APLIKASI PENDUKUNG DESAIN INTERIOR DENGAN SISTEM REKOMENDASI BERDASARKAN NAMA BRAND PERABOT MENGGUNAKAN ALGORITMA *CONTENT-BASED FILTERING* BERBASIS WEB

Aditya Raka Harischandra[1], M. Fikri Akbar Pratama[2], Felix[3], Albert Prima Laia[4]
Universitas Mikroskil, Jl. Thamrin No. 112, 124, 140, Telp. (061) 4573767, Fax. (061) 4567789
Fakultas Informatika, Program Studi Teknik Informatika, Universitas Mikroskil, Medan
[1]171112402@students.mikroskil.ac.id, [2]171112313@students.mikroskil.ac.id,
[3]felix.pandi@mikroskil.ac.id, [4]albert.laia@mikroskil.ac.id

Abstrak

Aplikasi perancangan desain interior dengan 3D Model memiliki kekurangan dalam memberikan panduan ide desain kepada pengguna baru dan tidak terdapat rekomendasi perabot yang dapat digunakan dalam desain berdasarkan selera pengguna. Sudah semestinya aplikasi 3D Model ruang menyediakan kemudahan bagi pengguna untuk mendapatkan ide desain dan rekomendasi perabot berdasarkan selera pengguna. Hal inilah yang melatarbelakangi penulis untuk memberikan solusi aplikasi pendukung desain interior. Penelitian ini bertujuan untuk membantu para pengguna agar menemukan ide desain ruangan yang sesuai dengan kebutuhan dan keinginan.

Algoritma *Text Preprocessing* adalah tahapan untuk mempersiapkan teks menjadi data yang akan mengalami pengolahan pada tahapan berikutnya. Sistem rekomendasi *Content-Based Filtering* dengan algoritma *Term Frequency – Inverse Document Frequency* (TF-IDF) digunakan untuk memberikan bobot pada tiap kata yang terdapat pada nama item perabot. Algoritma *Cosine Similarity* diterapkan untuk menemukan kemiripan antara nama item perabot berdasarkan hasil pembobotan pada metode *Term Frequency – Inverse Document Frequency* (TF-IDF), pada hasil akhir diharapkan dapat membantu pengguna untuk mendapatkan hasil rekomendasi dari perabot yang disukai melalui ide desain perabot yang telah disediakan.

Hasil pengujian dari algoritma ini adalah dengan menggunakan metode *confusion matrix* yang menunjukkan bahwa nilai *precision* 72%, *recall* 72%, *accuracy* 72%, dan *error rate* 51%, lalu berdasarkan hasil uji perangkat lunak menggunakan *Black-Box Testing* dapat ditarik kesimpulan bahwa perangkat lunak secara fungsional mengeluarkan hasil 96% sesuai dengan yang diharapkan.

Kata kunci: 3D Model, *Text Preprocessing*, TF-IDF, *Cosine Similarity*, CBF

Abstract

Interior design applications with 3D Models have shortcomings in providing design idea guidance to new users and there are no recommendations for furniture that can be used in designs based on user tastes. The 3D room model application should make it easy for users to get design ideas and furniture recommendations based on user tastes. This is the background of the author to provide application solutions to support interior design. This study aims to help users find room design ideas that suit their needs and desires.

The Text Preprocessing Algorithm is a step to prepare text into data that will undergo processing at the next stage. Content-Based Filtering recommendation system with Term Frequency – Inverse Document Frequency (TF-IDF) algorithm is used to give weight to each word contained in the name of the furniture item. The Cosine Similarity Algorithm is applied to find similarities between the names of furniture items based on the weighting results of the Term Frequency – Inverse Document Frequency (TF-IDF) method, in the result it is hoped that it can help users to get recommendations from the preferred furniture through the furniture design ideas that have been provided.

The test results of this algorithm are using the confusion matrix method which shows that the value of precision 72%, recall 72%, accuracy 72%, and error rate 51%, then based on the results of software testing using Black-Box Testing it can be concluded that the software functionally produces 96% results as expected.

Keywords: *3D Model, Text Preprocessing, TF-IDF, Cosine Similarity, CBF*

1. PENDAHULUAN

Desain interior adalah proses merencanakan, menata dan merancang ruang-ruang interior dalam bangunan. Tujuan desain interior adalah untuk memperbaiki fungsi, memperkaya nilai estetika dan meningkatkan aspek psikologis dari ruang interior tersebut [1]. Masalah yang sering terjadi pada aplikasi perancangan desain interior adalah tidak adanya panduan dalam menemukan ide desain dan tidak memiliki rekomendasi perabot sesuai selera pengguna. Penting untuk menentukan elemen pengisi berupa perabot yang dirasa perlu digunakan pada ruangan [2]. Desain yang baik adalah desain yang mengutamakan proses penyelesaian terhadap problematik yang terjadi dalam sebuah ruangan [3]. Oleh sebab itu diperlukan pengembangan aplikasi yang terfokus pada desain interior dimana dapat memberikan ide-ide gambar dalam desain kepada pengguna dan dilengkapi dengan fitur pendukung untuk mengekspresikan ide desain berupa simulasi desain *3D model*.

Selain pengembangan konsep simulasi dengan *3D model*, sistem rekomendasi adalah salah satu inovasi dalam revolusi tersebut. Sistem rekomendasi menyediakan pendekatan dalam memfasilitasi keinginan dari pengguna sistem dan secara rutin digunakan untuk menyarankan berdasarkan *history* perabot yang disukai pengguna dengan aksi *like* pada gambar [4]. Dalam upaya untuk memberi solusi pada kendala pemilihan perabot sesuai *brand*, maka solusi yang ditawarkan berupa sebuah sistem yang dapat merekomendasikan *item* interior berdasarkan *brand*. Pemilihan metode *Content-Based Filtering* merupakan metode yang paling cocok berdasarkan permasalahan diatas. Dimana *Content-Based Filtering* menggunakan ketersediaan konten yaitu *brand* sebagai basis dalam pemberian rekomendasi [5].

Metode yang digunakan untuk mendukung sistem rekomendasi *Content-Based Filtering* adalah *Term Frequency – Inverse Document Frequency* (TF-IDF) untuk memberikan bobot pada tiap kata yang terdapat pada nama *item*. Kemudian metode *Cosine Similarity* diterapkan untuk menemukan kemiripan antara nama *item* perabot berdasarkan hasil pembobotan pada metode *Term Frequency – Inverse Document Frequency* (TF-IDF). Alasan penggunaan metode *Term Frequency – Inverse Document Frequency* (TF-IDF) dan *Cosine Similarity*, karena merupakan metode pembobotan kata yang terkenal efisien, mudah dan memiliki hasil yang akurat. Sedangkan metode *Cosine Similarity* digunakan dengan alasan bahwa, metode ini mempunyai nilai akurasi yang tinggi dimana kelebihan utama dari metode *Cosine Similarity* adalah tidak terpengaruh pada panjang pendeknya suatu dokumen. Sehingga, dengan melakukan perbandingan *keyword* yang dihasilkan, maka kedekatan antara *item* juga dapat dipastikan [6].

2. TINJAUAN PUSTAKA

2.1 Text Processing

Text preprocessing adalah tahapan untuk mempersiapkan teks menjadi data yang akan mengalami pengolahan pada tahapan berikutnya. *Input*an awal pada proses ini adalah berupa dokumen utuh [7]. *Text preprocessing* pada penelitian ini terdiri dari beberapa tahapan, yaitu: proses *case folding*, proses *tokenizing*, proses *filtering*, dan proses *stemming*.

1. *Case folding*
   Case folding merupakan proses mengonversi semua huruf dalam suatu dokumen menjadi huruf kapital atau huruf kecil dan menghilangkan tanda baca untuk mempercepat proses komputasi.
2. *Tokenizing*

Tokenization merupakan proses memotong *string Input* menurut kata penyusunnya dengan memanfaatkan tanda baca koma sebagai pembatas atau *delimiter* antar kata dan menghilangkan angka, hasil dari *tokenization* adalah token atau *term*.

3. *Filtering*
   Filtering adalah proses menghilangkan *stopword* yang tidak penting, di dalam bahasa Indonesia *stopword* adalah kata penghubung dan dapat disebut sebagai kata tidak penting, misalnya di, oleh, pada, yang, dari dan lain sebagainya.
4. *Stemming*
   Stemming adalah proses mencari akar (*root*) kata dari tiap token kata yaitu dengan pengembalian suatu kata berimbuhan ke bentuk dasarnya (*stem*).

2.2 Content-Based Filtering

Sistem rekomendasi ini menggunakan ketersediaan konten (sering juga disebut dengan fitur, atribut atau karakteristik) sebuah *item* sebagai basis dalam pemberian rekomendasi [5]. Sebagai contoh, sebuah film mempunyai konten seperti *genre*, *author*, tahun rilis, dan lain-lain, atau sebuah *file* dokumen memiliki konten berupa tulisan yang ada di dalamnya. Sistem rekomendasi berbasis konten mencoba untuk melakukan pencocokan (*matching*) antara profil *user* (*user profile*) dengan konten *item* (*item content*).

Misalnya, seorang *user* telah memberi *rating* kepada 5 buah *item* $i1$, $i2$, $i3$, $i4$, $i5$ yang memiliki empat buah fitur $j1$, $j2$, $j3$, $j4$, $j5$. Angka 1 menandakan bahwa *item* tersebut memiliki atribut yang bersangkutan, sedangkan angka 0 menunjukkan bahwa *item* tersebut tidak memiliki atribut tersebut.

Tabel 1. Item Feature [8]

	Fitur j_1	Fitur j_2	Fitur j_3	Fitur j_4	*Rating*
Item i_1	1	1	1	1	4
Item i_2	1	0	1	1	3
Item i_3	1	0	0	0	2
Item i_4	1	0	1	1	3
Item i_5	1	1	0	1	3

Untuk menghitung bobot dari masing-masing atribut dapat dengan menggunakan rumus berikut [8].

$$w(u,jk) = \frac{1}{|Iu|} \sum_{i \in Iu} x\ (i,j)\ r(u,i) \quad (1)$$

Keterangan:
$w(u,j_k)$ = Bobot yang dimiliki oleh *user u* terhadap fitur j_k.
I_u = Satu set *item* yang telah di *rating* oleh *user u*.
$x(i,j)$ = Nilai kehadiran (angka 1 atau 0) sebuah fitur di dalam sebuah *item*.
$r(u,i)$ = *Rating* yang diberikan *user u* terhadap *item i*.

Sedangkan untuk menghitung prediksi *rating* yang akan diberikan seorang *user* terhadap sebuah *item* dapat menggunakan *formula* berikut ini [8].

$$R\ (u,i) = \frac{1}{|Di|} \sum_{j \epsilon Di} w\ (u,j) \quad (2)$$

Keterangan:
$R\ (u,i)$ = Prediksi *rating user u* terhadap *item i*.
Di = Fitur yang muncul di dalam *item i*.

2.3 Term Frequency – Inverse Document Frequency (TF-IDF)

Algoritma *Term Frequency – Inverse Document Frequency* (TF-IDF) merupakan algoritma yang berasal dari bidang *information retrieval*, namun saat ini semakin banyak digunakan dalam perbandingan dokumen [9]. Algoritma ini digunakan untuk menentukan bobot dari suatu *term* (kata) t, pada suatu dokumen d.

$$tfidf_{t.d} = tf_{t.d}\ .idf_t \quad (3)$$

Keterangan:

$tfidf_{t.d}$ = Bobot *term*
$tf_{t.d}$ = *Term frequency* t pada dokumen d
idf_t = *Inverse document frequency* t

Term Frequency (TF) adalah bobot dari suatu kata t, dalam suatu dokumen d dan dilambangkan dengan $tf_{t.d}$. Pendekatan paling sederhana dari konsep ini adalah dengan menyatakan bobot suatu kata t sebagai jumlah kemunculannya pada dokumen d.

$$tf_{t.d} = \log f_{t.d} + 1 \quad Jika\ f_{t.d} > 0 = jika\ kata\ tidak\ muncul \tag{4}$$

Keterangan:
$tf_{t.d}$ = *Term frequency*
$f_{t.d}$ = Jumlah kemunculan kata/*term* t di dalam dokumen d

Konsep *term frequency* memandang suatu dokumen sebagai *bag of words* (kantong kata) di mana urutan dari kemunculan suatu kata diabaikan dan hanya jumlah kemunculan dari kata itu saja yang penting.

Konsep *Inverse Document Frequency* (IDF) dibuat untuk mengurangi efek dari kata yang frekuensinya terlalu tinggi dalam kumpulan dokumen. Ide dasarnya adalah untuk menurunkan bobot dari kata dengan frekuensi kolektif (frekuensi total kemunculan kata di semua dokumen) yang tinggi. Dengan kata lain, semakin banyak dokumen kata tersebut pada suatu kumpulan dokumen, maka semakin rendah bobotnya.

$$idf_t = \log \frac{N}{N_t} \tag{5}$$

Keterangan:
idf_t = *Inverse document frequency*
N = Jumlah keseluruhan dokumen
N_t = Jumlah dokumen yang memuat *term* t

2.4 Cosine Similarity

Cosine Similarity merupakan metode pengukuran yang banyak digunakan di *pattern recognition* dan *text classification* [10]. *Cosine Similarity* mengukur kemiripan dua buah vektor dalam sebuah *product space* dengan mengukur *cosine* dari sudut kedua vektor [11]. Rumus perhitungan *cosine similarity* sebagai berikut:

$$Cosine(\vec{x}, \vec{y}) = \frac{\vec{x}.\vec{y}}{[\vec{x}][\vec{y}]} \tag{6}$$

Keterangan:
$\vec{x}$ = Vektor X yang dibentuk dari dokumen X
$\vec{y}$ = Vektor Y yang dibentuk dari dokumen Y
$[\vec{x}]$ = Panjang vektor X
$[\vec{y}]$ = Panjang vektor Y

Kemiripan antar dokumen dihitung menggunakan suatu fungsi ukuran kemiripan (*similarity measure*). Semakin besar hasil fungsi *similarity*, maka kedua objek yang dievaluasi semakin mirip demikian pula sebaliknya. Ukuran ini memungkinkan kedudukan dokumen sesuai dengan kemiripannya (relevansi) terhadap *query*. Kualitas hasil dari dokumen yang didapatkan sangat tergantung pada fungsi *similarity* yang digunakan.

2.5 Confusion Matrix

Confusion Matrix adalah sebuah metode yang biasa digunakan untuk perhitungan akurasi. Dalam pengujian keakuratan hasil pencarian akan dievaluasi nilai *recall*, *precision*, *accuracy*, dan *error rate*. Dimana *precision* mengevaluasi kemampuan sistem untuk menemukan peringkat yang paling relevan, dan didefinisikan sebagai presentasi dokumen yang di *retrieve* dan benar-benar relevan terhadap *query*. *Recall* mengevaluasi kemampuan sistem untuk menemukan semua *item* yang relevan dari koleksi

dokumen dan didefinisikan sebagai presentasi dokumen yang relevan terhadap *query*. *Accuracy* merupakan perbandingan kasus yang diidentifikasi benar dengan jumlah seluruh kasus dan *error rate* merupakan kasus yang diidentifikasi salah dengan jumlah seluruh kasus [6].

Tabel 2. Rumus *confusion matrix*

Dokumen	Nilai Sebenarnya	
	Relevant	*Nonrelevant*
Retrieved	*True positive (tp) Correct result*	*False positive (fp) Unexpected result*
Not retrieved	*False negative (fn) Missing result*	*True negative (tn) Correct absence of result*

Keterangan:
TP (*True Positive*) = Jumlah prediksi yang benar dari data yang relevan
FP (*False Positive*) = Jumlah prediksi yang salah dari data yang tidak relevan
FN (*False Negative*) = Jumlah prediksi yang salah dari data yang tidak relevan
TN (*True Negative*) = Jumlah prediksi yang benar dari data yang relevan

Sehingga rumus *confusion matrix* adalah sebagai berikut:

$$Precision = \frac{tp}{tp + fp} \quad (7)$$

$$Recall = \frac{tp}{tp + fn} \quad (8)$$

$$Accuracy = \frac{(tp + tn)}{(tp + fp + tn + fn)} \quad (9)$$

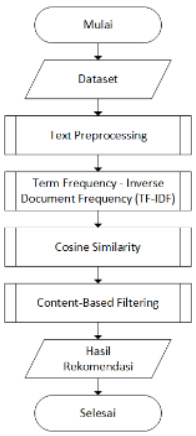
$$Error\ rate = \frac{(fn + tn)}{(tp + fp + tn + fn)} \quad (10)$$

Sistem yang dikatakan baik adalah sistem yang memiliki nilai *recall* dan *precision* tinggi.

3. METODE PENELITIAN

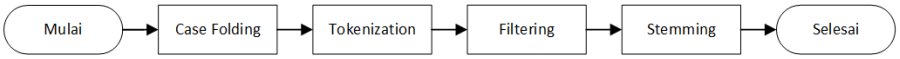
3.1 Analisis Proses

Analisis proses digunakan untuk menjelaskan proses kerja pada perangkat lunak untuk menyelesaikan permasalahan yang ada, meliputi tahapan *Text Preprocessing*, pembobotan dengan *Term Frequency-Inverse Document Frequency* (TF-IDF), mengidentifikasi kemiripan dengan *Cosine Similarity,* dan hasil akhir rekomendasi menggunakan *Content-Based Filtering* (CBF).



Gambar 1. *Flowchart* dari Analisis proses

3.1.1 Text Preprocessing



Gambar 2. Tahap *Text Preprocessing*

Pada tahapan *text preprocessing* pada penelitian ini akan menggunakan *dataset* berbentuk .csv lalu di-*input* ke *database,* dengan mengambil sampel sebanyak 5 (lima) *item* pertama dari *Nama_Item* pada *database* yang telah dipersiapkan sebelumnya.

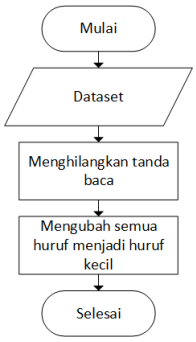
Tabel 3. *Load Dataset*

	Nama_Item
0	Heim Studio Oda Sofa Bed biru
1	Heim Studio Hara Sofa 2 Dudukan Biru
2	Heim Studio Hiro Sofa 3 Seater Abu-abu
3	Heim Studio Saki Sofa Bed Abu-Abu Muda
4	Ezma Olaf Sofa 3 Dudukan Biru

Tahapan *text preprocessing* meliputi:

1. *Case Folding*

   Pada penelitian ini, semua huruf dikonversi menjadi huruf kecil. Proses yang berjalan saat tahap *case folding*, akan digambarkan dalam *flowchart* berikut:



Gambar 3. *Flowchart Case Folding*

Gambar 3 di atas menggambarkan proses yang dilakukan saat tahap *case folding*, dimana data pada *database* dilakukan penghilangan tanda baca dan mengubah semua huruf menjadi huruf kecil.

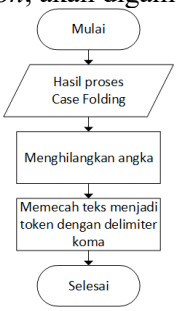
Tabel 4. Tahap *Case Folding*

	Nama_Item
0	heim studio oda sofa bed biru
1	heim studio hara sofa 2 dudukan biru
2	heim studio hiro sofa 3 seater abu-abu
3	heim studio saki sofa bed abu-abu muda
4	ezma olaf sofa 3 dudukan biru

Tabel 4 di atas adalah hasil dari tahap *case folding* telah dilakukan pada *database*. Dimana semua tanda baca telah dihilangkan dan semua huruf telah diubah menjadi huruf kecil.

2. *Tokenization*

   Proses yang berjalan saat tahap *tokenization*, akan digambarkan dalam *flowchart* berikut:



Gambar 4. *Flowchart Tokenization*

Gambar 4 di atas menggambarkan proses yang dilakukan saat tahap *tokenization*, dimana hasil dari proses *case folding* yang dilakukan sebelumnya menjadi data yang akan dilakukan proses penghilangan angka dan pemecahan teks menjadi token dengan *delimiter* koma.

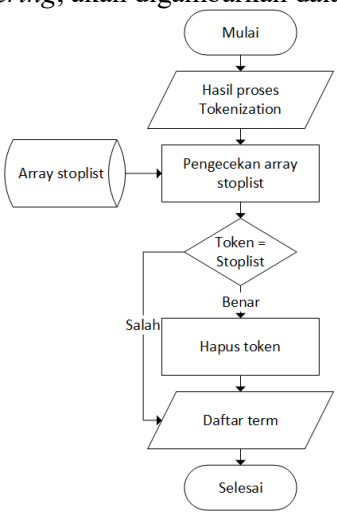
Tabel 5. Tahap *Tokenization*

	Nama_Item
0	[heim, studio, oda, sofa, bed, biru]
1	[heim, studio, hara, sofa, dudukan, biru]
2	[heim, studio, hiro, sofa, seater, abuabu]
3	[heim, studio, saki, sofa, bed, abuabu, muda]
4	[ezma, olaf, sofa, dudukan, biru]

Tabel 5 di atas adalah hasil dari tahap *tokenization* telah dilakukan pada *dataset*. Dimana semua angka telah dihilangkan dan semua kata telah menjadi token dengan *delimiter* koma.

3. *Filtering*

Proses yang berjalan saat tahap *filtering*, akan digambarkan dalam *flowchart* berikut:



Gambar 5. *Flowchart Filtering*

Gambar 5 di atas menggambarkan proses yang dilakukan saat tahap *stopword removal* atau *filtering*, dimana hasil dari proses *tokenizing* yang dilakukan sebelumnya, akan di cocokkan dengan *array stoplist* yang ada, apabila token yang dicek merupakan *stoplist* maka token akan dihapus, apabila token bukan termasuk *stoplist* maka token akan dibiarkan tetap ada.

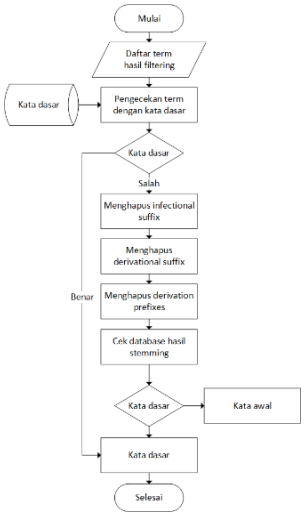
Tabel 6. Tahap *Filtering*

	Nama_Item
0	[heim, studio, oda, sofa, bed, biru]
1	[heim, studio, hara, sofa, dudukan, biru]
2	[heim, studio, hiro, sofa, seater, abuabu]
3	[heim, studio, saki, sofa, bed, abuabu, muda]
4	[ezma, olaf, sofa, dudukan, biru]

Tabel 6 di atas adalah hasil dari tahap *filtering* atau *stopword removal* telah dilakukan pada *dataset*. Dimana semua token yang merupakan *stoplist* maka token tersebut telah terhapus.

4. *Stemming*

Proses yang berjalan saat tahap *stemming*, akan digambarkan dalam *flowchart* berikut:



Gambar 6. *Flowchat Stemming*

Gambar 6 di atas menggambarkan proses yang dilakukan saat tahap *stemming*, dimana *term* hasil dari proses sebelumnya dilakukan pengecekan terhadap *database* kata dasar apakah *term* sudah kata dasar atau kata berimbuhan. Apabila *term* merupakan kata berimbuhan maka akan dilakukan *stemming* dengan melalui 3 (tiga) tahapan yaitu menghapus *inflectional suffix*, menghapus *derivation suffix*, dan menghapus *derivation prefix.*

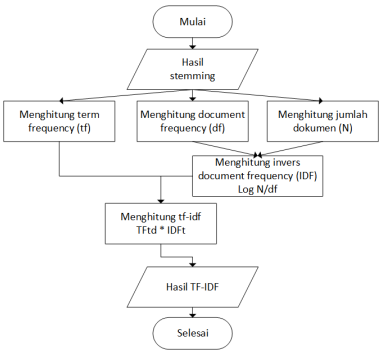
Tabel 7. Tahap *Stemming*

	Nama_Item
0	[heim, studio, oda, sofa, bed, biru]
1	[heim, studio, hara, sofa, duduk, biru]
2	[heim, studio, hiro, sofa, seater, abuabu]
3	[heim, studio, saki, sofa, bed, abuabu, muda]
4	[ezma, olaf, sofa, duduk, biru]

Tabel 7 di atas adalah hasil dari tahap *stemming* telah dilakukan pada *database*. Dimana tahap *stemming* dilakukan pada *Nama_Item*. Setelah itu hasil *stemming* disimpan ke *variable* yang sama. Hasil dari proses *stemming* ini akan dilanjutkan dengan tahap berikutnya untuk dilakukan pembobotan kata menggunakan algoritma *Term Frequency-Inverse Document Frequency* (TF-IDF).

3.1.2 Term Frequency – Inverse Document Frequency (TF-IDF)

Pada proses pembobotan kata *Term Frequency-Inverse Document Frequency* (TF-IDF) akan digambarkan dalam *flowchart* berikut:



Gambar 7. *Flowchart Term Frequency-Inverse Document Frequency* (TF-IDF)

Gambar 7 di atas menggambarkan tahap pembobotan kata dengan menggunakan metode *Term Frequency-Inverse Document Frequency* (TF-IDF), dimana daftar *term* hasil *stemming* dilakukan

perhitungan untuk mengetahui bobot per-kata dengan menghitung jumlah *Term Frequency* (TF) dokumen terlebih dahulu, kemudian menghitung nilai *Document Frequency* (DF), dan selanjutnya menghitung nilai *Inverse Document Frequency* (IDF), dimana *N* merupakan jumlah seluruh dokumen yang ada. Setelah nilai TF dan IDF sudah didapat, maka langkah terakhir adalah menentukan bobot kata dengan mengalikan TF dan IDF. Hasil dari proses perhitungan ini akan dilanjutkan pada tahap berikutnya untuk dilakukan perhitungan *Cosine Similarity* yang merupakan tahap menghitung kesamaan antar dokumen.

Berikut adalah dokumen yang akan diberi pembobotan dengan metode *Term Frequency-Inverse Document Frequency* (TF-IDF).

Tabel 8. Sampel dokumen

Dokumen	Term Yang Mewakili Dokumen
Item1	heim studio oda sofa bed biru
Item2	heim studio hara sofa duduk biru
Item3	heim studio hiro sofa seater abuabu
Item4	heim studio saki sofa bed abuabu muda
Item5	ezma olaf sofa duduk biru

Pada tabel 8 adalah dokumen sampel untuk dilakukan tahapan pembobotan yang menggunakan 5 (lima) *item* pertama dari *Nama_Item* pada *dataset* sebagai dokumen. Berikut adalah sebagai *query* adalah *Item1* dari dokumen.

Tabel 9. Dokumen query

Q	heim studio oda sofa bed biru

Pada tabel 9 adalah *query* (Q) *"heim studio oda sofa bed biru"* yang akan menjadi penentu daftar dari 5 (lima) *item* dokumen yang paling relevan dengan *query* tersebut.

Pembobotan kata dimulai dari mencari nilai TF yaitu menghitung jumlah kata yang muncul pada suatu dokumen. Contohnya kata "heim" muncul empat kali pada *item1*, *item2*, *item3* dan *item4*. Dari nilai TF itu akan didapatkan nilai DF yang didasarkan pada jumlah kata muncul pada semua dokumen. Kata "heim" muncul pada *item1*, *item2*, *item3* dan *item4* maka didapatkan nilai DF adalah 4.

Tabel 10. Perhitungan TF & DF

	tf						
Term	*Q*	Item1	Item2	Item3	Item4	Item5	*df*
heim	1	1	1	1	1	0	4
studio	1	1	1	1	1	0	4
oda	1	1	0	0	0	0	1
sofa	1	1	1	1	1	1	5
bed	1	1	0	0	1	0	2
biru	1	1	1	0	0	1	3
hara	0	0	1	0	0	0	1
duduk	0	0	1	0	0	1	2
hiro	0	0	0	1	0	0	1
seater	0	0	0	1	0	0	1
abuabu	0	0	0	1	1	0	2
saki	0	0	0	0	1	0	1
muda	0	0	0	0	1	0	1
ezma	0	0	0	0	0	1	1
olaf	0	0	0	0	0	1	1

Pada tabel 10 merupakan banyaknya frekuensi kemunculan kata pada *dataframe Nama_Item*. Contohnya kata "*heim*" muncul sebanyak empat kali pada *Item1*, *Item2*, *Item3* dan *Item4*. Dari nilai TF itu akan didapatkan nilai DF yang didasarkan pada jumlah kata muncul pada semua *item*. Kata "*heim*" muncul pada *Item1*, *Item2*, *Item3* dan *Item4* maka didapatkan nilai DF adalah 4 (empat).

Contoh menghitung nilai IDF dari *term* "heim" yaitu dengan menggunakan rumus (11).

$$idf_t = \log\frac{5}{4} = 0{,}097 \tag{11}$$

Tabel 11. Perhitungan IDF

	tf								idf
Term	Q	Item1	Item2	Item3	Item4	Item5	df	d/df	log(n/df)
heim	1	1	1	1	1	0	4	1,25	0,097
studio	1	1	1	1	1	0	4	1,25	0,097
oda	1	1	0	0	0	0	1	5	0,699
sofa	1	1	1	1	1	1	5	1	0,000
bed	1	1	0	0	1	0	2	2,5	0,398
biru	1	1	1	0	0	1	3	1,666667	0,222
hara	0	0	1	0	0	0	1	5	0,699
duduk	0	0	1	0	0	1	2	2,5	0,398
hiro	0	0	0	1	0	0	1	5	0,699
seater	0	0	0	1	0	0	1	5	0,699
abuabu	0	0	0	1	1	0	2	2,5	0,398
saki	0	0	0	0	1	0	1	5	0,699
muda	0	0	0	0	1	0	1	5	0,699
ezma	0	0	0	0	0	1	1	5	0,699
olaf	0	0	0	0	0	1	1	5	0,699

Selanjutnya dari nilai IDF tersebut akan dicari lagi nilai TF-IDF yaitu perkalian antara TF dan IDF. Dengan rumus sebagai berikut:

$$tfidf_{t.d} = tf_{t.d} \,.\, idf_t \tag{112}$$

Tabel 12. Perhitungan TF-IDF

	tf								idf	tfidf					
Term	Q	Item1	Item2	Item3	Item4	Item5	df	d/df	log(n/df)	Q	Item1	Item2	Item3	Item4	Item5
heim	1	1	1	1	1	0	4	1,25	0,097	0,097	0,097	0,097	0,097	0,097	0,000
studio	1	1	1	1	1	0	4	1,25	0,097	0,097	0,097	0,097	0,097	0,097	0,000
oda	1	1	0	0	0	0	1	5	0,699	0,699	0,699	0,000	0,000	0,000	0,000
sofa	1	1	1	1	1	1	5	1	0,000	0,000	0,000	0,000	0,000	0,000	0,000
bed	1	1	0	0	1	0	2	2,5	0,398	0,398	0,398	0,000	0,000	0,398	0,000
biru	1	1	1	0	0	1	3	1,666667	0,222	0,222	0,222	0,222	0,000	0,000	0,222
hara	0	0	1	0	0	0	1	5	0,699	0,000	0,000	0,699	0,000	0,000	0,000
duduk	0	0	1	0	0	1	2	2,5	0,398	0,000	0,000	0,398	0,000	0,000	0,398
hiro	0	0	0	1	0	0	1	5	0,699	0,000	0,000	0,000	0,699	0,000	0,000
seater	0	0	0	1	0	0	1	5	0,699	0,000	0,000	0,000	0,699	0,000	0,000
abuabu	0	0	0	1	1	0	2	2,5	0,398	0,000	0,000	0,000	0,398	0,398	0,000
saki	0	0	0	0	1	0	1	5	0,699	0,000	0,000	0,000	0,000	0,699	0,000
muda	0	0	0	0	1	0	1	5	0,699	0,000	0,000	0,000	0,000	0,699	0,000
ezma	0	0	0	0	0	1	1	5	0,699	0,000	0,000	0,000	0,000	0,000	0,699
olaf	0	0	0	0	0	1	1	5	0,699	0,000	0,000	0,000	0,000	0,000	0,699

Setelah hasil pembobotan TF-IDF selesai dilakukan, didapatkanlah hasil seperti tabel 12 diatas. Selanjutnya dari hasil TF-IDF tersebut dilakukan perangkingan untuk mencari dokumen mana yang paling relevan dengan *query* Q.

Tabel 13. Nilai TF-IDF *query* Q

Q	Nilai
Item1	1,5126

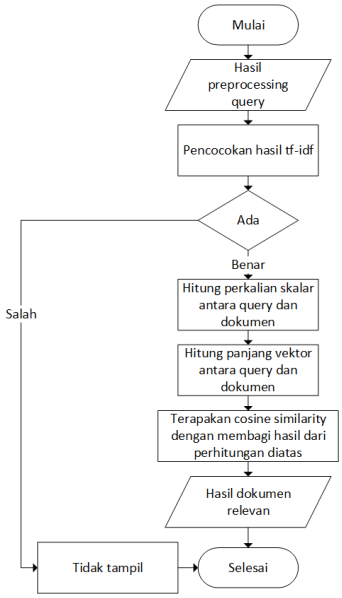
Tabel 14. Hasil perangkingan tiap dokumen terhadap *query* Q

Rangking		Nilai
1	Item1	1,5126
2	Item4	0,5918
3	Item2	0,4157
4	Item5	0,2218
5	Item3	0,1938

Tabel 14 merupakan hasil perangkingan dari tiap dokumen yang paling relevan berdasarkan bobot TF-IDF dengan bobot TF-IDF *query*. Dimana *Item1* merupakan *item* yang paling relevan diikuti *Item4*, *Item2*, *Item5* dan *Item3*, dikarenakan *item1* merupakan *item query* Q.

3.1.3 Cosine Similarity

Pada proses menghitung kemiripan antar dokumen dengan metode *Cosine Similarity* akan digambarkan dalam *flowchart* berikut:



Gambar 8. *Flowchart Cosine Similarity*

Gambar 8 di atas menggambarkan proses untuk menemukan dokumen yang relevan dengan *query user* menggunakan metode *Cosine Similarity*, dimana *query* yang dimasukkan *user* dilakukan tahap *preprocessing* yang hasilnya dicocokkan dengan hasil perhitungan *Term Frequency-Inverse Document Frequency* (TF-IDF), apabila *term* ditemukan maka akan diproses ke tahap selanjutnya untuk mendapatkan dokumen yang relevan.

Tahap pertama yaitu menentukan perkalian skalar antara *query Q* dan 5 dokumen *item* tersebut. Hasil perkalian dari setiap dokumen dengan *query Q* lalu ditotalkan.

Tabel 15. Perkalian tiap dokumen terhadap *query* Q

	Q*WDi				
	Item1	Item2	Item3	Item4	Item5
	0,009	0,009	0,009	0,009	0
	0,009	0,009	0,009	0,009	0
	0,489	0	0	0	0
	0	0	0	0	0
	0,158	0	0	0,158	0
	0,049	0,049	0	0	0,049
	0	0	0	0	0
	0	0	0	0	0
	0	0	0	0	0
	0	0	0	0	0
	0	0	0	0	0
	0	0	0	0	0
	0	0	0	0	0
	0	0	0	0	0
Total	0,71	0,07	0,02	0,18	0,05

Tabel 15 adalah perkalian antara *query Q* dengan dokumen *item*. Dan total merupakan hasil penjumlah perkalian *query Q* dengan dokumen *item*. Selanjutnya adalah menentukan panjang vektor pada setiap dokumen, termasuk *query Q*. Dengan cara mengkuadratkan bobot setiap *term* dalam semua dokumen, lalu jumlahkan dan terakhir lakukan pengakaran.

Tabel 16. Panjang vektor pada tiap dokumen

	Panjang Vektor					
	Q	Item1	Item2	Item3	Item4	Item5
	0	0,009	0,009	0,009	0,009	0
	0	0,009	0,009	0,009	0,009	0
	0	0,489	0	0	0	0
	0	0	0	0	0	0
	0	0,158	0	0	0,158	0
	0,049	0,049	0,049	0	0	0,049
	0	0	0,489	0	0	0
	0	0	0,158	0	0	0,158
	0	0	0	0,489	0	0
	0	0	0	0,489	0	0
	0	0	0	0,158	0,158	0
	0	0	0	0	0,489	0
	0	0	0	0	0,489	0
	0	0	0	0	0	0,489
	0	0	0	0	0	0,489
Total	0,715	0,715	0,715	1,154	1,313	1,185
Akar	0,846	0,846	0,846	1,074	1,146	1,088

Tabel 16 adalah hasil pencarian panjang vektor pada setiap dokumen. Dari hasil perkalian antara *query Q* dan dokumen serta hasil panjang vektor yang didapatkan maka tahap selanjutnya adalah menghitung kemiripan *query Q* dengan semua dokumen *item* yang ada dengan menerapkan rumus *Cosine Similarity* diatas.

Tabel 17. Nilai *Cosine Similarity* tiap dokumen terhadap *query* Q

Cosine Similarity with Q	
Cos(Q,Item1)	1,000000
Cos(Q,Item2)	0,095116
Cos(Q,Item3)	0,020677
Cos(Q,Item4)	0,182860
Cos(Q,Item5)	0,053479

Tabel 17 adalah hasil *Cosine Similarity* antara *query Q* dengan dokumen *item* yang ada. Lalu lakukan perangkingan untuk mencari dokumen mana yang paling relevan dengan *query Q*.

Tabel 18. Nilai *Cosine Similarity query Q*

Q	Nilai
Item1	1,0000

Tabel 18 merupakan nilai kemiripan dokumen *Cosine Similarity* dari *query Q*.

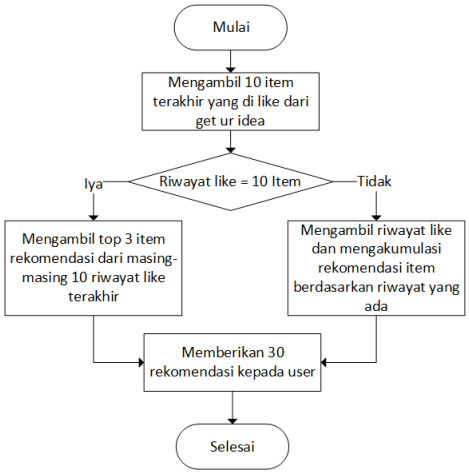
Tabel 19. Hasil perangkingan tiap dokumen terhadap *query* Q

Rangking		Nilai
1	Item1	1,000000
2	Item4	0,182860
3	Item2	0,095116
4	Item5	0,053479
5	Item3	0,020677

Tabel 19 merupakan hasil perangkingan dari dokumen yang paling relevan dengan *query*. Dimana *Item1* merupakan *item* yang paling relevan diikuti *Item4*, *Item2*, *Item5* dan *Item3*, dikarenakan *item1* merupakan *item query Q*.

3.1.4 Content-Based Filtering

Skenario rekomendasi dapat dilihat pada gambar 9 di bawah.



Gambar 9. *Flowchart* Skenario Rekomendasi

Pada skenario di atas, sistem memberikan rekomendasi berdasarkan *like* yang diberikan *user* terhadap ide-ide desain pada menu *Get Your Idea*. Dimana *like* yang diberikan akan dijadikan *query*. Pada gambar 9, skenario yang digunakan adalah sebagai berikut:

1. Sistem mengambil 10 *like* terakhir pada *Get Your Idea*.
2. Sistem mengambil *top* 3 rekomendasi pada 10 *query* atau *like* tersebut.
3. Jika riwayat *like* kurang dari 10, maka sistem mengakumulasi rekomendasi berdasarkan riwayat yang ada.

4. Sistem memberikan 3 rekomendasi pada tiap *query* atau *like* pada *user*.

Tabel 20 Hasil rekomendasi CBF

Hasil Rekomendasi		
Top	Dokumen	Term Yang Mewakili Dokumen
1	Item1	heim studio oda sofa bed biru
2	Item4	heim studio saki sofa bed abuabu muda
3	Item2	heim studio hara sofa duduk biru
4	Item5	ezma olaf sofa duduk biru
5	Item3	heim studio hiro sofa seater abuabu

Query	heim studio oda sofa bed biru

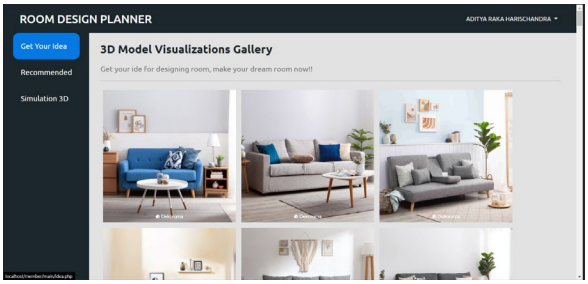
Tabel 20 adalah hasil rekomendasi yang didapatkan dari *query* dengan hasil rekomendasi urutan pertama adalah *Item1*, urutan kedua adalah *Item4*, urutan ketiga adalah *Item2*, urutan keempat adalah *Item5* dan terakhir adalah *Item3*. Rekomendasi yang akan diberikan kepada *user* adalah *top* 3 *item* dari tabel di diatas, yaitu *item1*, *item4* dan *item2*.

4. HASIL DAN PEMBAHASAN

4.1 Hasil

Hasil penelitian ini adalah sebuah sistem berupa *website* yang didukung oleh sistem rekomendasi *Content Based Filtering* dengan penerapan algoritma *Term Frequency – Inverse Document Frequency* (TF-IDF) dan *Cosine Similarity*.

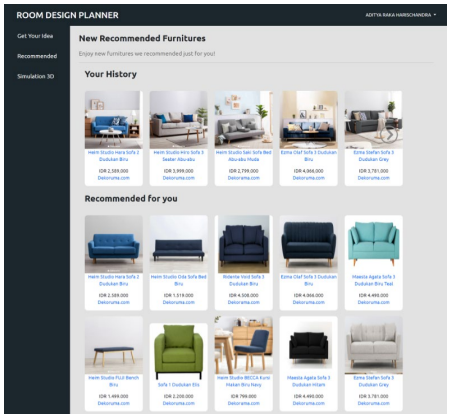
4.1.1 Hasil Tampilan Get Your Idea (Query)



Gambar 10 Halaman *Get Your Idea*

Merupakan tampilan yang dapat digunakan oleh *member* untuk mencari referensi ide gambaran konsep desain ruangan berdasarkan perabot dan dapat melakukan aksi *like* pada tiap gambar yang disukai.

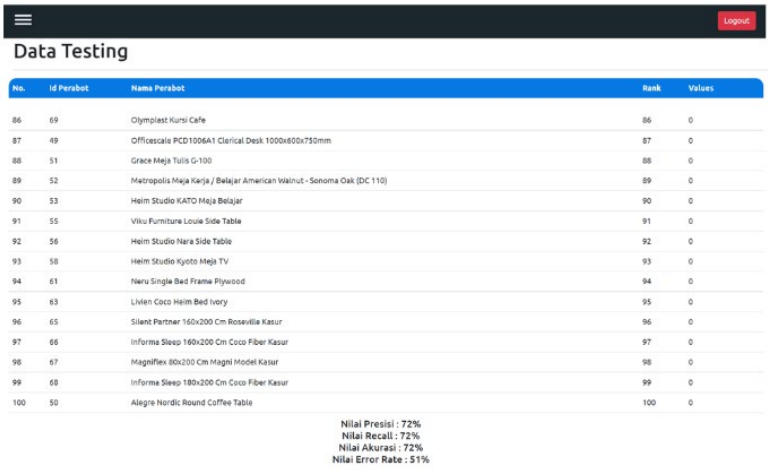
4.1.2 Hasil Tampilan Rekomendasi



Gambar 11 Halaman Rekomendasi

Merupakan tampilan yang dapat digunakan oleh *member* untuk mendapatkan sejumlah rekomendasi perabot berdasarkan *history like* yang sebelumnya dilakukan *member* pada halaman *get your idea*.

4.1.3 Hasil Perhitungan Confusion Matrix



Gambar 12 Halaman Pengujian Akurasi

Merupakan tampilan yang dapat digunakan oleh *admin* untuk melihat hasil nilai keakuratan menggunakan *confusion matrix* pada sistem rekomendasi.

4.2 Pembahasan

Dalam fase *testing* ini untuk melakukan pengujian sistem adalah menggunakan pengujian *Confusion Matrix* yang biasa digunakan dalam perhitungan akurasi pada suatu sistem temu kembali informasi untuk mengevaluasi seberapa baik kemampuan sistem dalam pencarian dokumen.

Tabel 21 Nilai *Confusion Matrix*

Dokumen	Nilai Sebenarnya	
	Relevant	*Nonrelevant*
Retrieved	*True positive (tp)* 49	*False positive (fp)* 19
Not retrieved	*False negative (fn)* 19	*True negative (tn)* 51

Tabel diatas menunjukkan bahwa:

1. Perabot hasil rekomendasi yang termasuk ke dalam kategori yang cocok dengan *query* ada 49.
2. Perabot hasil rekomendasi tidak termasuk pada kategori yang cocok dengan *query* ada 19.
3. Perabot yang termasuk dalam kategori cocok dengan *query* tetapi tidak ditampilkan sebagai rekomendasi sebanyak 19.
4. Perabot yang sama sekali tidak termasuk dalam hasil rekomendasi maupun kategori yang cocok dengan *query* ada 51.

Selanjutnya ketika tabel didapat, maka hasil pengujian *Confusion Matrix* sesuai dengan pengujian keakuratan hasil pencarian akan dievaluasi nilai *recall*, *precision*, *accuracy*, dan *error rate*. Berikut hasil *Confusion Matrix* berdasarkan nilai *recall*, *precision*, *accuracy*, dan *error rate*:

$$Precision = \frac{49}{49 + 41} = 0{,}720588235 \quad (13)$$

$$Recall = \frac{49}{49 + 19} = 0{,}720588235 \quad (14)$$

$$Accuracy = \frac{(49 + 51)}{(49 + 19 + 51 + 19)} = 0{,}724637681 \quad (15)$$

$$Error\ rate = \frac{(19 + 51)}{(49 + 19 + 51 + 19)} = 0{,}507246377 \quad (16)$$

Setelah nilai *precision*, *recall*, *accuracy* dan *error rate* didapat lalu nilai tersebut akan diubah menjadi persen. Maka dalam penelitian ini didapatkan nilai *precision* 72%, *recall* 72%, *accuracy* 72%, dan *error rate* 51%. Dan berdasarkan hasil pengujian dengan kasus uji perangkat lunak dengan *Black-Box Testing* diatas dapat ditarik kesimpulan bahwa perangkat lunak secara fungsional mengeluarkan hasil 96% sesuai dengan yang diharapkan. Hasil 96% didapatkan dari 26 uji *testing* 1 yang tidak berjalan dengan semestinya.

5. KESIMPULAN

Berdasarkan pengujian yang dilakukan terhadap Implementasi Sistem Rekomendasi *Content-Based Filtering* beserta penerapan algoritma *Term Frequency – Inverse Document Frequency* (TF-IDF) dan *Cosine Similarity* pada aplikasi ini diperoleh beberapa kesimpulan:

1. Penerapan algoritma *Term Frequency – Inverse Document Frequency* (TF-IDF) dan *Cosine Similarity* telah berhasil diterapkan dalam sistem dengan baik dimana sistem dapat memberikan *output* berupa dokumen yang relevan yaitu perabot sesuai dengan *query* yang di-*Input* kan.
2. Pengujian sistem yang dilakukan dengan menggunakan *confusion matrix* dalam penelitian ini didapatkan nilai *precision* 72%, *recall* 72%, *accuracy* 72%, dan *error rate* 51%. Sehingga, sistem dapat dikatakan baik, dikarenakan sistem yang baik adalah sistem yang memiliki nilai *precision, recall* dan *accuracy* yang tinggi.
3. Aplikasi pendukung desain interior dengan sistem rekomendasi berdasarkan nama *brand* perabot menggunakan algoritma *content-based filtering* berbasis web telah berhasil menerapkan *text preprocessing*, *Term Frequency – Inverse Document Frequency* (TF-IDF), dan *Cosine Similarity* untuk memberikan rekomendasi perabot pada pengguna berdasarkan *like* yang diberikan pada menu *get your idea*.

6. SARAN

Untuk pengembangan selanjutnya, maka saran-saran yang dapat diberikan adalah sebagai berikut:

1. Mengembangkan halaman *profile* sehingga pengguna dapat melakukan sunting data *profile.*
2. Diperlukan pengembangan fitur simulasi *3D model* yang dibangun khusus agar peneliti selanjutnya tidak perlu menggunakan aplikasi *open-source.*
3. Perlunya pembuatan *object* 3D pada perabot-perabot yang ada pada *dataset*, agar bisa ditampilkan pada fitur simulasi *3D model* sehingga pengguna dapat melihat kecocokan perabot tersebut dengan desainnya.
4. Perlunya dikembangkan versi aplikasi *mobile*, agar lebih praktis.
5. Penerapan metode sistem rekomendasi yang lebih kompleks dari metode *content-based filtering*, agar hasil rekomendasi yang diberikan semakin lebih baik lagi.

DAFTAR PUSTAKA

[1] F. D. Ching, Ilustrasi Desain Interior, Erlangga Jakarta, 2000.

[2] A. A. Wicaksono dan E. Tisnawati, Teori Interior, Yogyakarta, 2014.

[3] C. T. Widiyanti dan R. Firmansyah, "Spatial Design Analysis Dalam Proses Perancangan Dan Perancangan Interior," Idealog, vol. Vol 3, 2018.

[4] P. Nasiti, "Penerapan Metode Content Based Filtering Dalam Implementasi Sistem Rekomendasi Tanaman Pangan," Teknika, 2019.

[5] F. Ricci, L. Rokarch dan B. Shapira, Recommender Systems Handbook, 2011.

[6] R. Melita, V. Amrizal, H. B. Suseno dan T. Dirjam, "Penerapan Metode Term Frequency Inverse Document Frequency (TF-IDF) dan Cosine Similarity Pada Sistem Temu Kembali Informasi

Untuk Mengetahui Syarah Hadits Berbasis Web," Jurnal Teknik Informatika, vol. VOL 11 No. 2, 2018.

[7] M. Mustaqhfiri, Peringkasan Teks Otomatis Berita Olahraga Berbahasa Indonesia Menggunakan Metode Maximum Marginal Relevance, Malang: Universitas Islam Negri Malang, 2011.

[8] M. Uluyagmur, Z. Cataltepe dan E. Tayfur, "Proceedings of the World Congress on Engineering and Computer Science," Content-Based Movie Recommendation Using Different Feature Sets, 2012.

[9] D. Micol, Ó. Ferrández, F. Llopis dan R. Muñoz, "A Textual-Based Similarity Approach for Efficient and Scalable External Plagiarism Analysis Lab Report for PAN at CLEF," 2010.

[10] S. A. Salloum, M. Al-Emran, A. A. Monem dan K. Shaalan, "Using Text Mining Techniques for Extracting Information from Research Articles," Studies in Computational Intelligence, 2018.

[11] G. S. Lehal dan V. Gupta, "A Survey of Text Mining Techniques and Applications," JOURNAL OF EMERGING TECHNOLOGIES IN WEB INTELLIGENCE, 2009.