**PERANCANGAN SISTEM REKOMENDASI ARTIKEL ONLINE DENGAN *CONTENT BASED FILTERING***

**(Brilio.net)**

**SKRIPSI**

Digunakan Sebagai Syarat Maju Ujian Diploma IV

Politeknik Negeri Malang

**Oleh:**

**MUHAMMAD IQBAL ROFIKURRAHMAN. NIM. 1941727019**



**PROGRAM STUDI TEKNIK INFORMATIKA**

**JURUSAN TEKNOLOGI INFORMASI**

**POLITEKNIK NEGERI MALANG**

**JULI 2020**

HALAMAN PENGESAHAN

**PERANCANGAN SISTEM REKOMENDASI ARTIKEL ONLINE DENGAN *CONTENT BASED FILTERING***

**(Brilio.net)**

Disusun oleh:

MUHAMMAD IQBAL ROFIKURRAHMAN. NIM. 1941727019

Laporan Akhir ini telah diuji pada tanggal ... Juni 2020

Disetujui oleh:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1. | Penguji I | : | NIP. | ........................... |
|  |  |  |  |  |
| 2. | Penguji II | : | NIP. | ........................... |
|  |  |  |  |  |
| 3. | Pembimbing I | : | Rudy Ariyanto, S.T., M.Cs  NIP. 197903132008121002 | ........................... |
|  |  |  |  |  |
| 4. | Pembimbing II | : | Yoppy Yunhasnawa, S.T., M.Sc  NIP. | ........................... |

Mengetahui,

|  |  |
| --- | --- |
| Ketua Jurusan  Teknologi Informasi | Ketua Program Studi  Teknik Informatika |
| Rudy Ariyanto, S.T., M.Cs. | Imam Fahrur Rozi, S.T., M.T. |
| NIP. 19711110 199903 1 002 | |  | | --- | | NIP. 19840610 200812 1 004 | |

**PERNYATAAN**

Dengan ini saya menyatakan bahwa pada Skripsi ini tidak terdapat karya, baik seluruh maupun sebagian, yang sudah pernah diajukan untuk memperoleh gelar akademik di Perguruan Tinggi manapun, dan sepanjang pengetahuan saya juga tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini serta disebutkan dalam daftar sitasi/pustaka.

|  |  |
| --- | --- |
|  | Malang, … Juni 2020  M IQBAL R. |

**ABSTRAK**

***ABSTRACT***

**KATA PENGANTAR**

Puji Syukur kami panjatkan kehadirat Allah SWT/Tuhan YME atas segala rahmat dan hidayah-Nya penulis dapat menyelesaikan skripsi dengan judul “PERANCANGAN SISTEM REKOMENDASI ARTIKEL ONLINE DENGAN *CONTENT BASED FILTERING*”. Skripsi ini penulis susun sebagai persyaratan untuk menyelesaikan studi program Diploma IV Program Studi Teknik Informatika, Jurusan Teknologi Informasi, Politeknik Negeri Malang.

Kami menyadari bahwasannya dengan tanpa adanya dukungan dan kerja sama dari berbagai pihak, kegiatan laporan akhir ini tidak akan dapat berjalan baik. Untuk itu, kami ingin menyampaikan rasa terima kasih kepada:

1. Bapak Rudy Ariyanto, ST., M.Cs., selaku ketua jurusan Teknologi Informasi
2. Bapak Imam Fahrur Rozi, ST., MT., selaku ketua program studi Manajemen Informatika
3. Bapak Yoppy Yunhasnawa, S.ST., M.Sc, selaku dosen pembimbing II
4. Dan seluruh pihak yang telah membantu dan mendukung lancarnya pembuatan Laporan Akhir dari awal hingga akhir yang tidak dapat kami sebutkan satu persatu.

Penulis menyadari bahwa dalam penyusunan laporan akhir ini, masih banyak terdapat kekurangan dan kelemahan yang dimiliki penulis baik itu sistematika penulisan maupun penggunaan bahasa. Untuk itu penulis mengharapkan saran dan kritik dari berbagai pihak yang bersifat membangun demi penyempurnaan laporan ini. Semoga laporan ini berguna bagi pembaca secara umum dan penulis secara khusus. Akhir kata, penulis ucapkan banyak terima kasih.

Malang, … Juni 2020

M IQBAL R

**DAFTAR ISI**

Halaman

[**BAB I. PENDAHULUAN** 1](#_Toc40397660)

[**1.1** **Latar Belakang** 1](#_Toc40397661)

[**1.2** **Rumusan Masalah** 2](#_Toc40397662)

[**1.3** **Tujuan** 2](#_Toc40397663)

[**1.4** **Batasan Masalah** 3](#_Toc40397664)

[**1.5** **Sistematika Penulisan** 3](#_Toc40397665)

[BAB II. LANDASAN TEORI 5](#_Toc40397666)

[2.1 State-of-the-Art Penelitian Terdahulu 5](#_Toc40397667)

[2.2 *Text* *Mining* 8](#_Toc40397668)

[2.3 *Content Based Filtering* 8](#_Toc40397669)

[2.4 *Preprocessing* 8](#_Toc40397670)

[2.5 Tokenisasi 8](#_Toc40397671)

[2.6 *Stemming* 9](#_Toc40397672)

[2.7 Penghapusan *Stopword* 9](#_Toc40397673)

[**2.8**  **Pembobotan *Term Frequency - Inverse Document Frequency*** 9](#_Toc40397674)

[**2.9**  ***Dice Similarity*** 10](#_Toc40397675)

[BAB III. METODELOGI PENELITIAN 12](#_Toc40397676)

[3.1 Data 12](#_Toc40397677)

[3.2 Metode Pengambilan Data 13](#_Toc40397678)

[3.3 Metode Pengolahan Data 13](#_Toc40397679)

[3.4 Metode Pengembangan Perangkat Lunak 14](#_Toc40397680)

[3.5 Metode Pengujian Data 15](#_Toc40397681)

[BAB IV. ANALISIS DAN PERANCANGAN SISTEM 16](#_Toc40397682)

[4.1 Analisis Permasalahan 16](#_Toc40397683)

[4.2 Analisis Sistem 16](#_Toc40397684)

[4.2.1 Sistem Rekomendasi Terdahulu 17](#_Toc40397685)

[4.2.2 Sistem Rekomendasi Terbaru 18](#_Toc40397693)

[4.3 Analisis Data Masukan 18](#_Toc40397694)

[4.4 Analisis Algoritma 24](#_Toc40397695)

[4.4.1 Klasifikasi dengan algoritma *Support Vector Machine* 24](#_Toc40397696)

[4.5 Analisis Kebutuhan Non Fungsional 24](#_Toc40397697)

[4.6 Analisis Kebutuhan Fungsional 25](#_Toc40397698)

[4.6.1 *Database* 25](#_Toc40397699)

[4.6.2 *Use* *Case* 26](#_Toc40397700)

[4.6.3 *Activity Diagram* 29](#_Toc40397701)

[4.6.4 *Sequence Diagram* 31](#_Toc40397702)

[BAB V. IMPLEMENTASI DAN PENGUJIAN 36](#_Toc40397703)

[5.1 Implementasi *Database* 36](#_Toc40397704)

[5.2 Implementasi Proses Klasifikasi 36](#_Toc40397705)

[5.2.1 *Load Dataset* 36](#_Toc40397706)

[5.2.2 *Preprocessing* 36](#_Toc40397707)

[5.2.3 TF IDF 36](#_Toc40397708)

[5.2.4 *Training* 36](#_Toc40397709)

[5.2.5 *Testing* 36](#_Toc40397710)

[5.2.6 Pengujian 36](#_Toc40397711)

[5.3 Implementasi *Interface* 36](#_Toc40397712)

[5.4 Pengujian Fungsional 36](#_Toc40397713)

[5.5 Pengujian Akurasi 36](#_Toc40397714)

[BAB VI. HASIL DAN PEMBAHASAN 37](#_Toc40397715)

**DAFTAR GAMBAR**

**DAFTAR TABEL**

**DAFTAR LAMPIRAN**

**BAB I. PENDAHULUAN**

* 1. **Latar Belakang**

Artikel merupakan karangan tertulis yang panjangnya tidak dapat ditentukan, dimana tujuannya untuk menyampaikan gagasan maupun fakta dengan maksud meyakinkan, mendidik, ataupun menghibur [1]. Di era modern, artikel sering ditulis di berbagai media online seperti website. Artikel online dapat dibagi menjadi berbagai kategori sesuai pembahasannya. Dalam sebuah kategori pastinya akan memiliki jumlah artikel yang sangat banyak. Sehingga pengkategorian artikel, tidak dapat digunakan sebagai acuan sarana untuk mempermudah pengguna dalam memilih artikel yang tepat. Sistem rekomendasi hadir sebagai sarana agar pengguna dapat lebih mudah memilih artikel yang relevan dengan artikel lain sesuai yang ingin pengguna akses, tanpa perlu harus melakukan pencarian di kolom pencarian website.

Sistem rekomendasi merupakan sistem yang sering diterapkan oleh *website* penyedia artikel [2], karena sistem rekomendasi dapat ini memberikan dampak yang baik terhadap statistik *website*. Dalam sistem ini tingkat relevansi terhadap artikel yang direkomendasikan harus diperhatikan, agar para pengguna dapat memberikan respon yang baik pada *website* dan artikel yang sedang diakses. Pada sistem rekomendasi terdapat berbagai teknik dalam penerapannya, yaitu terdiri dari *content based filtering, collaborative filtering,* dan *hybrid filtering*.

Brilio.net merupakan sebuah perusahaan media penyedia artikel yang sudah cukup dikenal di Indonesia. *website* ini menyediakan artikel tentang gaya hidup, berita *viral*, teknologi dan artikel-artikel jenaka yang sering diperbincangkan oleh banyak netizen di Indonesia, dan memiliki jumlah pengunjung yang banyak dengan rata-rata sepuluh ribu pengunjung setiap harinya, brilio.net belum menerapkan sistem rekomendasi (*content based filtering*) yang berdasarkan sesuai dengan kemiripan artikel yang sedang diakses oleh pengguna, hal ini dapat mengurangi ketertarikan pengguna untuk membaca artikel lainnya, karena artikel yang direkomendasikan hanya berdasarkan artikel dengan jumlah akses terbanyak saja, sehingga tidak ada peningkatan dalam statistik *website* dibagian lama pengguna mengakses.

Pada penelitian sebelumnya telah dibuat sistem rekomendasi artikel yang serupa dengan menggunakan metode yang berbeda yaitu metode *cosine similarity* dengan hasil menunjukkan bahwa artikel yang direkomendasikan adalah relevan [3], kelemahannya sistem ini beroperasi kurang cepat, dikarenakan alur sistem yang akan melakukan operasi pengecekan kemiripan artikel setiap  pengguna mengakses, sehingga semakin banyak pengguna dan artikel yang diakses, semakin lama juga proses perhitungannya.

Brilio.net membutuhkan sebuah sistem rekomendasi artikel online dengan proses perhitungan yang lebih cepat dan dapat menampilkan artikel yang relevan sesuai dengan artikel yang diakses oleh pengguna. Sebuah artikel perbandingan menyebutkan bahwa metode *dice similarity* dan *jaccard similarity* tidak membutuhkan komputasi yang terlalu berat [4]. Oleh sebab itu dibuatlah perancangan sistem rekomendasi artikel online dengan *content based filtering*.

* 1. **Rumusan Masalah**

Berdasarkan latar belakang di atas, maka rumusan masalah yang dapat diambil, yaitu :

1. Bagaimana membuat sebuah sistem rekomendasi artikel yang relevan berdasarkan artikel yang sedang diakses oleh pengguna ?
2. Bagaimana mempercepat proses pemberian rekomendasi artikel kepada pengguna, tanpa terpengaruh dari jumlah artikel dan pengguna yang mengakses ?
   1. **Tujuan**

Berdasarkan rumusan masalah yang telah dijelaskan, maka tujuan dari penelitian ini yaitu :

1. Membuat prototipe sistem dalam memberikan rekomendasi artikel yang relevan sesuai dengan artikel yang sedang diakses oleh pengguna.
2. Mempercepat proses perhitungan dalam memberikan rekomendasi artikel tanpa terpengaruh dengan banyaknya artikel yang sedang diakses.
   1. **Batasan Masalah**

Agar skripsi penulis yang berjudul “Perancangan Sistem Rekomendasi Artikel Online dengan *Content Based Filtering*” dapat berjalan sesuai dengan rencana dan tujuan awal, maka penulis memberikan batasan-batasan masalah yaitu:

1. Sistem hanya memberikan rekomendasi artikel dari website Brilio.net saja.
2. Sistem ini memberikan rekomendasi sesuai kemiripan artikel yang sedang diakses, tanpa memperhatikan perilaku pengguna sebelumnya.
3. Artikel yang direkomendasikan adalah artikel yang telah diambil dan disimpan ke penyimpanan lokal sistem sebelumnya.
4. Sistem ini memberikan rekomendasi tidak bergantung kepada kategori artikel.
   1. **Sistematika Penulisan**

Uraian dalam laporan Skripsi penulis menyusun dengan Sistematika penulisan sebagai berikut:

**BAB I : PENDAHULUAN**

Pendahuluan berisikan tentang latar belakang, rumusan masalah, batasan masalah, tujuan dan manfaat.

**BAB II : LANDASAN TEORI**

Landasan teori berisikan tentang tinjauan pustaka dari aplikasi yang penulis buat.

**BAB III : METODELOGI PENELITIAN**

Berisi mengenai tahapan yang dilakukan untuk menyelesaikan masalah pada tugas akhir yang bersumber dari proses dalam perencanaa tugas akhir. Metode penelitian berisi urauian tentang metode pengmbilan data, metode pengembangan sistem, fase-fase pengembangan sistem.

**BAB IV : ANALISIS DAN PERANCANGAN**

Analisa dan Perancangan berisikan tentang analisa sistem aplikasi dan perancangannya.

**BAB V : IMPLEMENTASI DAN PENGUJIAN**

Implementasi dan pengujian berisikan penerapan/implementasi dan pengujian dari aplikasi yang telah penulis buat.

**BAB VI : HASIL DAN PEMBAHASAN**

Hasil dan Pembahasan berisikan tentang pembahasan serta analisa dari hasil proses tersebut.

**BAB VII : KESIMPULAN dan SARAN**

Kesimpulan berisikan tentang kesimpulan dari penelitian dan saran.

# BAB II. LANDASAN TEORI

## 2.1 State-of-the-Art Penelitian Terdahulu

Penelitian yang dilakukan oleh (Bening, Herwijayanti, 2017) berupa sebuah sistem klasifikasi berita online dengan menggunakan pembobotan TF.IDF dan cosine similarity, sistem tersebut melakukan klasifikasi berita dari sebuah situs yaitu kompas.com pengelompokan berita dianggap penting untuk kenyamanan user dalam mencari berita sesuai kategori. Hasil dari pengelompokan ini akurat 91.25 % akan lebih akurat apabila data yang digunakan lebih banyak.

Penelitian yang dilakukan oleh (Fikri, dan Ahmad Dzul, 2018) berupa perbandingan metode dice similarity dengan cosine similarity menggunakan query expansion pada pencarian ayatul ahkam dalam terjemah Al-quran berbahasa Indonesia. Undergraduate thesis, Universitas Islam Negeri Maulana Malik Ibrahim. Tujuan penelitian ini adalah untuk mengetahui perbandingan akurasi dan relevansi dokumen yang dihasilkan di antara kedua metode tersebut dengan atau tanpa menggunakan query expansion. Nilai yang menjadi ukuran adalah recall, precision dan f-measure. Dengan hasil dice similarity lebih baik.

Penelitian yang dilakukan oleh (Mas'udiah, P.E., Atmaja, M.D. & Mustafa, L.D. 2017) berupa sebuah sistem Information Retrieval Tugas Akhir dan Perhitungan Kemiripan Dokumen Mengacu Pada Abstrak Menggunakan Vector Space Model. Sistem ini merupakan sebuah sistem temu kembali informasi judul tugas akhir dan perhitungan kemiripan dokumen menggunakan vector space model. Sistem secara otomatis akan melakukan indexing secara offline dan temu kembali (retrieval) secara real time. Proses retrieval dimulai dengan mengambil query dari pengguna, menerapkan stop word removal sehingga dihasilkan keyword yang compaq tetapi dapat mewakili query tersebut, kemudian sistem menghitung kemiripan antara keyword dengan daftar dokumen yang diwakili oleh term-term di dalam index. hasil pengujian terlihat ketika kata kunci “android” dimasukkan maka akan tampil empat dokumen yang diurutkan sesuai tingkat kemiripannya, yaitu docId 3 dengan tingkat kemiripan 0.9512, docId 4 dengan tingkat kemiripan 0.5020, docId 2 dengan tingkat kemiripan 0.2671, docId 8 dengan tingkat kemiripan 0.1522.

Penelitian yang dilakukan oleh (Prasetyo, V.R., 2019) berupa sebuah sistem Penentuan Pembimbing Tugas Akhir Mahasiswa Jurusan Teknik Informatika Universitas Surabaya Dengan Metode Dice Coefficient, pada sistem ini berasal dari sebuah masalah dalam proses pemilihan pembimbing dalam mahasiswa untuk menyelesaikan tugas akhir yang takut salah dalam memilih pembimbing, hasil dari penelitian dengan sistem ini dengan metode K-Fold Cross Validation dengan nilai k = 100 untuk 1000 data TA yang ada, didapatkan nilai akurasi sistem sebesar 36,25%. Apabila diambil 100 data TA yang memiliki dosen pembimbing yang ideal, sistem memberikan nilai akurasi yang lebih baik yaitu sebesar 45,5%.

Penelitian yang dilakukan oleh (Yasni, L., Subroto, I.M.I. & Haviana, S.F.C. 2018) berupa sebuah sistem Implementasi Cosine Similarity Matching Dalam Penentuan Dosen Pembimbing Tugas Akhir. Pada penelitian ini dihitung tingkat kesamaan antara judul, topik, dan abstrak tugas akhir mahasiswa dibandingkan dengan data dosen pembimbing berupa keahlian dosen pembimbing, tugas akhir yang pernah dibimbing oleh dosen. Kemudian metode Cosine similarity akan menghitung tingkat kesamaan kedua query tersebut. Nilai kemiripan yang tertinggi akan dimunculkan sebagai dosen pembimbing yang direkomendasikan. Hasil dari penelitian ini sistem rekomendasi dosen pembimbing tugas akhir dengan menerapkan metode Cosine Similarity membantu mahasiswa mendapatkan dosen pembimbing sesuai dengan tugas akhir yang diajukan.

Tabel 2. 1 State-of-the-Art Penelitian Terdahulu

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Judul | Penulis  Univ/ Tahun | Permasalahan | Metode | Hasil |
| 1 | Klasifikasi Berita Online dengan menggunakan Pembobotan TF-IDF dan Cosine Similarity | Bening Herwijayanti  Brawijaya/ 2017 | Pengelompokan berita yang belum sesuai dan masih banyak human error saat proses pengelompokan berita | Cosine Similarity | akurat 91.25 % akan lebih akurat apabila data yang digunakan lebih banyak. |
| 2 | Implementasi Cosine Similarity Matching Dalam Penentuan Dosen Pembimbing Tugas Akhir | Yasni L, Subroto.  Universitas Islam Sultan Agung/ 2018 | Proses penentuan dosen pembimbing yang sesuai | Cosine Similairty | Sukses,Hal ini ditunjukkan dengan pengujian sistem yang dilakukan memperoleh performa yang cukup baikyaitu hasil pengujian *precision* dan *recall* memiliki performa rata-rata 0.74 |

## 2.2 *Text* *Mining*

*Text* *mining* merupakan proses analisis dalam data yang berupa teks dimana sumber data didapatkan dari dokumen (Ronen Feldman, 2007). Tujuan text mining adalah mencari kata-kata yang dapat mewakili isi dari dokumen sehingga dapat dilakukan analisis keterhubungan antar dokumen. Text mining sering digunakan untuk beberapa proses seperti pencocokan dokumen.

## 2.3 *Content Based Filtering*

Sistem rekomendasi berbasis konten (*Content-based Recommendation System*) menggunakan ketersediaan konten (sering juga disebut dengan fitur, atribut atau karakteristik) sebuah item sebagai basis dalam pemberian rekomendasi (Ricci, 2011). Sistem ini cocok digunakan untuk membuat rekomendasi yang tidak membutuhkan histori atau aktifitas pengguna sebelumnya.

Perbedaan metode *Content Based Filtering* dengan collaborative filtering adalah persyaratan yang harus dipenuhi yaitu rekaman hasil pengguna selama mengakses sehingga bisa dikelompokkan tema atau judul artikel apa saja yang pengguna sukai. Oleh sebab itu metode collaborative filtering ini tidak dapat diterapkan pada studi kasus ini.

## 2.4 *Preprocessing*

*Preprocessing* merupakan suatu proses yang dilakukan untuk mendapatkan data yang bersih sehingga proses pencarian data menjadi lebih akurat. Perbandingan dokumen atau data tanpa dilakukan *preprocessing* maka tidak dapat dinilai sebagai hasil yang relevan. preprocessing sendiri terdiri dari beberapa tahapan seperti Penghapusan *stopword*, *stemming*, dan tokenisasi*.*

*Preprocessing* dilakukan sebelum proses pembobotan kata, sehingga kata yang akan diolah menjadi sebuah kata baku dan memiliki arti sendiri, tanpa adanya imbuhan atau kata bantu.

## 2.5 Tokenisasi

Tokenisasi adalah proses pemecahan dokumen teks menjadi token-token kata. Tiga metode yang sering digunakan untuk melakukan tokenisasi, yaitu: Unigram (satu kata), Bigram (dua kata), dan Trigram (tiga kata). tokenisasi dilakukan biasanya di awal *preprocessing*.

Contoh terdapat sebuah kalimat “kuliah kerja di polinema” dilakukan tokenisasi dua kata sehingga menjadi :

* kuliah kerja
* di polinema

Pada penelitian ini proses tokenisasi dilakukan dalam bentuk satu kata sehingga dapat meningkatkan tingkat relevansi terhadap hasil perbandingan dokumen.

## 2.6 *Stemming*

*Stemming* memiliki 2 tahapan utama yaitu pengecekan kata dasar dan penghapusan afiks, prefiks, dan sufiks pada suatu kata. dengan proses *stemming* kata dasar dapat diperoleh seperti contoh “meramal” menjadi “ramal”, “penyaringan” menjadi “saring”.

*Stemming* dapat dilakukan dengan menyiapkan perpustakaan kata terlebih dahulu, semakin banyak jumlah kata semakin baik. Hal yang harus diperhatikan adalah bahasa yang akan digunakan dalam teknik *stemming* ini.

## 2.7 Penghapusan *Stopword*

*Stopword* merupakan kata-kata yang terdapat pada suatu kalimat, di mana kata tersebut memiliki pengaruh yang kecil pada proses information retrieval [9]*.* Penghapusan *stopword* ini adalah proses awal dalam *preprocessing.* Kata-kata yang termasuk ke dalam *stopword* adalah kata ganti (saya, kamu,dia) dan kata hubung (di, ke, dari, tetapi).

Contoh terdapat sebuah kalimat “saya kuliah di polinema” maka hasilnya setelah dilakukan penghapusan stopword akan menjadi “kuliah” dan “polinema”. Sama hal nya dengan *stemming*, penghapusan *stopword* juga memerlukan kumpulan kata yang sudah tersimpan sebelumnya.

**2.8 Pembobotan *Term Frequency - Inverse Document Frequency***

*Term Frequency - Inverse Document Frequency* atau TF - IDF adalah suatu metode algoritma yang berguna untuk menghitung bobot setiap kata yang diperoleh dari hasil *preprocessing* sebelumnya. Metode ini akan menghitung nilai Term Frequency (TF) dan Inverse Document Frequency (IDF) pada setiap token (kata) di setiap dokumen dalam korpus. Secara sederhana, metode TF-IDF digunakan untuk mengetahui seberapa sering suatu kata muncul di dalam dokumen.

Hasil perhitungan TF-IDF dapat memberikan sebuah nilai bobot setiap dokumen sesuai dengan kata kunci yang dicari, setelah dilakukan *sorting* dengan bentuk vektor untuk mengetahui dokumen yang paling mirip.

Rumus umum untuk *Term* *Weighting* TF-IDF adalah penggabungan dari formula perhitungan raw TF dengan formula IDF dengan cara mengalikan nilai TF dengan nilai IDF :

Wdt = tfdt \* Idft

Idft = (log(N/df))

(1).

Dimana:

Wdt = bobot dokumen ke­-d terhadap kata ke­-t.

tfdt = banyaknya kata yang dicari pada sebuah dokumen.

N = total dokumen.

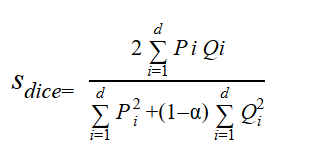
df = banyak dokumen yang mengandung kata yang dicari.

Setelah dilakukan pembobotan maka dokumen akan dihitung tingkat kemiripannya dengan metode *dice similarity.*

**2.9  *Dice Similarity***

*Dice Similarity* adalah metode untuk melihat tingkat kedekatan atau kesamaan (similarity) term dengan cara pembobotan term*.* Dokumen dipandang sebagai sebuah vektor yang memiliki *magnitude* (jarak) dan direction (arah). Kelebihan dari metode ini adalah proses komputasi yang tidak terlalu berat namun akurat dan dapat menentukan prioritas antara precision dan recall berdasarkan kebutuhan melalui nilai pada persamaan (2).

Untuk perhitungan metode *dice similarity* ini dapat dilakukan dengan rumus sebagai berikut :



(2).

Dimana P dan Q adalah dokumen yang berbeda. Pi adalah term i yang ada di dokumen P sedangkan Qi adalah term i yang ada di dokumen Q, dan α adalah nilai konstanta antara 0 dan 1. Apabila nilai α lebih besar dari 0,5 sehingga nilai precision akan lebih tinggi. Sedangkan jika nilai α lebih kecil dari 0,5 maka nilai recall akan lebih tinggi.

Sebagai contoh apabila **P** dan **Q** adalah himpunan set string (bukan vektor), jumlah unigram dari “Photography” and “Photographic” masing-masing adalah **|P| = 9** dan **|Q| = 10** dimana **|P n Q| = 8**, sehingga dapat diperoleh **dice(P,Q) = 0.842** atau persentase similarity dari kedua kata tersebut adalah 84.2%.

**2.10  *Jaccard Similarity***

*Jaccard similarity* adalah metode yang digunakan untuk membandingkan suatu kemiripan sebuah kata, kalimat atau dokumen. Metode *jaccard similarity* dapat digunakan tanpa harus melalui propses pembobotan.

(3).

Pada persamaan (3) metode jaccard yaitu jumlah gabungan antara dokumen satu dan dokumen dua dibagi dengan irisan dokumen satu dan dua.

# BAB III. METODELOGI PENELITIAN

## 3.1 Data

Data yang akan diolah merupakan data yang berasal dari artikel yang telah dibuat sebelumnya pada situs brilio.net (data uji), pengolahan data berfokus kepada isi artikel seperti judul, gambar, isi konten, tanggal terbit dan alamat URL. Dalam proses perhitungan algoritma, semua data kecuali alamat URL, gambar, dan tanggal terbit akan digabung menjadi satu dokumen untuk dibandingkan dengan dokumen yang telah dibuat sebelumnya.

Akan ada proses pengambilan data secara terjadwal pada sistem untuk memperoleh data baru pada situs brilio.net, data baru yang diperoleh akan di catat jumlahnya dalam sebuah *history.*

Tabel 3. 1 Contoh data hasil crawl

|  |  |  |  |
| --- | --- | --- | --- |
| Thumbnail | Title | Date | Url |
| https://cdn-brilio-net.akamaized.net/news/2020/05/13/184420/gara-gara-corona-mira-lesmana-ungkap-produser-film-rugi-miliaran-200513t.jpg | Mira Lesmana sebut produser film sudah rugi miliaran karena corona | 2020-05-13 | https://brilio.net/selebritis/gara-gara-corona-mira-lesmana-ungkap-produser-film-rugi-miliaran-200513t.html |
| https://cdn-brilio-net.akamaized.net/news/2020/05/13/184399/10-momen-kebersamaan-ranbir-kapoor-alia-bhatt-makin-mesra-200513v.jpg | 10 Momen kebersamaan Ranbir Kapoor & Alia Bhatt, makin mesra | 2020-05-13 | https://brilio.net/selebritis/10-momen-kebersamaan-ranbir-kapoor-alia-bhatt-makin-mesra-200513v.html |

Pada tabel 3.1 terdapat beberapa kolom yang tidak ditampilkan, sebagai contoh adalah kolom *content*, pada kolom content semua *text* akan diambil.

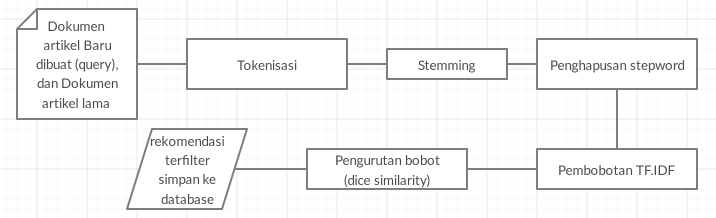
## 3.2 Metode Pengambilan Data

Metode yang digunakan dalam proses pengambilan data adalah metode *crawling,*  dimana metode ini juga dikenal dengan metode *scrapping* yang intinya melakukan ekstraksi sebuah halaman pada situs tertentu untuk memperoleh data yang dibutuhkan, khususnya pada penelitian kali ini adalah situs brilio.net.

Metode ini memiliki tingkat kesulitan tersendiri tergantung situs yang akan dilakukan ekstraksi. Semua data yang diperoleh adalah data legal, dan semua data yang tampil saat membuka situs secara manual adalah data yang dapat dilakukan proses *crawling.*

## 3.3 Metode Pengolahan Data

Metode pengolahan data dimulai saat pertama kali data diperoleh dari proses *crawling*. Data yang akan diperoleh untuk diolah adalah judul artikel dan isi konten, yang telah diformat menjadi satu dokumen tanpa adanya tag html menjadi teks keseluruhan. Sebelumnya akan disediakan beberapa data sebagai data training untuk mengenalkan sistem.



Gambar 3. 1 Alur pengolahan data

Disaat ada artikel baru yang terbaca dari sistem *crawling* maka data tersebut akan dilakukan pemrosesan seperti proses *preprocessing* dan dilanjutkan dengan melakukan perhitungan pembobotan dengan TF.IDF, setelah menentukan bobot dari TF.IDF selanjutnya bobot tersebut akan dihitung nilai kedekatannya dengan metode *dice similarity* dimana metode ini adalah metode yang sering digunakan untuk mencari nilai kedekatan suatu dokumen.

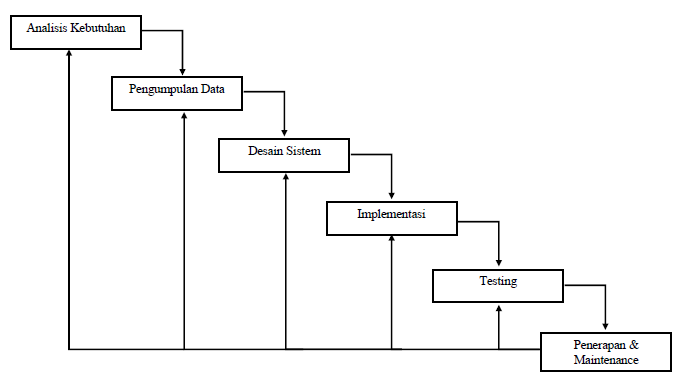
Keunggulan dari *dice similarity* dibandingkan metode lain yang pernah digunakan yaitu dapat menentukan prioritas *precision* dan *recall* sesuai kebutuhan, dan keunggulan metode *jaccard similairty* dibandingkan dengan metode yang pernah digunakan adalah proses perhitungan yang cepat, Maka *dice similarity* dan *jaccard* similarityj menjadi metode yang dirasa tepat untuk menjalankan sebuah sistem rekomendasi artikel berita yang cepat.

Setelah data rekomendasi ditentukan dan difilter sesuai kebutuhan, selanjutnya data akan disimpan kedalam database dengan *id* sesuai dengan *id* artikel yang baru, dimana artikel baru ini digunakan sebagai *query* pembanding dengan artikel lainnya.

Untuk mengatasi keterlambatan pembaharuan data rekomendasi, maka dibuatlah sebuah alur sistem dimana ketika artikel lama di akses kembali oleh pengguna, perhitungan rekomendasi akan dilakukan pembaharuan berdasarkan data artikel terbaru berdasarkan tanggal pembuatan artikel.

## 3.4 Metode Pengembangan Perangkat Lunak

Metode dalam penelitian ini menggunakan metode waterfall. Dalam metode ini terdapat 6 tahap, langkah-langkahnya adalah sebagai berikut:



Gambar 3. 2 Diagram Alur Waterfall

1. *Requirement Analysis*

Pada tahap ini akan dijelaskan tentang semua kebutuhan sistem untuk menjalankan sistem rekomendasi artikel ini, baik kebutuhan fungsional dan kebutuhan non fungsional**.**

1. Pengumpulan Data

Proses pengumpulan data dilakukan setiap hari dan dilakukan oleh sistem, sehingga data yang diolah adalah data dinamis yang sifatnya tidak tetap.

1. *Design*

Desain sistem akan terbagi menjadi beberapa bagian, seperti bagian admin atau sistem kontrol dan bagian rekomendasi atau desain untuk pengguna pada situs brilio.net

1. *Implementation*

Dalam tahap ini dilakukan pembuatan aplikasi. Pembuatan perangkat lunak dipecah menjadi modul-modul kecil yang nantinya akan digabungkan dalam tahap berikutnya. Selain itu dalam tahap ini juga dilakukan pemeriksaaan terhadap modul yang dibuat, apakah sudah memenuhi fungsi yang diinginkan*.*

1. *Testing*

Pada tahap testing dilakukan uji coba pada aplikasi yang telah dibuat untuk mengetahui apakah perangkat lunak yang dibuat telah sesuai dengan desainnya dan apakah masih terdapat kesalahan

1. *Maintenance*

Tahap ini dilakukan jika perangkat lunak sudah jadi danmengalami kerusakan. Maka akandilakukan pemeliharaan. Pemeliharaan termasuk dalam memperbaiki kesalahan yang tidak ditemukan pada langkah sebelumnya.

## 3.5 Metode Pengujian Data

Pengujian yang dilakukan untuk menguji sistem dilakukan dengan 2 langkah, yaitu pengujian fungsionalitas dan pengujian performa sistem. Pengujian bertujuan untuk memastikan bahwa sistem yang dibangun telah berjalan sesuai dengan yang diharapkan.

1. Pengujian Fungsionalitas

Uji coba fungsionalitas dikatakan berhasil apabila fungsi yang ada pada sistem sesuai dengan yang diharapkan. Uji fungsionalitas yang diterapkan pada penelitian ini ialah dengan mencoba menjalankan setiap fitur yang ada dan yang disediakan pada sistem ini

1. Pengujian Akurasi Sistem

Dalam pengujian ini, dilakukan untuk mengetahui keakuratan aplikasi dalam menampilkan data hasil pencarian oleh pengguna.

# BAB IV. ANALISIS DAN PERANCANGAN SISTEM

## 4.1 Analisis Permasalahan

Sebuah situs besar penyedia artikel akan terus menerbitkan artikel setiap hari nya, jumlah artikel yang terbit tidak dapat diperkirakan. Hasil rata-rata jumlah artikel harian pada situs brilio.net bisa mencapai 20 - 30 artikel perharinya. Tema atau kategori artikel yang di unggah tidak selalu sama. Penerapan sistem rekomendasi artikel merupakah suatu hal yang sangat penting agar para pengguna bisa terus mengakses dalam waktu yang lama.

Permasalahan yang terjadi adalah situs brilio.net saat ini menerapkan rekomendasi artikel manual berdasarkan editor atau penulis artikel, dimana artikel yang direkomendasikan belum tentu berkaitan satu sama lain. Dalam hal ini pembuatan sistem rekomendasi artikel dibutuhkan, sehingga sistem dapat memberikan rekomendasi artikel secara otomatis berdasarkan tingkat kemiripan artikel yang telah dibuat.

Dengan adanya sistem rekomendasi otomatis daftar rekomendasi sistem yang telah dibuat sebelumnya akan terus diperbaharui mengikuti hasil perhitungan tingkat kemiripan artikel yang baru. Pada sistem ini jumlah rekomendasi dibatasi menjadi 5 artikel terbaik, dan perubahan data rekomendasi akan dilakukan pada artikel yang telah diunggah 30 hari terakhir.

## 4.2 Analisis Sistem

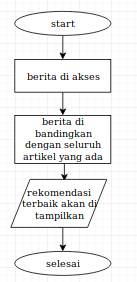
Sistem rekomendasi artikel yang telah dibuat merupakan sebuah sistem yang terpisah dari situs asli yaitu brilio.net (sistem utama). sehingga dibutuhkan sebuah komunikasi antar sistem agar pengolahan dan pemindahan data dapat berjalan dengan baik, dalam praktik nya sistem rekomendasi terpisah ini memiliki beberapa keuntungan seperti berikut:

1. Perubahan atau perbaikan sistem dapat lebih mudah
2. Saat sistem error sistem utama tidak memperoleh dampak yang siginfikan
3. Proses sistem tidak membrikan beban pada sistem utama

Gambaran sistem secara teknis dapat digolongkan menjadi *server* dan *client* sistem. Dimana situs utama akan menjadi bagian *client* yang akan melakukan request kepada *server*, begitu juga sebaliknya *server* akan memberikan respon terharap request yang ada.

### Sistem Rekomendasi Terdahulu

Pada penelitan sebelumnya telah dibuat beberapa macam sistem rekomendasi artikel yang memiliki tujuan yang sama yaitu untuk memberikan rekomendasi artikel sesuai dengan artikel yang sedang diakses, berikut adalah alur sistem penelitian terdahulu



Gambar 4. 1 alur sistem terdahulu

Pada sistem rekomendasi sebelumnya, proses perhitungan dilakukan ketika berita saat itu diakses oleh pengguna, ketika seorang pengguna mengakses satu artikel maka prosesnya akan cepat, namun hal ini berbeda ketika ribuan pengguna mengakses ribuan artikel yang berbeda. maka dibutuhkan sebuah sistem yang baru dengan alur sistem yang berbeda sehingga dapat mempercepat proses pemberian rekomendasi tanpa terpengaruh oleh banyaknya pengguna dan artikel.

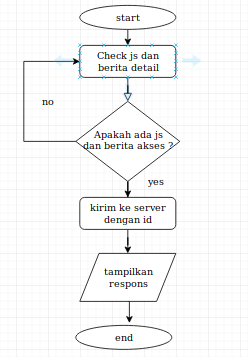
Kelebihan dari sistem yang terdahulu semua rekomendasi yang diberikan adalah berasal dari semua artikel yang terbaru, namun hal ini dapat membuat proses sistem menjadi sangat lama terutama dalam proses perhitungan.



### Sistem Rekomendasi Terbaru

Sistem rekomendasi terbaru menerapkan sebuah alur yang melakukan sebuah perhitungan atau kalkulasi kemiripan terlebih dahulu pada saat pembuatan artikel, sehingga saat artikel yang dibuat dan diakses oleh pengguna sistem tidak memerlukan lagi proses perhitungan kemiripan ulang. Hal ini dapat meringankan dan mempercepat kinerja sistem.

Berikut adalah alur sistem yang terletak pada bagian situs utama atau *client* sistem:

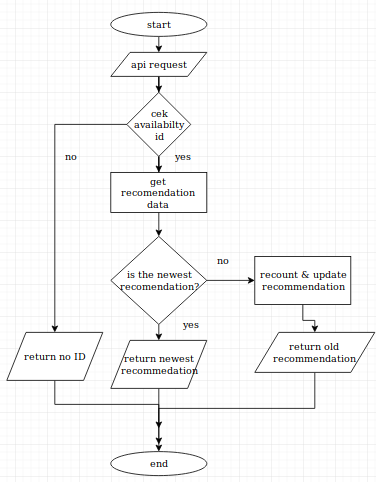


Gambar 4. 2 Alur sistem *client*

Pada gambar 4.2 merupakan alur sistem disaat pengguna mulai mengakses berita pada situs brilio.net. Sistem akan melakukan pengecekan apakah ada file javascript yang telah disediakan untuk melakukan pengiriman data kepada server, jika ada, maka id artikel akan dikirimkan dan menunggu server akan memberikan sebuah jawaban berupa data rekomendasi artikel yang telah dilakukan perhitungan sebelumnya. Sehingga akan ditampilkan ke dalam sebuah tag atau elemen yang sudah disediakan di halaman brilio.net.

Dalam pemasangan kode javascript ini, pihak brilio dapat menentukan sesuai parameter jenis rekomendasi apa yang akan ditampilkan, bisa berupa gambar atau hanya list saja. namun hal ini tidak berpengaruh terhadap proses perhitungan rekomendasi, hanya tipe respon dari server saja yang berbeda.

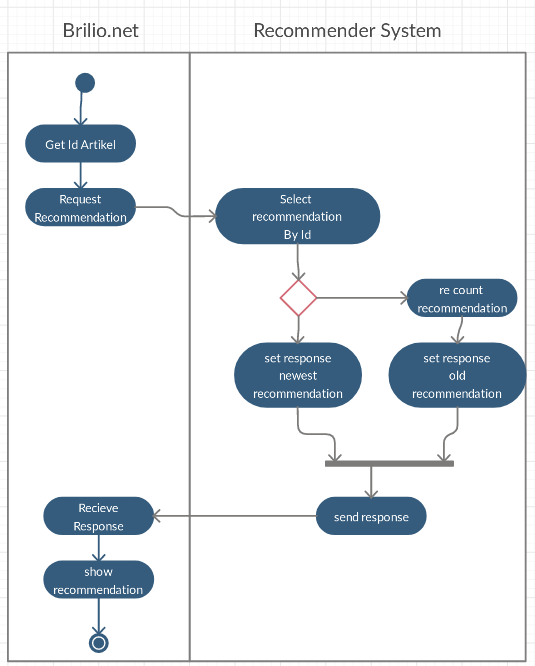
Berikut adalah alur sistem yang terletak pada bagian *server*:



Gambar 4. 3 Alur sistem *client*

Pada gambar 4.3 dijelaskan alur tentang proses sistem memberikan rekomendasi kepada situs brilio, sistem akan berjalan otomatis ketika ada request dari pihak brilio berupa ID artikel yang sedang pengguna akses. Dijelaskan bahwa sistem melakukan pengecekan ketersediaan ID yang diminta, setelah sukses maka sistem akan memperoleh data rekomendasi yang telah dihitung sebelumnya, dan dilakukan perbandingan apakah data ini merupakan data rekomendasi lama atau tidak. sehingga jika data termasuk data lama maka sistem akan melakukan perhitungan ulang dengan artikel yang lebih terbaru.

Berikut adalah *activity diagram* dari dua entitas sebagai client dan server:



Gambar 4. 4 activity diagram client server

Entitas brilio.net selaku client yang akan merequest setiap ada pengguna yang mengakses artikel, lalu *recommender system* selaku server yang akan menyediakan list rekomendasi yang sudah dihitung sebelumnya saat pertama kali artikel dimasukkan ke dalam sistem.

## 4.3 Analisis Kebutuhan Non Fungsional

Analisis kebutuhan non fungsional merupakan analisis untuk mengetahui spesifikasi kebutuhan sistem yang dibangun. Spesifikasi kebutuhan,yaitu analisis perangkat lunak dan perangkat keras.

1. Analisis Kebutuhan Perangkat Lunak

Tabel 4. 1 Kebutuhan perangkat lunak

|  |  |  |
| --- | --- | --- |
| **No.** | **Perangkat** | **Keterangan** |
| 1. | *Web Browser* | *Google Chrome / Firefox* |
| 2. | *Web Server* | *Apache / Xampp* |
| 3. | *DBMS* | *PhpMyAdmin* |
| 4. | *Text Editor* | *Visual Studio Code* |
| 5. | *Framework* | *Python & laravel* |

1. Analisis Kebutuhan Perangkat Keras

Tabel 4. 2 Kebutuhan perangkat keras

|  |  |  |
| --- | --- | --- |
| **No.** | **Perangkat** | **Keterangan** |
| 1. | *Processor* | *Intel Core i7* |
| 2. | *Memory* | *8GB* |
| 3. | *Hardisk* | *100GB* |

## 4.4 Analisis Kebutuhan Fungsional

Analisa kebutuhan fungsional berisikan proses-proses yang dilakukan oleh sistem. Dalam hal ini user sebagai pengguna dapat menggunakan layanan-layanan pada sistem, antara lain :

Sistem dapat mengumpulkan data artikel terbaru dan menyimpannya kedalam database

Sistem dapat melakukan *preprocessing* data yang disimpan didalam database

Sistem dapat melakukan proses pembobotan

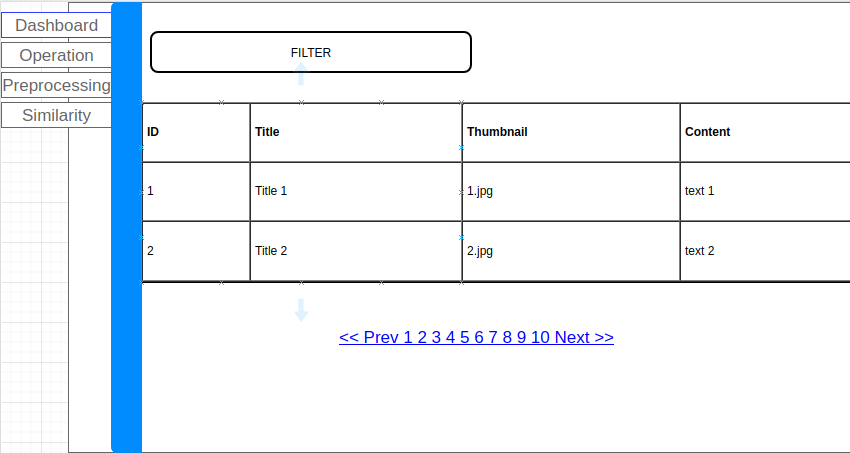
Sistem dapat melakukan perhituangan *similarity*

Sistem dapat melakukan pengurutan artikel dengan kemiripan tertinggi dan menampilkan nya saat artikel diakses

## 4.5 Perancangan Antarmuka

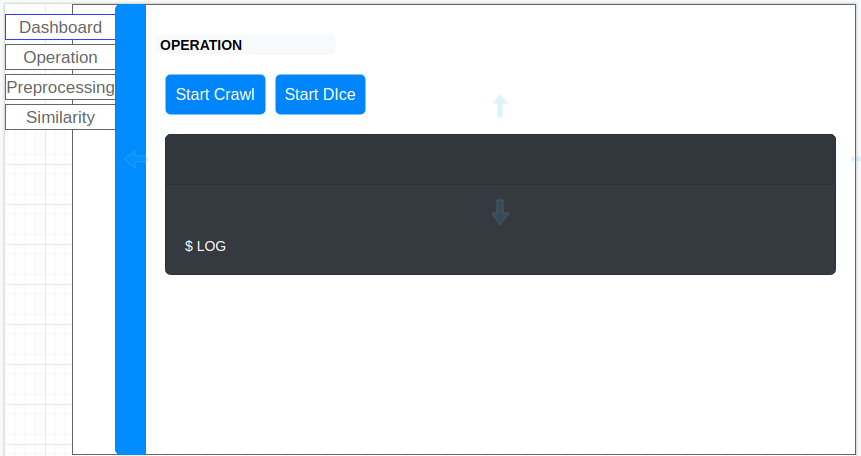
Perancangan antarmuka sistem rekomendasi terdiri dari beberapa menu yang memiliki tujuan tersendiri.

Bagian pertama adalah halaman *dashboard*



Gambar 4. 5 Halaman *dashboard*

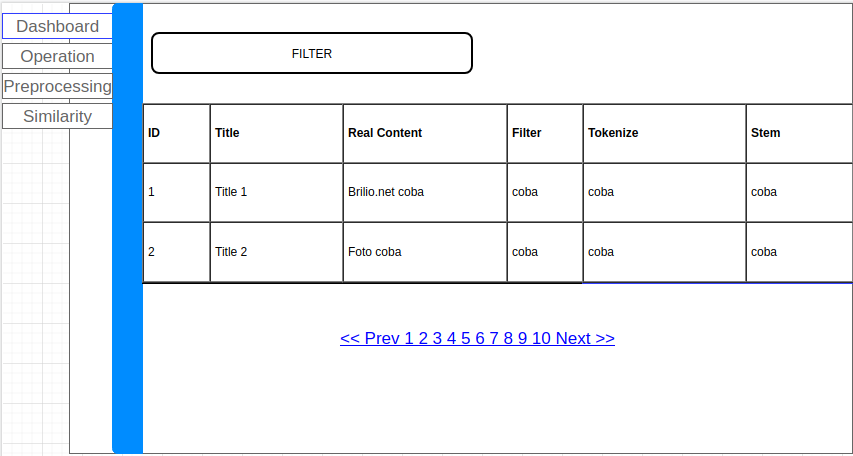
Pada halaman *dashboard* akan disediakan hasil *crawling* artikel dari situs brilio.net namun hanya di tampilkan kolom yang dibutuhkan saja. Akan ada fungsi untuk filter pada halaman ini.



Gambar 4. 6 Halaman *operation*

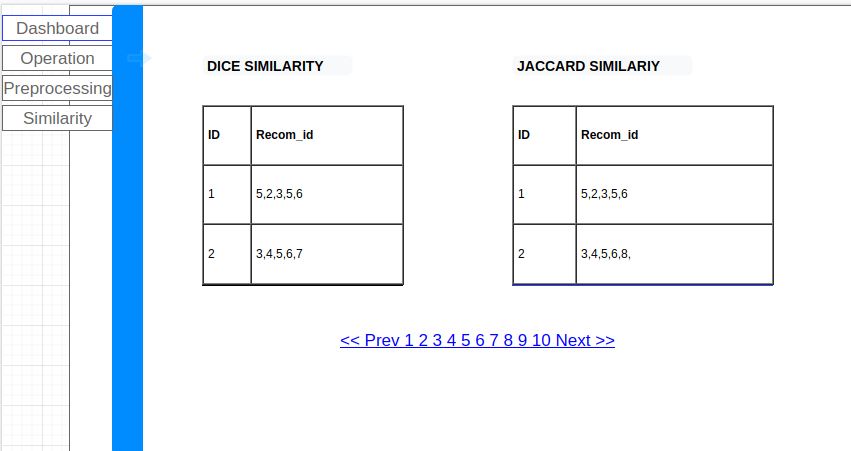
Pada halaman *operation* akan ada dua aksi untuk memulai proses *crawling* dan perhitungan *similarity*. hasil operasi akan ditampilkan pada bagian log. Log yang ditampilkan merupakan hasil *output* dari pemrograman *python* yang telah disiapkan dan di panggil sebelumnya.

Pada bagian selanjutnya adalah halaman *preprocessing*



Gambar 4. 7 Halaman *preprocessing*

Pada halaman *preprocessing* akan ada tabel untuk menampilkan hasil pengolahan kata dari artikel yang telah diperoleh sebebelumnya. Artikel tersebut akan diolah agar proses perhitungan dapat lebih maksimal. Pengolahan berupa *filtering* yang berguna untuk menhilangkan karakter yang tidak dibutuhkan dan mengubah semua kata menjadi huruf kecil, pengolahan selanjutnya adalah tokenisasi yang berguna untuk menghapus beberapa kata yang tidak terpakai, dan yang terakhir adalah proses pengolahan *stemming* yang berguna untuk mengubah kata kerja menjadi kata baku.



Gambar 4. 8 Halaman *similarity*

Pada halaman *similarity* akan menampilkan hasil rangking artikel dari proses perhitungan.

## 4.6 Perancangan Basisdata

Perancangan basisdata pada sistem rekomendasi ini terdiri dari beberapa tabel dan menggunakan Mysql sebagai *database management system*.

Tabel 4. 3 Tabelartikel

|  |  |  |
| --- | --- | --- |
| **No** | **Atribut** | **Tipe Data** |
| 1 | Id | Int(11) PK |
| 2 | Thumbnail | Text |
| 3 | Title | Varchar(500) |
| 4 | Real\_content | Text |
| 5 | Date | Date |
| 6 | Url | Varchar(225) |
| 7 | Filter | Text |
| 8 | Tokenize | Text |
| 9 | Term | Text |
| 10 | Content | Text |

Pada tabel artikel memiliki sepuluh kolom yang berguna menyimpan data artikel yang diperoleh hasil dari *crawling* dan hasil *preprocessing* sebelumnya.

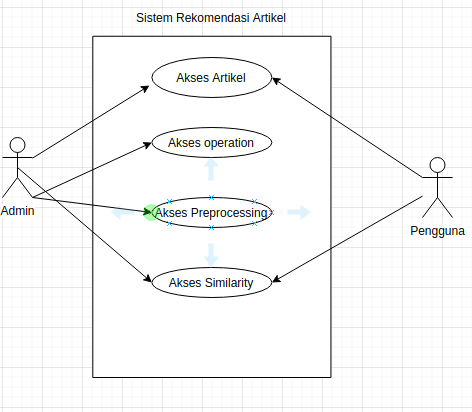
Tabel 4. 4 Tabel similairty

|  |  |  |
| --- | --- | --- |
| **No** | **Atribut** | **Tipe Data** |
| 1 | Article\_id | Int(11) FK id |
| 2 | Recommendation\_id | Varchar(50) |
| 3 | Similarity | Varchar(100) |
| 4 | Percent | Varchar(100) |

Pada tabel selanjutnya akan ada dua tabel yang mirip yaitu tabel *similarity* dan tabel *similarity\_other*, dua tabel ini berguna untuk menyimpan hasil perhitungan dan perangkingan artikel yang telah diproses sebelumnya. Perbedaan dari dua tabel ini adalah metode perhitungan yang digunakan, yaitu menggunakan metode dice dan jaccard *similarity.*

## 4.7 *Use Case*

*Use case* merupakan diagram yang terdiri dari aktor dan proses yang dapat dilakukan oleh aktor tersebut. *Use case* dibuat berdasarkan layanan-layanan yang dimiliki oleh sistem. Berikut adalah *Use case* pada sistem rekomendasi ini.



Gambar 4. 9 Use Case Sistem

Deskripsi *use case*

Tabel 4. 5 Deskripsi use case

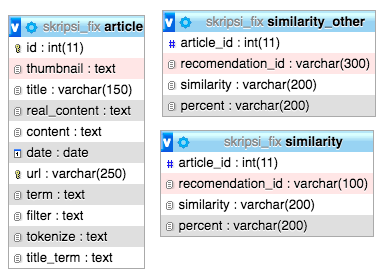
|  |  |  |
| --- | --- | --- |
| **No** | **Proses** | **Deskripsi** |
| 1 | Akses Artikel | Dua aktor dapat mengakses proses ini, namun perebedaanya adalah aktor pengguna hanya mengakses artikel melalui portal web utama atau halaman homepage. |
| 2 | Akses Operation | Proses ini berisi tentang bagaimana artikel akan di crawling dan diranking sesuai hasil perhitungan |
| 3 | Akses *preprocessing* | Proses ini berisi tentang hasil *preprocessing* yang telah dioperasikan sebelumnya. |
| 4 | Akses *Similarity* | Proses ini juga dapat diakses dua aktor namun perbedaanya aktor pengguna hanya mengakses hasil berupa artikel lengkap rekomendasi saja. |

Aktor penguna tidak dapat mengakses proses keseluruhan pada halaman admin.

# BAB V. IMPLEMENTASI DAN PENGUJIAN

## 5.1 Implementasi *Database*

Sistem ini menggunakan sebuah database yang terdiri dari tiga tabel, dengan dua tabel yang memiliki struktur yang sama berguna untuk memberikan hasil perhitungan kemiripan dari berbagai artikel.



Gambar 5. 1 Desain database

Pada gambar desain database terdapat tabel similarity dan similarity\_other yang memiliki struktur data yang sama. dengan tabel article yang menjadi tabel utama tempat untuk menyimpan data artikel yang diperoleh dari sistem. Pada tabel article kolom yang akan dibuat sebagai bahan perhitungan adalah kolom term, dimana nantinya berisi data gabungan dari kolom title dan kolom content yang telah di *preprocessing* sebelumnya. Untuk kolom bagian thumbnail, real\_content dan kolom lainnya akan di tampilkan saat artikel di akses di situs resminya.

Tabel similarity dan similarity\_other menyimpan data hasil perhitungan kecocokan dari setiap artikel. Kolom article\_id menandakan artikel yang dibandingnkan dan kolom recommendation\_id menyimpan id artikel lain yang telah diurutkan sesuai dengan hasil kecocokan yang paling tinggi, kolom similarity adalah skor akhir dari hasil perhitungan dan kolom percent merupakan hasil persentase dari kolom similarity.

Tabel similarity digunakan untuk menyimpan hasil perhitungan dari metode dice similarity, sedangan tabel similarity\_other digunakan untuk menyimpan hasil perhitungan dari metode jaccard\_coefficient.

## 5.2 Implementasi *Crawling* Artikel

Sistem ini melakukan pengolahan data artikel yang berasal dari situs brilio.net, artikel yang diolah merupakan artikel hasil *crawling* sistem terpisah dari situs brilio.net itu sendiri. Teknologi yang digunakan dalam proses *crawling* menggunakan bahasa pemrograman pythton dengan *library* BeautifulSoup4.

Adapun data yang diperoleh berupa link, judul, tanggal, konten, dan gambar artikel, gambar yang dimaksud hanya berupa link dan bukan berupa file, sehingga apabila ada gambar yang dihapus dari situs brilio maka gambar dari hasil *crawling* juga akan terhapus. Kelemahan menggunakan sistem *crawling* ini adalah data yang proses tidak selalu update, karena proses crawling ini dilakukan hanya sehari sekali saja sesuai jadwal. Kelemahan lainnya adalah apabila artikel ada yang terhapus maka sistem ini tidak dapat mendeteksi dan data yang terakhir ambil akan tetap diproses.

**articles = soup.find\_all('p',class\_='text-article')**

**links = []**

**for article in articles :**

**url = ''**

**el = article.a.attrs['href']**

**if 'http' in el :**

**url = el**

**else :**

**url = 'https://brilio.net' + el**

**links.append(url)**

**if(len(links) > 9 ):**

**break**

**cursor = mydb.cursor()**

**for link in links:**

**cursor.execute('select count(url) from article where url ="'+link+'" ')**

**total=cursor.fetchone()**

**if(total[0] > 0):**

**links.remove(link)**

**cursor.close ()**

Kutipan kode diatas merupakan contoh kode untuk melakukan pengambilan data di bagian halaman depan situs brilio.net

Berikut adalah potongan kode untuk medapatkan data detail dari data halaman depan yang telah diperoleh pada proses *crawling* halaman depan sebelumnya.

**for link in links :**

**try:**

**if (link[0:10] != "https://s."):**

**content = urllib.request.urlopen(link).read()**

**soup = BeautifulSoup(content, 'html.parser')**

**konten = soup.find('div', class\_='main-content').find\_all('p')**

**# GET DETAIL**

**titles.append(soup.find('title').get\_text())**

**# DATE**

**try :**

**time\_convert = timeConvert(soup.find('p', class\_='index-date').get\_text())**

**except :**

**time\_convert = datetime.datetime.today().strftime('%Y-%m-%d')**

**dates.append(time\_convert)**

**temp\_img = soup.find('figure', class\_='headline-index').find('img')**

**thumbnails.append(temp\_img.attrs['src'])**

**urls.append(link)**

**text\_temp = ""**

**for p in konten :**

**if(p.find('div', class\_='detail-box')):**

**continue**

**else :**

**text\_temp += ' '+p.get\_text()**

**contents.append(text\_temp)**

**print("crawled "+ str(urut))**

**#  time.sleep(1)**

**urut+= 1**

## 5.3 Implementasi *Preprocessing* Data

Implementasi preprocessing data dilakukan saat data telah melalui proses *crawling* sebelumnya. Preprocessing dibagi menjadi tiga tahap yaitu *filter, tokenize, stem*.

## 5.3.1 *Filtering*

Proses *filtering* dilakukan untuk menyaring data dari berbagai karakter dan nomor yang tidak digukanan dalam proses perhitungan.

**def filtering(kalimat) :**

**removed\_number = re.sub(r"\d+", "",kalimat)**

**removed\_unused\_char = re.sub(r'[^\w\s]',' ',removed\_number)**

**ready\_to\_tokenize = removed\_unused\_char.lower().strip()**

**return ready\_to\_tokenize**

Pada proses filtering ini juga dilakukan pengubahan bentuk kaliamat dari huruf besar manjadi huruf kecil semua.

### 5.3.2 *Tokenizing*

Setelah prosef *filtering* telah dilakukan maka proses selanjutnya adalah *tokenizing*.

**def tokenize(ready\_to\_tokenize):**

**stop\_factory = StopWordRemoverFactory().get\_stop\_words() #load defaul stopword**

**more\_stopword = ['brilio','net','com','foto'] #menambahkan stopword**

**data = stop\_factory + more\_stopword**

**dictionary = ArrayDictionary(data)**

**str = StopWordRemover(dictionary)**

**ready\_to\_stem = nltk.tokenize.word\_tokenize(str.remove(ready\_to\_tokenize))**

**return ready\_to\_stem**

Proses *Tokenizing* dilakukan untuk mengubah kaliamat menjadi satuan kata (token) dan juga pada pada potongan kode diatas terdapat proses menyeleraskan atau menghilangkan beberapa kata yang tidak digunakan dalam perhitungan (*stopwords*) seperti contoh kata “brilio”, karena semua artikel yang ada pasti diawali dengan kata brilio.

### 5.3.3 *Stemming*

Proses *stemming* dijalankan saat proses *tokenzing* selesai dijalankan.

**def stem(ready\_to\_stem):**

**factory = StemmerFactory()**

**stemmer = factory.create\_stemmer()**

**hasil = stemmer.stem(" ".join(ready\_to\_stem))**

**return hasil**

Pada proses ini setiap kata yang telah di *filter dan tokenize* akan dipangkas menjadi kata baku seperti contoh “mencintai” menjadi “cinta”, dengan mengubahnya menjadi kata baku maka proses perhitungan perbandingan artikel dapat memperoleh hasil yang baik.

## 5.4 Implementasi Pembobotan TFIDF

Proses pembobotan dilakukan menggunakan metode TF IDF untuk dapat menghitung jumlah term dalam sebuah dokumen (artikel) dan untuk mengetahui nilai bobot setiap term nya sehingga selanjutnya diproses dalam perhitungan algoritma *dice similarity.*

**total = len(get\_articles)**

**a = get\_articles**

**features = AllSimilarity()**

**tfidfs = AllSimilarity()**

**for i in range(0,total) :**

**for j in range(0,total) :**

**if i == j :**

**continue**

**else :**

**tfidf = TfidfVectorizer(use\_idf=True, lowercase=True, preprocessor=None, tokenizer=None, stop\_words=None)**

**texts = [a[i]['term'].replace(',', ' '),a[j]['term'].replace(',', ' ')]**

**feature = tfidf.fit\_transform(texts)**

**tfidfs.add(str(a[i]['id'])+'-'+str(a[j]['id']),tfidf)**

**# print pd.DataFrame(feature.toarray(),columns=tfidf.get\_feature\_names())**

**features.add(str(a[i]['id'])+'-'+str(a[j]['id']),feature)**

Potongan kode diatas menjalankan perintah untuk melakukan pembobotan dari setiap artikel yang diperoleh.

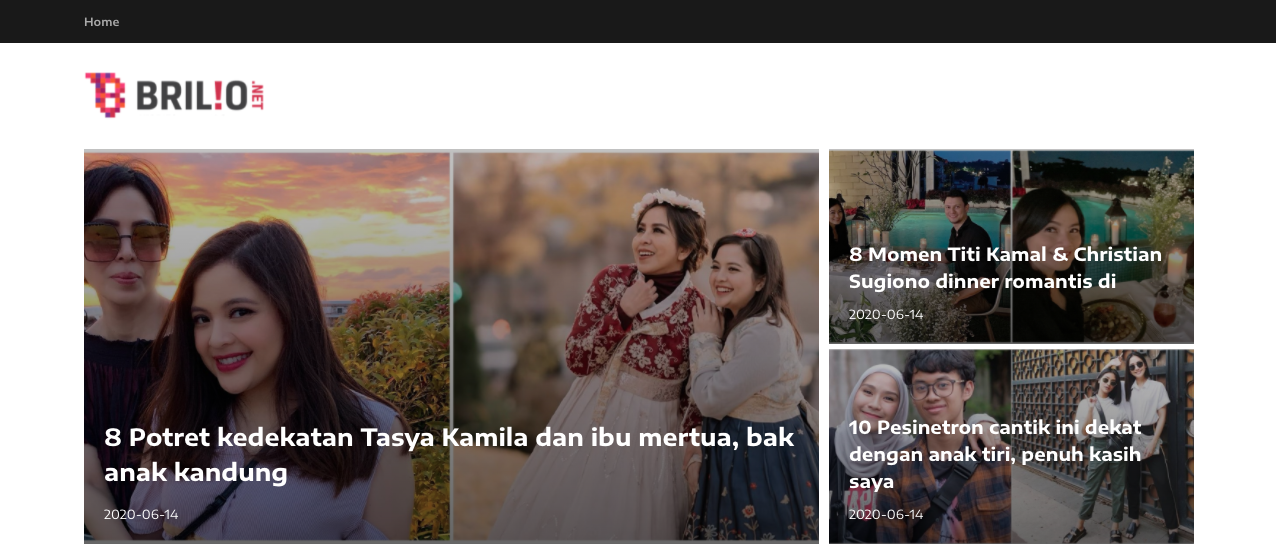
Pembobotan pada sistem ini tidak menggabung semua dokumen menjadi satu, melainkan pembobotan dilakukan disetiap dokumen. Hal ini dilakukan karena setiap proses perhitungan berjalan jumlah dokumen yang digunakan tidak selalu sama, bergantung dengan jumlah artikel yang diperoleh.

## 5.5 Implementasi Tampilan

Sistem ini dirancang dalam desain berbasis website dengan menggunakan *framework* CSS berupa bootstrap.

### 5.5.1 *Homepage*

Pada halaman awal atau *homepage* akan ditampilkan hasil artikel terbaru yang telah melalui berbagai proses mulai dari *crawling* sampai proses perhitungan akhir.

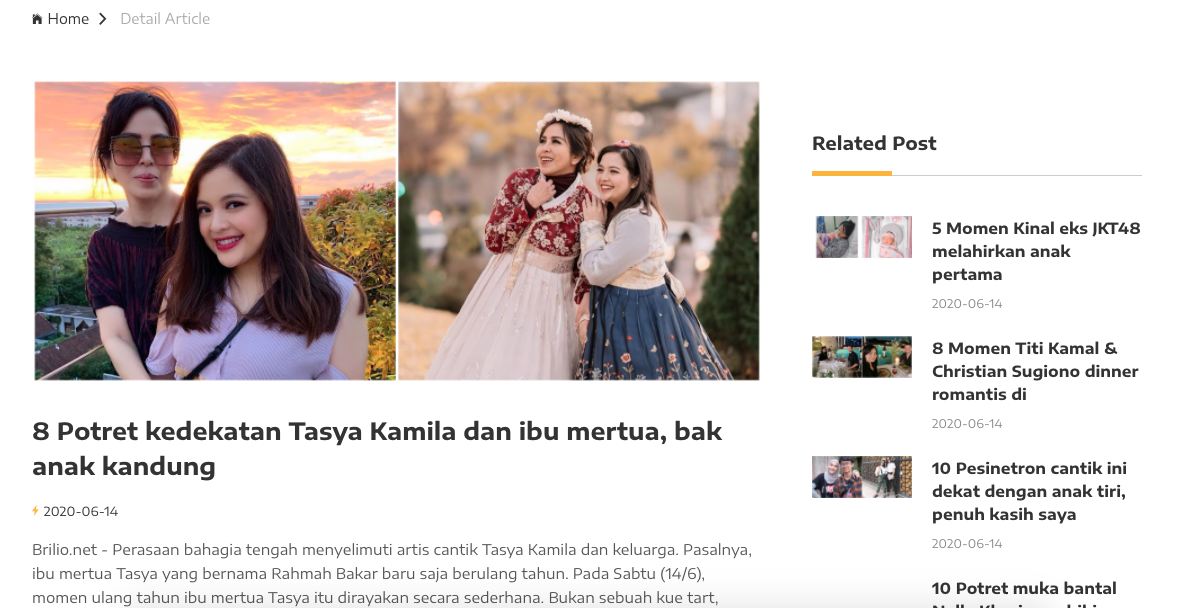


Gambar 5. 2 Homepage

Pada halaman awal menampilkan beberapa data berupa judul artikel, gambar dan tanggal terbit artikel. Saat judul di klik maka akan mengarah kehalaman selanjutnya yaitu halaman *detail* artikel.

### 5.5.2 *Detail Page*

Halaman *detail* menampilkan seluruh konten berupa teks yang telah di *crawling* sebelumnya dari situs brilio.net.

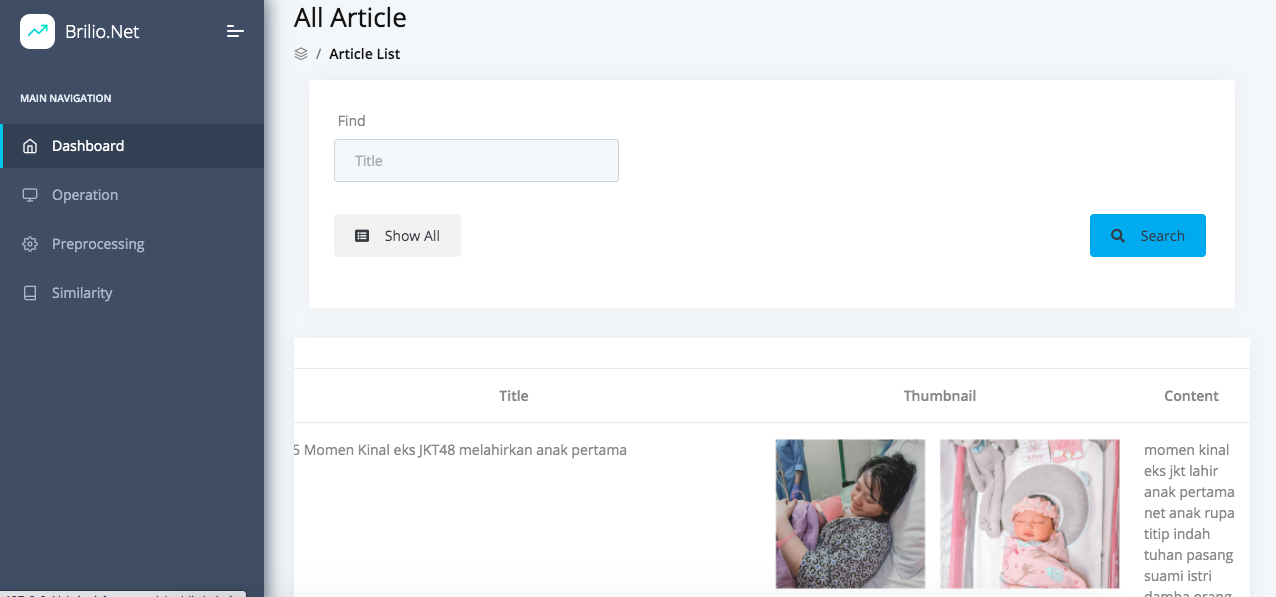


Gambar 5. 3 Detail page

Halaman ini memiliki dua bagian yaitu bagian kiri yang menampilkan data berupa gambar artikel, judul, tanggal dan isi konten artikel. Sedangkan pada bagian kanan menampilkan rekomendasi artikel hasil perhitungan yang telah diurutkan sesuai dengan hasil yang paling tinggi.

### 5.5.3 *Dashboard*

Halaman *dashboard* meampilkan keseluruhan artikel yang telah tersimpan didalam baisdata.

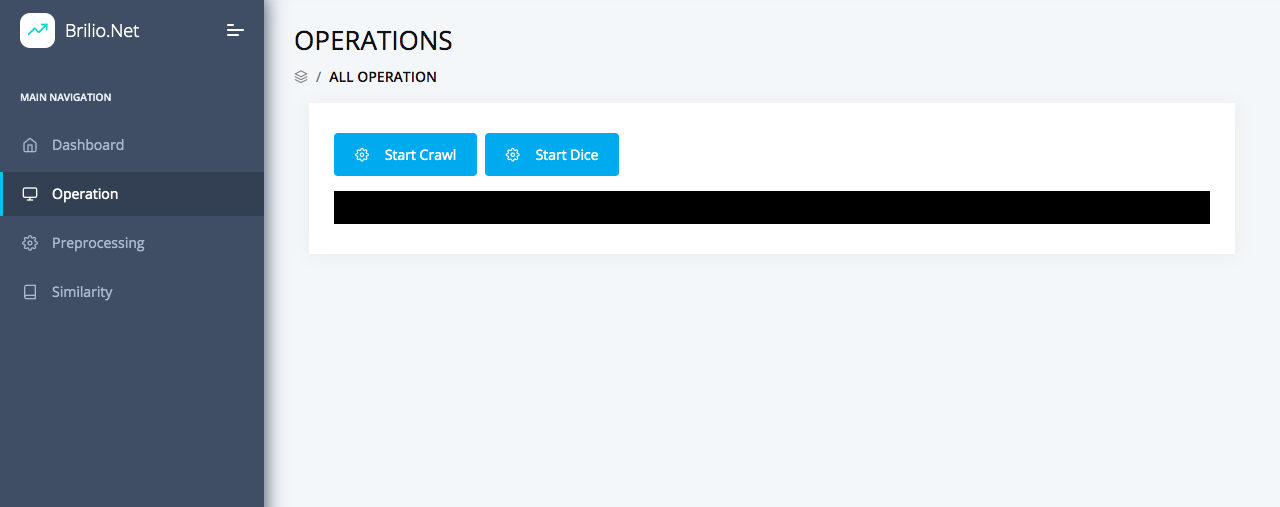


Gambar 5. 4 Dashboard

Halaman dashboard memiliki bagian untuk melakukan filter pencarian sesuai dengan judul yang akan dicari, sehingga dapat memudahakan pihak admin untuk melakukan pencarian artikel.

### 5.5.4 *Operation*

Pada halaman *operation* berguna untuk menjalankan perintah *crawling* dan perhitungan secara manual.

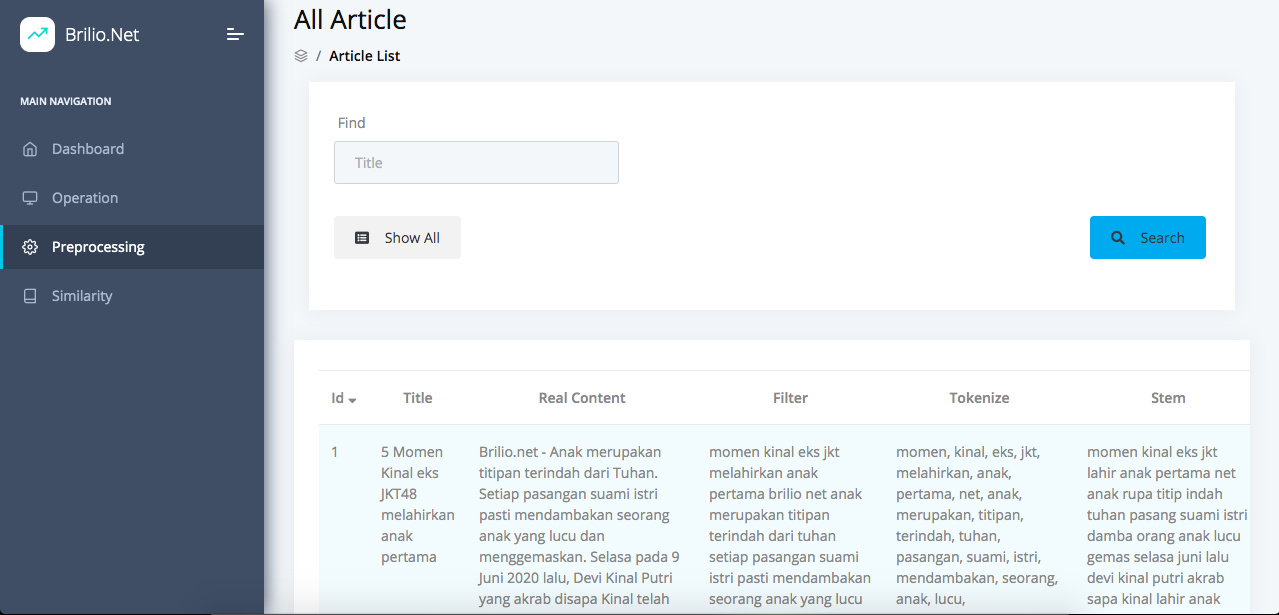


Gambar 5. 5 Opration

Halaman *Operation* memiliki sebuah kota berwarna hitam yang berguna untuk menampilkan hasil log dari perintah yang dijalankan, teradapat dua tombol yang masing-masing berfungsi untuk menjalankan *crawling* dan perhitungan secara manual.

### 5.5.5 *Preprocessing*

Halaman *preprocessing* menampilkan data hasil proses *preprocessing* yang telah tersimpan didalam basisdata.

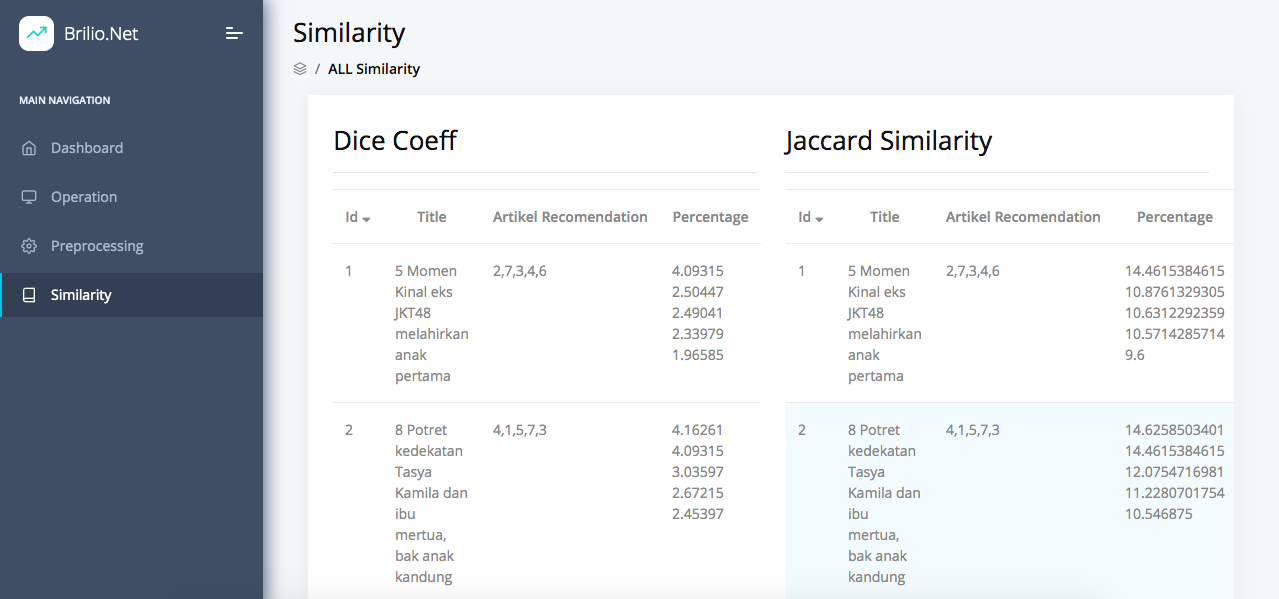


Gambar 5. 6 Preprocessing

Terdapat enam kolom yang akan ditampilkan secara urut Id artikel, judul, konten artikel hasil *crawling*, proses hasil filter, tokenize, dan yang terakhir adalah hasil proses stem.

### 5.5.6 *Similarity*

Halaman *similarity* menampilkan hasil perhitungan dari dua metode yaitu dice similarity dan jaccard similairy.



Gambar 5. 7 Similarity

Terdapat dua bagian yang membagi antar metode yaitu bagian kiri hasil perhitungan dari dice similarity, dan bagian kanan adalah hasil dari perhitungan dari jaccard similariy.

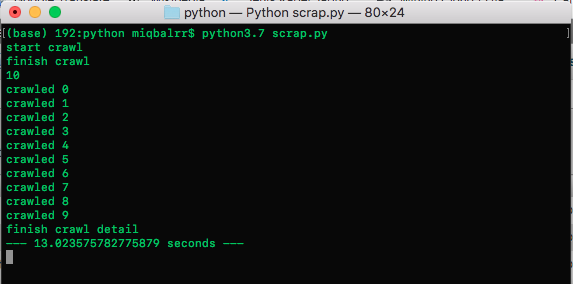
Masing-masing bagian menampilkan kolom diantaranya id dan judul artikel merupakan artikel pembanding, recomendasi id merupakan hasil rekomendasi yang sudah diurutkan dari hasil yang tertinggi, kolom similarity yang menampilkan hasil perhitungan berupa angka, dan kolom yang terakhir adalah kolom persentase yaitu hasil persentase similairy secara urut.

## 5.6 Pengujian Funsgional

Sistem rekomendasi artikel memiliki banyak fungsi yang dijalankan, sehingga pengujian dilakukan agar alur sistem secara keseluruhan dapat berjalan dengan baik.

### 5.6.1 *Pengujian crawling*

Funsgi *crawling* berguna untuk mengambil data artikel yang terdapat pada situs brilio.net, pengujian *crawling* dilakukan dengan cara mengambil data sebanyak sepuluh artikel.

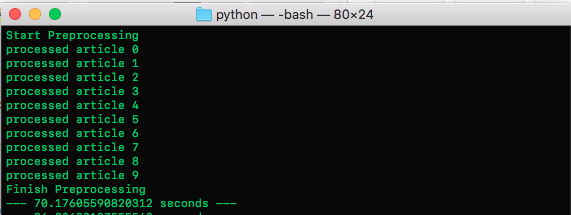


Gambar 5. 8 Pengujian crawling

Pengujian sukses dengan melakukan pengambilan sepuluh data artikel pertama pada situs brilio.net. waktu yang dibutuhkan untuk melakukan *crawling* dengan *default* *network caching* sebanyak13 - 15 detik bergantung kepada koneksi internet yang digunakan.

### 5.6.2 Pengujian *Preprocessing*

Data yang diuji pada fungsi *preprocessing* merupakan sepuluh data yang telah ter*crawling* sebelumnya dan telah tersimpan kedalam sebuah variabel.



Gambar 5. 9 pengujian preprocessing

Rentan waktu yang diberikan untuk *preprocessing* sepuluh data adalah sekitar 60 – 70 detik. hasil preprocessing akan tersimpan kedalam database dan tidak akan dilakukan *preprocessing* ulang untuk data yang telah tersimpan.

### 5.6.1 *Pengujian* Pembobotan TF IDF

Pengujian pembobotan TF IDF dilakukan dengan menguji hasil pembobotan dari artikel yang telah dilakukan *preprocessing* dengan adanya penyaringan yaitu untuk data dari 7 hari terakhir.

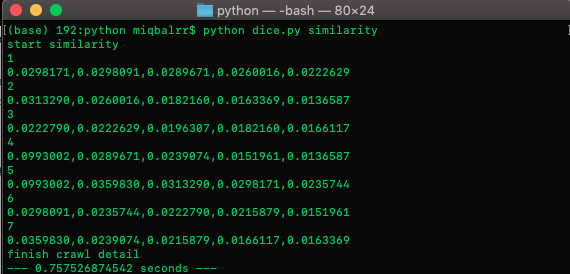


Gambar 5. 10 Pengujian pembobotan

Total artikel yang dilakukan pembobotan ada 8 artikel yang dibandingkan masing-masing dua artikel. Proses pembobotan menghabiskan rentan waktu antara 10 – 15 detik, bergantung pada jumlah artikel yang diproses.

### 5.6.2 Pengujian metode

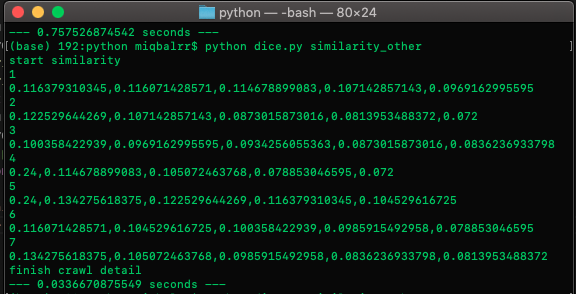
Pengujian metode dilakukan disetiap metode yang diterapkan dalam sistem.



Gambar 5. 11 pengujian metode dice

Dengan menggunakan metode dice similiarty hasil yang dperoleh untuk perhitungan kemiripan dari sepuluh artikel menghabiskan waktu ratarata 0.7 detik.

Pengujian dengane metode jaccard\_similarity

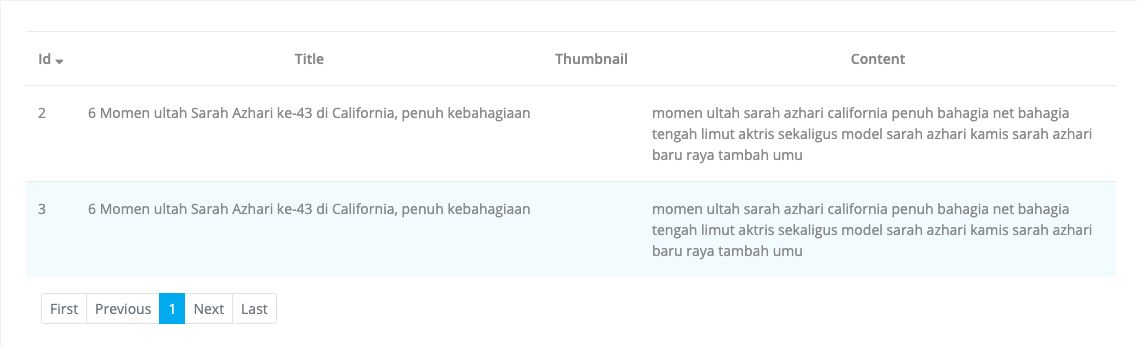


Gambar 5. 12 pengujian jaccard similairty

Berdasarkan hasil pengujian metode jaccard dengan untuk mencari tingkat kemiripan dari sepuluh artikel menghabiskan waktu rata-rata 0.03 detik.

## 5.5 Pengujian Akurasi

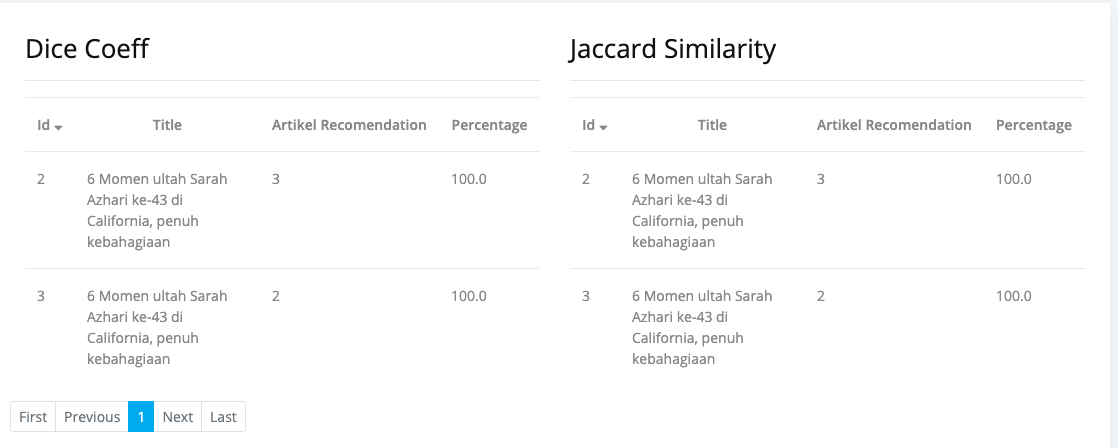
Proses pengujian akurasi dilakukan dengan cara membandingan dua artikel yang sama persis, dengan hasil yang harus dihasilkan adalah 100%. Artikel yang akan digunakan adalah artikel asli dan di duplikasi dengan mengubah id dan url yang berbeda saja. Karena id dan url merukpakan data unik disetiap baris dan juga tidak digunakan dalam proses perhitungan.



Gambar 5. 13 data duplikasi

Proses pengujian akurasi tidak memperhatikan waktu pengujian, dan hanya berfokus pada hasil perbandingan.

Proses pengujian akurasi akan berhasil saat hasil perhitungan dari data yang mirip bernilai 100%.



Gambar 5. 14 pengujian akurasi

Pada hasil pengujian akurasi metode dice dan jaccard menunjukkan hasil yang sama yaitu akurat dengan persentasi sesusai harapan yaitu 100% antara dua artikel yang sama persis.

# BAB VI. HASIL DAN PEMBAHASAN