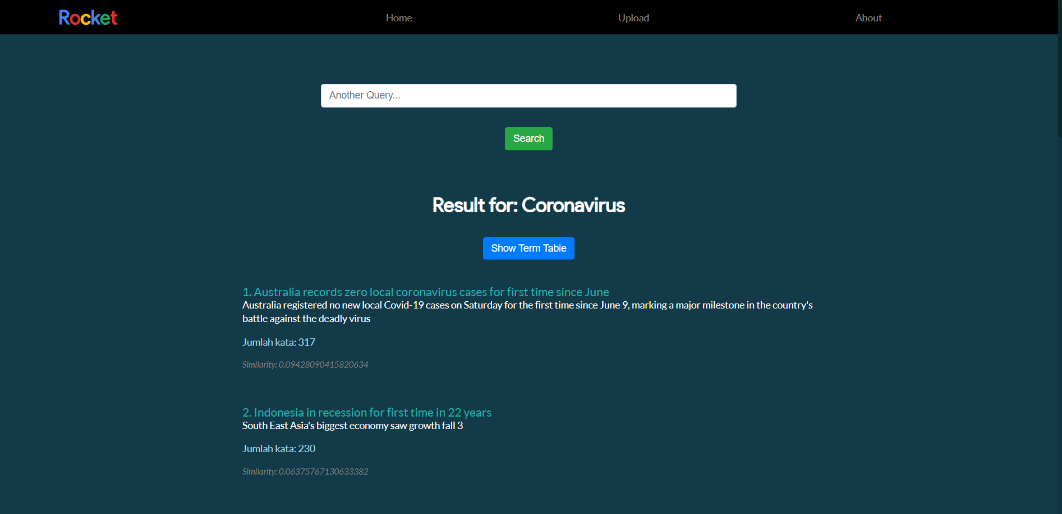
* Tampilan *home page*



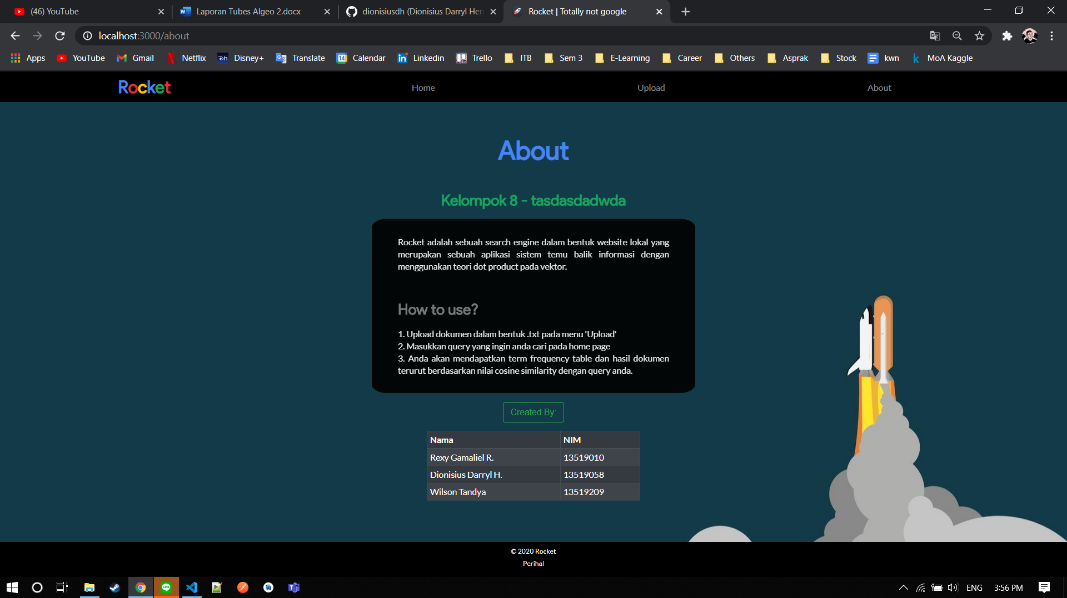
* Tampilan upload dokumen



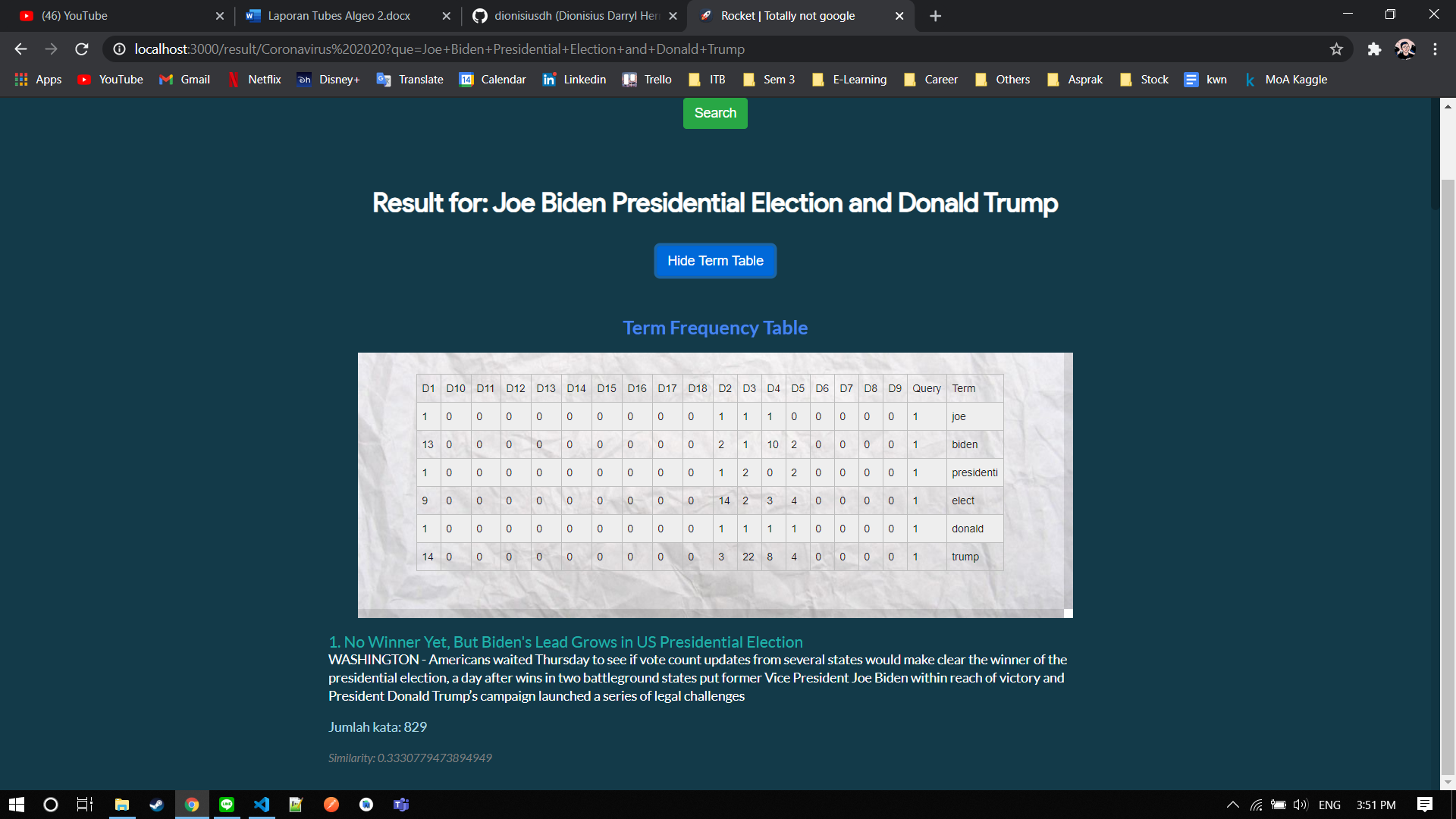
* Tampilan hasil pencarian query



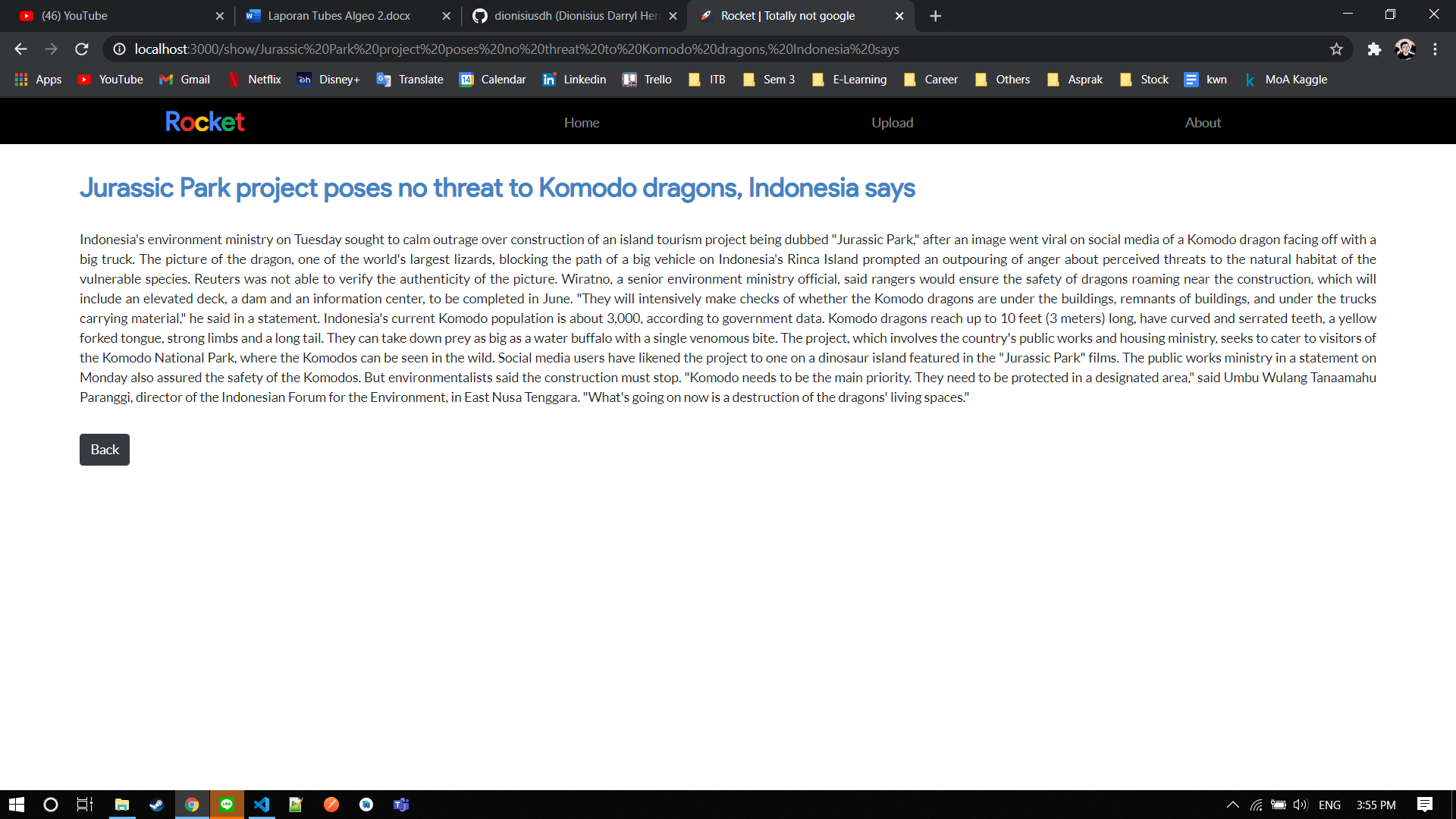
* Tampilan a*bout page*



* Tampilan *term frequency table*



* Tampilan isi dokumen



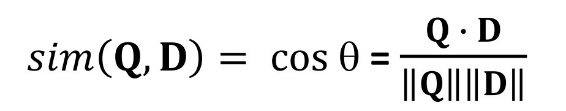
Arsitektur *website* yang dibuat dibagi ke dalam 2 komponen yaitu, komponen *front-end* dan *back-end.* Komponen frontend dibuat menggunakan React.js dengan bahasa JavaScript dan dihubungkan kepada backend yang diimplementasikan menggunakan framework Flask dengan bahasa Python.

Pada pemrosesan backend ada beberapa kelas dan modul yang dibuat. Modul dan kelas tersebut berjumlah 6, dengan fungsionalitas yang berbeda-beda. Berikut ini adalah penjelasan dari modul dan kelas yang ada tersebut secara lebih rinci:

|  |  |
| --- | --- |
| **Nama Fungsi/Prosedur** | **Keterangan** |
| **Main** | |
| Berisi sebuah aplikasi Flask yang digunakan sebagai API untuk melakukan *fetch* data dari *front-end* untuk diproses dan ditampilkan. Terdapat berbagai REST API yang didefinisikan menggunakan fungsi yang terdapat pada modul-modul di bawah ini. | |
| **Program** | |
| query\_sim | Mencari similarity dari query dari kumpulan dokumen yang ada di folder test. Output hasil query yang telah terurut berdasarkan value dalam bentuk dictionary |
| term\_frequency\_table | Menghasilkan term\_frequency\_table dari query terhadap dokumen yang ada dalam format HTML |
| sort\_dict | Sorting sebuah dictionary berdasarkan value |
| make\_json | Membuat dictionary menjadi berformat json. Mereturn hasil dalam bentuk  [ {'title' : ... ,  'sim': ...,  'first\_sentence':...}  , … ] |
| **Document** | |
| *Berisi semua hal yang berkaitan dengan dokumen. Kelas document, preprocessing document, dan menghasilkan token dari sekumpulan dokumen* | |
| \_\_init\_\_ | Inisialisasi class document dengan atribut-atributnya |
| preprocess | Preprocessing text dengan menghapus stopwords dan stemming |
| get\_tokens | Melakukan preprocessing dan menghasilkan token dari sekumpulan dokumen |
| get\_first\_sentence | Melakukan preprocessing dan menghasilkan kalimat pertama dari sekumpulan dokumen |
| get\_num\_words | Melakukan preprocessing dan menghasilkan jumlah kata dari sekumpulan dokumen |
| get\_table | Menghasilkan tabel perhitungan kata dalam format .json. Hanya menampilkan term global dari query  [{  "Term": ...,  "Query": ...,  "D1":...,  ...  }] |
| get\_table\_html | Menghasilkan format HTML dari tabel |
| **Reader** | |
| *Berisi semua hal yang berkaitan dengan pembacaan dokumen dari file* | |
| get\_files | Mereturn semua file yang ada dalam sebuah folder |
| read\_txt | Membaca file .txt |
| write | Writing sebuah document |
| **Vector** | |
| *Berisi semua hal yang berkaitan dengan vektor. Operasi vektor, menghitung magnitude, dot product, cosine similarity, dan mekonversi token kata menjadi vektor* | |
| vectorize | Membuat vektor token kata dokumen dari sekumpulan token kata dokumen. |
| term\_frequency | Membuat map frekuensi kata dokumen dari sekumpulan token kata dokumen |
| dot | Melakukan dot product pada vektor a dan b |
| magnitude | Melakukan dot product pada vektor a dan b |
| sim | Mencari nilai cosine similarity dari vektor a dan b. Prasyarat: Ukuran a dan b harus sama |

**BAB IV**  
**EKSPERIMEN**

Berdasarkan hasil eksperimen, dapat dianalisis bahwa frekuensi kesamaan kata bukanlah satu-satunya parameter dalam menghitung *cosine similarity*, namun juga jumlah kata unik pada dokumen yang bersangkutan. Semakin banyak kata unik yang ada, maka magnitudo vektor-vektor dokumen pun semakin besar. Hal ini berarti, sesuai dengan rumus:



nilai pembagi akan semakin besar, sehingga nilai *similarity* semakin berkurang.

Berikut merupakan hasil eksekusi *website* dan analisis terhadap contoh-contoh dokumen yang diberikan:

|  |  |
| --- | --- |
| **Query** | **Hasil** |
| “Coronavirus 2020” | Urutan dokumen: |
| Term frequency table: |
| "Joe Biden Presidential Election and Donald Trump” | Urutan dokumen: |
| Term frequency table: |
| “Building Komodo Jurassic Park in Indonesia” | Urutan dokumen: |
| Term frequency table: |
| “US Election 2020 Affecting Indonesian Stock Market” | Urutan dokumen: |
| Term frequency table: |
| “Donald Trump or Covid Cause Indonesia's Economic Recession” | Urutan dokumen: |
| Term frequency table: |

**BAB V**  
**KESIMPULAN DAN SARAN**

**5.1. Kesimpulan**

Kesimpulan yang dapat ditarik dari tugas besar ini yaitu sebuah sistem temu balik informasi dapat dirancang menggunakan teori aljabar vektor yaitu ruang vektor dan *dot product* dengan indikator utama *cosine similarity* serta dapat diimplementasikan secara nyata dalam bentuk sebuah *website* lokal menggunakan teknologi Flask dan React.js.

**5.2. Saran**

Untuk pengembangan lagi kedepannya, penulis memberikan saran sebagai berikut:

1. Mendeploy website ke internet agar dapat diakses khalayak umum.
2. Memperbanyak fitur-fitur *search engine* pada website.

**5.3. Refleksi**

Melalui pengerjaan tugas besar 2 IF2123 Aljabar Linier dan Geometri ini, penulis memperoleh banyak hal baik dari segi akademik maupun non-akademik. Penulis dapat memahami materi tentang vektor serta aplikasinya secara lebih mendalam dan jelas. Melalui implementasi algoritma-algoritma secara nyata dalam program, hal ini membantu penulis dalam memahami materi yang ada. Selain itu, penulis juga belajar untuk bekerja sama dalam tim terutama dalam mengerjakan sebuah proyek serta dapat melatih skill manajemen waktu. Seluruh hasil yang telah direfleksikan ini, diharapkan dapat membantu penulis untuk berkembang ke arah yang lebih baik lagi.

**DAFTAR REFERENSI**

Akeela, K. 2020. *Implementing the TF-IDF Search Engine*. Dilansir dari https://medium.com/@kartheek\_akella/implementing-the-tf-idf-search-engine-5e9a42b1d30b.

Grimberg, M. 2020. *How To Create a React + Flask Project.* Dilansir dari https://blog.miguelgrinberg.com/post/how-to-create-a-react--flask-project.

Khalid, I. A. 2020. *Create A Simple Search Engine Using Python*. Dilansir dari <https://towardsdatascience.com/create-a-simple-search-engine-using-python-412587619ff5>.

Munir, R. 2020. *Aplikasi Dot Product pada Sistem Temu-balik Informasi.* Bandung: Institut Teknologi Bandung.

N.N. 2020. *Tugas Besar 2 IF 2123 Aljabar Linier dan Geometri Aplikasi Dot Product pada Sistem Temu-balik Informasi Semester I Tahun 2020/2021*. Bandung: Institut Teknologi Bandung.

Symth, P. 2018. *Creating Web APIs with Python and Flask*. Dilansir dari <https://programminghistorian.org/en/lessons/creating-apis-with-python-and-flask>.

Wahyuni, R. T. 2017. *Penerapan Algoritma Cosine Similarity dan Pembobotan TF-IDF pada Sistem Klasifikasi Dokumen Skripsi.* Semarang: Universitas Negeri Semarang.