**LAPORAN TUGAS BESAR 2**

**IF2123 ALJABAR LINIER DAN GEOMETRI**

**APLIKASI DOT PRODUCT PADA SISTEM TEMU-BALIK INFORMASI**

**SEMESTER I TAHUN 2020/2021**





Disusun oleh:

Ruhiyah Faradishi Widiaputri (13519034)

Melita (13519063)

Akifa Nabil Ufairah (13519179)

**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA**

**INSTITUT TEKNOLOGI BANDUNG**

**2020**

**DAFTAR ISI**

[**BAB I**](#_v0j3899yzlc4) **2**

[**BAB II**](#_pbwfp1o840t0) **3**

[**BAB III**](#_vguqgg519jhi) **6**

[**BAB IV**](#_39gadgvaue9i) **9**

[**BAB V**](#_3rgph193m4k1) **15**

[**REFERENSI**](#_p5b6hfafwao) **16**

# 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 di dalam , dimana nilai 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:



Pada kesempatan ini, kalian ditantang untuk membuat sebuah *search engine* sederhana dengan model ruang vektor dan memanfaatkan *cosine similarity*.

# BAB II

**Teori Singkat**

**2.1. Vektor**

2.1.1. Pengertian vektor

Vektor merupakan kuantitas fisik yang memiliki besar dan arah. Vektor dilambangkan dengan huruf-huruf kecil yang dicetak tebal atau diberikan tanda panah (jika berupa tulisan tangan), misalkan **u**, **v**, **w**. Vektor di ruang 2D atau 3D dapat direpresentasikan dengan menggunakan panah.



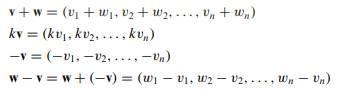
→

Jika suatu vektor **v** mempunyai titik asal A dan titik terminal B, maka **v** = AB. Vektor-vektor **u** = (u1, u2,...,un) dan **v** = (v1, v2,...,vn) di dikatakan sama (**u** = **v**) jika:



2.1.2. Operasi dasar vektor

Jika **u** = (u1, u2,...,un) dan **v** = (v1, v2,...,vn) adalah vektor-vektor di , dan jika k adalah sembarang skalar maka didefinisikan:



2.1.2. Panjang vektor

Panjang dari suatu vektor **v** = (v1,v2,...,vn) disebut juga norma dari **v**, dilambangkan dengan ||**v**|| didefinisikan sebagai berikut:



Jika **v** adalah vektor di Rn dan jika k adalah skalar maka:

* ||**v**|| ≥ 0
* ||**v**|| = 0 jika dan hanya jika **v** = **0**
* ||k**v**|| = |k|||**v**||

2.1.3. Dot Product

Jika **u** dan **v** adalah vektor-vektor tidak nol di R2 dan R3, dan jika θ adalah sudut di antara **u** dan **v**  maka dot product (perkalian titik) dari **u** dan **v**, dilambangkan dengan **u** ⋅ **v** didefinisikan sebagai:



Jika **u** = **0** atau **v** = **0**, maka **u** ⋅ **v** = 0.

Dari formula di atas kita dapat menentukan besarnya sudut yang dibentuk oleh 2 vektor **u** dan **v**, yaitu dari

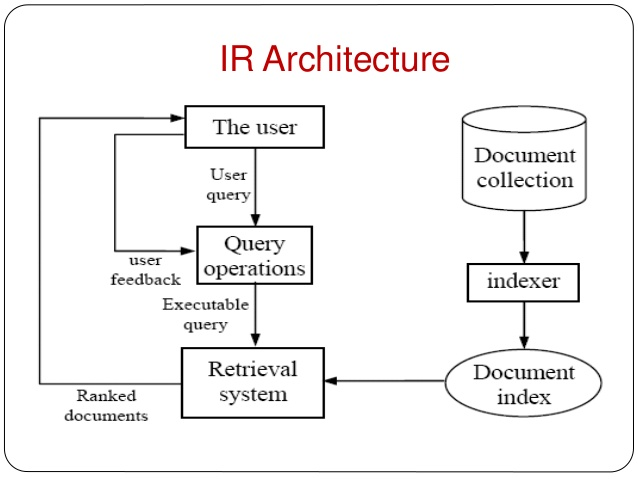


Jika **u** = (u1, u2,...,un) dan **v** = (v1, v2,...,vn) adalah vektor-vektor di Rn, maka dot product dari **u** dan **v** juga dapat dinyatakan sebagai berikut:



**2.2. Sistem Temu Balik Informasi**

Sistem Temu Kembali Informasi (STKI) atau *Information Retrieval System* (IRS) digunakan untuk menemukan kembali *(retrieve)* informasi-informasi yang relevan terhadap kebutuhan pengguna dari suatu kumpulan informasi secara otomatis. Sistem temu balik informasi umumnya digunakan pada pencarian informasi yang isinya tidak terstruktur, seperti dokumen dan laman web.



Gambar: arsitektur sistem temu balik informasi

Salah satu penerapan dari sistem temu balik informasi ini adalah pada *search engine*. Sistem temu balik informasi membantu pengguna dalam menemukan informasi yang dibutuhkannya. Selain itu sistem temu balik informasi ini juga digunakan dalam filter spam pada email program untuk mengklasifikasikan email mana yang merupakan spam dan mana yang bukan spam.

Sistem temu balik informasi mempunyai kemampuan untuk menampilkan, menyimpan, mengatur, dan mengakses *item-item* informasi. Untuk melakukan pencarian dibutuhkan *query*, yaitu sekumpulan *keyword*.

**2.3. Cosine Similarity**

*Cosine similarity* adalah salah satu cara untuk menentukan seberapa besar kesamaan antara 2 buah dokumen. Model ini menggunakan teori di dalam aljabar vektor.

Misalkan terdapat n kata berbeda sebagai kamus kata (*vocabulary*) atau indeks kata (*term index*). Kata-kata tersebut membentuk ruang vektor berdimensi n. Setiap dokumen maupun *query* dinyatakan sebagai vektor **w** = (w1, w2,..., wn) di dalam dengan adalah jumlah kemunculan setiap kata *i* di dalam dokumen.

Secara matematis, jika **d** adalah vektor yang kita peroleh dari suatu dokumen dan **q** adalah vektor yang kita dapatkan dari *query* seperti di atas, *cosine similarity* mengukur kosinus dari sudut yang dibentuk di antara 2 vektor **d** dan **q.**

Dengan kata lain kesamaan (sim) antara dua vektor **Q** = (q1, q2, …, qn) dan **D** = (d1, d2, …, dn) dapat ditentukan dengan:



Dengan **Q**,**D** adalah perkalian titik yang didefinisikan sebagai:



# BAB III

**Implementasi Program**

**3.1. Struktur Class yang Didefinisikan**

1. Flask
2. StemmerFactory
3. StopWordRemoverFactory
4. Counter
5. BeautifulSoup

**3.2. Garis Besar Program**

Berikut adalah *working directory* program *search engine* pada *website* yang kami buat.

* doc
* src
  + static
    - images
    - styles
      * styles.css
    - uploads
  + templates
    - about.html
    - index.html
    - upload.html
  + index.py
  + index2.py
* test
* README.md
* screenshot.png
* screenshot2.png

Terdapat file index.py dan index2.py yang merupakan *backend* dari *website*. File index.py berisi program yang melakukan proses terhadap data yang didapat dari web scrapping, sedangkan index2.py melakukan pemrosesan terhadap data yang diterima dari input manual dokumen. Pada program *file* index.py dan index2.py kami membuat 13 buah fungsi dan prosedur, yaitu:

1. getdata()

* Prosedur ini hanya ditemui di index.py dan berfungsi untuk mengumpulkan data dari *web scraping* dengan bantuan BeautifulSoup pada *website* alodokter.

1. countWordsArticles(df)

* Fungsi untuk menghitung banyak kata pada setiap artikel dan mengembalikan sebuah array yang menyimpan banyak kata dari setiap artikel yang telah dikumpulkan sebelumnya.

1. clean\_text(text)

* Fungsi yang menerima parameter string ini akan melakukan proses *cleaning data* berupa penghapusan karakter-karakter yang tidak diperlukan seperti angka dan tanda baca menggunakan bantuan modul string, serta menghapus *stopwords* dan melakukan *stemming* menggunakan *library* sastrawi.

1. clean\_articles(Articles)

* Fungsi yang menerima *input* sebuah *array* yang berisi artikel-artikel yang digunakan pada program ini, lalu melakukan pembersihan data pada setiap artikel pada *array* tersebut dengan memanggil kembali fungsi clean\_text(text)

1. nilaidot(vec,q\_vec)

* Fungsi yang menerima 2 buah *array* yang merepresentasikan vektor sebagai parameternya dan mengembalikan hasil perkalian dot dari kedua vektor tersebut.

1. panjangvektor(vector)

* Fungsi yang menerima sebuah *array* yang merepresentasikan vektor sebagai parameter *input* dan mengembalikan panjang dari vektor tersebut.

1. get\_sorted\_sim(q,df)

* Fungsi ini terlebih dahulu akan mengubah *query* q ke dalam bentuk vektor, lalu menghitung nilai similarity dari *query* q dengan setiap artikel yang terdapat pada df dengan memanggil fungsi-fungsi sebelumnya. Setelah mendapat nilai *similarity* dari *query* dengan tiap artikel yang disimpan dalam sebuah *array* sim, *array* ini akan *di-sorting* mulai dari nilai *similarity* terbesar hingga terkecil. Kemudian, fungsi akan mengembalikan *array* yang sudah di*-sorting* ini.

1. getdatafile()

* Fungsi ini dipanggil saat mengumpulkan data dari *file* .txt.

1. get\_unique\_words(articles)

* Fungsi untuk mencari kata-kata unik dalam artikel (len>=2).

1. vectorize(articles,unique\_words)

* Fungsi untuk membuat DataFrame dari artikel.

1. vektorquery(query,unique\_words)

* Fungsi untuk mengubah *query* menjadi vektor.

1. listterm(qkata)

* Fungsi untuk membuat *array* berisi *query.*

1. kolterm(kata, arrterm)

* Fungsi untuk menghitung kata pada *query.*

Selain fungsi di atas, juga terdapat 4 fungsi dari website ini yang dibuat dengan menggunakan Web Framework Flask, yaitu:

1. index()

Fungsi ini akan dipanggil saat halaman /index diakses. Saat *method* yang terbaca adalah POST, yaitu saat user memasukan input *query* maka fungsi ini akan melakukan perhitungan *similarity* untuk menampilkan *output* yang sesuai dengan menggunakan fungsi-fungsi yang sebelumnya sudah dijabarkan. Kemudian index() akan mengembalikan data yang akan digunakan pada *file* index.html untuk ditampilkan ke *website.*

1. about()

Fungsi akan dipanggil saat halaman /about diakses, yaitu saat pengguna mengklik pada tulisan perihal di halaman indeks.

1. upload\_form()

Fungsi dipanggil saat halaman /upload diakses.

1. upload\_file()

Fungsi dipanggil saat *submit* file .txt.

Selanjutnya pada folder templates, terdapat 3 *file* html, yaitu index.html, upload.html, dan about.html yang masing-masing akan ditampilkan sesuai dengan halaman *website* yang diakses. Secara garis besar kedua html ini terbagi menjadi 2 bagian yaitu head dan body. Bagian head berisi title dan link ke stylesheets dari web page terkait. Sedangkan pada bagian body, terdapat informasi yang ingin ditampilkan ke *website*.

Pada folder static, terdapat *resources* tambahan yang kami gunakan pada *website*, yang terbagi menjadi 2 folder, yaitu images dan styles. Pada folder images terdapat *image file* yang ditampilkan pada *html file*. Sedangkan pada folder style terdapat file styles.css yang digunakan sebagai stylesheet di *file* html. Selain itu, pada static ada satu folder lagi bernama uploads yang menyimpan data .txt yang pernah di*-upload* ke *website.*

# BAB IV

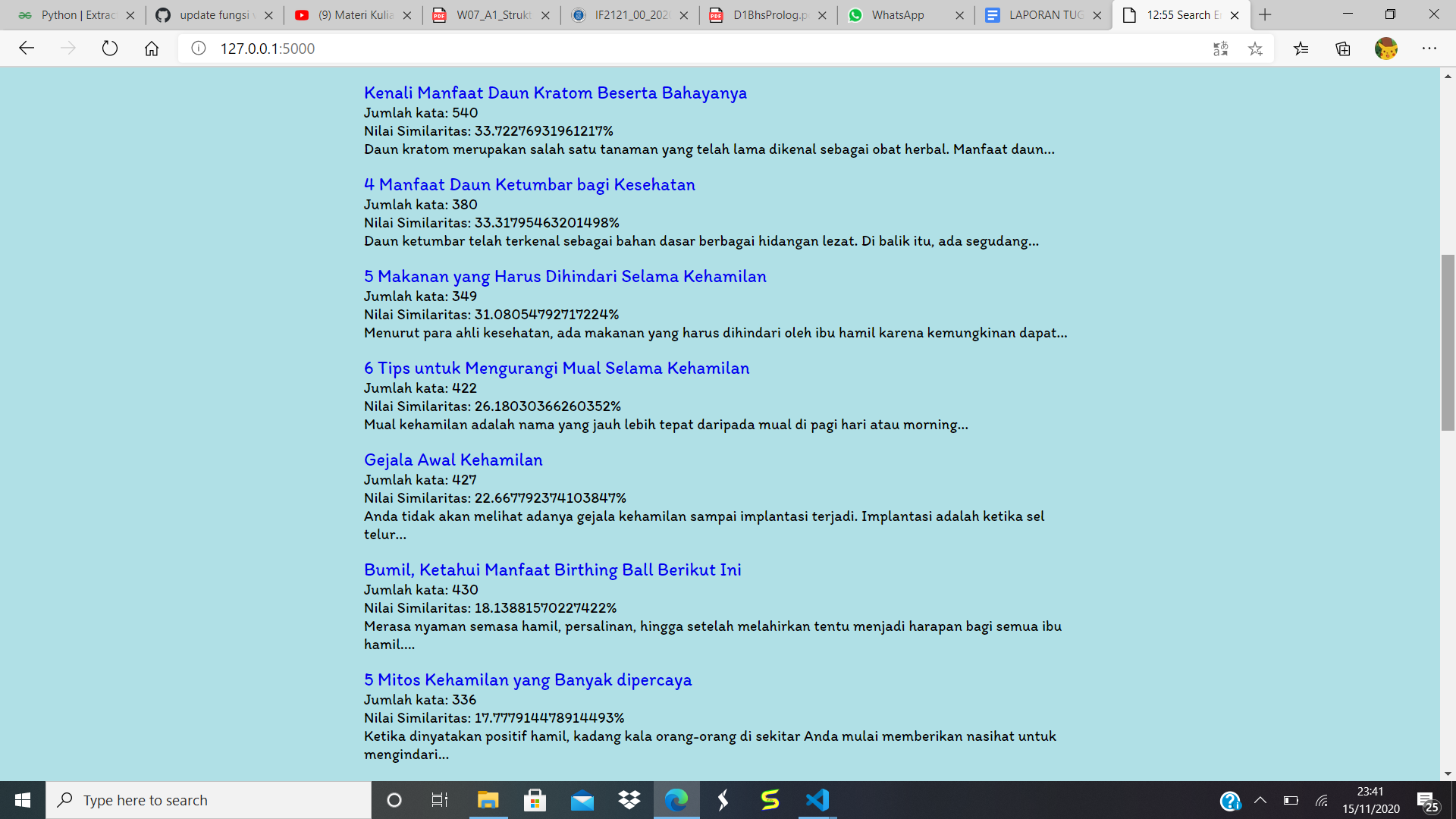
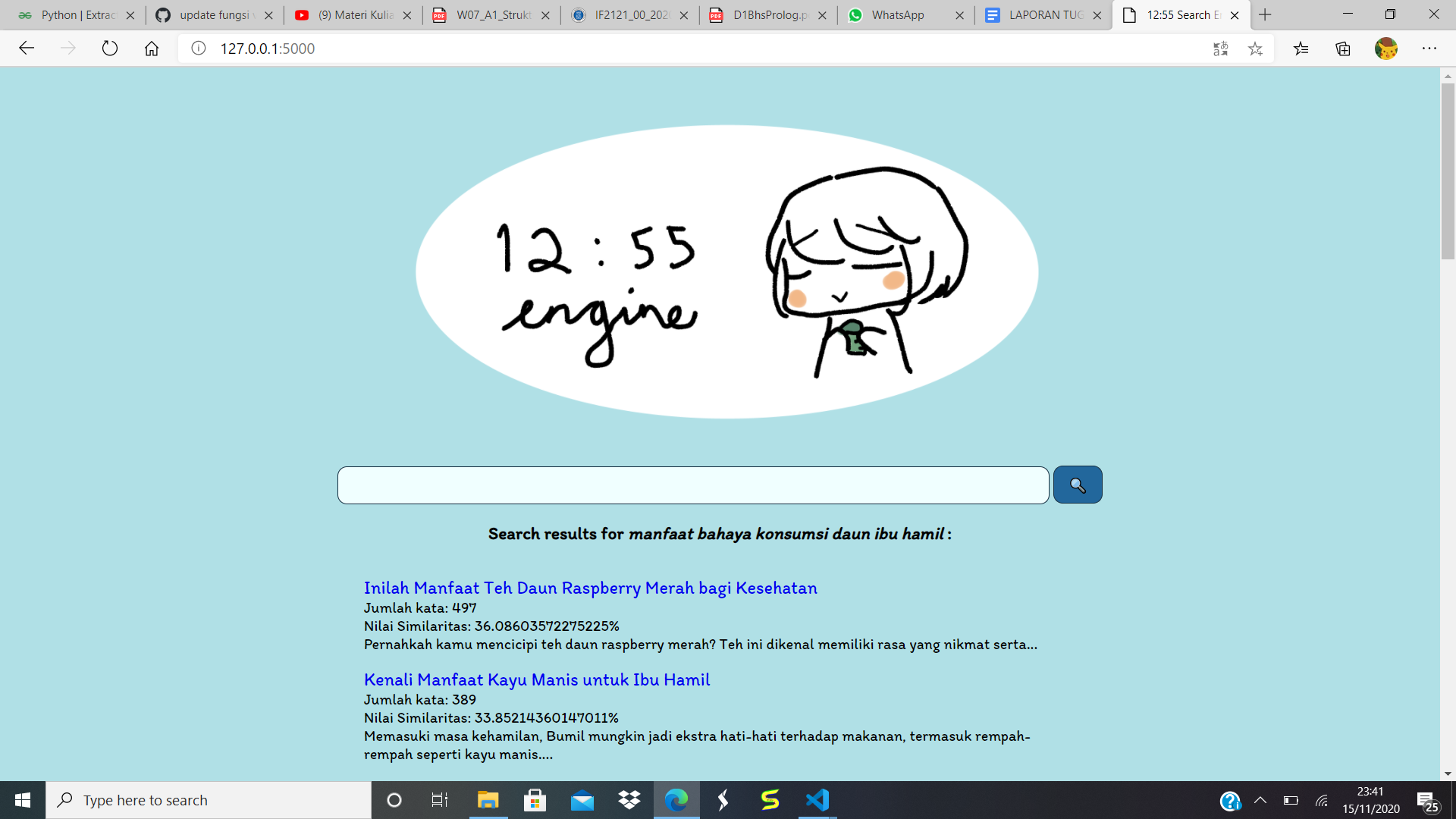
**Eksperimen**

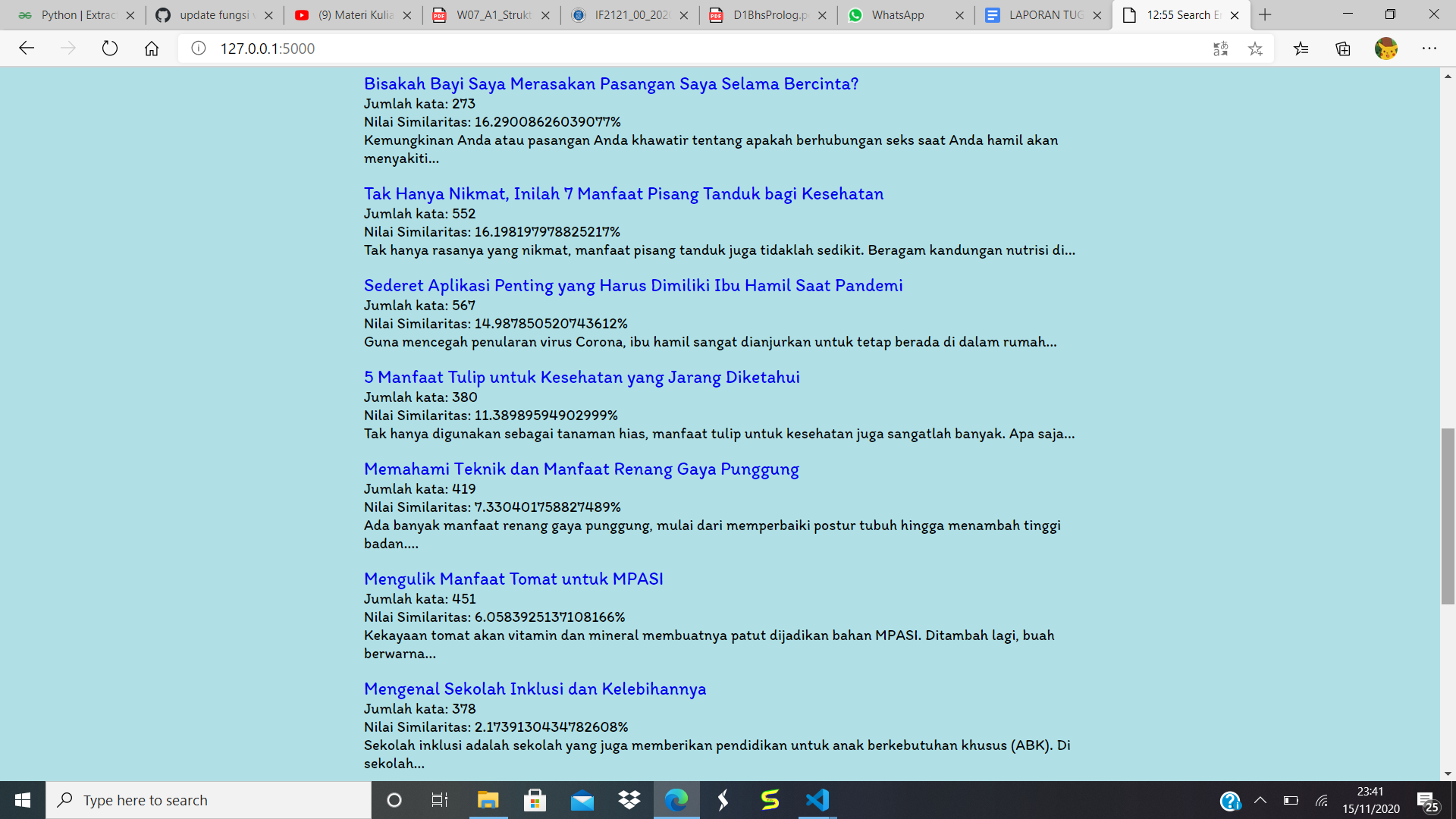
**4.1. Eksperimen dengan Dokumen dari *Web Scraping***

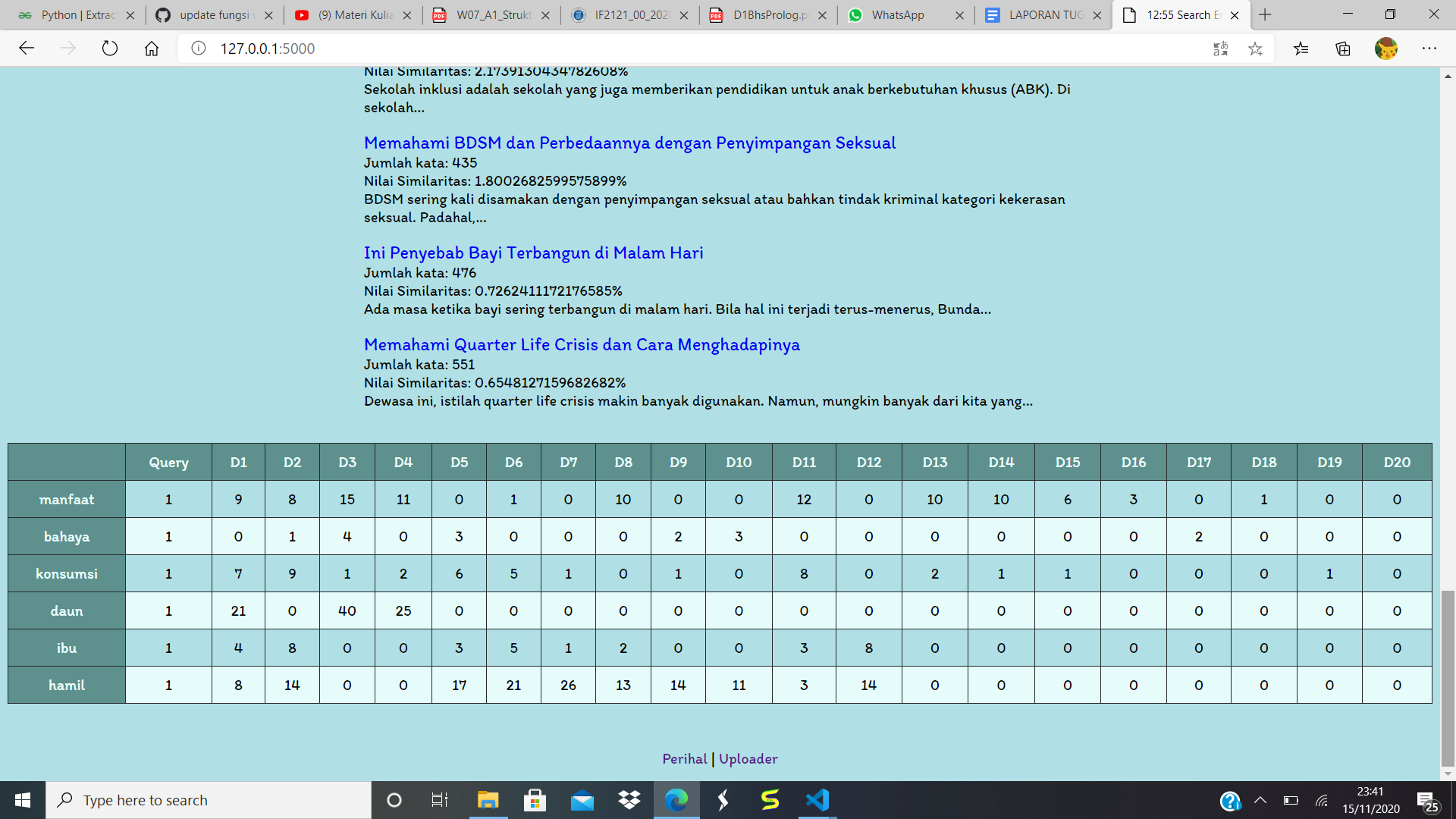
4.1.1. Kasus 1:

Query = manfaat dan bahaya konsumsi daun-daunan untuk ibu hamil

Hasil:



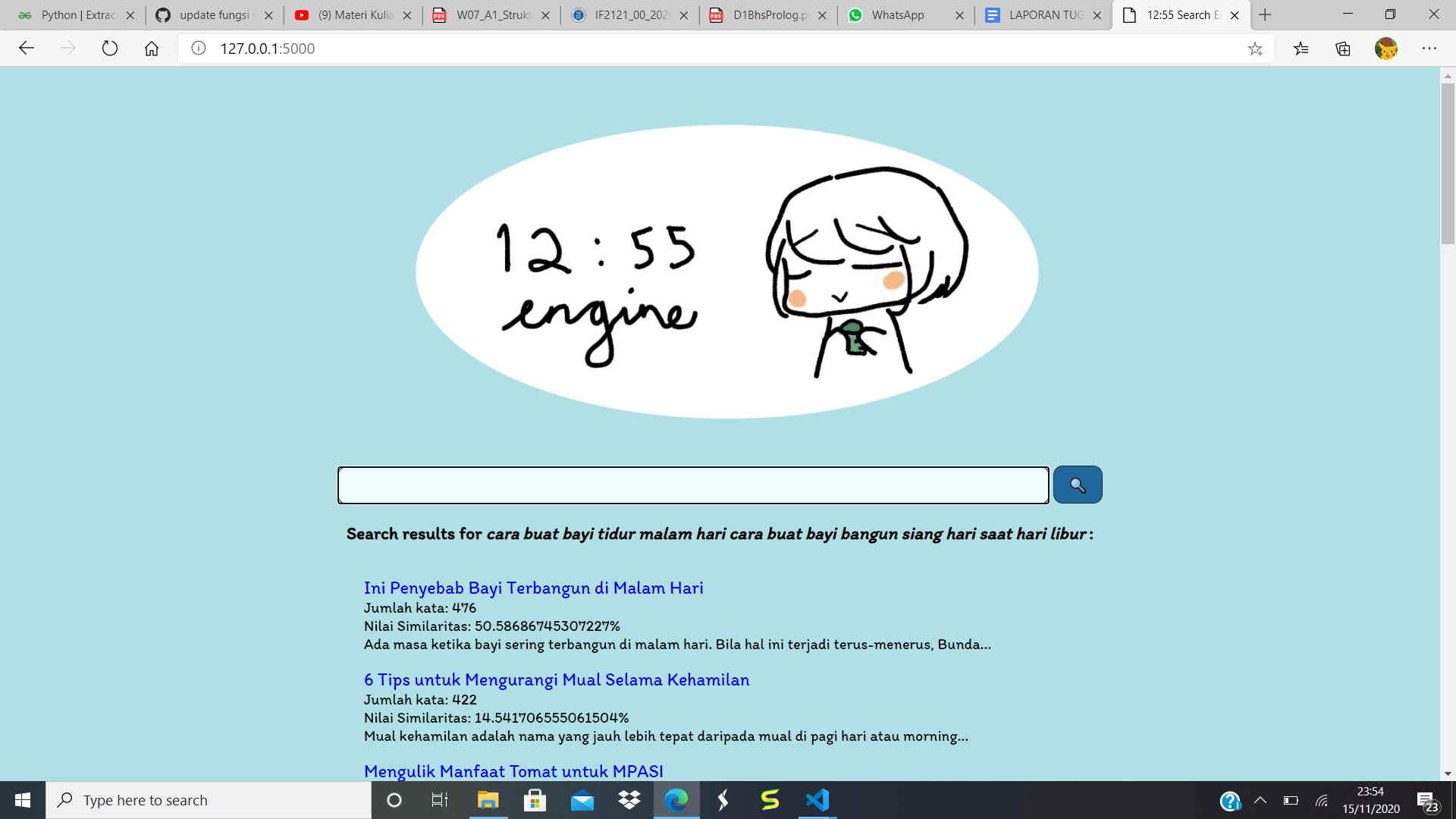


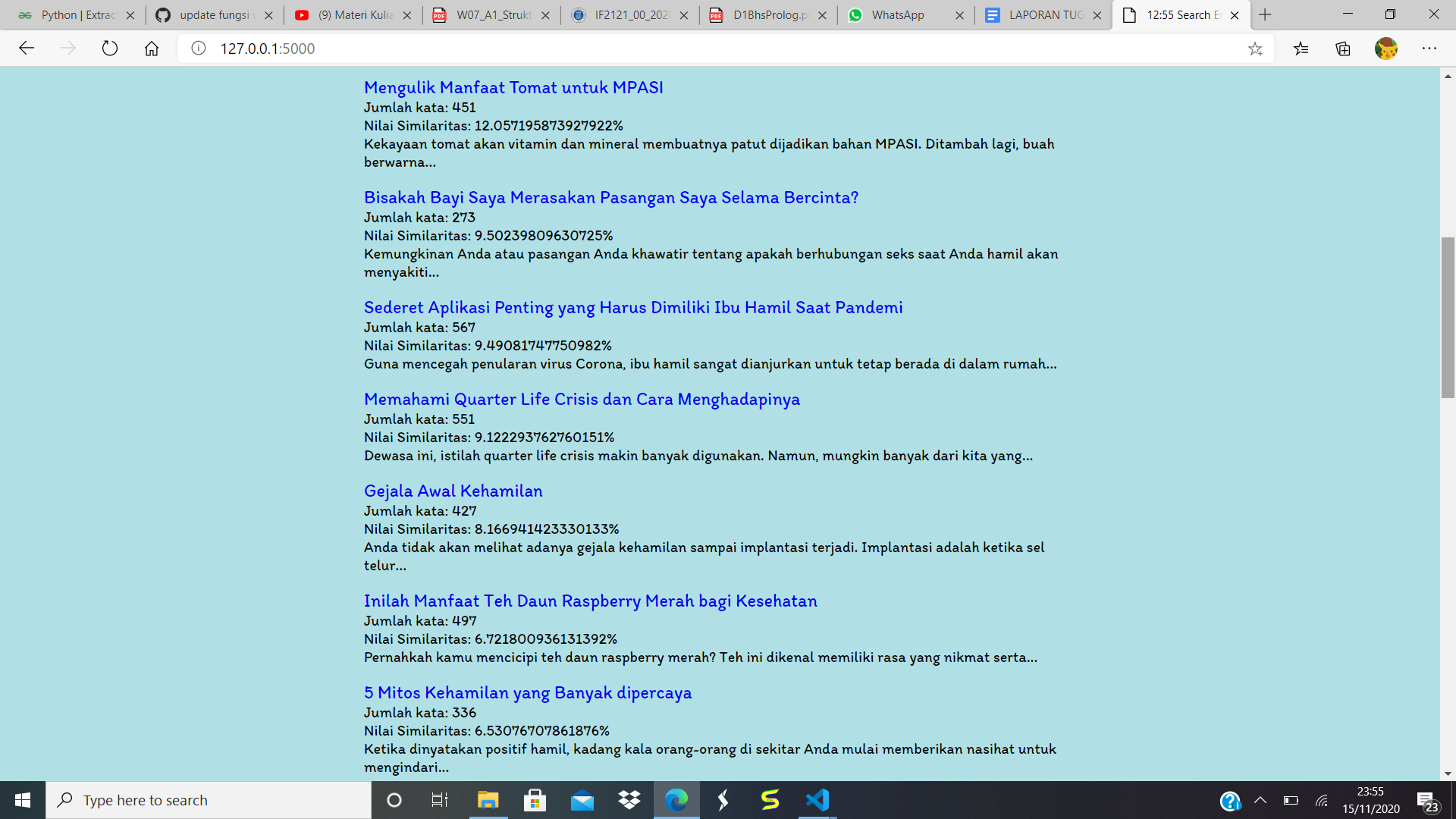


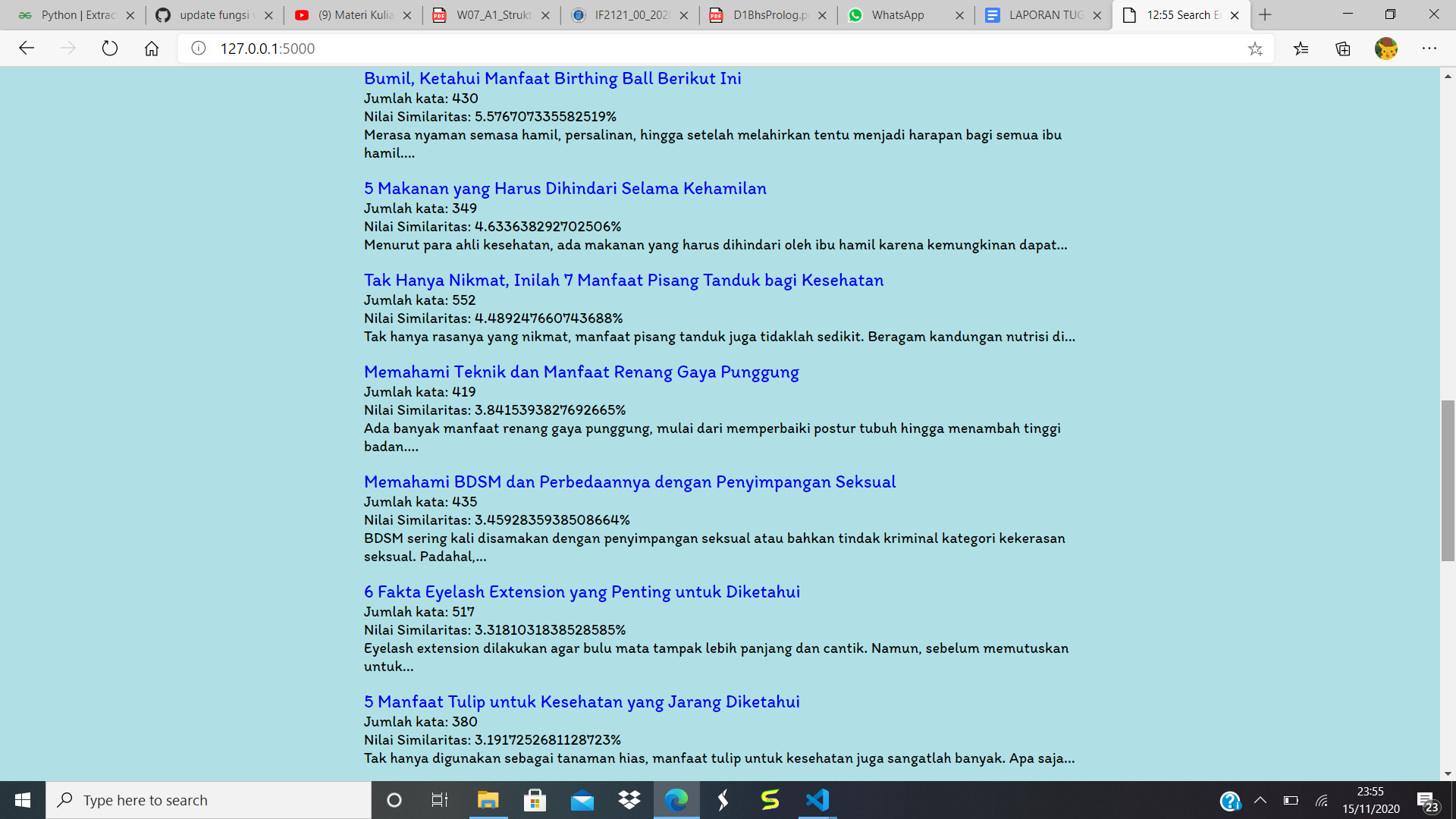
4.1.2. Kasus 2:

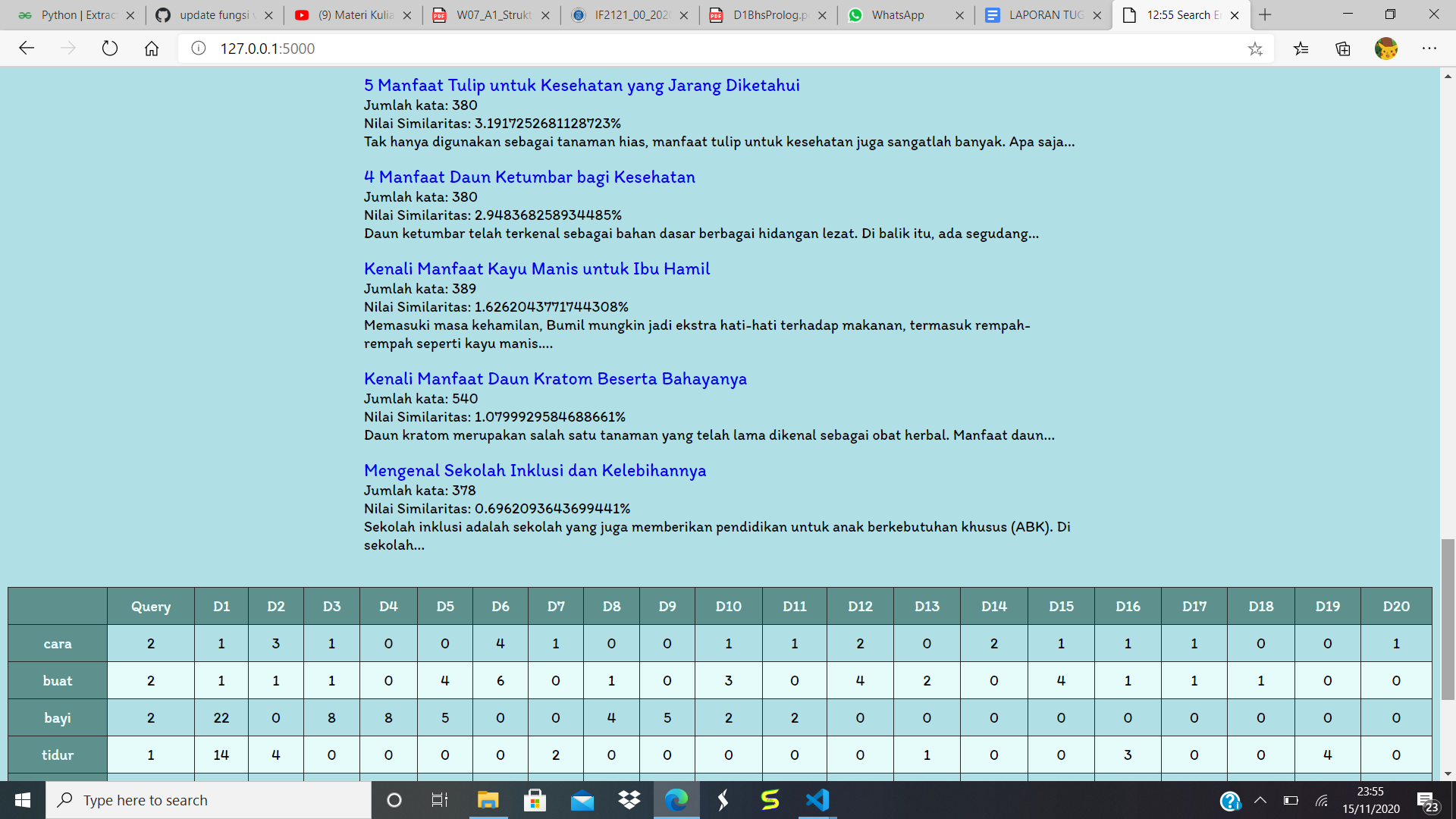
Query = cara membuat bayi tidur pada malam hari dan cara membuat bayi terbangun pada siang hari pada saat hari libur

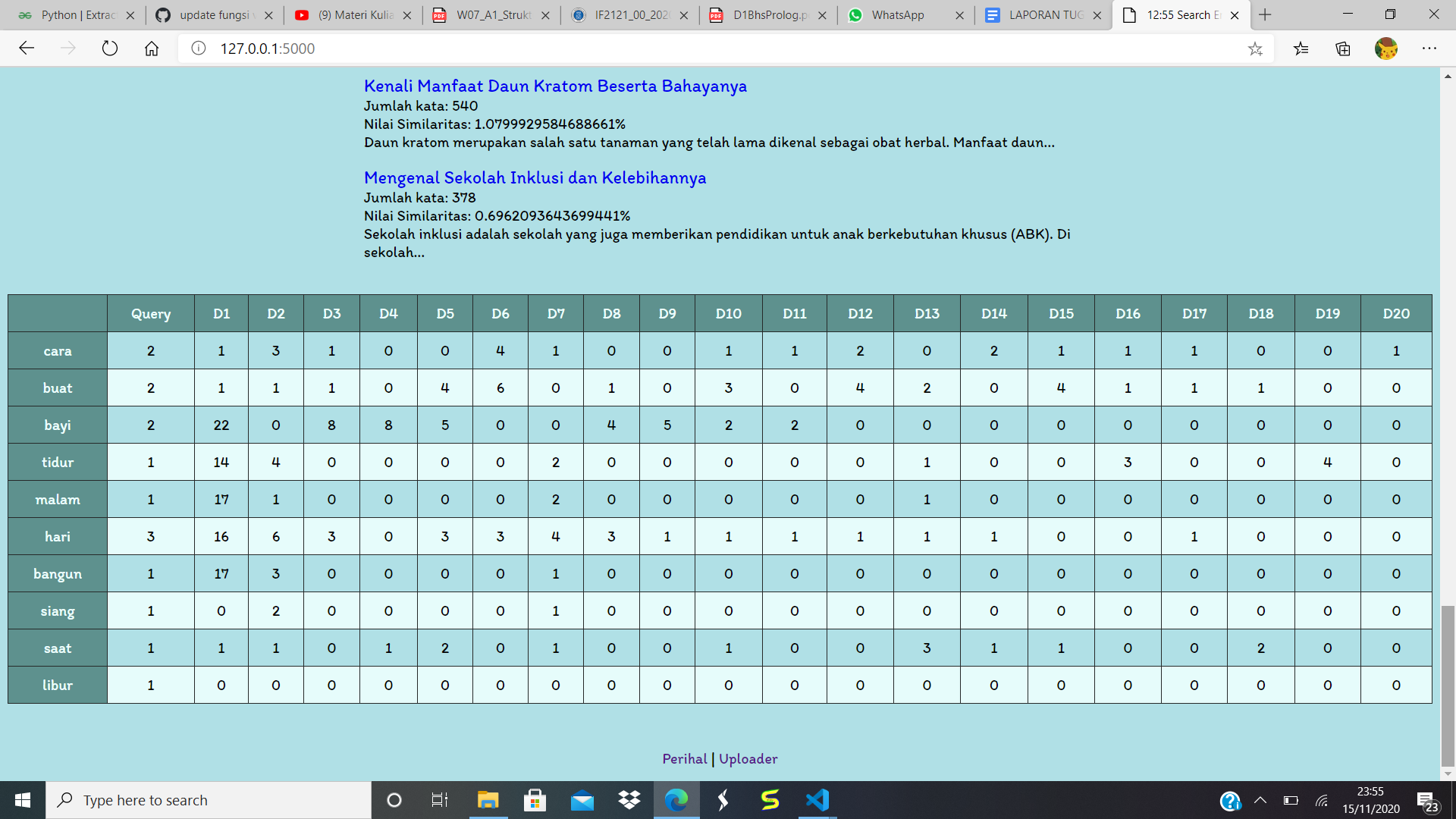
Hasil:









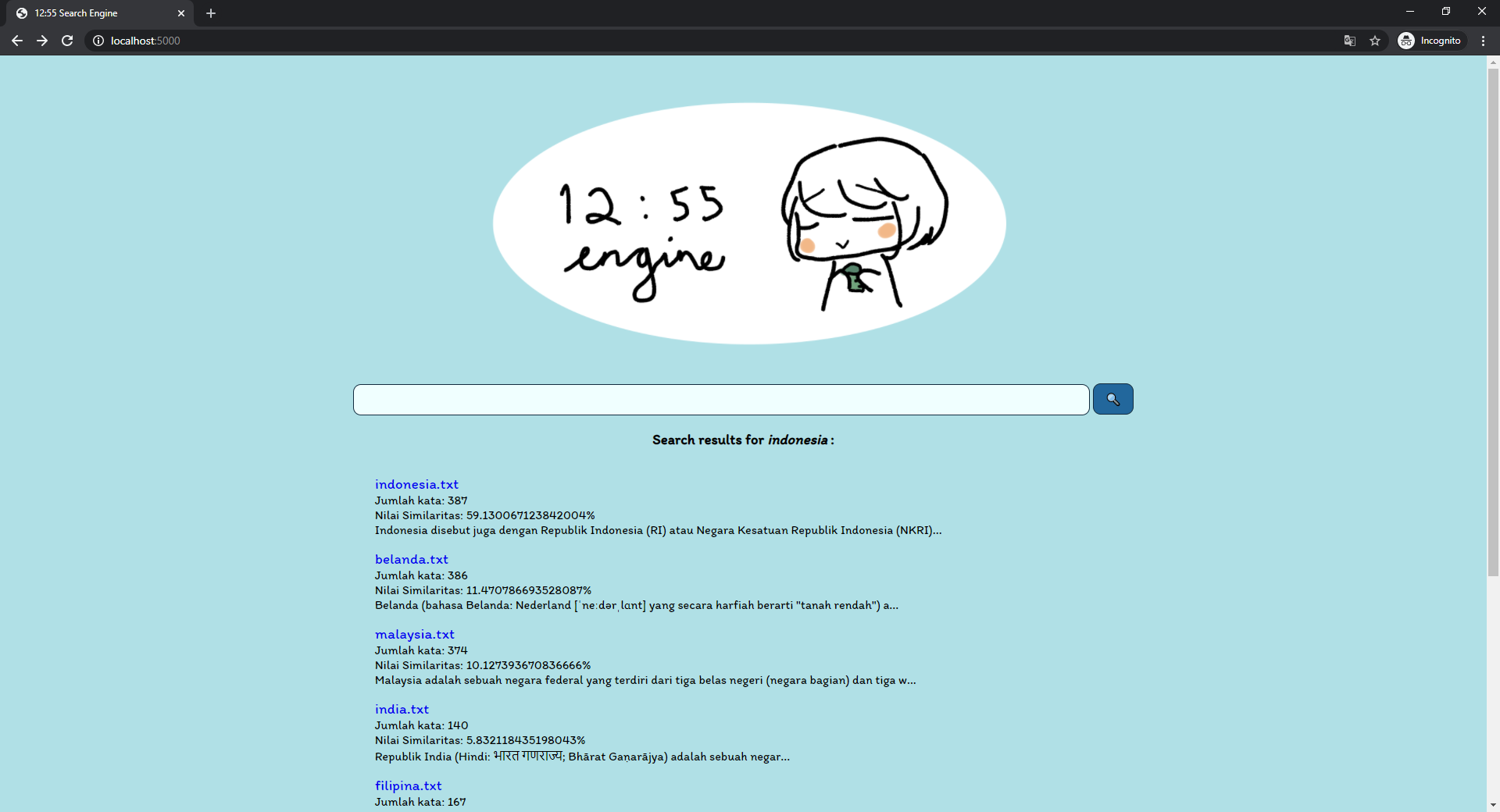


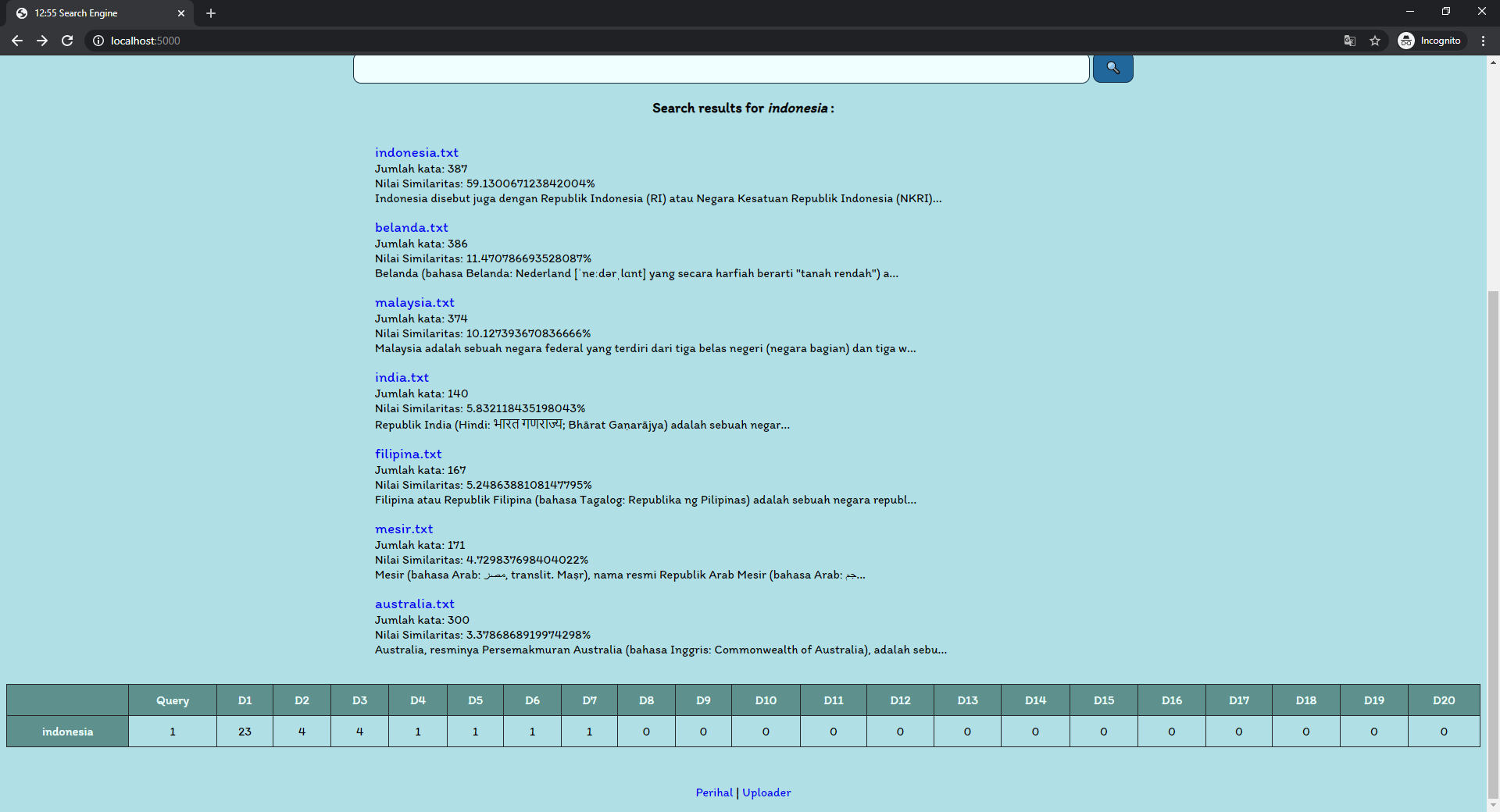
**4.2. Eksperimen dengan Dokumen .txt**

4.2.1. Kasus 1:

Query = indonesia

Hasil:

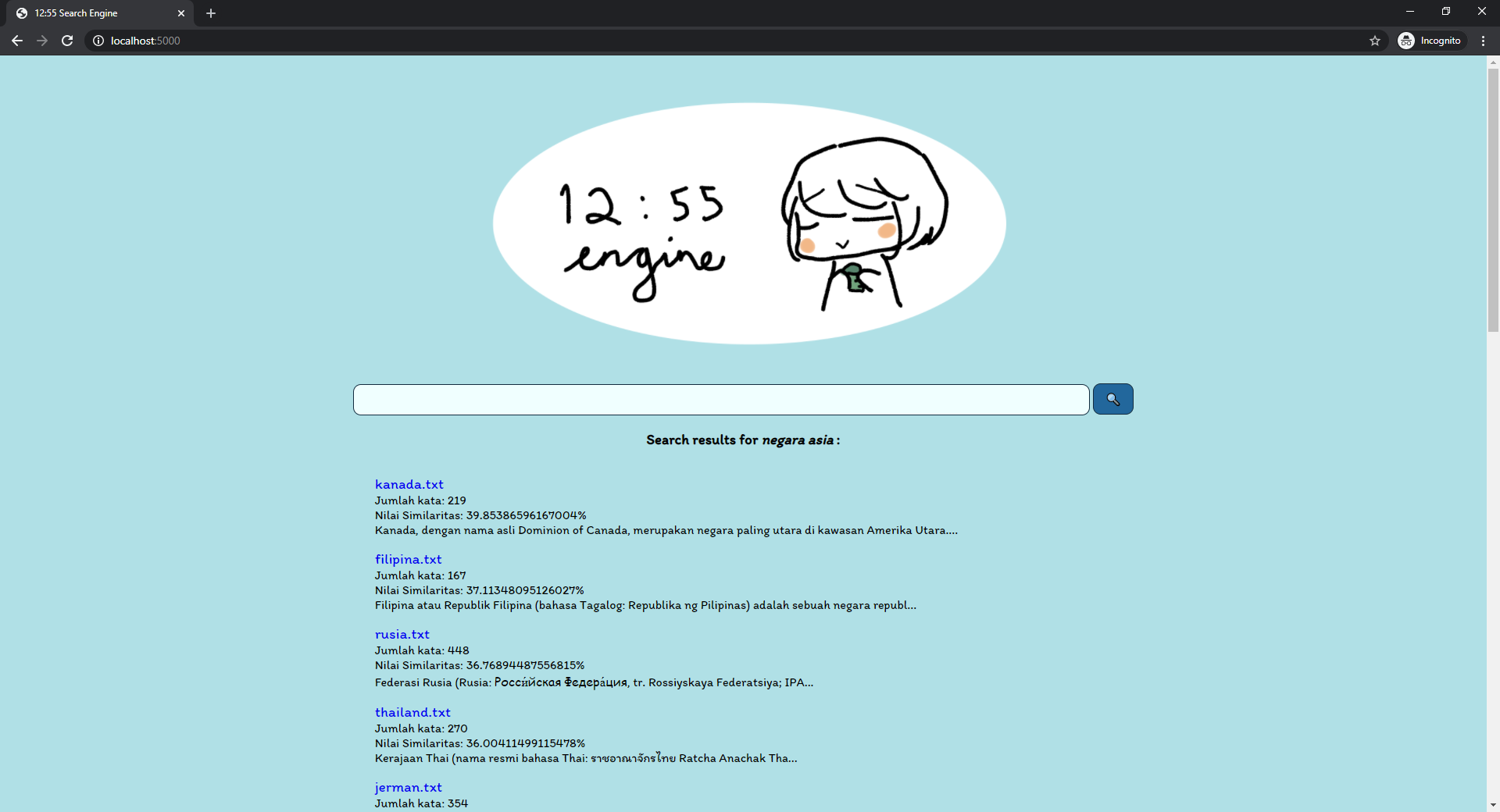


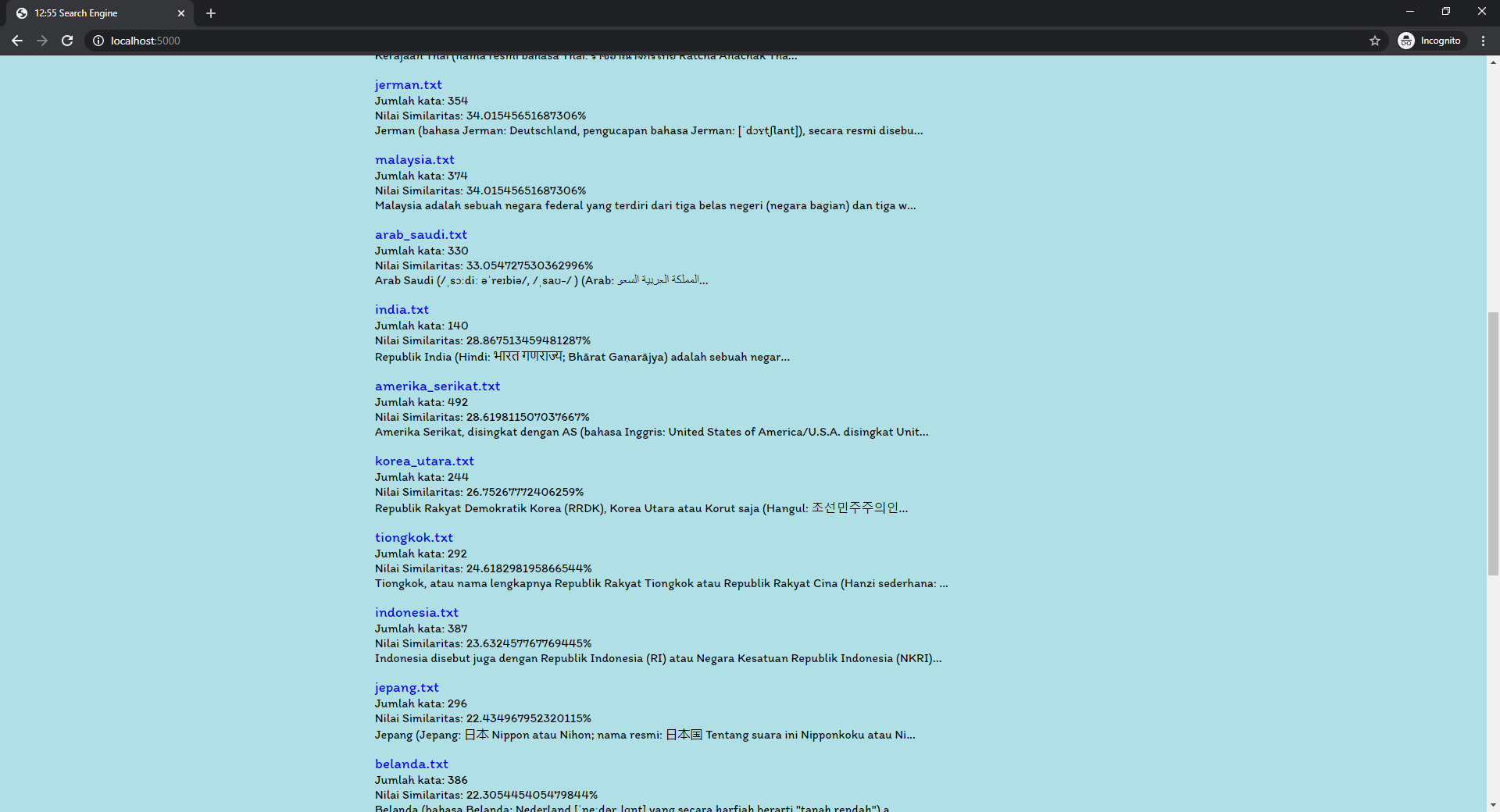


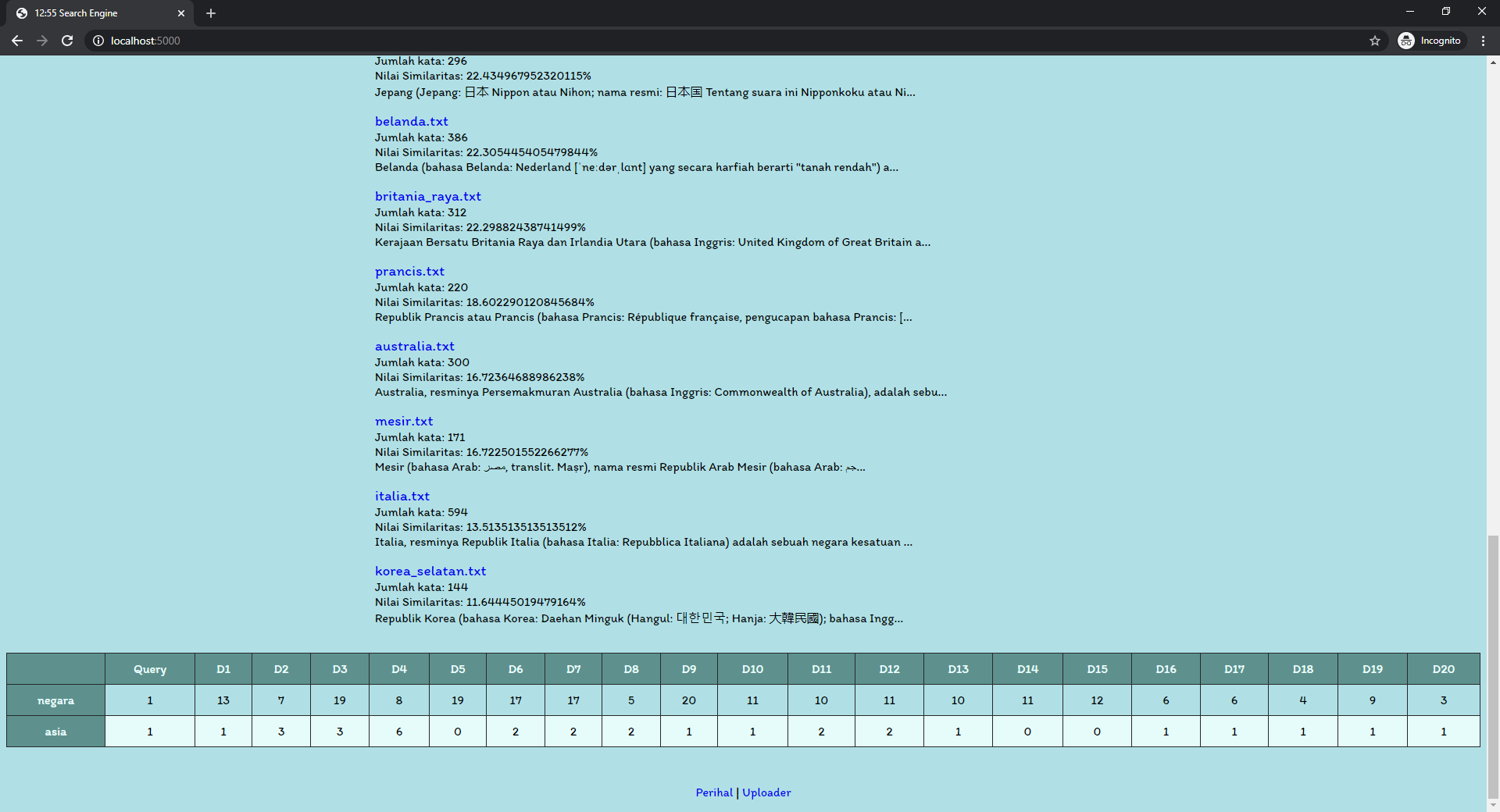
4.2.2. Kasus 2:

Query = negara di asia

Hasil:







# 

# BAB V

**Kesimpulan, Saran, dan Refleksi**

**5.1. Kesimpulan**

*Search engine* yang dibuat dapat mencari informasi menggunakan *cosine similarity,* baik dari *file* .txt ataupun *web scraping* dari situs alodokter. Hasil pencarian diurutkan berdasarkan nilai *similarity* terbesar sampai terkecil. Data yang ditampilkan adalah judul artikel berupa *link* menuju artikel tersebut, nilai *cosine similarity,* jumlah kata, dan deskripsi singkat artikel. Ditampilkan juga tabel yang berisi seluruh kata dalam *query* dan jumlah kata-kata tersebut di masing-masing dokumen.

**5.2. Saran**

Untuk pengembangan di masa depan, dapat digunakan *parser* lain agar proses *web scraping* menjadi lebih cepat. Selain itu, proses *file* dapat dilakukan langsung, sehingga pengguna tidak perlu me-*restart* aplikasi setelah meng-*upload file* baru.

**5.3. Refleksi**

Penanganan kasus *query* kosong sebaiknya tidak dilupakan. Penentuan *path* menggunakan Flask juga harus diperhatikan.

# REFERENSI

Anton, Howard dan Chris Rorres. 2014. *Elementary Linear Algebra 11th Edition.* USA : Wiley

<https://informatikalogi.com/sistem-temu-kembali-informasi/> diakses 14 November 2020 pukul 22.14

<https://image.slidesharecdn.com/tdminformationretrieval-150803041801-lva1-app6891/95/tdm-information-retrieval-13-638.jpg?cb=1438575616> diakses 14 November 2020 pukul 22.19

<https://www.geeksforgeeks.org/what-is-information-retrieval/> diakses 14 November 2020 pukul 22.29

<https://medium.com/dev-genius/get-started-with-multiple-files-upload-using-flask-e8a2f5402e20> diakses 15 November 2020 pukul 17.55

<https://id.wikipedia.org/wiki/Daftar_negara_menurut_jumlah_penduduk> diakses 15 November 2020 pukul 16.48

<http://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2020-2021/Tubes2-Algeo-2020.pdf>

<http://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2020-2021/Algeo-10-Vektor-di-Ruang-Euclidean-Bag1.pdf>

<http://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2020-2021/Algeo-11-Vektor-di-Ruang-Euclidean-Bag2.pdf>

<http://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2020-2021/Algeo-12-Aplikasi-dot-product-pada-IR.pdf>