**PROPOSAL**

## PENERAPAN *VECTOR SPACE MODEL* (VSM) PADA SISTEM PENCARIAN ARTIKEL ARKEOLOGI (STUDI KASUS: JURUSAN ARKEOLOGI UNIVERSITAS HALU OLEO)

Diajukan untuk memenuhi salah satu syarat memperoleh derajat Sarjana Teknik



**WA ODE WAHYUNI MAKMUN**

**E1E1 16 084**

## JURUSAN TEKNIK INFORMATIKA

## FAKULTAS TEKNIK

## UNIVERSITAS HALU OLEO

## KENDARI

### 2020

**BAB I**

**PENDAHULUAN**

## 1.1 Latar Belakang

Mengembangkan Tridharma perguruan tinggi merupakan tanggung jawab dari suatu perguruan tinggi. Salah satu tridharma perguruan tinggi yaitu penelitian. Dalam melakukan suatu penelitian seorang peneliti perlu untuk mencari artikel ilmiah yang berkaitan untuk dipelajari maupun sebagai pendukung penelitiannya. Artikel ilmiah adalah tulisan yang berisi laporan sistematis mengenai hasil kajian atau hasil penelitian yang dilakukan oleh dosen, mahasiswa, peneliti dan ilmuwan (Suryoputro dkk., 2012).

Jurusan Arkeologi Universitas Halu Oleo telah memiliki koleksi artikel Arkeologi. Namun, koleksi artikel tersebut masih tersimpan pada komputer secara manual dengan nama file yang tidak teratur dan artikel tersebut dikelola oleh dosen yang telah diberi tanggung jawab. Dengan itu, masih banyak kendala-kendala yang dihadapi seperti apabila dosen atau mahasiswa yang membutuhkan artikel sebagai referensi, harus meminta kepada dosen yang memiliki wewenang untuk mengelolanya. Kendala lain yang dihadapi yaitu proses pencarian dokumen membutuhkan waktu yang relatif lama (± 15 hingga 20 menit), karena banyaknya koleksi artikel yang dimiliki (± 215 dokumen) serta harus membuka satu per satu dokumen untuk mendapatkan artikel yang berkaitan dengan penelitian selanjutnya. Berdasarkan hal tersebut dibutuhkan suatu sistem pencarian yang dapat membantu menemukan dokumen yang dibutuhkan.

Seiring dengan perkembangan teknologi saat ini, pembuatan sistem pencarian informasi menjadi semakin modern dan canggih. Sistem pencarian informasi dapat dibedakan menjadi beberapa model, diantaranya sistem pencarian biasa dalam hal ini pencarian berdasarkan kata kunci yang dihubungkan dengan basis data dan sistem temu kembali informasi.

Sistem pencarian biasa lebih terstruktur, serta menggunakan *query* pada *database*, namun metode pencariannya memiliki beberapa kelemahan seperti informasi yang didapatkan ialah informasi yang benar – benar mirip dengan kata

1

kunci, sehingga informasi yang didapatkan hanya terbatas pada kata kunci yang digunakan. Hal ini menyebabkan pengguna harus benar-benar memasukkan kata kunci yang tepat untuk mendapatkan informasi yang diinginkan (Maskur & Andriansyah, 2015). Sedangkan sistem temu kembali informasi tidak terstruktur, serta menggunakan teks bebas seperti bahasa manusia sehari-hari, sehingga mampu melakukan pencarian infomasi yang relevan dengan kata kunci yang dimaksud serta menampilkan sesedikit mungkin dokumen yang tidak relevan.

Sistem Temu Kembali Informasi (*Information Retrieval*) adalah proses menemukan materi yang biasanya dokumen dari data yang tidak terstruktur yang memenuhi kebutuhan informasi dari sebuah kumpulan koleksi yang tersimpan pada komputer (Christioko & Daru, 2019).

Dalam sistem temu kembali informasi, diperlukan perankingan hasil pencarian dokumen sesuai dengan tingkat relevansinya dengan kata kunciyang dimasukkan. *Vector Space Model* (VSM) merupakan salah satu model dalam sistem temu kembali informasi yang mampu melakukan perankingan terhadap dokumen – dokumen yang relevan terhadap kata kunci yang dimasukkan. *Vector Space Model* (VSM) adalah metode untuk mengetahui tingkat kesamaan (*similarity*) *term* dengan cara pembobotan *term*. *Vector Space Model* (VSM) merepresentasikan suatu dokumen dan *query* dalam sebuah bentuk *vector* yang memiliki jarak (*magnitude*) dan arah (*direction*) (Sanjaya, 2018).

Berdasarkan latar belakang yang telah diuraikan diatas, maka penulis mengambil topik penelitian dengan judul “**Penerapan *Vector Space Model* (VSM) pada Sistem Pencarian Artikel Arkeologi. (Studi Kasus: Jurusan Arkeologi Universitas Halu Oleo)**”.

## 1.2 Rumusan Masalah

Berdasarkan latar belakang di atas, maka permasalahan yang dapat dirumuskan adalah bagaimana kinerja *Vector Space Model* (VSM) dalam menemukan dokumen artikel yang relevan dan melakukan perangkingan berdasarkan tingkat kecocokan antara *query* dan dokumen pada *database*.

## 1.3 Batasan Masalah

Adapun batasan masalah untuk penelitian ini, sebagai berikut:

1. Data yang digunakan dalam penelitian ini adalah kumpulan artikel Arkeologi yang ada di Jurusan Arkeologi Universitas Halu Oleo.
2. Sistem ini dibuat khusus untuk pencarian artikel Arkeologi dengan jumlah 215 artikel.
3. *Query* yang di-*input*-kan oleh *user* akan dicocokkan berdasarkan tingkat kemiripan dengan isi dari dokumen pada *database*.
4. Format file yang digunakan bertipe .pdf

## 1.4 Tujuan Penelitian

Adapun tujuan penelitian ini adalah menerapkan *Vector Space Model* (VSM) untuk melakukan pencarian dokumen artikel yang relevan dan melakukan perangkingan berdasarkan tingkat kecocokan antara *query* dan dokumen pada *database*.

## 1.5 Manfaat Penelitian

Adapun manfaat yang diharapkan dalam penelitian ini adalah sebagai berikut:

1. Dapat membantu dosendan mahasiswa jurusan Arkeologi dalam pencarian informasi tentang artikel yang dibutuhkan.
2. Pencariannya akan lebih cepat dan tepat.
3. Mencegah adanya kesamaan judul penelitian.

**1.6 Sistematika Penulisan**

Sistematika penulisan laporan tugas akhir ini adalah sebagai berikut :

## BAB I PENDAHULUAN

Bab ini berisi penjelasan mengenai latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, sistematika penulisan dan tinjauan pustaka.

## BAB II LANDASAN TEORI

Bab ini berisi pengertian – pengertian dan teori – teori yang berkaitan untuk pembuatan sistem pencarian artikel Arkeologi, yaitu penjelasan tentang artikel ilmiah, Arkeologi, *Text Mining*, Sistem Temu Kembali Informasi, *Vector Space Model* (VSM).

## BAB III METODOLOGI PENELITIAN

Bab ini menjelaskan mengenai langkah – langkah pengumpulan data, metode pengembangan perangkat lunak, tempat dan waktu penelitian.

## BAB IV ANALISIS DAN PERANCANGAN

Bab ini akan membahas tentang analisis dan perancangan dari sistem yang akan dibuat, dengan menggunakan desain *Unified Modelling Language* (UML) serta desain *user interface*.

## BAB V IMPLEMENTASI DAN PENGUJIAN SISTEM

Bab ini membahas mengenai implementasi hasil perancangan sistem yang dibuat serta melakukan pengujian terhadap sistem.

## BAB VI PENUTUP

Bab ini menguraikan tentang kesimpulan dari keseluruhan bab serta memberikan saran-saran yang mungkin berguna untuk mengatasi masalah yang dihadapi.

### 1.7 Tinjauan Pustaka

Pelaksanaan dari penelitian ini menggunakan referensi yang berhubungan. Penelitian ini didasarkan pada penelitian sebelumnya mengenai sistem pencarian yang menggunakan metode *Vector Space Model* (VSM).

Penelitian sebelumnya oleh Yudhanto (2019) dalam penelitiannya yang berjudul Penerapan *Vector Space Model* (VSM) Untuk Perangkingan Saran Jawaban Pada Aplikasi Tanya Ustadz menyimpulkan bahwa Metode *Vector Space Model* (VSM) dapat berjalan dengan baik dalam melakukan perangkingan saran jawaban, terlihat dari persentase saran jawaban yang ditampilkan. Semakin besar tingkat kecocokannya maka semakin besar pula nilai persentasenya demikian sebaliknya, saat di-*input*-kan kalimat acak dan tidak bermakna, sistem tidak menampilkan saran jawaban apapun. Hal ini menunjukkan bahwa metode *Vector Space Model* (VSM) berjalan sebagaimana mestinya yaitu melihat tingkat kecocokan kata yang telah terindeks dalam KBBI.

Penelitian sebelumnya oleh Sanjaya (2018) dalam penelitiannya yang berjudul Pemanfaatan Sistem Temu Kembali Informasi dalam Pencarian Dokumen Menggunakan Metode *Vector Space Model* (VSM) menyimpulkan bahwa hasil pencarian dokumen pada sistem dapat mengembalikan hampir semua dokumen yang relevan jika dokumen tersebut ada, hasil pengujian didapatkan nilai *precision* yang rendah hal ini dipengaruhi oleh banyaknya variasi *term* dalam data karena sistem melakukan *indexing* dari semua kata yang ada dalam dokumen. Semakin banyak *term* yang dihasilkan sebuah dokumen dalam proses *indexing* maka semakin sering dokumen tersebut dikembalikan oleh sistem terhadap *query* yang sebenarnya kurang relevan (*low precision*). Tetapi hasil tersebut telah diatasi pada sistem dengan memberi nilai *threshold* pada nilai perhitungan kemiripan (*similarity*), *Term frequency* dan *Inverse document frequency* berperan penting dalam hasil perangkingan dokumen yang dikembalikan. Dengan kata lain hasil rangking dokumen sangat dipengaruhi oleh data term pada dokumen dan juga *query*.

Penelitian selanjutnya oleh Amin (2011) dalam penelitiannya yang berjudul Implementasi *Search Engine* (Mesin Pencari) Menggunakan Metode *Vector Space Model* menyimpulkan bahwa berdasarkan implementasi contoh kasus *keyword* atau kata kunci (*query*) sistem dengan 3 dokumen yang ada, *Vector Space Model* menghasilkan Dokumen 2 (D2) sebagai dokumen yang paling mendekati kemiripannya atau tingkat kedekatannya dengan *query* diikuti dengan Dokumen 1 (D1) dan Dokumen 3 (D3). *Recall* yang dihasilkan rendah dan presisi yang dihasilkan tinggi, artinya dokumen yang diharapkan muncul dengan tingkat akurasi tinggi dapat ditemukan dengan tepat dan dokumen yang dihasilkan sedikit.

Penelitian selanjutnya oleh Putung dkk (2016) dalam penelitiannya yang berjudul Penerapan Sistem Temu Kembali Informasi Pada Kumpulan Dokumen Skripsi menyimpulkan bahwa Berdasarkan hasil penelitian yang dilakukan sistem temu kembali informasi berhasil dibangun dan dapat mengimplementasikan pengindeksan otomatis dengan menggunakan metode pembobotan TF-IDF dan dapat menemukan dokumen yang relevan terhadap kata kunci dengan menghitung nilai *cosine similarity*. Waktu proses *indexing* sistem berdasarkan pengujian waktu *indexing*, waktu proses *indexing* terus meningkat bersamaan dengan banyaknya dokumen yang di *index* dengan waktu pengujian *indexing* pada 100 dokumen skripsi 328,85 detik. Data hasil pengujian menghasilkan nilai *recall* 1 pada setiap pengujian yang menunjukkan bahwa semua dokumen yang relevan dapat ditemukan oleh sistem, nilai *precision* antara 0,346154 sampai dengan 1 yang menunjukkan terdapat dokumen lain selain dokumen relevan yang ikut ditemukan oleh sistem.

Penelitian selanjutnya oleh Somu (2017) dalam penelitiannya yang berjudul Sistem Temu Balik Informasi Dokumen Teks Menggunakan *Vector Space Model* (VSM) (Studi Kasus : Pencarian Buku Cetak Di Toko Buku Suci – Kupang) menyimpulkan bahwa Sistem temu balik dokumen teks dengan menggunakan *Vector Space Model* (VSM) (studi kasus: pencarian buku di Toko Buku Suci Kupang) mampu membantu pelayan Toko Buku Suci dan pembeli dalam proses pencarian buku cetak berdasarkan *query* dan kebutuhan pembeli sehingga lebih efisien. Hal ini dapat dilihat dari rata-rata waktu pencarian informasi yang berlangsung cepat yaitu 0,01454 detik.

Dari beberapa penelitian di atas, dapat disimpulkan bahwa terdapat banyak penelitian yang telah dilakukan menggunakan *Vector* *Space* *Model* (VSM), namun belum ada penelitian yang menggunakan *Vector* *Space* *Model* (VSM) dalam sistem pencarian artikel ilmiah tentang Arkeologi. Serta dapat diketahui bahwa sistem temu kembali informasi yang menggunakan *Vector Space Model* (VSM) dapat bekerja dengan baik yaitu mampu menemukan dokumen – dokumen yang relevan dengan kata kunci yang dimasukkan.

**BAB II**

**LANDASAN TEORI**

## 2.1 Pengertian Artikel Ilmiah

Menurut Kamus Besar Bahasa Indonesia, artikel adalah karya tulis lengkap, misalnya laporan berita atau esai dalam majalah, surat kabar dan sebagainya. Sedangkan ilmiah adalah memenuhi syarat (kaidah) ilmu pengetahuan. Jadi, artikel ilmiah adalah karya tulis lengkap yang memenuhi syarat ilmu pengetahuan.

Dalam (Suryoputro dkk., 2012) Adnan dkk. (2005) menyatakan Artikel ilmiah (*research articles*) adalah tulisan yang berisi laporan sistematis mengenai hasil kajian atau hasil penelitian yang disajikan bagi masyarakat ilmiah tertentu, yang merupakan audiens khusus dengan tujuan menyampaikan hasil kajian dan kontribusi penulis artikel kepada mereka untuk dipikirkan, dikaji kembali, dan didiskusikan, baik secara lisan maupun tertulis. Yang dimaksud dengan audien khusus antara lain seperti mahasiswa, dosen, peneliti dan ilmuwan.

Dengan demikian dapat disimpulkan bahwa artikel ilmiah merupakan sebuah media komunikasi yang digunakan oleh dosen, mahasiswa, peneliti dan ilmuwan untuk menyampaikan hasil kajian ilmu atau penelitian.

## 2.2 Pengertian Arkeologi

Menurut Kamus Besar Bahasa Indonesia, Arkeologi adalah ilmu tentang kehidupan dan kebudayaan zaman kuno berdasarkan benda peninggalannya, seperti patung dan perkakas rumah tangga.

Arkeologi merupakan satu disiplin ilmu yang mengkaji tentang manusia dan kebudayaan masa silam berdasarkan kepada tinggalan dengan cara penyelidikan yang sistematik dengan menggunakan berbagai pendekatan disiplin-disiplin ilmu yang lain seperti sejarah, antropologi, geologi, geografi, fisika, matematis dan lainlainnya (Manalu, 2013).

7

## 2.3 *Text Mining*

Dalam (Amin, 2011) sesuai buku *The Text Mining Handbook*, *text mining* dapat didefinisikan sebagai suatu proses menggali informasi dimana seorang *user* berinteraksi dengan sekumpulan dokumen menggunakan *tools* analisis yang merupakan komponen – komponen dalam *data mining* yang salah satunya adalah kategorisasi. Tujuan dari *text mining* adalah untuk mendapatkan informasi yang berguna dari sekumpulan dokumen.

Dalam (Indriani dkk., 2018) Prakasa (2016) menyatakan permasalahan yang biasa ditangani oleh *text mining* selain klasifikasi yaitu *information extraction*, *clustering*, dan *information retrieval*.

## 2.4 *Text Preprocessing*

Menurut (Sanjaya, 2018) *text preprocessing*, merupakan tahapan awal pada proses merepresentasikan koleksi dokumen kedalam bentuk tertentu untuk memudahkan dan mempercepat proses pencarian dan penemuan kembali dokumen yang relevan. Terdapat beberapa tahapan pada fase ini antara lain:

a. *Case Folding* dan *Tokenization*, *Case folding* dilakukan untuk merubah huruf besar dari setiap kata diubah menjadi huruf kecil dan menghilangkan karakter selain huruf seperti angka dan tanda baca (*delimiter*). Sedangkan *tokenization* yaitu memecah semua kalimat pada isi dokumen menjadi kata per kata.

Contoh *Case Folding* dan *Tokenization* dapat di lihat sebagai berikut :

*Input*:

1. Aspek – Aspek Arkeologis Pada Situs – Situs Bercorak Megalitik Di Kawasan Bantarkalong Tasikmalaya
2. Ragam Bentuk Tinggalan Budaya Megalitik Di Papua

*Output*

:

Dokumen

Dokumen 1

Aspek

–

Aspek Arkeologis

Pada Situs

–

Situs Bercorak

Megalitik

Di

Kawasan

Bantarkalong Tasikmalaya

Hasil

*Case Folding*

dan

*Tokenization*

aspek

arkeologis

pada

situs

bercorak

megalitik

di

kawasan

bantarkalong

tasikmalaya

ragam

bentuk

tinggalan

budaya

Dokumen 2

Ragam Bentuk Tinggalan

Budaya Megalitik Di Papua

## Gambar 2.1 Contoh Proses *Case Folding* dan *Tokenization*

b. *Filtering*, dilakukan dengan metode *stopword* yaitu menghilangkan semua kata yaitu kata sambung, kata depan, kata ganti, dan lain-lain. Sebagian contoh dalam bahasa Indonesia : yang, juga, dari, dia, kami, kamu, aku, saya, ini, itu, atau, dan, tersebut, pada, dengan, adalah, yaitu, ke, tak, tidak, di, pada, jika, maka, ada, pun, lain, saja, hanya, namun, seperti, kemudian, karena, untuk, dan lain-lain.

Contoh *Filtering* dapat dilihat pada Gambar 2.2.

aspek

arkeologis

pada

situs

bercorak

megalitik

di

kawasan

bantarkalong

tasikmalaya

ragam

bentuk

tinggalan

budaya

Hasil

*Case Folding*

dan

*Tokenization*

*Stopword List*

pada

di

Hasil

*Filtering*

aspek

arkeologis

situs

bercorak

megalitik

kawasan

bantarkalong

tasikmalaya

ragam

bentuk

tinggalan

budaya

## Gambar 2.2 Contoh Proses *Filtering*

c. *Stemming*, dilakukan dengan cara menghilangkan semua imbuhan (*affixes*) baik yang terdiri dari awalan (*prefixes*), sisipan (*infixes*), akhiran (*suffixes*) dan *confixes* (kombinasi dari awalan dan akhiran) pada kata turunan. *Stemming* digunakan untuk mengganti bentuk dari suatu kata menjadi kata dasar dari kata tersebut yang sesuai dengan struktur morfologi Bahasa Indonesia.

Contoh *Stemming* dapat dilihat pada Gambar 2.3.

**Gambar 2.3**

**Contoh**

**Proses**

***S***

***temming***

Hasil

*Filtering*

aspek

arkeologis

situs

bercorak

megalitik

kawasan

bantarkalong

tasikmalaya

ragam

bentuk

tinggalan

budaya

Hasil

*Stemming*

aspek

arkeologis

situs

corak

megalitik

kawasan

bantarkalong

tasikmalaya

ragam

bentuk

tinggal

budaya

### 2.5 Indexing

Dalam (Somu, 2017) Salton (1989) menyatakan bahwa *indexing* merupakan sebuah proses untuk melakukan pengindeksan terhadap kumpulan dokumen yang akan disediakan sebagai informasi kepada pengguna. Pengindeksan dilakukan dengan membuat *inverted index* (atau disebut juga *inverted file*). *Inverted index* merupakan konversi dari dokumen asli yang mengandung sekumpulan kata ke dalam daftar kata (*term*) yang behubungan dengan dokumen terkait dimana kata – kata tersebut muncul.

Pada *information retrieval system*, indeks ini nantinya yang digunakan untuk merepresentasikan informasi di dalam sebuah dokumen. Elemen dari indeks adalah *index term* yang didapatkan dari teks yang dipecah di dalam sebuah dokumen. Elemen lainnya adalah *term weighting* sebagai penentuan rangking dari kriteria relevan sebuah dokumen yang memiliki istilah yang sama selain itu dapat digunakan untuk pengurutan saat penemukembalian dengan susunan menurun dari bobot yang besar ke kecil sesuai dengan bobot istilah-istilah yang sama antara *query* dan dokumen.

### 2.6 Sistem Temu Kembali Informasi (Information Retrieval)

Sistem temu kembali informasi (*Information retrieval*) adalah proses menemukan materi yang biasanya dokumen dari data yang tidak terstruktur yang memenuhi kebutuhan informasi dari sebuah kumpulan koleksi yang tersimpan pada komputer (Christioko & Daru, 2019).

*Information Retrieval* adalah aktifitas utama yang dilakukan oleh sebuah penyedia informasi atau pusat pelayanan informasi, termasuk perpustakaan dan jenis dari layanan lainnya yang menyediakan informasi. Sistem temu kembali informasi tidak memberitahu (yakni tidak mengubah pengetahuan) pengguna mengenai masalah yang ditanyakannya. Sistem tersebut hanya memberitahukan keberadaan atau ketidakberadaan dan keterangan dokumen – dokumen yang berhubungan dengan permintaannya (Sanjaya, 2018).

Dalam (Amin, 2011) Salton (1989) berpendapat bahwa sistem temu kembali informasi merupakan bagian dari ilmu komputer yang berhubungan dengan pengambilan informasi dari dokumen-dokumen yang didasarkan pada isi dan konteks dari dokumen-dokumen itu sendiri. Proses dalam sistem temu kembali dapat digambarkan sebagai sebuah proses untuk mendapatkan dokumen yang relevan dari koleksi dokumen melalui pencarian *query* yang di-*imput*-kan *user*. Salton menjelaskan bahwa sistem temu kembali informasi bertujuan untuk menjembatani kebutuhan informasi user dengan sumber informasi yang tersedia dalam situasi seperti dikemukakan sebagai berikut:

1. Mempresentasikan sekumpulan ide dalam sebuah dokumen menggunakan sekumpulan konsep.
2. Terdapat beberapa pengguna yang memerlukan ide, tapi tidak dapat mengidentifikasikan dan menemukannya dengan baik.
3. Sistem temu kembali informasi bertujuan untuk mempertemukan ide yang dikemukakan oleh penulis dalam dokumen dengan kebutuhan informasi pengguna yang dinyatakan dalam bentuk *key word query*/istilah penelusuran.

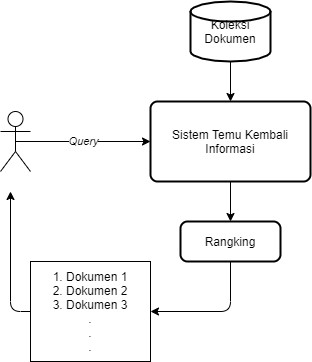
Dalam (Putung dkk., 2016) Kurniasi, N (2014) berpendapat bahwa *Information retrieval* adalah sebuah proses untuk menemukan kembali informasi yang dibutuhkan dari sebuah sistem penyimpanan dan penelusuran informasi. Sistem temu kembali informasi mensyaratkan ada kebutuhan informasi dari pengguna, ada dokumen atau data yang berisi informasi yang diorganisasikan dalam sebuah sistem yang memudahkan temu kembali informasi dan strategi penelusuran yang tepat sehingga dokumen yang sesuai dengan kebutuhan dapat ditemukan kembalikan. Sistem temu balik informasi terdiri dari tiga komponen utama, yaitu masukan (*input*), pemroses (*processor*), dan keluaran (*output*). Menurut Manning, dkk (2009) *Information Retrieval* (IR) adalah menemukan bahan (biasanya dokumen) yang bersifat tidak terstruktur (biasanya teks) yang memenuhi kebutuhan informasi dari dalam koleksi besar (biasanya disimpan di komputer).

Definisi *query* dalam *information* *retrieval* merupakan sebuah formula yang digunakan untuk mencari informasi yang dibutuhkan oleh *user*, dalam bentuk yang paling sederhana, sebuah *query* merupakan suatu *keywords* (kata kunci) dan dokumen yang mengandung *keywords* merupakan dokumen yang dicari dalam sistem temu kembali informasi.

*Information* *retrieval* memiliki 3 model yang dapat digunakan, yaitu model boolean, model *probabilistic* (*Binary Independence Model*, Model Okapi BM25, dan *Bayesian Network Model*), dan model ruang vector (Purnama, 2012).

Adapun proses sistem temu kembali informasi dapat dilihat pada Gambar

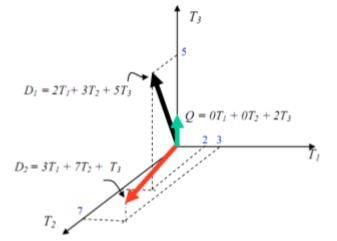
2.4.



## Gambar 2.4 Proses Sistem Temu Kembali Informasi

### 2.7 Vector Space Model (VSM)

Dalam (Wicaksosno dkk., 2016) Turney, P. D., & Pantel, P. (2010) berpendapat bahwa *Vector Space Model* (VSM) merupakan model yang digunakan untuk mengukur kemiripan antara dokumen dan *query* yang mewakili setiap dokumen dalam sebuah koleksi sebagai sebuah titik dalam ruang. Dalam metode *Vector Space Model* dihitung bobotdari setiap *term* yang terdapat dalam semua dokumen dan *query* dari *user*. *Term* adalah kata atau kumpulan kata yang merupakan ekspresi verbal dari suatu pengertian. Penentuan relevansi dokumen dengan *query* dipandang sebagai pengukuran kesamaan antara vektor dokumen dengan vektor *query*. Contoh representasi relevansi antara dokumen dan *query* dapat digambarkan pada Gambar 2.5. Q merupakan *query* pembanding, D1 dan D2 adalah dua dokumen yang akan dibandingkan, sedangkan T1, T2 dan T3 adalah tiga *term* pada dokumen tersebut.



## Gambar 2.5 Representasi Dokumen dan *Query* Pada Ruang Vektor (Turney & Pantel, 2010)

(Somu, 2017) menyatakan dalam *Vector Space Model* (VSM) koleksi dokumen direpresentasikan sebagai sebuah matriks *term document* (matriks *term frequency*). Setiap sel dalam matriks bersesuaian dengan bobot yang diberikan dari suatu *term* dalam dokumen yang ditentukan. Nilai nol berarti bahwa *term* tersebut tidak ada dalam dokumen. Matriks *term document* dengan n dokumen dan t *term* dapat dilihat pada Gambar 2.6.

T1 T2 T3 T… Tt

D1 W11 W21 W31 … Tt1

D2 W12 W22 W32 … Tt2

D3 W13 W23 W33 … Tt3

D… … … … … …

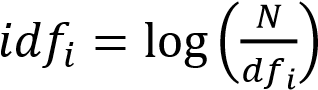
D4 W1n W2n W3n … Ttn

## Gambar 2.6 Matriks *Term Document*

Proses perhitungan *vektor space model* (VSM) melalui tahapan perhitungan *term frequency* (tf), *inverse document frequency* (idf), *term frequency inverse document frequency* (tf – idf), jarak *query* dan dokumen, pengukuran similaritas *query document* (*inner product*), dan pengukuran *Cosine Similarity* (menghitung nilai *cosine* sudut antara dua vektor).

### 2.7.1 Menghitung bobot dokumen dengan term – frequency dan inverse document frequency (tf – idf)

*Term frequency–inverse document frequency* atau biasa sering disebut tf – idf adalah metode pembobotan kata dengan menghitung nilai tf dan juga menghitung kemunculan sebuah kata pada koleksi dokumen teks secara keseluruhan. *Inverse document frequency* (idf) adalah jumlah dokumen yang mengandung sebuah *term* didasarkan pada seluruh dokumen yang ada pada data set (Ahmad dkk, 2012). Terdapat beberapa cara atau metode dalam melakukan pembobotan kata pada metode tf – idf yaitu melalui skema pembobotan *query* dan dokuman. Formula yang digunakan pada *term frequency* (tf) diberikan berdasarkan jumlah kemunculan suatu kata di dalam dokumen. *Inverse dokument frequency* (idf) dihitung dengan menggunakan Persamaan 2.1 (Manning dkk, 2009):

 (2.1)

Dimana :

N = total jumlah dokumen dalam koleksi dfi = jumlah dokumen yang berisi *term* j

idfi = frekuensi kemunculan suatu istilah dalam dokumen i

Dengan demikian rumus untuk tf – idf adalah penggabungan dari formula perhitungan *raw* tf dengan formula idf dengan cara mengalikan nilai *term frequency* (tf) dengan nilai *inverse document frequency* (idf). Formula yang biasa digunakan dapat dilihat pada Persamaan 2.2.

Wij = tfij \* idfi (2.2)

Dimana:

Wij = bobot *term* j dalam dokumen i tfij = frekuensi *term* j dalam dokumen i

idfi = frekuensi kemunculan suatu istilah dalam dokumen i

Pada algoritma tf – idf digunakan rumus untuk menghitung bobot (W) masing-masing dokumen terhadap kata kunci dengan rumus pada Persamaan (2.1) dan (2.2). Setelah bobot (W) masing-masing dokumen diketahui, maka dilakukan proses *sorting* atau pengurutan dimana semakin besar nilai W, semakin besar tingkat similaritas dokumen tersebut terhadap kata kunci, demikian sebaliknya.

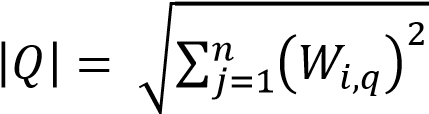
### 2.7.2 Menghitung Jarak Query dan Dokumen

Perhitungan jarak dokumen dan jarak *query* dilakukan setelah perhitungan bobot. Jika, Q adalah vektor *query* dan D adalah vektor dokumen, merupakan 2 buah vektor dalam ruang berdimensi-n, dan θ adalah sudut yang dibentuk oleh 2 vektor tersebut. Perhitungan jarak *query* dan dokumen menggunakan Persamaan

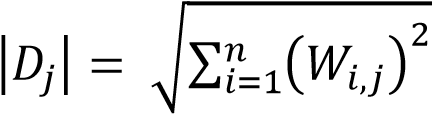
2.3 (Baeza-Yates et al, 1999).

𝑄 ∙ 𝐷 = |𝑄||𝐷|cos 𝜃 (2.3)

Dimana 𝑄 ∙ 𝐷 adalah hasil perkalian dalam (*inner product*) kedua vector, sedangkan perhitungan jarak *query* menggunakan Persamaan 2.4 dan perhitungan jarak dokumen menggunakan Persamaan 2.5.

 (2.4)

Dengan |𝑄| adalah jarak *query* dan 𝑊𝑖,𝑞 adalah bobot *query* dokumen ke – i, maka jarak *query* dari (|𝑄|) dihitung untuk didapatkan jarak *query* dari bobot *query* dokumen (𝑊𝑖,𝑞) yang terambil oleh sistem. Jarak *query* bisa dihitung dengan persamaan akar jumlah kuadrat dari *query*.

 (2.5)

Dengan |𝐷𝑗| adalah jarak dokumen dan 𝑊𝑖,𝑗 adalah bobot dokumen ke – i, maka jarak dokumen (|𝐷𝑗|) dihitung untuk didapatkan jarak dokumen dari bobot dokumen (𝑊𝑖,𝑗) yang terambil oleh sistem. Jarak dokumen bias dihitung dengan persamaan akar jumlah kuadrat dari dokumen. Persamaan di atas merupakan panjang *vector* atau jarak *euclidean* suatu vektor dengan titik nol.

### 2.7.3 Menghitung Similaritas Query dan Dokumen

Setelah jarak dari tiap dokumen dan *query* didapatkan, dilakukan perhitungan similaritas antara *query* dengan dokumen. Perhitungan kesamaan kedua vektor menggunakan Persamaan 2.6 (Cios et al, 2007).

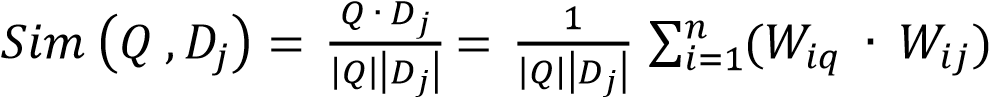
 (2.6)

Dengan 𝑊𝑖𝑗 adalah bobot *term* dalam dokumen, 𝑊𝑖𝑞 adalah bobot *query*, dan 𝑆𝑖𝑚 (𝑄, 𝐷𝑗) adalah similaritas antara *query* dan dokumen. Similaritas antara *query* dan dokumen (*inner product*) atau 𝑆𝑖𝑚 (𝑄, 𝐷𝑗) digunakan untuk mendapatkan bobot dengan didasarkan pada bobot *term* dalam dokumen 𝑊𝑖𝑗 dan bobot *query* 𝑊𝑖𝑞 atau dengan cara menjumlah bobot Q dikalikan dengan bobot dokumen.

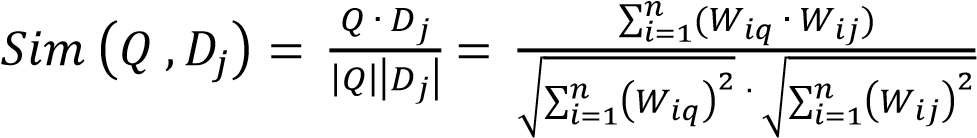
### 2.7.4 Menghitung Cosine Similarity

Melalui *vector space model* dan tf-idf *weighting* akan didapatkan representasi nilai numerik dokumen sehingga dapat dihitung kedekatan antar dokumen. Semakin dekat dua vektor didalam suatu *vector space model*, maka semakin mirip dua dokumen yang diwakili oleh dua vektor tersebut. Kemiripan antar dokumen dapat dihitung menggunakan suatu fungsi ukuran kemiripan (*similarity measure*). Ukuran ini memungkinkan perankingan dokumen sesuai dengan kemiripannya atau relevansinya terhadap *query*.

Pengukuran *cosine similarity* (menghitung nilai *cosine* sudut antara dua vektor) menggunakan Persamaan 2.7 dan 2.8.

 (2.7)

Dari formula diatas dapat dinormalisasi menjadi (Cios, 2007) :

 (2.8)

Similaritas antara *query* dan dokumen atau 𝑆𝑖𝑚 (𝑄 , 𝐷𝑗) berbanding lurus terhadap jumlah bobot *query* (Q) dikali bobot dokumen (𝐷𝑗) dan berbanding terbalik terhadap akar jumlah kuadrat (|𝑄|) dikali akar jumlah kuadrat dokumen (|𝐷𝑗|). Perhitungan similaritas menghasilkan bobot dokumen yang mendekati nilai

1 atau menghasilkan bobot dokumen yang lebih besar dibandingkan dengan nilai yang dihasilkan dari perhitungan *inner product*.

### 2.8 Unified Modeling Language (UML)

*Unified Modeling Language* (UML) adalah sebuah standarisasi Bahasa pemodelan untuk pembangunan perangkat lunak yang dibangun dengan menggunakan teknik pemrograman berorientasi objek (S. & Shalahuddin, 2015).

#### 2.8.1 *Use Case Diagram*

*Use case diagram* merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. *Use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi – fungsi itu.

Simbol – simbol yang ada pada diagram *use case* dapat dilihat pada Tabel 2.1.

# Tabel 2.1 Simbol-simbol *Use Case Diagram*

|  |  |
| --- | --- |
| **Simbol** | **Deskripsi** |
| Aktor | Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat, jadi walaupun actor19 dari actor adalah gambar orang, tapi actor belum tentu merupakan orang. |
| *Use case* | Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau actor. |
| Asosiasi/*association* | Komunikasi antara actor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan actor. |
| Generalisasi/*generalization* | Hubungan generalisasi dan spesialisasi (umumkhusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya. |
| Ekstensi/*extend* | Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu. |
| Menggunakan/*include*/*uses* | Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini. |

### 2.8.2 Activity Diagram

*Activity Diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak.

Yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem.

Simbol – simbol yang ada pada *activity diagram* dapat dilihat pada Tabel 2.2.

# Tabel 2.2 Simbol-simbol *Activity Diagram*

|  |  |
| --- | --- |
| **Simbol** | **Deskripsi** |
| Status awal | Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal. |
| Aktivitas | Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja. |
| Percabangan/*decision* | Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu. |
| Penggabungan/*join* | Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu. |
| Status akhir | Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir. |

### 2.8.3 Sequence Diagram

*Sequence Diagram* menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambar *sequence diagram* maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu.

Simbol – simbol yang ada pada *sequence diagram* dapat dilihat pada Tabel

2.4.

# Tabel 2.3 Simbol-simbol *Sequence Diagram*

|  |  |  |  |
| --- | --- | --- | --- |
| **Simbol** | | | **Deskripsi** |
| Aktor | | | Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat, jadi walaupun simbol dari aktor adalah gambar orang, tapi actor belum tentu merupakan orang. |
| Garis hidup/*lifeline* | | | Menyatakan kehidupan suatu objek. |
| Objek   |  | | --- | | nama objek : nama kelas | | | | Menyatakan objek yang berinteraksi pesan. |
| Waktu aktif | | | Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan didalamnya. |
|  |  |  |
| Pesan tipe *create*  <<*create*>> | | | Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat. |
| Pesan tipe *call*  1 : nama\_metode () | | | Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri. |
| Pesan tipe *send*  1 : masukan | | | Menyatakan bahwa suatu objek mengirimkan data/masukan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirimi. |

### 2.8.4 Class Diagram

*Class Diagram* menggambarkan struktur sistem dari segi pendefinisian kelas – kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi.

1. Atribut merupakan variabel – variabel yang dimiliki suatu kelas.
2. Operasi atau metode adalah fungsi – fungsi yang dimiliki oleh suatu kelas.

Simbol – simbol yang ada pada *class diagram* dapat dilihat pada Tabel 2.5.

# Tabel Tabel 2.4 Simbol-simbol *Class Diagram*

|  |  |
| --- | --- |
| **Simbol** | **Deskripsi** |
| Kelas   |  | | --- | | nama\_kelas | | +atribut | | +operasi() | | Kelas pada struktur sistem. |
| antarmuka/*interface*  nama\_interface | Sama dengan konsep *interface* |
| Asosiasi/*association* | Relasi antarkelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity*. |
| Asosiasi berarah/*directed association* | Relasi antarkelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*. |
| Generalisasi/*generalization* | Relasi antarkelas dengan makna generalisasispesialisasi (umum khusus). |
| Kebergantungan/*dependency* | Relasi antarkelas dengan makna kebergantungan antarkelas. |
| Agregasi/*aggregation* | Relasi antarkelas dengan makna semua-bagian (*whole-part*). |

### 2.9 Metode Pengembangan Sistem

Metode pengembangan sistem yang digunakan pada penelitian ini adalah *Rational Unified Process* (RUP). Dalam (Ependi dkk., 2014) Rosa dan Shalahuddin (2011) berpendapat bahwa *Rational Unified Process* (RUP) adalah pendekatan pengembangan perangkat lunak yang dilakukan berulang-ulang (*iterative*), fokus pada arsitektur (*architecture-centric*), lebih diarahkan berdasarkan penggunaan kasus (*use case driven*).

Metode RUP mempunyai empat fase, yaitu:

1. Fase *inception*

Tahap dimana kita memodelkan proses bisnis yang dibutuhkan (*business modeling*) dan mendefinisikan kebutuhan akan sistem yang akan dibuat (*requirements*).

1. Fase *elaboration*

Tahap ini lebih difokuskan pada perencanaan arsitektur sistem. Tahap ini lebih pada analisis dan desain sistem.

1. Fase *construction*

Tahap dimana kita mengembangkan komponen dan fitur-fitur sistem. Implementasi dan pengujian sistem yang fokus pada implementasi perangkat lunak pada kode program. 4. Fase *transition*

Tahap dimana kita *deployment* atau instalasi sistem agar dapat dimengerti oleh user. Aktifitas pada tahap ini termasuk pada pelatihan user dan pemeliharaan.

### 2.10 Metode Pengujian Sistem

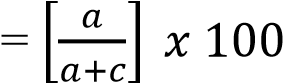
Dalam mengukur efektivitas sistem temu kembali informasi biasanya digunakan *recall* dan *precision*. *Recall* adalah proporsi jumlah dokumen yang dapat ditemukan kembali oleh sebuah proses pencarian di sistem IR. *Precision* adalah proporsi jumlah dokumen yang ditemukan dan dianggap relevan untuk kebutuhan si pencari informasi. *Recall* (perolehan) berhubungan dengan kemampuan sistem untuk memanggil dokumen yang relevan. *Precision* (ketepatan) berkaitan dengan kemampuan sistem untuk tidak memanggil dokumen yang tidak relevan (Lestari, 2016).

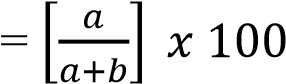
Lancaster (1991) merumuskan matriks terkenal berikut ini sebagai ukuran *recall* – *precision*:

# Tabel 2.5 Matriks *Recall* dan *Precision* Lancaster

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Relevan** | **Tidak Relevan** | **Total** |
| **Ditemukan** | a | b | a + b |
| **Tidak Ditemukan** | c | d | c + d |
| **Total** | a + c | b + d | a + b + c + d |

Lalu, berdasarkan tabel tersebut, rumus *recall* dan *precision* dapat dilihat pada persamaan 2.9 dan 2.10.

*Recall*  (2.9)

*Precision*  (2.10)

Dimana : a = Data relevan yang ditemukan sistem b = Data tidak relevan yang ditemukan sistem c = Data relevan yang tidak ditemukan sistem d = Data tidak relevan yang tidak ditemukan sistem

**BAB III METODOLOGI PENELITIAN**

## 3.1 Metode Pengumpulan Data

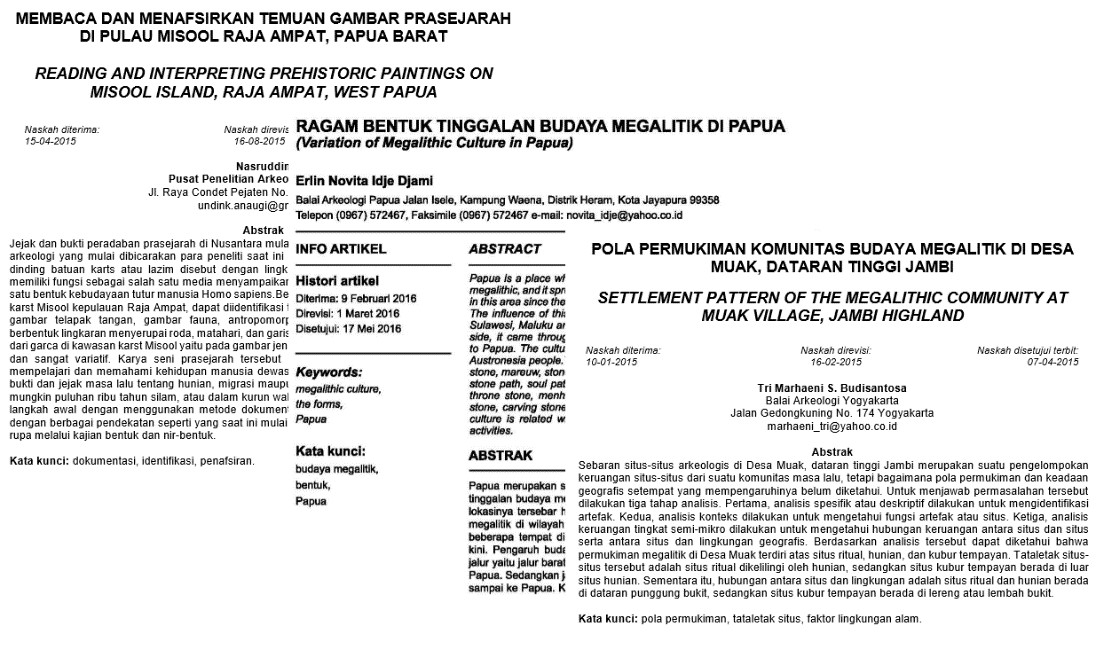
Metode pengumpulan data yang digunakan yaitu studi literatur, digunakan untuk mendapatkan dasar – dasar referensi yang berkaitan dengan judul tugas akhir. Studi literatur dilakukan dengan mengumpulkan literatur yang berkaitan dengan sistem pencarian dan metode *Vector Space Model* (VSM). Sedangkan data artikel arkelogi yang digunakan diambil dari Jurusan Arkeologi Universitas Halu Oleo, sebanyak 215 artikel. Untuk menghitung tingkat kemiripan antara *query* yang di*input*-kan oleh *user* dengan dokumen pada *database* dihitung berdasarkan isi pdf.

Untuk lebih detaildapat dilihat pada Tabel 3.1.

# Tabel 3.1 Detail Data Artikel Arkeologi

|  |  |
| --- | --- |
| Jumlah Dokumen | 215 |
| Format Dokumen | PDF |
| Ukuran Total Dokumen | 376 MB |
| Ukuran Rata-rata Dokumen | ± 2 MB |

Adapun contoh data artikel Arkeologi dapat dilihat pada Gambar 3.1.



## Gambar 3.1 Contoh Data Artikel Arkeologi

25

### 3.2 Metode Pengembangan Sistem

Metode pengembangan sistem yang digunakan pada penelitian ini adalah *Rational Unified Process* (RUP). Proses yang dilakukan pada empat fase RUP dapat dilihat pada Tabel 3.2.

# Tabel 3.2 Fase *Rational Unified Process*

|  |  |
| --- | --- |
| **Fase *Rational Unified Process*** | **Proses yang dilakukan** |
| *Inception* | Analisis pada aplikasi yang akan dibuat. Seperti manfaat dan tujuan pembuatan aplikasi serta batasan masalah yang akan diterapkan. |
| *Elaboration* | Mendesain *UML* sistem, seperti *usecase diagram,* *activity diagram, class diagram* dan *sequence diagram* dari perangkat lunak yang akan dibuat, serta desain antarmuka aplikasi yang akan dibuat. |
| *Construction* | Membuat *interface* aplikasi dan *coding* perangkat lunak. |
| *Transition* | Pada tahap ini meliputi *testing* dan pengujian sistem. Memperbaiki masalah-masalah yang muncul saat pembuatan dan setelah pengujian aplikasi. |

#### 3.3 Waktu dan Tempat Penelitian

##### 3.3.1 Tempat Penelitian

Penelitian tugas akhir bertempat di Jurusan Arkeologi Universitas Halu Oleo dan Laboratorium *Artificial Intelligence* Jurusan Teknik Informatika Universitas Halu Oleo.

##### 3.3.2 Waktu Penelitian

Waktu pelaksanaan penelitian tugas akhir dilaksanakan mulai dari bulan Maret 2020. Rincian kegiatan dapat dilihat pada Tabel 3.3.

27

# Tabel 3.3 Tabel Waktu Penelitian

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Uraian** |  | |  |  | |  |  |  | |  |  | **Waktu (2020)** | | | | | | |  | | | |  | | | |  | | | |
| **April** | |  | **Mei** | |  |  | **Juni** | |  |  | **Juli** | | | **Agustus** | | | | **September** | | | | **Oktober** | | | | **November** | | | |
| **3** | **4** | **1** | **2** | **3** | **4** | **1** | **2** | **3** | **4** | **1** | **2** | **3** | **4** | **1** | **2** | **3** | **4** | **1** | **2** | **3** | **4** | **1** | **2** | **3** | **4** | **1** | **2** | **3** | **4** |
| **1** | ***Inception*** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **2** | ***Elaboration*** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **3** | ***Construction*** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **4** | ***Transition*** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

## 3.1 Analisis Sistem

Analisis sistem adalah suatu tahapan yang bertujuan untuk mengetahui dan mengamati siapa saja yang terlibat dalam sistem. Pembahasan yang ada pada analisis sistem ini yaitu analisis kebutuhan fungsional meliputi perancangan sistem menggunakan bahasa pemodelan UML (*Unified Modeling Language*), perancangan tampilan *interface* serta analisis kebutuhan nonfungsional yang meliputi kebutuhan perangkat keras dan perangkat lunak yang akan digunakan.

### 4.1.1 Kebutuhan Nonfungsional

Kebutuhan nonfungsional adalah sebuah langkah dimana pembangun aplikasi menganalisis sumber daya kebutuhan untuk membangun aplikasi yang akan dibangun. Analisis kebutuhan nonfungsional yang dilakukan dibagi dalam dua tahap, yaitu analisis kebutuhan perangkat keras dan analisis kebutuhan perangkat lunak.

## 1. Kebutuhan Perangkat Keras

Kebutuhan perangkat keras yaitu kebutuhan perangkat atau komponen yang dibutuhkan pada sistem. Untuk menerapkan rancangan yang akan dibangun, dibutuhkan beberapa perangkat keras sebagai sarana untuk mengimplementasikan aplikasi yang dibangun. Spesifikasi minimum laptop atau komputer yang dibutuhkan dapat dilihat pada Tabel 4.1.

# Tabel 4.1 Spesifikasi Minimum Perangkat Keras

|  |  |  |
| --- | --- | --- |
| **No.** | **Nama Perangkat** | **Spesifikasi** |
| 1. | *Processor* | AMD E1-2100 APU |
| 2. | *Memory* | RAM 2.00 GB |

28

### 2. Kebutuhan Perangkat Lunak

Kebutuhan perangkat lunak yaitu kebutuhan perangkat lunak untuk membantu agar komponen perangkat keras dapat berfungsi dan dapat dijalankan pada system. Adapun perangkat lunak atau *software* yang dibutuhkan dapat dilihat pada Tabel 4.2.

# Tabel 4.2 Spesifikasi Perangkat Lunak

|  |  |  |
| --- | --- | --- |
| **No.** | **Nama Perangkat Lunak** | **Spesifikasi** |
| 1. | *Operating System* | *Windows 7, 10* |
| 2. | *Sublime* | *Versi 3.0* |
| 3. | *Xampp* | *Versi 3.2.1* |
| 4. | *Browser* | *Opera, Mozilla, Chrome* |

### 4.1.2 Kebutuhan Fungsional

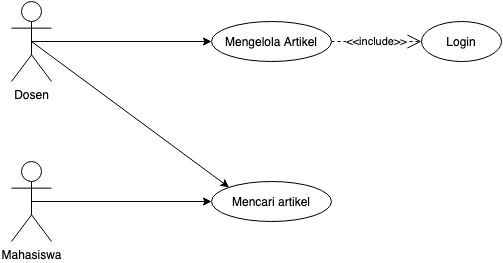
Kebutuhan fungsional adalah segala bentuk data yang dibutuhkan oleh sistem agar sistem dapat berjalan sesuai dengan prosedur yang dibangun melalui perancangan sistem. Adapun kebutuhan fungsional dari sistem yang akan dibangun yaitu Perancangan diagram sistem menggunakan bahasa pemodelan UML (*Unified Modeling Language*) yang meliputi *use case* diagram, *activity* diagram, *sequence* diagram, *class* diagram, dan *design interface* sistem.

### 1. Unified Modeling Language (UML)

Aplikasi dibangun dengan menggunakan *Unified Modeling Language* (UML). UML merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram yang terdiri dari *Use Case* Diagram, *Activity* Diagram, *Sequence Diagram* dan *Class* Diagram.

### a. Use Case Diagram

*Use Case* diagram menjelaskan apa yang akan dilakukan oleh sistem yang akan dibangun dan siapa yang berinteraksi dengan sistem. Berikut adalah *use case* diagram untuk sistem yang akan dibangun :



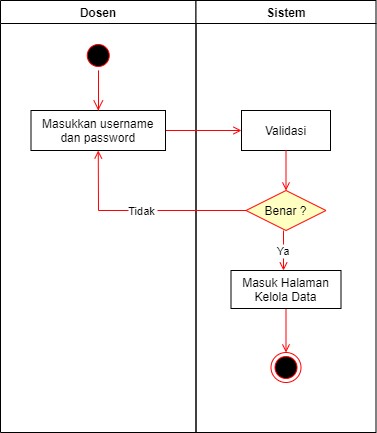
**Gambar 4.1 *Use Case* *Diagram***

### b. Activity Diagram

*Activity* diagram menjelaskan tentang aktivitas apa saja yang terjadi dalam sebuah sistem.

### 1) Activity Diagram Login

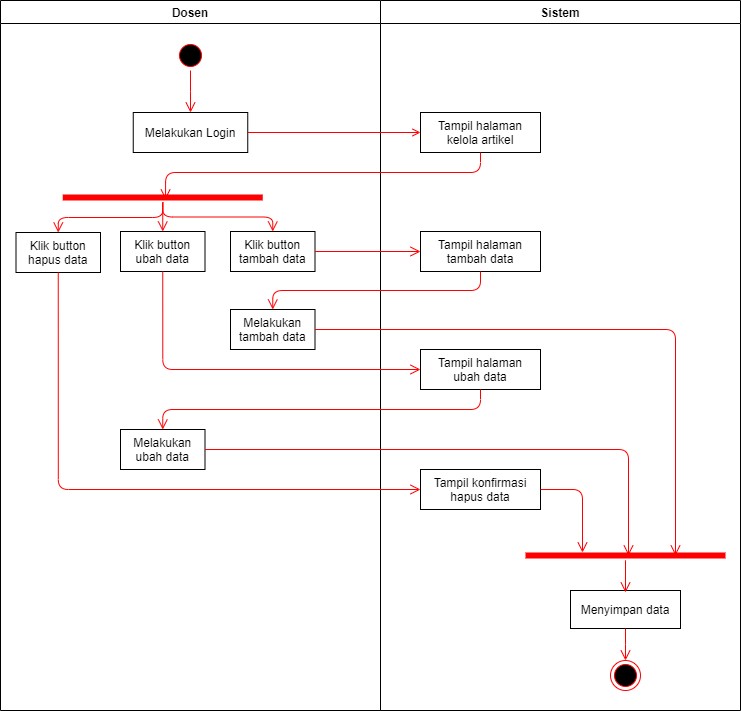
Pada diagram ini dimulai dengan *user* yaitu dosen yang akan melakukan kelola data artikel, mengisi *username* dan *password* kemudian sistem melakukan pengecekan *username* dan *password*, apabila sesuai maka dosen dapat melakukan kelola data, namun apabila data yang dimasukkan tidak *valid* maka sistem akan menampilkan halaman *login*.



**Gambar 4.2 *Activity* *Diagram* *Login***

## 2) *Activity* *Diagram* Kelola Artikel

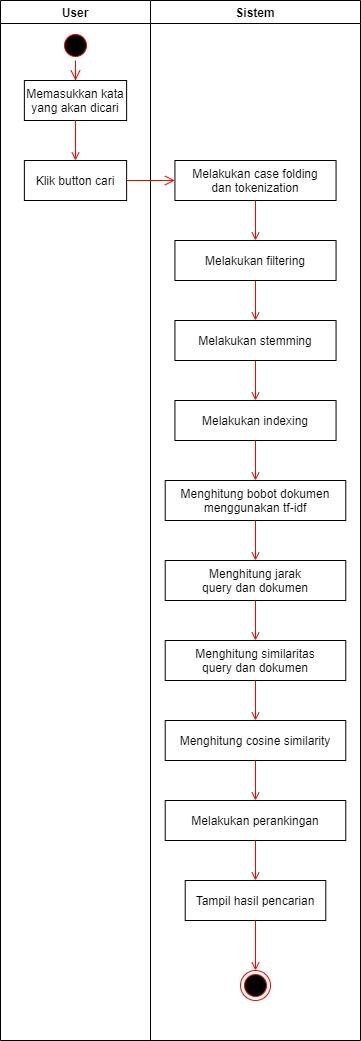
Pada diagram ini menggambarkan dosen yang akan melakukan kelola data artikel, harus melakukan *login* terlebuh dahulu. Apabila akan menambahkan data, *user* mengklik *button* tambah data, kemudian melakukan tambah data, setelah itu data akan tersimpan. Apabila akan mengubah data, *user* mengklik *button* ubah data, kemudian melakukan data data, setelah itu data akan tersimpan. Apabila akan menghapus data, *user* mengklik *button* hapus data, kemudian tampil konfirmasi hapus data, setelah itu data terhapus.



## Gambar 4.3 *Activity* *Diagram* Kelola Artikel

### 3) Activity Diagram Pencarian

Pada diagram ini menggambarkan *user* yang akan melakukan pencarian data artikel, memasukkan kata kunci yang akan dicari, kemudian mengklik *button* cari, lalu sistem akan melakukan *text preprocessing, indexing*, dan perhitungan kemiripan menggunakan *Vector Space Model*, kemudian menampilkan hasil pencarian.



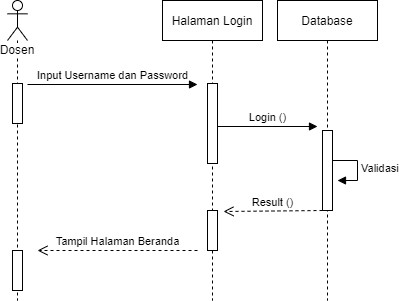
## Gambar 4.4 *Activity* *Diagram* Pencarian

### c. Sequence Diagram

*Sequence* diagram menjelaskan tentang kolaborasi dinamis antara sejumlah *object* untuk menunjukkan rangkaian pesan yang dikirim antar *object* serta interaksi antara *object*.

### 1) Sequence Diagram Login

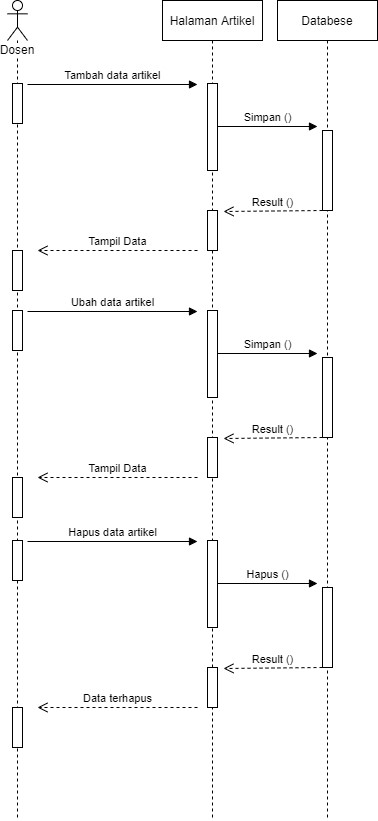
Pada diagram ini digambarkan dosen memasukkan *username* dan *password* pada halaman *login*. Kemudian sistem melakukan fungsi *login* yang dimana data yang diinputkan melewati proses validasi, setelah divalidasi sistem akan mengirimkan hasilnya, jika berhasil maka akan menampilkan halaman beranda.



**Gambar 4.5 *Sequence* *Diagram* *Login***

## 2) *Sequence* *Diagram* Kelola Artikel

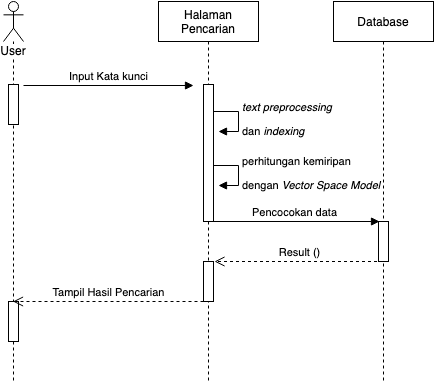
Pada diagram ini menggambarkan dosen yang akan melakukan kelola data artikel, harus melakukan *login* terlebuh dahulu. Apabila akan menambahkan data, *user* mengklik *button* tambah data, kemudian melakukan tambah data, setelah tambah data maka sistem akan melakukan fungsi simpan kemudian sistem akan mengirim hasilnya, jika berhasil maka akan tampil data yang telah diinput. Apabila akan mengubah data, *user* mengklik *button* ubah data, kemudian melakukan data data, setelah ubah data maka sistem akan melakukan fungsi simpan kemudian sistem akan mengirim hasilnya, jika berhasil maka akan tampil data yang telah diubah. Apabila akan menghapus data, *user* mengklik *button* hapus data, kemudian tampil konfirmasi hapus data, setelah itu sistem akan melakukan fungsi hapus kemudian sistem akan mengirim hasilnya, jika berhasil maka data akan terhapus.



## Gambar 4.6 *Sequence* *Diagram* Kelola Artikel

### 3) Sequence Diagram Pencarian

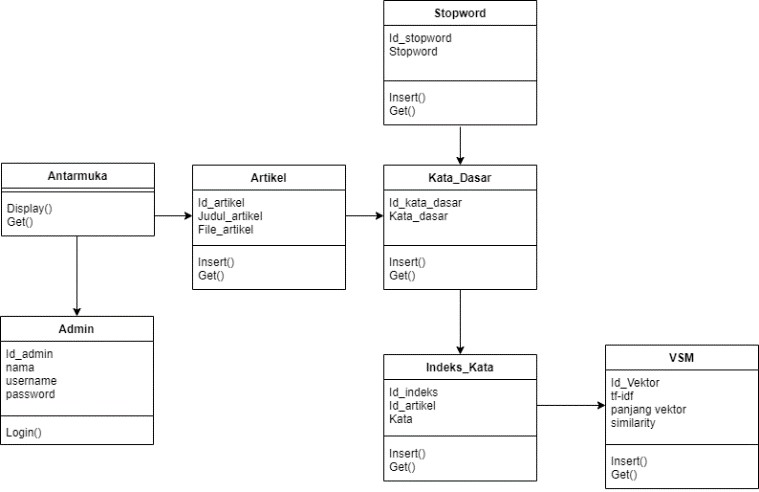
Pada diagram ini menggambarkan *user* yang akan melakukan pencarian data artikel, maka *user* memilih halaman pencarian, kemudian memasukkan kata kunci yang akan dicari, kemudian mengklik *button* cari, lalu sistem akan melakukan *text preprocessing, indexing*, dan perhitungan kemiripan menggunakan *Vector Space Model*, kemudian menampilkan hasil pencarian.



## Gambar 4.7 *Sequence* *Diagram* Pencarian

### d. Class Diagram

Pada diagram ini terdapat 7 *Class* yaitu, *class* *admin*, *class* antarmuka, *class* artikel, *class stopword*, *class* kata dasar, *class* indeks kata, dan *class* VSM.



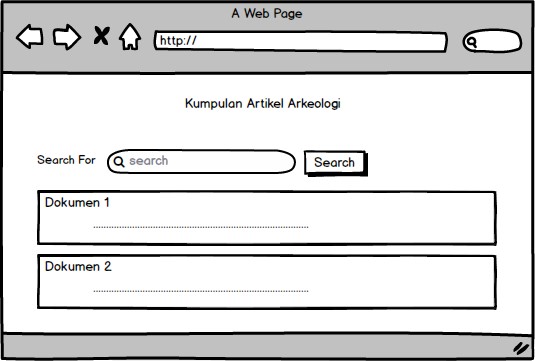
### Gambar 4.8 Class Diagram

#### 2. *User Interface*

Rancangan antarmuka pengguna atau *design user interface* merupakan penggambaran tampilan yang digunakan secara langsung oleh pengguna terhadap sistem. Rancangan *user interface* ini dibuat sederhana agar mudah dimengerti pengguna dan tidak ada kerumitan dalam menjalankannya sehingga mencapai tujuan perangkat lunak yang *user friendly*.

## a. Halaman Utama

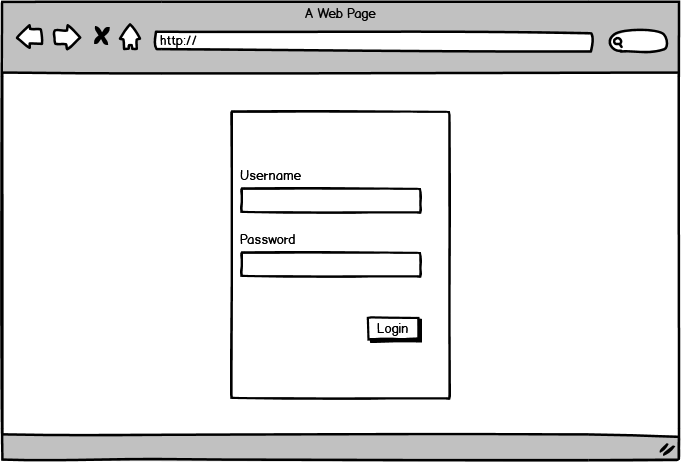
Halaman ini merupakan halaman untuk melakukan pencarian artikel. Pada halaman ini berisi *textfield, button search* dan tabel yang berisi data hasil pencarian.



## Gambar 4.9 Halaman Utama

**b. Halaman *Login***

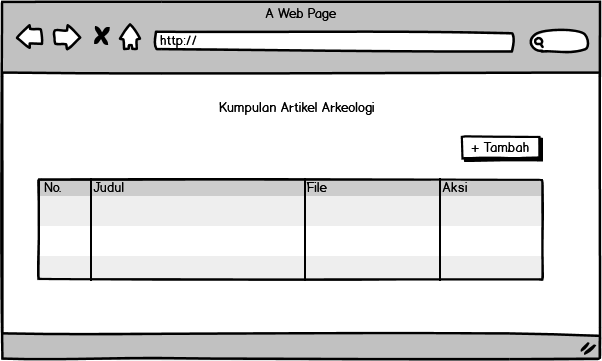
Halaman ini berisi *textfield username*, *textfield password* dan *button login*.



## Gambar 4.10 Halaman *Login*

### c. Halaman Kelola Artikel

Halaman ini berisi *button* tambah dan tabel yang berisi data yang telah diinputkan.



**Gambar 4.11 Halaman Kelola Artikel**