

Perancangan Arsitektur Search Engine dengan Mengintegrasikan Web Crawler, Algoritma Page Ranking, dan Document Ranking

Lazuardy Khatulistiwa^{1, a)}, Muhammad Eka Suryana^{2, b)}, Mulyono^{3, c)}

¹*Program Studi Ilmu Komputer Fakultas Matematika dan Ilmu Pengetahuan Alam.*

²*Universitas Negeri Jakarta.*

³*Jalan Raya Rawamangun Muka, Jakarta Timur DKI Jakarta 13220.*

Email: ^{a)}lazdevs@gmail.com, ^{b)}eka-suryana@unj.ac.id, ^{c)}mulyono@unj.ac.id

Abstract

The search engine was created in 1990 and continues to grow to this day. Google is a search engine that was built in 1998 and is a search engine the most popular at the moment. The main function of the search engine is as provider of information based on certain keywords. This research aims to create a search engine architecture using web crawlers, page ranking, and document rankings. The algorithm used in the web crawler is breadth first search and modified similarity based crawling, on page ranking is Google PageRank, and on document ranking is TF IDF. The process of developing this system uses the method Scrum and all programs created using the Python programming language. The end result of this architecture includes project structure, diagram design, server configuration, web service in the form of REST API, console program and simple front-end website, as well as complete documentation of the project.

Keywords: search engine, information retrieval, web crawler, page rank, TF IDF.

Abstrak

Search engine diciptakan sejak 1990 dan terus berkembang hingga hari ini. Google adalah *search engine* yang dibangun pada tahun 1998 dan merupakan *search engine* yang paling populer saat ini. Fungsi utama dari *search engine* adalah sebagai penyedia informasi berdasarkan kata kunci tertentu. Penelitian ini bertujuan untuk membuat arsitektur *search engine* dengan menggunakan *web crawler*, *page ranking*, dan *document ranking*. Algoritma yang digunakan pada web crawler adalah *breadth first search* dan *modified similarity based crawling*, pada *page ranking* adalah *Google PageRank*, dan pada document ranking adalah *TF IDF*. Proses pengembangan sistem ini menggunakan metode Scrum dan seluruh program yang dibuat menggunakan bahasa pemrograman Python. Hasil akhir dari arsitektur ini mencakup struktur *project*, rancangan diagram, konfigurasi server, *web service* berupa *REST API*, program *console* dan tampilan web sederhana, serta dokumentasi lengkap project.

Kata-kata kunci: mesin pencari, temu kembali informasi, *web crawler*, *page rank*, *TF IDF*.

PENDAHULUAN

Saat ini internet merupakan suatu kebutuhan bagi masyarakat secara umum yang berimplikasi terhadap jumlah situs yang terdapat pada *World Wide Web*. Berdasarkan survei yang dilakukan NetCraft pada bulan April tahun 2022, terdapat 1 miliar situs yang terdapat di web dengan 202 juta situs yang aktif. Dengan banyaknya situs tersebut, maka pengguna internet memerlukan sebuah mesin pencari.

Mesin pencari atau yang biasa disebut *Search Engine* merupakan sebuah program komputer yang berguna untuk membantu pengguna dalam mencari situs web berdasarkan permintaan pencarian pengguna. Mesin pencari sebenarnya tidak berbeda dengan *website* pada umumnya, hanya saja perannya lebih terfokus pada pengumpulan dan pengorganisasian berbagai informasi di internet sesuai dengan kebutuhan penggunaannya. Selain untuk memudahkan pencarian, mesin pencari juga berguna untuk meningkatkan pengunjung sebuah situs web.

Dalam arsitektur pembuatan *search engine*, terdapat salah satu bagian yang disebut *crawler*. *Crawler* berfungsi untuk mengumpulkan data-data yang akhirnya data-data tersebut akan diproses dan ditampilkan ke pengguna. *Crawler* ini perlu dirancang sebaik mungkin untuk mengumpulkan data yang dapat mendukung *search engine*. *Web crawler* pertama kali dimunculkan pada tahun 1944 bernama RBSE dengan dasar dua program yaitu *Spider* dan *Mite* (Eichmann, 1994). Dua program dasar ini berguna untuk membentuk antrean data dalam database serta untuk mengunduh halaman dari sebuah web. Guna dari *web crawler* sendiri adalah sebagai 3 perangkat lunak yang secara otomatis bekerja untuk menjelajahi website dengan cara terorganisir.

Dalam jurnal tahun 2020 oleh Kaur dan Geetha, mereka membuat sebuah *web crawler* untuk *hidden web* menggunakan *SIM+HASH* disertai dengan *Redis Server* (Kaur dan Geetha, 2020). *Web crawler* ini mampu menghasilkan hasil halaman dengan tingkat kecocokan yang maksimum. Meski hasil yang ditampilkan sangat relevan dengan *keyword* yang dimasukkan, namun dalam pembangunannya, *web crawler* ini sangatlah rumit meski komponennya hampir sama dengan *web crawler* pada umumnya.

Penelitian yang dilakukan Muhammad Fathan menghasilkan *web crawler* yang dapat berjalan dengan baik di situs yang menggunakan html versi 4 maupun html versi 5. Algoritma *crawler* yang digunakan pada penelitian tersebut adalah algoritma *breadth first search crawling* dan algoritma *modified similarity based crawling* yang dikembangkan oleh Google (Qoriiba, 2021).

Dalam menghasilkan informasi berdasarkan *keyword* yang dimasukkan pengguna pada *search engine*, diperlukan sebuah sistem temu kembali informasi atau yang biasa disebut dengan *Information Retrieval*. Temu kembali informasi ini dimaksudkan untuk menemukan material atau data-data (sekumpulan dokumen) pada penyimpanan yang tak beraturan (biasanya berbentuk teks) di mana menjadi kebutuhan akan sebuah informasi dari kumpulan data berukuran besar (dalam 4 penyimpanan komputer) (Manning dkk., 2009). *Information retrieval* mampu menemukan informasi yang tersimpan dalam sistem sesuai dengan *keyword* tertentu yang dimasukkan pengguna. Teknologi ini mencakup 2 hal, yaitu pembobotan dan perangkingan. Peringkat tertinggi akan diisi oleh data yang paling relevan dengan *keyword* masukan pengguna dan akan terus diurutkan berdasarkan relevansinya.

Google PageRank adalah salah satu algoritma perangkingan situs (*page ranking*) yang berfungsi untuk menentukan situs web mana yang lebih populer. Algoritma ini diperkenalkan oleh Sergey Brin dan Larry Page pada tahun 1998 dalam jurnalnya yang berjudul *The Anatomy of a Large-Scale Hypertextual Web Search Engine*. Konsep dari perangkingan *PageRank* adalah semakin banyak situs lain yang memasang link ke situsnya, maka situs tersebut akan semakin populer, dengan anggapan bahwa konten situs tersebut lebih bermanfaat daripada konten situs lain.

Dalam pengembangan *search engine*, selain perangkingan situs web diperlukan sebuah metode *document ranking*, yaitu untuk memverifikasi apakah konten dalam situs web tersebut relevan atau tidak. Salah satu metode yang paling umum digunakan adalah metode *TF-IDF*. *TF-IDF* atau *Term Frequency-Inversed Document Frequency* adalah suatu metode untuk memberikan bobot kata yang berhubungan terhadap suatu dokumen. Metode ini menggunakan algoritma yang menggabungkan 2 nilai yaitu *Term Frequency (TF)* yang merupakan nilai frekuensi kemunculan kata dan *Inverse Document Frequency (IDF)* yang merupakan nilai untuk mengukur seberapa penting sebuah kata. Dengan menggunakan metode ini pencarian pada *search engine* akan memiliki hasil yang relevan sesuai dengan *keyword* masukan pengguna.

Dari penelitian yang dilakukan oleh Juan Ramos yang berjudul "*Using 5 TF-IDF to Determine Word Relevance in Document Queries*", terdapat kesimpulan bahwa *TF-IDF* merupakan algoritma sederhana dan efisien yang dapat menghasilkan kumpulan dokumen dengan tingkat kesesuaian yang tinggi sesuai dengan *keyword* yang dimasukkan.

Penelitian ini akan merancang dan membangun *search engine* berdasarkan arsitekturnya dengan mengintegrasikan penelitian dari Muhammad Fathan yang berfokus pada perancangan *crawler* menggunakan algoritma *modified similarity based crawling*, penelitian Mohammad Riza yang

mengimplementasikan algoritma perankingan *Google PageRank*, dan penelitian Savira Rahmayanti yang mengembangkan sistem pencarian teks menggunakan metode *TF-IDF*.

KAJIAN PUSTAKA

A. Search Engine

Search engine adalah sebuah mesin atau program yang mampu menampilkan banyak informasi yang relevan terkait dengan *keyword* masukan pengguna. Sistem ini adalah sebuah sistem yang mengindeks, mengumpulkan segala informasi yang tersimpan dalam internet, menyaringnya berdasarkan *keyword* input, dan menampilkan halaman *website* terkait dengan perankingan guna memudahkan dalam mendapatkan informasi. Secara singkat, *search engine* juga dapat disebut sebagai sebuah navigasi atau penunjuk dalam melakukan surfing dalam dunia internet. Dengan memanfaatkan *search engine*, kita dengan mudah akan mendapatkan informasi yang diperlukan dengan rekomendasi halaman yang paling terkenal berada dalam ranking paling atas.

B. Web Crawler

Web crawler adalah program yang mengambil halaman web, biasanya untuk digunakan oleh mesin pencari (Pinkerton, 1994). *Web crawler* akan mengindeks dan mengunduh semua konten dari internet, lalu menyimpannya ke dalam *database*. *Web crawler* juga biasa dikenal sebagai *web spider*, *web bot*, dan *bot crawler*. Secara kasar, *crawler* dimulai dari *URL* halaman awal P_0 . Kemudian *crawler* merayap ke P_0 , mengekstrak semua *URL* yang terdapat di dalam P_0 , dan menambahkannya ke antrean *URL* untuk dipindai. Setelah itu, *crawler* mengambil *URL* dari antrean (dengan urutan tertentu), dan mengulangi prosesnya (Cho dkk., 1998).

C. Pembobotan Term Frequency-Inverse Document Frequency

TF-IDF adalah metode pemberian bobot pada hubungan antara kata (*word*) dan dokumen. *TF-IDF* merupakan ukuran statistik yang digunakan untuk mengevaluasi pentingnya sebuah kata dalam dokumen atau kumpulan kata. Berikut rumus dari *TF-IDF* (Manning dkk., 2009).:

$$tfidf_{t,d} = \frac{\text{Jumlah kata } t \text{ di dokumen } d}{\text{Jumlah kata di dokumen } d} \times \log \left(\frac{\text{Jumlah dokumen}}{\text{Jumlah dokumen yang terdapat kata } t} \right)$$

C. Google PageRank

Google PageRank menganalisis jumlah tautan masuk ke setiap halaman di web, dan menetapkan setiap halaman sebuah peringkat yang ditentukan oleh peringkat setiap halaman yang menautkannya. Berikut rumus dari *PageRank*:

$$PR(u) = (1 - d) + d \sum_{v \in B(u)} \frac{PR(v)}{N_v}$$

Di mana u merepresentasikan halaman web. $B(u)$ adalah kumpulan halaman yang merujuk ke u . $PR(u)$ dan $PR(v)$ masing-masing merupakan skor peringkat halaman u dan v . N_v menunjukkan jumlah tautan keluar dari halaman v . Sedangkan d merupakan *damping factor* yang biasanya bernilai 0.85 (Page dkk., 1999).

C. Cosine Similarity

Metode *cosine similarity* merupakan metode yang digunakan untuk menghitung tingkat kesamaan (similarity) antar dua buah objek. Dalam *search engine*, metode ini dapat digunakan untuk sebagai ukuran kemiripan antara kueri dari pengguna dengan masing-masing dokumen yang tersimpan. Setelah itu, skor kemiripan tersebut dapat ditambahkan dengan skor *PageRank* sehingga mendapatkan hasil yang lebih relevan (Roul dkk., 2019). Nilai *cosine similarity* dapat didefinisikan sebagai berikut:

$$Sim_{A,B} = \frac{AxB}{|A||B|} = \frac{\sum_{i=1}^n A_i x B_i}{\sqrt{\sum_{i=1}^n A_i^2} \times \sqrt{\sum_{i=1}^n B_i^2}}$$

Pada persamaan di atas, AxB merupakan *dot product* antara dokumen A dan B, sedangkan $|A||B|$ merupakan perkalian panjang vektor antara dokumen A dan B. Dot product dilakukan dengan melakukan perkalian biasa antara nilai *TF IDF* dokumen A dengan dokumen B untuk setiap dimensi. Sedangkan panjang vektor A dan B, dapat diketahui dengan cara menghitung akar kuadrat dari penjumlahan pangkat dua masing-masing nilai *TF IDF* pada setiap dokumen A dan B (Firdaus dkk., 2019).

C. Metode Scrum

Metode Scrum diciptakan oleh Jeff Sutherland dan tim pengembangnya pada awal 1990-an. Sutherland, Viktorov, Blount dan Puntikov menggambarkan Scrum sebagai “Proses pengembangan perangkat lunak Agile yang dirancang untuk menambah energi, fokus, kejelasan, dan transparansi bagi tim proyek yang mengembangkan sistem perangkat lunak” (Sutherland dkk., 2007). Proses Scrum mematuhi prinsip-prinsip pendekatan Agile.

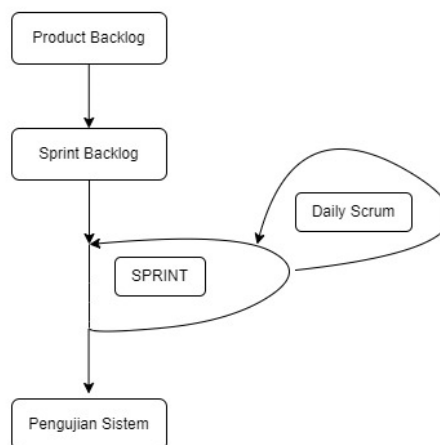
Prinsip pendekatan Agile berfokus pada memuaskan pelanggan melalui pengiriman awal perangkat lunak yang berharga; membolehkan perubahan kebutuhan; kolaborasi antara pengembang dan pelaku bisnis; perangkat lunak yang berfungsi (sebagai ukuran kemajuan); menjaga desain tetap sederhana; dan secara berkala, membuat tim merenungkan bagaimana menjadi lebih efektif selama proses pengembangan.

Scrum menawarkan cara yang dapat disesuaikan untuk bekerja pada proyek yang berbeda yang memiliki berbagai persyaratan dan Scrum memiliki keunggulan seperti pemilihan persyaratan atau spesifikasi kebutuhan yang fleksibel untuk sprint dan tidak ada prosedur khusus yang harus diikuti. Prinsipnya adalah bekerja secara iteratif hingga mencapai waktu yang ditentukan dan dapat memenuhi kebutuhan konsumen.

METODE PENELITIAN

A. Desain Penelitian

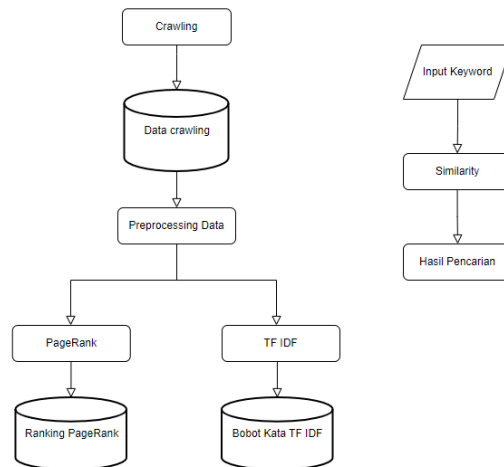
Desain penelitian mencakup tahapan-tahapan yang dilakukan penulis dalam pembuatan sistem dengan menggunakan metode Scrum. Tahapan tersebut dapat dilihat pada gambar 1.



GAMBAR 1. Desain Penelitian

Tahap pertama yang dilakukan pada penelitian ini adalah mendefinisikan *product backlog* dari sistem yang akan dibuat, lalu membuat jadwal pengerjaan atau *sprint backlog*. Setelah itu, sprint dimulai dengan diselingi *daily scrum* untuk melaporkan *progress* dan menentukan task selanjutnya. Setelah sistem selesai dikerjakan di dalam *sprint*, maka dilakukan pengujian sistem.

B. Flowchart Sistem



GAMBAR 2. Flowchart Sistem

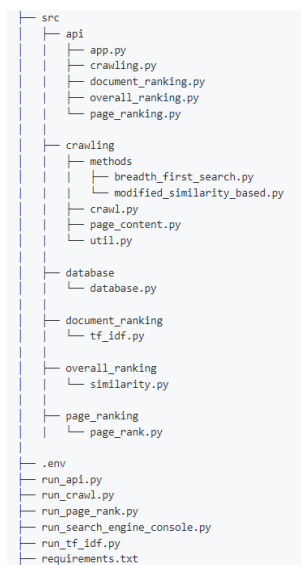
Proses dari sistem *search engine* dimulai dengan mengumpulkan data ke dalam *database* sebanyak-banyaknya menggunakan *crawler* dan data tersebut akan diolah melalui proses *preprocessing* data, *PageRank*, dan *TF IDF*. Setelah data sudah siap, maka sistem dimulai dengan meminta *input* berupa *keyword* dari pengguna lalu *keyword* tersebut akan diproses melalui proses *similarity*, yaitu proses mengambil website yang relevan dengan cara menghitung *cosine similarity* berdasarkan bobot *TF IDF* yang ada lalu dijumlahkan dengan nilai *PageRank*. Setelah proses *similarity* selesai, hasil pencarian akan muncul sesuai dengan *keyword* yang dimasukkan.

C. Alat Penelitian

Dalam penelitian ini, alat berupa perangkat keras yang digunakan untuk menunjang pembuatan sistem adalah laptop yang mempunyai *CPU Intel i7-6700hq* dengan *RAM 20 GB* dan koneksi berbasis Wi-Fi. Selain itu, arsitektur ini di-deploy pada sebuah *Virtual Private Server (VPS)* yang disewa di *niagahoster.co.id*.

HASIL DAN PEMBAHASAN

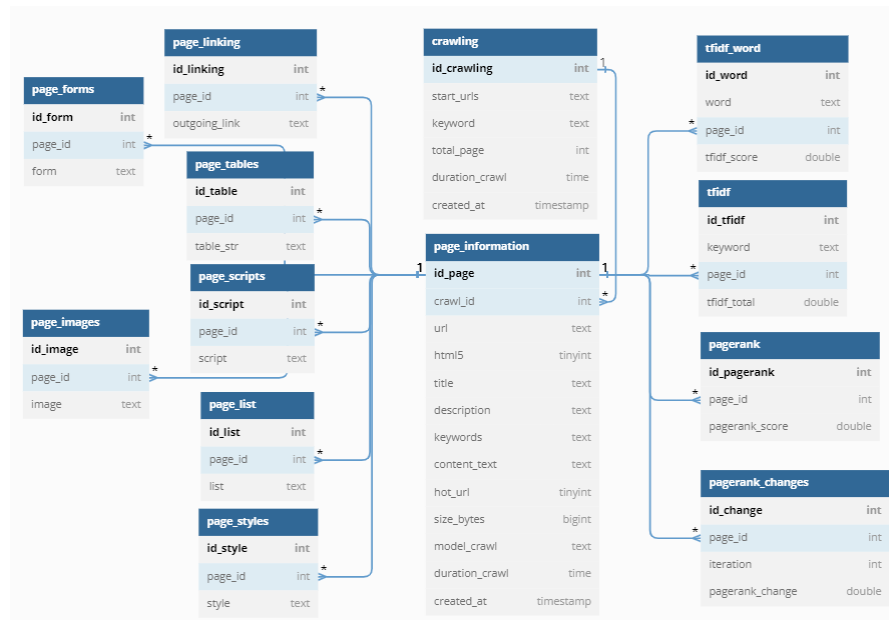
A. Struktur Direktori dan File



GAMBAR 3. Struktur Direktori dan File

Spesifikasi struktur direktori dan *file search engine* dibagi menjadi beberapa *package* atau folder sesuai dengan komponennya masing-masing. Pada direktori terluar atau *root project* dikhususkan pada *file* konfigurasi *search engine* dan *script* yang nantinya dapat dijalankan secara langsung. Sedangkan algoritma atau inti kode dari tiap komponennya terdapat di dalam direktori *src*

B. Desain Database

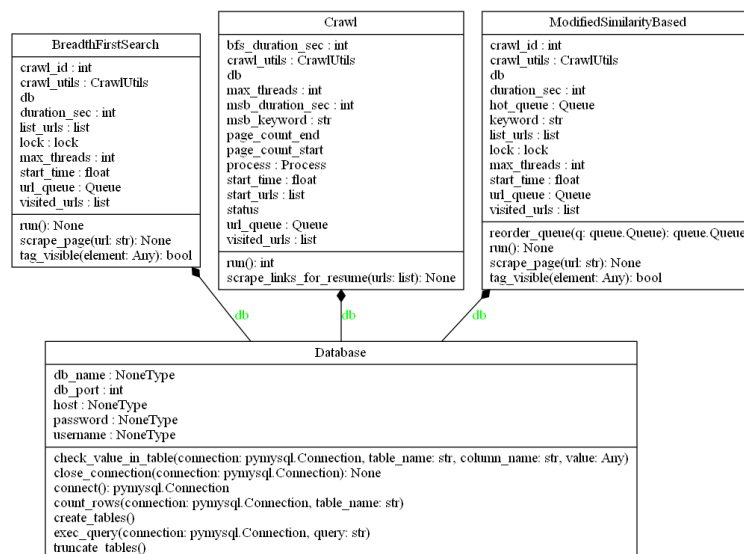


GAMBAR 4. Desain Database

Gambar 4 merupakan hasil dari desain *database* untuk *search engine*. Tabel yang diperlukan pada proses crawling adalah *page_form*, *page_images*, *page_linking*, *page_tables*, *page_scripts*, *page_list*, *page_styles*, *page_information*, dan *crawling*. Tabel yang diperlukan pada proses *TF IDF* adalah *tfidf_word* dan *tfidf*. Sedangkan tabel yang diperlukan pada proses *PageRank* adalah *pagerank* dan *pagerank_changes*.

C. Class Diagram

Kelas-kelas pada program yang didesain pada *search engine* ini terdapat pada gambar 5.



GAMBAR 5. Class Diagram

D. Routing Table REST API

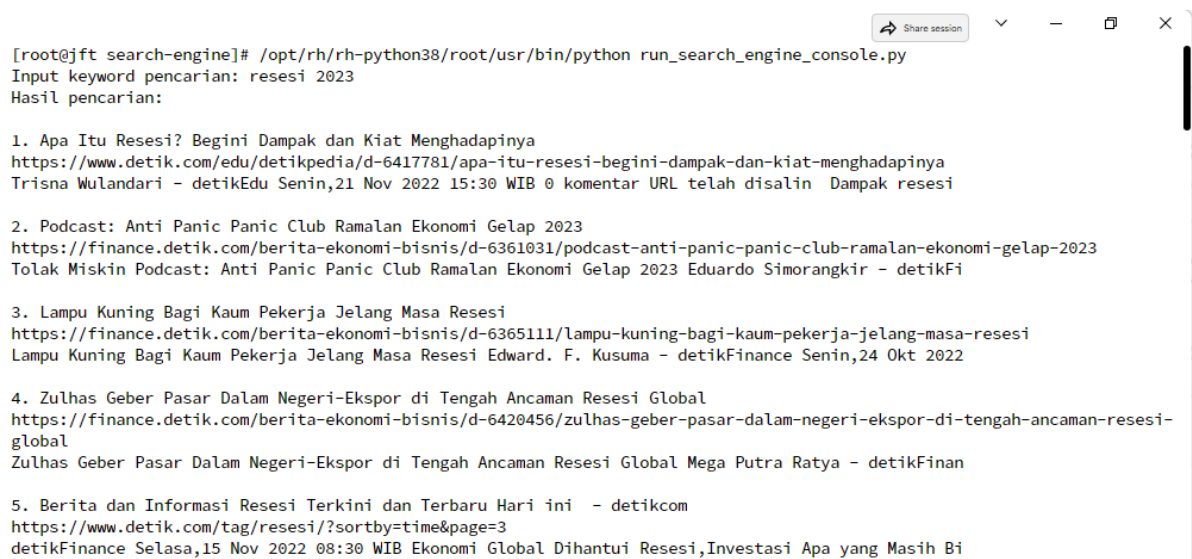
Daftar URL yang tersedia pada *backend search engine* beserta dengan deskripsinya terdapat pada tabel 1.

TABEL1. Routing Table REST API

Route	Method	Description
/api/v1.0/crawling/crawl	GET	Untuk menjalankan proses crawling
/api/v1.0/crawling/stop	GET	Untuk menghentikan proses crawling
/api/v1.0/document_ranking/tf_idf	GET	Untuk mendapatkan ranking halaman menggunakan metode TF IDF
/api/v1.0/page_ranking/page_rank	GET	Untuk mendapatkan ranking halaman menggunakan metode Page Rank
/api/v1.0/overall_ranking/similarity	GET	Untuk mendapatkan ranking halaman secara overall dari pengintegrasian TF IDF dan Page Rank
/api/v1.0/crawling/pages	GET	Untuk mendapatkan data halaman yang sudah dicrawling
/api/v1.0/crawling/page_information	POST	Untuk mendapatkan informasi lengkap halaman dari page id
/api/v1.0/crawling/start_insert	POST	Untuk mendapatkan crawl id yang nantinya digunakan untuk memasukkan data halaman ke database crawling
/api/v1.0/crawling/insert_page	POST	Untuk memasukkan data halaman ke database crawling

E. Program Console

Program *console search engine* dibuat seperti pada gambar 7, kegunaan program ini dapat menerima input sebuah keyword dan menampilkan hasil pencarian yang terdiri dari URL, judul, dan deskripsi singkat dari URL tersebut.



```
[root@jft search-engine]# /opt/rh/rh-python38/root/usr/bin/python run_search_engine_console.py
Input keyword pencarian: resesi 2023
Hasil pencarian:

1. Apa Itu Resesi? Begini Dampak dan Kiat Menghadapinya
https://www.detik.com/edu/detikpedia/d-6417781/apa-itu-resesi-begini-dampak-dan-kiat-menghadapinya
Trisna Wulandari - detikEdu Senin,21 Nov 2022 15:30 WIB 0 komentar URL telah disalin Dampak resesi

2. Podcast: Anti Panic Panic Club Ramalan Ekonomi Gelap 2023
https://finance.detik.com/berita-ekonomi-bisnis/d-6361031/podcast-anti-panic-panic-club-ramalan-ekonomi-gelap-2023
Tolak Miskin Podcast: Anti Panic Panic Club Ramalan Ekonomi Gelap 2023 Eduardo Simorangkir - detikFi

3. Lampu Kuning Bagi Kaum Pekerja Jelang Masa Resesi
https://finance.detik.com/berita-ekonomi-bisnis/d-6365111/lampu-kuning-bagi-kaum-pekerja-jelang-masa-resesi
Lampu Kuning Bagi Kaum Pekerja Jelang Masa Resesi Edward. F. Kusuma - detikFinance Senin,24 Okt 2022

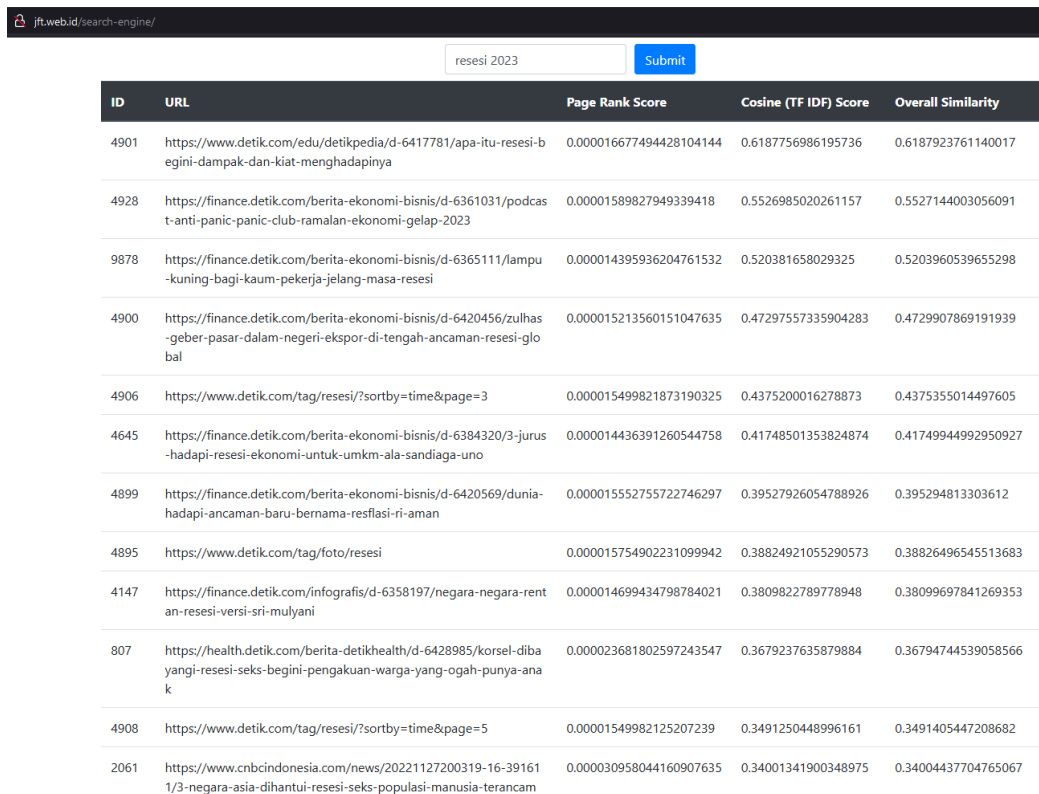
4. Zulhas Geber Pasar Dalam Negeri-Ekspor di Tengah Ancaman Resesi Global
https://finance.detik.com/berita-ekonomi-bisnis/d-6420456/zulhas-geber-pasar-dalam-negeri-ekspor-di-tengah-ancaman-resesi-global
Zulhas Geber Pasar Dalam Negeri-Ekspor di Tengah Ancaman Resesi Global Mega Putra Ratya - detikFinan

5. Berita dan Informasi Resesi Terkini dan Terbaru Hari ini - detikcom
https://www.detik.com/tag/resesi/?sortby=time&page=3
detikFinance Selasa,15 Nov 2022 08:30 WIB Ekonomi Global Dihantui Resesi,Investasi Apa yang Masih Bi
```

GAMBAR 7. Program Console

F. Tampilan Web

Tampilan web dibuat seperti pada gambar 6 yang menerima input berupa *keyword* dan menampilkan hasil relevansi beserta dengan skornya masing-masing.



ID	URL	Page Rank Score	Cosine (TF IDF) Score	Overall Similarity
4901	https://www.detik.com/edu/detikpedia/d-6417781/apa-itu-resesi-begini-dampak-dan-kiat-menghadapinya	0.000016677494428104144	0.6187756986195736	0.6187923761140017
4928	https://finance.detik.com/berita-ekonomi-bisnis/d-6361031/podcasts-anti-panic-panic-club-ramalan-ekonomi-gelap-2023	0.00001589827949339418	0.5526985020261157	0.5527144003056091
9878	https://finance.detik.com/berita-ekonomi-bisnis/d-6365111/lampu-kuning-bagi-kaum-pekerja-jelang-masa-resesi	0.000014395936204761532	0.520381658029325	0.5203960539655298
4900	https://finance.detik.com/berita-ekonomi-bisnis/d-6420456/zulhas-geber-pasar-dalam-negeri-ekspor-di-tengah-ancaman-resesi-global	0.000015213560151047635	0.47297557335904283	0.4729907869191939
4906	https://www.detik.com/tag/resesi/?sortby=time&page=3	0.000015499821873190325	0.4375200016278873	0.4375355014497605
4645	https://finance.detik.com/berita-ekonomi-bisnis/d-6384320/3-jurus-hadapi-resesi-ekonomi-untuk-umkm-ala-sandiaga-uno	0.000014436391260544758	0.41748501353824874	0.41749944992950927
4899	https://finance.detik.com/berita-ekonomi-bisnis/d-6420569/dunia-hadapi-ancaman-baru-bernama-resesi-ri-aman	0.000015552755722746297	0.39527926054788926	0.395294813303612
4895	https://www.detik.com/tag/foto/resesi	0.000015754902231099942	0.38824921055290573	0.38826496545513683
4147	https://finance.detik.com/infografis/d-6358197/negara-negara-rentan-resesi-versi-sri-mulyani	0.000014699434798784021	0.3809822789778948	0.38099697841269353
807	https://health.detik.com/berita-detikhealth/d-6428985/korsel-dibayangi-resesi-seks-begitu-pengakuan-warga-yang-ogah-punya-anak	0.000023681802597243547	0.3679237635879884	0.36794744539058566
4908	https://www.detik.com/tag/resesi/?sortby=time&page=5	0.00001549982125207239	0.3491250448996161	0.3491405447208682
2061	https://www.cnbciindonesia.com/news/20221127200319-16-391611/3-negara-asia-dihantui-resesi-seks-populasi-manusia-terancam	0.000030958044160907635	0.34001341900348975	0.34004437704765067

GAMBAR 6. Tampilan Web

G. Source code dan dokumentasi lengkap

Source code keseluruhan arsitektur *search engine* dan dokumentasi lengkap beserta cara penggunaannya terdapat di Github, yaitu pada link <https://github.com/lazuardyk/search-engine>

H. Pengujian Sistem

Pengujian arsitektur *search engine* dilakukan menggunakan teknik *functional* dan *non-functional testing*. *Functional testing* dilakukan bersama dengan *Scrum Master*.

1. Functional Testing

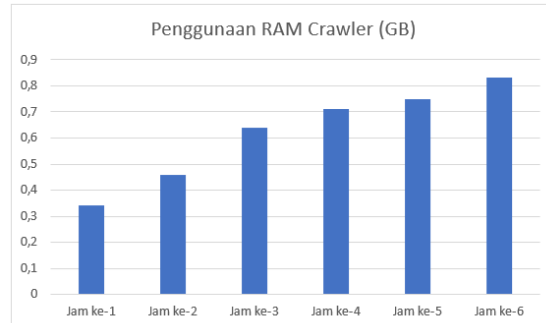
TABEL 2. Functional Testing

Skenario Pengujian	Yang Diharapkan	Pengamatan	Kesimpulan
Menjalankan program crawler, page rank dan TF IDF di lokal komputer dan melihat hasilnya di database	Data tersimpan dengan baik di database	Data yang dihasilkan di database sudah benar dan sesuai	Diterima
Melakukan start dan stop pada setiap background service di server	Start dan stop pada background service di server dapat bekerja dengan baik	Start dan stop pada background service bekerja dengan baik	Diterima
Menjalankan semua routes yang ada pada REST API dengan Postman	Kode dan isi response tiap endpoint benar dan sesuai fungsinya	Tiap response pada endpoint API sudah benar	Diterima
Melakukan pencarian pada tampilan web dan program console search engine	Mempunyai hasil pencarian yang relevan	Hasil pencarian sudah relevan sesuai keyword	Diterima

2. Non-functional Testing

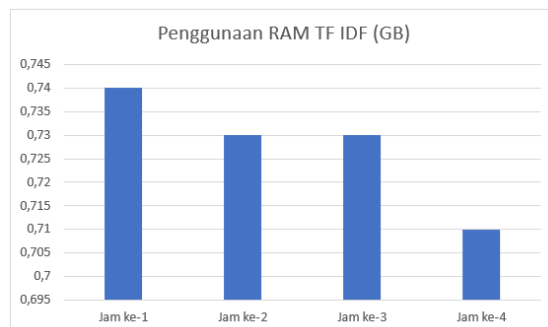
- Lama waktu dan penggunaan RAM

Saat menjalankan *background service crawler* di *server* situs awal yang dipilih adalah detik.com, lalu durasinya diatur pada 6 jam dan dengan jumlah 1 thread. Grafik penggunaan RAM tiap jam saat *crawler* berjalan dapat dilihat pada gambar 7.



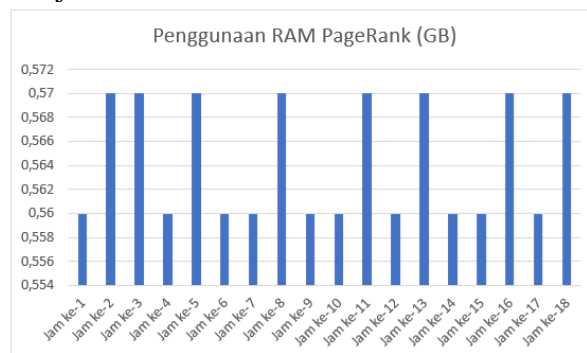
GAMBAR 7. Penggunaan RAM Crawler

Berdasarkan grafik, penggunaan RAM pada *crawler* bertambah seiring berjalannya waktu, hal ini dikarenakan terdapat antrean halaman yang ingin di-*crawl*. Saat *crawler* memproses setiap halaman dan mengurangi antrean, *crawler* juga menambahkan semua URL yang ada pada halaman tersebut ke dalam antrean, sehingga antrean halaman akan lebih banyak bertambah dibandingkan berkurang yang mengakibatkan 82 bertambahnya konsumsi RAM. Hasil halaman yang telah di-*crawl* dari proses *background service* ini berjumlah 10.714 halaman.



GAMBAR 8. Penggunaan RAM TF IDF

Setelah proses *crawler* selesai, *background service TF IDF* dijalankan untuk memproses data halaman yang dihasilkan dari *crawler* tersebut. Grafik penggunaan RAM tiap jam saat *TF IDF* berjalan dapat dilihat pada gambar 8 di mana penggunaan RAM-nya cenderung menurun. *Background service TF IDF* yang memproses 10.714 halaman ini memakan waktu selama 4 jam 23 menit.



GAMBAR 9. Penggunaan RAM PageRank

Background service PageRank juga dijalankan memakai data halaman dari proses *crawler* yang telah dilakukan sebelumnya. Grafik penggunaan *RAM* tiap jam saat *PageRank* berjalan dapat dilihat pada gambar 9 di mana penggunaan *RAM*-nya cenderung naik turun tetapi tetap stabil. Proses *PageRank* berhenti atau konvergen pada iterasi ke-8 di mana rata-rata perbedaan nilai *PageRank* di iterasi terakhir bernilai 0.0000636. *Background service PageRank* yang memproses 10.714 halaman ini memakan waktu selama 18 jam 3 menit.

- *Response Time API*

Pengujian *REST API* dilakukan dengan *membandingkan response time tiap endpoint API* pada saat *background service* berjalan dan pada saat *background service* tidak berjalan. Hasil dari perbandingan ini dapat dilihat pada tabel 3.

TABEL 3. *Response Time API*

Route	Response Time	
	Berjalan	Tidak Berjalan
/api/v1.0/crawling/crawl	659 ms	302 ms
/api/v1.0/document_ranking/tf_idf	323 ms	304 ms
/api/v1.0/page_ranking/page_rank	534 ms	292 ms
/api/v1.0/overall_ranking/similarity	77 ms	64 ms
/api/v1.0/crawling/pages	92 ms	82 ms
/api/v1.0/crawling/page_information	62 ms	60 ms
/api/v1.0/crawling/start_insert	63 ms	62 ms
/api/v1.0/crawling/insert_page	78 ms	76 ms

- Keandalan *Server*

Keandalan *server* diuji dengan cara melakukan sejumlah request yang telah ditentukan ke salah satu route *API*. Hasil dari pengujian ini terdapat pada tabel 4.

TABEL 4. Uji Keandalan *Server*

Jumlah Request	Rata-rata Response Time
100	581 ms
1000	796 ms
10000	1826 ms

Berdasarkan hasil uji keandalan *server* pada tabel 4.14, dapat disimpulkan bahwa perbedaan *response time* yang signifikan dapat dialami ketika melakukan *request* atau melakukan kunjungan ke *server* selama 10000 kali.

KESIMPULAN DAN SARAN

A. Kesimpulan

Berdasarkan hasil implementasi dan pengujian arsitektur *search engine* yang telah dirancang, maka diperoleh kesimpulan sebagai berikut:

1. Perancangan arsitektur *search engine* yang mengintegrasikan *web crawler*, algoritma *page ranking*, dan *document ranking* dirancang menggunakan metode pengembangan Scrum.
2. Arsitektur *search engine* yang dibuat mencakup struktur *project*, rancangan diagram, konfigurasi *server*, *web service* berupa *REST API*, program *console* dan tampilan web sederhana, serta dokumentasi lengkap *project*.
3. Berdasarkan hasil pengujian, didapatkan bahwa fungsi-fungsi yang terdapat pada sistem berjalan seluruhnya dengan baik.

B. Saran

Adapun saran untuk penelitian selanjutnya antara lain:

1. Untuk penelitian selanjutnya agar membuat tampilan atau frontend website yang menarik dan mudah untuk digunakan.
2. Meningkatkan kinerja *web crawler* agar memiliki penggunaan *RAM* yang lebih efisien dan dapat menyimpan dokumen yang berbentuk file.
3. Meningkatkan kinerja *PageRank* atau *TF IDF* agar proses perankingan menjadi lebih cepat.

REFERENSI

- Brin, S. dan Page, L. (1998). The anatomy of a large-scale hypertextual web search engine.
- Cho, J., Garcia-Molina, H., dan Page, L. (1998). Efficient crawling through url ordering.
- Eichmann, D. (1994). The rbse spider - balancing effective search against web load. *Proceedings of the first World Wide Web Conference*.
- Firdaus, Pasnur, dan Wabdillah (2019). Implementasi cosine similarity untuk peningkatan akurasi pengukuran kesamaan dokumen pada klasifikasi dokumen berita dengan k nearest neighbour. *Inspiration: Jurnal Teknologi Informasi dan Komunikasi*, 9.
- Kaur, S. dan Geetha, G. (2020). Simhar-smart distributed web crawler for the hidden web using sim+ hash and redis server. *IEEE Access*, 8:117582–117592.
- Manning, C., Raghavan, P., dan Schutze, H. (2009). An introduction to information retrieval (online edition). *Cambridge: Cambridge University Press*.
- Page, L., Brin, S., Motwani, R., dan Winograd, T. (1999). The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab.
- Pinkerton, B. (1994). Finding what people want: Experiences with the webcrawler. *Proc. 2nd WWW Conf., 1994*.
- Qoriiba, Muhammad, F. (2021). Perancangan crawler sebagai pendukung pada search engine.
- Roul, R., Sahoo, J., dan Arora, K. (2019). Query-Optimized PageRank: A Novel Approach, pages 673–683.
- Sutherland, J., Viktorov, A., Blount, J., dan Puntikov, N. (2007). Distributed scrum: Agile project management with outsourced development teams. *2007 40th Annual Hawaii International Conference on System Sciences (HICSS'07)*, 274a–274a.