**PROPOSAL**

**PENERAPAN METODE N-GRAM UNTUK MEMPERBAIKI**

**KESALAHAN PENULISAN EJAAN KATA KUNCI (*SPELLING***

***CORRECTION*) PADA APLIKASI PENCARIAN HADIS RIWAYAT BUKHARI**

Diajukan Untuk Memenuhi Salah Satu Syarat Memperoleh Derajat Sarjana

Teknik



**RIFA’ATUS SHALIHAH**

**E1E116028**

**JURUSAN TEKNIK INFORMATIKA**

**FAKULTAS TEKNIK**

**UNIVERSITAS HALU OLEO**

**KENDARI**

**2020**

**DAFTAR ISI**

**HALAMAN JUDUL .......................................................................... i**

**DAFTAR ISI ...................................................................................... xii**

**DAFTAR GAMBAR .......................................................................... xv**

**DAFTAR TABEL .............................................................................. xvii**

[BAB I PENDAHULUAN](#_Toc175016)

[1.1 Latar Belakang 1](#_Toc175017)

[1.2 Rumusan Masalah 3](#_Toc175018)

[1.3 Batasan Masalah 3](#_Toc175019)

[1.4 Tujuan 3](#_Toc175020)

[1.5 Manfaat 3](#_Toc175021)

[1.6 Sistematika Penulisan 4](#_Toc175022)

[1.7 Tinjauan Pustaka 4](#_Toc175023)

[BAB II LANDASAN TEORI](#_Toc175024)

[2.1 Hadis 7](#_Toc175025)

[2.1.1 Imam Bukhari 8](#_Toc175026)

[2.2 *Text Mining* 9](#_Toc175027)

[2.2.1 Tahapan *Text Mining* 10](#_Toc175028)

[2.3 Sistem Temu Kembali Informasi 12](#_Toc175029)

[2.4 *Typographical Error* 14](#_Toc175030)

[2.5 Metode N-gram 14](#_Toc175031)

[2.6 *Jaccard Similarity* 16](#_Toc175032)

[2.7 *Unified Modeling Language* (UML) 18](#_Toc175033)

[2.7.1 Diagram-diagram UML 18](#_Toc175034)

[2.8 *Flowchart* *Diagram* 22](#_Toc175035)

[2.9 Metode Pengembangan Sistem 23](#_Toc175036)

[2.10 Metode Pengujian Sistem 24](#_Toc175037)

[BAB III METODELOGI PENELITIAN](#_Toc175038)

[3.1 Metode Pengumpulan Data 25](#_Toc175039)

[3.2 Metode Pengembangan Sistem 25](#_Toc175040)

[3.3 Waktu dan Tempat Penelitian 26](#_Toc175041)

[3.3.1 Tempat Penelitian 26](#_Toc175042)

[3.3.2 Waktu Penelitian 26](#_Toc175043)

[3.4 Analisis Sistem 28](#_Toc175045)

[3.4.1 Analisis Kebutuhan Non-Fungsional 28](#_Toc175046)

[3.4.2 Analisis Kebutuhan Fungsional 29](#_Toc175047)

[3.2 Perancangan Antar Muka (*Interface*) 42](#_Toc175048)

**DAFTAR PUSTAKA**

**DAFTAR GAMBAR**

**Gambar 2.1** Pembagian Hadits ............................................................................... 8

**Gambar 2.2** Proses *Case Folding* ......................................................................... 10

**Gambar 2.3** Proses *Tokenizing* ............................................................................. 11

**Gambar 2.4** Proses *Filtering* ................................................................................. 11

**Gambar 2.5** Proses *Stemming* ............................................................................... 12

**Gambar 2.6** Sistem Temu Kembali Informasi ...................................................... 13

**Gambar 3.1** Buku Hadis Riwayat Imam al-Bukhari dan Muslim ........................ 25

**Gambar 3.1** *Flowchart Diagram*Sistem .............................................................. 31

**Gambar 3.2** *Flowchart Case Folding* ................................................................... 32

**Gambar 3.3** *Flowchart Tokenizing* ....................................................................... 33

**Gambar 3.4** *Flowchart Filtering* .......................................................................... 34

**Gambar 3.5** *Flowchart Stemming* ......................................................................... 35

**Gambar 3.6** *Flowchart Indexing* ........................................................................... 36

**Gambar 3.7** *Flowchart Diagram* Metode ............................................................. 37

**Gambar 3.8** *Use Case Diagram* ............................................................................ 38

**Gambar 3.9** *Activity Diagram* Cari Hadits ........................................................... 39

**Gambar 3.10** *Activity* *Diagram* Hadits ................................................................. 40

**Gambar 3.11** *Activity* *Diagram* Tentang............................................................... 40

**Gambar 3.12** *Class Diagram* ................................................................................ 41

**Gambar 3.13** *Sequence Diagram* .......................................................................... 42

**Gambar 3.14** Form Cari Hadits ............................................................................ 43

**Gambar 3.15** Form Hadits .................................................................................... 43

**Gambar 3.16** Form Tentang.................................................................................. 43

**DAFTAR TABEL**

**Tabel 2.1** Contoh Proses *Indexing* ......................................................................... 12

**Tabel 2.2** Contoh Penggunaan Bi-gram ................................................................ 15

**Tabel 2.3** Contoh Kata Perhitungan *Jaccard Similarity* ........................................ 16

**Tabel 2.4** Hasil Perhitungan Bi-gram dan *Jaccard Similarity* .............................. 17

**Tabel 2.5** Simbol *Use Case Diagram* .................................................................... 19

**Tabel 2.6** Simbol *Activity* *Diagram* ....................................................................... 20

**Tabel 2.7** Simbol *Class Diagram* .......................................................................... 21

**Tabel 2.8** Simbol *Sequence Diagram* .................................................................... 22

**Tabel 2.9** Simbol *Flowchart Diagram* .................................................................. 22

**Tabel 2.10** Matriks *Recall* dan *precision* ............................................................... 24

**Tabel 3.1** Fase Pengembangan *Software* ............................................................... 26

**Tabel 3.2** Tabel Waktu Penelitian ......................................................................... 27

**Tabel 3.3** Tabel Spesifikasi Perangkat Keras ........................................................ 28

**Tabel 3.4** Tabel Spesifikasi Perangkat Lunak ....................................................... 29

# BAB I

# PENDAHULUAN

## 1.1 Latar Belakang

Hadis berasal dari bahasa Arab Al-Hadîts yang dalam Kamus Besar Bahasa Indonesia berarti sabda atau perbuatan. Hadis adalah perkataan dan perbuatan dari Nabi Muhammad SAW. Hadis secara harfiah berarti perkataan atau percakapan. Dalam terminologi Islam istilah hadis berarti melaporkan/mencatat sebuah pernyataan dan tingkah laku dari Nabi Muhammad SAW.

Hadis merupakan sumber ajaran Islam di samping Al-Qur’an. Tanpa menggunakan hadis, syari’at Islam tidak dapat dimengerti secara utuh dan tidak dapat dilaksanakan. Hadis sebagai sumber hukum kedua setelah Al-Qur’an memiliki otoritas dan posisi penting yang nampaknya sudah tidak perlu dipertanyakan lagi, karena selain sebagai penjelas bagi otoritas wahyu Allah (Al-

Qur’an) juga sebagai sumber hukum Islam ke dua yang menjadi rujukan para *fuqaha* yaitu ahli fiqih atau hukum Islam (ROSYID, 2016).

Pada era globalisasi saat ini informasi telah berkembang dengan pesat. Perkembangan dunia akan teknologi sangat mempengaruhi kebutuhan informasi masyarakat. Salah satunya dalam pengutipan hadis dan penyebaran hadis-hadis di internet dan media sosial yang diketahui begitu marak menyebarkan hadis. Sistem Temu Kembali Informasi (STKI) merupakan sebuah sistem yang mempelajari bagaimana menemukan kembali suatu informasi yang sesuai untuk kebutuhan suatu pengguna disekumpulan informasi secara otomatis. Oleh sebab hal itu perlu dikembangkannya sebuah sistem teknologi di bidang agama diharapkan dapat mempermudah dan memberikan kenyamanan bagi masyarakat luas untuk mencari dan mengetahui ilmu-ilmu Agama Islam pada sebuah buku Hadis. (Sri dkk, 2019). Pentingya penggunaan sistem temu kembali pada pencarian dokumen hadis dikarenakan sistem temu kembali mampu memproses Bahasa alami atau Bahasa manusia sehari-hari (*Natural* *Language*) pada kata kunci pencarian serta merangking daftar dokumen sesuai dengan tingkat relevansinya dengan *query* yang

1

dimasukkan berbeda dengan sistem pencarian biasa yang hanya menampilkan informasi yang benar-benar mirip dengan *query* sehingga informasi yang tidak relevan dengan *query* tidak akan ditampilkan.

Pada suatu sistem temu perlu adanya fitur *spelling correction* untuk memperbaiki kesalahan penulisan kata kunci. Salah satu hal yang sering terjadi pada sistem pencarian dokumen teks adalah kesalahan penulisan atau yang disebut dengan *typographical* *error*. Pada penulisan kata kunci pencarian (*query*) sering terjadi kesalahan penulisan yang diakibatkan oleh beberapa faktor diantaranya timbul ketidaktahuan penulis tentang ejaan bahasa yang benar serta pengguna yang salah menekan tombol *keyboard*.

Metode N-gram adalah salah satu alternatif untuk menyelesaikan masalah pengoreksiaan ejaan kata. N-gram adalah segmen teks yang terdiri dari n-karakter, termasuk pemisah antar kata (biasanya berupa spasi). Pendekatan N-Gram yaitu menggunakan *dictionary look-up* dengan cara membandingkan n-gram dari kata yang salah dengan n-gram kata yang ada pada *database* yang telah diinisialisasi sebelumnya (Wardhana dkk, 2011). Metode N-gram dengan beberapa macam pendekatan menghasilkan akurasi yang bagus. Hasil yang lebih unggul didapatkan dari nilai N sebesar 2, yang disebut dengan bigram, daripada dengan penggunaan N sebesar 3 (trigram). Penggunaan data uji berupa *dataset* Bahasa Inggris menghasilkan akurasi sebesar 84% untuk bigram dan 73% untuk trigram (Ahmed dkk, 2009).

Berdasarkan uraian di atas, maka penulis berkeinginan untuk mengambil judul tugas akhir **“Penerapan Metode N-Gram Untuk Memperbaiki Kesalahan Penulisan Ejaan Kata Kunci (*Spelling correction*) Pada Aplikasi Pencarian Hadis Riwayat Bukhari”.** Sistem pencarian ini dikhususkan hanya untuk hadis yang diriwayatkan oleh salah satu imam besar yaitu Bukhari dengan menerapkan metode N-gram sebagai fitur *spelling correction*. Diharapkan sistem ini dapat membantu memberikan gambaran yang cukup baik bagaimana kinerja dari metode N-gram sebagai fitur *spelling correction* dalam memperbaiki kesalahan penulisan kata kunci pada dokumen hadis riwayat Bukhari.

## 1.2 Rumusan Masalah

Berdasarkan latar belakang tersebut, maka rumusan masalah pada penelitian ini adalah bagaimana kinerja dari metode N-gram dalam memperbaiki kesalahan penulisan kata kunci pada sistem temu pencarian hadis riwayat Bukhari.

## 1.3 Batasan Masalah

Adapun batasan masalah yang akan dibahas dalam penelitian ini adalah sebagai berikut:

1. Menggunakan algoritma N-gram untuk memperbaiki kesalahan penulisan kata kunci pada sistem temu pencarian hadis riwayat bukhari.
2. Sistem ini dibuat menggunakan *python* 3.8.
3. Pengimplementasian metode N-gram pada tugas akhir ini dibatasi pada Bigram.
4. Dokumen yang akan diolah adalah dokumen Bahasa Indonesia dengan format txt.
5. Koreksi untuk huruf, tidak untuk angka.

## 1.4 Tujuan

Adapun tujuan penelitian ini adalah dapat menganalisis kinerja dari penerapan metode N-gram dalam memperbaiki kesalahan penulisan kata kunci pada sistem temu hadis riwayat Bukhari.

## 1.5 Manfaat

Adapun manfaat dari penelitian ini adalah sebagai berikut :

1. Mempermudah *user* dalam menemukan hadis Bukhari dan juga mempermudah untuk mengamalkan hadis Bukhari dalam kehidupan sehari-hari.
2. Diharapkan metode N-gram dapat memperbaiki kesalahan penulisan kata kunci pada sistem temu pencarian hadis riwayat imam Bukhari.

## 1.6 Sistematika Penulisan

Sistematika penulisan tugas akhir ini terdiri atas beberapa bagian utama sebagai berikut:

###### BAB I PENDAHULUAN

Merupakan bab pendahuluan yang menguraikan latar belakang masalah, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, dan sistematika penulisan.

###### BAB II LANDASAN TEORI

Membahas mengenai dasar-dasar teori pendukung yang berhubungan dengan masalah yang diambil dan program aplikasi yang akan digunakan dalam pembangunan sistem.

###### BAB III METODE PENELITIAN

Membahas mengenai waktu dan tempat penelitian, metode pengumpulan data, metode pembangunan sistem yang digunakan dalam penelitian tugas akhir ini.

## 1.7 Tinjauan Pustaka

Penelitian ini dibuat berdasarkan penelitian sebelumnya tentang analisis penerapan metode N-gram.

Penelitian sebelumnya oleh Aida Indriani, dkk (2018) dengan judul Implementasi *jaccard* *index* dan n-gram pada rekayasa aplikasi koreksi kata berbahasa Indonesia, data yang digunakan yaitu semua kata yang ada pada Kamus Besar Bahasa Indonesia. Setelah melakukan uji coba dan hasil analisa terhadap implementasi *jaccard* *index* dan fitur N-gram pada aplikasi koreksi kata, maka dapat disimpulkan bahwa aplikasi yang dibuat dapat membantu tim editor dalam melakukan koreksi kata.

Penelitian selanjutnya oleh Arina Indana Fahma, dkk (2018) dengan judul Identifikasi Kesalahan Penulisan Kata (*Typographical Error*) pada Dokumen Berbahasa Indonesia Menggunakan Metode N-gram dan *Levenshtein Distance*. Berdasarkan hasil pengujian dan analisis pada identifikasi *typographical error* pada dokumen bahasa Indonesia menggunakan metode N-gram dan *Levenshtein Distance* dapat diambil kesimpulan bahwa metode pendekatan *Dictionary Lookup* pada proses identifikasi *typographical* *error* pada dokumen bahasa Indonesia dapat diterapkan dengan baik untuk mencari kata *typographical* *error* dalam dokumen data *input*. Untuk menentukan kandidat kata, metode *Levenshtein Distance* dapat menghasilkan kandidat kata yang sesuai dengan nilai aktual yang diharapkan *user*. Namun untuk kata *typographical* *error* tertentu, jumlah kandidat kata yang ditampilkan dalam sistem terlalu banyak. Hasil presisi dan *recall* pada penelitian ini memiliki nilai yang beragam pada setiap skenario pengujian. Nilai presisi terbaik yang dihasilkan sistem sebesar 0.97 pada skenario pengujian *typographical* *error* jenis *insertion*. Sementara itu, nilai *recall* terbaik yang dihasilkan sistem sebesar 1 pada skenario pengujian *typographical* *error* jenis *substitution*.

Penelitian selanjutnya oleh Badruz Zaman dkk, (2015) dengan judul Sistem Deteksi Bahasa pada Dokumen menggunakan N-Gram. Data yang dikumpulkan berupa dokumen bahasa Indonesia dan Inggris sejumlah 110 dokumen, 50 dokumen untuk data *training*, dan 60 dokumen untuk data *testing*. Sistem deteksi bahasa menggunakan N-Gram menunjukkan kinerja yang sangat baik dalam mendeteksi bahasa dari dokumen. Hal ini ditunjukkan dengan hasil rata-rata *F-measures* sebesar 0,93.

Penelitian selanjutnya oleh Erick Alfons Lisangan (2015) dengan judul Implementasi N-gram *Technique* Dalam Deteksi Plagiarisme Pada Tugas Mahasiswa. Data yang dijadikan sebagai sampel adalah data tugas mata kuliah Jaringan Komputer pada Semester Akhir 2012/2013. Proses studi dokumentasi mengambil dua buah sampel tugas, yaitu A dan B dimana keduanya memiliki kesamaan isi dokumen sebesar 10%. Dari kedua dokumen tugas tersebut kemudian dijadikan acuan untuk isi dari beberapa dokumen. Kesimpulan yang dapat ditarik dari penelitian ini adalah N-gram *technique* dapat digunakan untuk mendeteksi plagiarisme tugas mahasiswa berdasarkan *similarity text* antar dokumen tugas. Semakin besar nilai n yang diberikan akan memberikan tingkat relevansi yang semakin baik dengan syarat nilai rata-rata selisih relevansi lebih besar dari 0. Dalam penelitian ini, nilai n terbaik yang diperoleh adalah 7.

Penelitian selanjutnya oleh Wedha Satya Wardhana, dkk (2011) dengan judul Pengoreksian ejaan kata menggunakan metode n-gram (studi kasus dokumen teks berbahasa indonesia). Dalam penelitian ini dapat disimpulkan bahwa metode bigram dan trigram dapat diimplementasikan untuk melakukan pengoreksian ejaan kata dalam teks berbahasa Indonesia. Jenis kesalahan kata mempengaruhi performansi, *error* kata yang disebabkan oleh *insertion* dan *deletion* dapat dikoreksi dengan baik oleh algoritma n-gram.

Berdasarkan penelitian-penelitian yang telah dipaparkan di atas dapat disimpulkan bahwa metode N-gram telah banyak digunakan pada sistem koreksi kata, identifikasi Bahasa, serta plagiarisme dokumen. Sehingga sangat memungkinkan menerapkan metode N-gram dan *Jaccard Similarity* pada sistem pencarian hadis yang menggunakan fitur *spelling correction* untuk mendeteksi kesalahan penulisan ejaan kata kunci pada sistem pencarian hadis riwayat Bukhari.

# BAB II

# LANDASAN TEORI

## 2.1 Hadis

Hadis merupakan sumber ajaran Islam disamping Al-Qur’an. Tanpa menggunakan hadis, syari’at Islam tidak dapat dimengerti secara utuh dan tidak dapat dilaksanakan. Hadis sebagai sumber hukum kedua setelah Al-Qur’an memiliki otoritas dan posisi penting yang nampaknya sudah tidak perlu dipertanyakan lagi, karena selain sebagai penjelas bagi otoritas wahyu Allah (Al-

Qur’an) juga sebagai sumber hukum Islam kedua yang menjadi rujukan para *fuqaha* yaitu ahli fiqih atau hukum Islam (ROSYID, 2016).

Apabila dilihat dari segi riwayat, penyampaian secara lisan sesuatu keterangan dari Rasulullah maka jadilah hadis yang mempunyai kualitas yang bertingkat-tingkat, ada yang kuat dan ada yang lemah. Sedangkan dalam menyampaikan sebuah hadis terkadang Nabi berhadapan dengan orang-orang yang jumlahnya amat banyak, terkadang dengan beberapa orang, terkadang pula hanya dengan satu atau dua orang saja.

Hadis sebagai sumber asasi dan sumber hukum Islam yang kedua setelah Al-Quran. Kedudukannya sebagai sumber setelah Al-Quran adalah disebabkan karena kedudukannya sebagai penafsir, dan pedoman pelaksanaan yang otentik terhadap Al-Quran. Ia menafsirkan dan menjelaskan ketentuan yang masih dalam garis besar atau membatasi keumuman, atau menyusuli apa yang disebut oleh AlQuran. Di samping itu hadis-hadis Nabi dalam kaitannya dengan Al-Quran mempunyai fungsi menetapkan dan memperkuat hukum-hukum yang telah ditentukan oleh Al-Quran. Maka dalam hal ini keduanya sama-sama menjadi sumber hukum, begitu pula hadis memberikan perincian dan penafsiran ayat-ayat Al-Quran yang masih global dan lain sebagainya.

Hadis yang dapat dijadikan pegangan dasar hukum sesuatu perbuatan haruslah diyakini benar-benar akan kebenarannya. Karena kita tidak mendengar

hadis itu langsung dari Nabi Muhammad SAW. Maka jalan penyampaian hadis itu harus dapat memberikan keyakinan tentang kebenaran hadis tersebut.

Kita telah memahami bahwa sejumlah hadis diriwayatkan oleh beberapa orang sahabat dan tabiin, namun sejumlah hadis lainnya hanya dinukilkan oleh seorang sahabat, kemudian diteruskan juga oleh seorang *tabiin*, yang hanya mempunyai seorang murid yang meriwayatkan hadis. Oleh sebab itu perlu melihat keberadaan hadis-hadis tersebut dinilai berdasarkan jumlah perawinya (Zuhri, 2008). Hadis dibagi berdasarkan kuantitas dan kualitasnya seperti pada gambar dibawah ini.

Hadis

ntitas

Kua

litas

Ku

*Mut*

*awatir*

*Ahad*

*Maknawi*

*Lafzi*

*Qarib*

*Azi*

*z*

*Masyhur*

*Shahih*

*Hasan*

*Daif*

###### Gambar 2.1 Pembagian Hadis

#### 2.1.1 Imam Bukhari

Nama lengkapnya adalah Abu Abdillah Muhammad bin Ismail bin Ibrahim bin al-Mughirah al-Ju’fi. Beliau dilahirkan hari Jum’at, 13 Syawal 194 H di

Bukhara. Ayahnya, Isma’il, adalah seorang ulama hadis pula yang berguru pada sejumlah ulama termasyhur, seperti Malik bin Anas, Hammad bin Zaid dan Ibn Mubarak. Imam Bukhari mulai belajar hadits pada saat beliau masih sangat remaja, bahkan belum mencapai usia sepuluh tahun. Sebelum mencapai usia 16 tahun, Bukhari telah berhasil menghafalkan beberapa buah buku ulama, seperti Ibn

Mubarak, Waqi’ dan lain-lain. Beliau tidak hanya menghafal matan hadits atau buku ulama terdahulu, tetapi juga mengenal betul biografi para perawi yang mengambil bagian dan penukilan sejumlah hadits, baik data tanggal dan tempat lahir, tanggal dan tempat meninggal dan sebagainya. Beliau menetap di Hijaz selama enam tahun untuk mempelajari hadits dan mengembara ke Baghdad sebanyak delapan kali. Suatu saat ulama Baghdad menguji kekuatan daya hafalan Imam Bukhari, yang konon pada waktu itu kemasyhuran hafalan beliau mengguncangkan banyak ulama. Mereka menunjuk sepuluh ulama untuk menguji hafalan Bukhari. Setiap ulama tersebut mengganti sanad hadits satu dan menempatkannya pada hadits lain secara acak pada matan yang berbeda. Satu demi satu penanya menyampaikan pertanyannya. Dan setelah semua penanya selesai membacakan dan menyampaikan pertanyaan, Imam Bukhari secara sistematis menerangkan kepada mereka sanad mana yang tepat untuk matan hadits yang mereka bacakan dan tanyakan. (Masrukhin, 2016)

Al-Bukhari juga sudah mulai menulis kitab-kitabnya sejak umur 18 tahun hingga dalam masa rihlah ilmunya ini. Sedangkan mengenai sebab penulisan alJâmi’ al-Sahîh, adalah karena anjuran dari gurunya, Imam Ishaq bin Rahawaih alHanzhali, untuk meringkas hadis-hadis Nabi SAW yang sahih dalam suatu kitab. Juga, mimpi al-Bukhari bertemu dengan Rasulullah SAW di mana ia melindungi Rasulullah SAW dengan sebuah kipas. Mimpi ini ditafsirkan bahwa ia kelak akan membela Rasulullah SAW dari para pendusta atas hadis-hadisnya. Murid-murid alBukhari antara lain, Imam al-Tirmidzi, Imam al-Marwazi, Imam Ibnu Khuzaimah,

Imam Abu Ali Salih al-Asadi, Imam Abu Ja’far al-Hadhrami, dan lain-lain.

Menjelang wafatnya, al-Bukhari mendapat cobaan dan fitnah sehingga diusir dari Naisabur dan Bukhara. Akhirnya, ia singgah di Khartank, salah satu desa pinggiran Samarkand dan tinggal di rumah kerabatnya, Ghalib bin Jibril. Di desa tersebut, ia berdoa: “Ya Allah! bumi ini sekarang menjadi sempit bagiku, maka cabutlah nyawaku.” Setelah beberapa hari, ia pun sakit dan meninggal di Khartank pada hari Jum’at malam Idul Fitri tahun 256 H./ 870 M. (Abdurrahman, 2012)

## 2.2 Text Mining

*Text* *Mining* atau sering disebut Pemrosesan Teks merupakan salah satu bidang pengetahuan pada *Artificial* *Intelligence* yang menerapkan konsep dan teknik data mining untuk mencari pola dalam teks dengan proses ekstraksi pola yang berupa informasi dan pengetahuan berguna dari sejumlah besar sumber data yang tidak terstruktur. Dalam *text* *mining* dilakukan penambangan data yang berupa teks dimana sumber data biasanya didapatkan dari dokumen, dan tujuannya adalah mencari kata-kata yang memberikan penjelasan isi dari dokumen sehingga dapat dilakukan analisa keterkaitan antar dokumen (Baskoro dkk, 2015).

#### 2.2.1 Tahapan Text Mining

Berdasarkan ketidakteraturan struktur data teks, maka proses *text* *mining* memerlukan beberapa tahap awal yang pada intinya adalah mempersiapkan agar teks dapat diubah menjadi lebih terstruktur. Menurut ketidakaturan struktur data dalam teks, maka proses *text* *mining* membutuhkan tahapan-tahapan awal agar teks berubah menjadi data yang lebih terstruktur. Salah satu tahapan tersebut adalah *preprocessing* yang meliputi *case* *folding*, *tokenizing*, *filtering* dan *stemming* (Fahma dkk, 2018).

1. *Case* *Folding*

*Case* *folding* yaitu merubah semua karakter huruf pada sebuah kalimat menjadi huruf kecil dan menghilangkan karakter yang dianggap tidak valid seperti angka, tanda baca, dan *Uniform Resources Locator* (URL) (Indraloka & Santosa, 2017).

Contoh penggunaan *case* *folding* dapat dilihat pada Gambar 2.2.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  | | --- | | Bagaimana cara menenangkan hati ? | |  | |  | | --- | | bagaimana cara menenangkan hati | |

###### Gambar 2.2 Proses *Case* *Folding*

2. *Tokenizing*

*Tokenizing* yaitu memotong sebuah kalimat berdasarkan tiap kata yang menyusunnya (Indraloka & Santosa, 2017). Contoh penggunaan *tokenizing* dapat dilihat pada Gambar 2.3.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  | | --- | | bagaimana cara menenangkan hati | |  | |  | | --- | | bagaimana  cara  menenangkan  hati | |

###### Gambar 2.3 Proses *Tokenizing*

3. *Filtering*

*Stopwords removal* atau *filtering* merupakan proses penghilangan kata tidak penting pada deskripsi melalui pengecekan kata-kata. Hasil dari *parsing* deskripsi apakah termasuk di dalam daftar kata tidak penting (*stoplist*) atau tidak (Melita,

2018). Contoh stopword adalah “yang”, “dan”, “di”, “dari”, dan lain-lain. Contoh penggunaan *filtering* dapat dilihat pada Gambar 2.4.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  | | --- | | bagaimana  cara  menenangkan  hati | |  | |  | | --- | | cara  menenangkan  hati | |

###### Gambar 2.4 Proses *Filtering*

4. *Stemming*

*Stemming* yaitu merubah berbagai kata berimbuhan menjadi kata dasarnya, tahap ini pada umumnya dilakukan untuk teks dengan bahasa Inggris, karena teks dengan bahasa Inggris memiliki struktur imbuhan yang tetap (Indraloka & Santosa, 2017). Contoh penggunaan *stemming* dapat dilihat pada Gambar 2.5.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  | | --- | | cara  menenangkan  hati | |  | |  | | --- | | cara tenang hati | |

###### Gambar 2.5 Proses *Filtering*

2.3 Indexing

*Indexing* adalah memetakan kembali dari setiap *term* ke dokumen dimana *term* tersebut muncul. Pengindeksan dilakukan dengan membuat *inverted* *index* (atau disebut juga *inverted* *file*) yang terdiri dari *dictionary* dan *postings*. *Inverted* *index* merupakan konversi dari dokumen asli yang mengandung sekumpulan kata ke dalam daftar kata (*dictionary*) yang berasosiasi dengan dokumen terkait dimana kata-kata tersebut muncul (*postings*). *Dictionary* adalah daftar kata yang diperoleh dari hasil pengindeksan dokumen (Bintana, 2012). Contoh *indexing* dapat dilihat pada Tabel 2.1.

###### Tabel 2.1 Contoh proses *indexing*

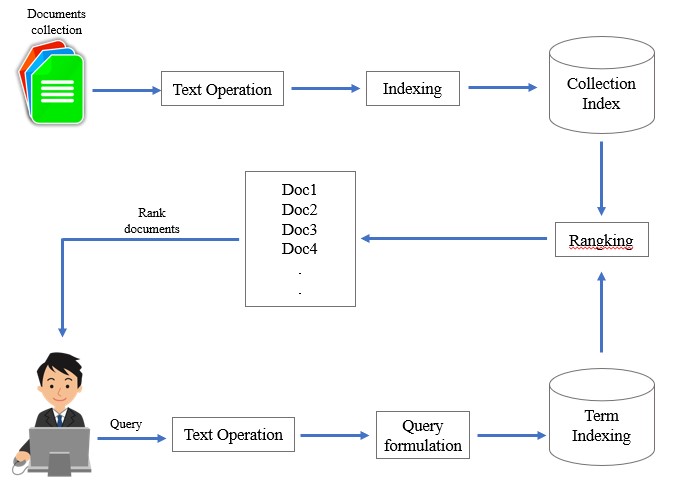
|  |  |  |
| --- | --- | --- |
| Nama Dokumen | Dok1 | Dok2 |
| Cara | 1 | - |
| Tenang | 1 | - |
| Hati | 1 | 1 |

## 2.4 Sistem Temu Kembali Informasi

*Information* *Retrieval* (IR) merupakan suatu cara yang digunakan untuk menemukan kembali (*retrieve*) informasi-informasi yang relevan terhadap kebutuhan pengguna dari suatu kumpulan informasi secara otomatis. Salah satu aplikasi umum dari IR *system* adalah *search* *engine* atau mesin pencarian yang terdapat pada jaringan internet. Pengguna dapat mencari halaman-halaman *web* yang dibutuhkannya melalui *search* *engine*. Contoh lain dari IR *system* adalah sistem informasi perpustakaan. IR *system* terutama berhubungan dengan pencarian informasi yang isinya tidak memiliki struktur. Ekspresi kebutuhan pengguna yang disebut *query*, juga tidak memiliki struktur. Hal ini yang membedakan IR *system* dengan sistem basis data. Dokumen adalah contoh informasi yang tidak terstruktur. Isi dari suatu dokumen sangat tergantung pada pembuat dokumen tersebut. Sebagai suatu sistem, IR *system* memiliki beberapa bagian yang membangun sistem secara keseluruhan. Berikut adalah bagian dari IR *system* (Bunyamin, 2005) :

1. *Text Operations* (operasi terhadap teks) yang meliputi pemilihan kata-kata dalam *query* maupun dokumen (*term* *selection*) dalam pentransformasian dokumen atau *query* menjadi *term* *index* (indeks dari kata-kata).
2. *Query* *formulation* (formulasi terhadap *query*) yaitu memberi bobot pada indeks katakata *query*.
3. Rangking (perangkingan), mencari dokumen-dokumen yang relevan terhadap *query* dan mengurutkan dokumen tersebut berdasarkan kesesuaiannya dengan *query*.
4. *Indexing* (pengindeksan), membangun basis data indeks dari koleksi dokumen.

Dilakukan terlebih dahulu sebelum pencarian dokumen dilakukan.



###### Gambar 2.6 Proses Sistem Temu Kembali Informasi

IR *system* menerima *query* dari pengguna, kemudian melakukan perangkingan terhadap dokumen pada koleksi berdasarkan kesesuaiannya dengan *query*. Hasil perangkingan yang diberikan kepada pengguna merupakan dokumen yang menurut sistem relevan dengan *query*. Namun relevansi dokumen terhadap suatu *query* merupakan penilaian pengguna yang subjektif dan dipengaruhi banyak faktor seperti topik, pewaktuan, sumber informasi maupun tujuan pengguna. Model IR *system* menentukan detail IR system yaitu meliputi representasi dokumen maupun *query*, fungsi pencarian (*retrieval* *function*) dan notasi kesesuaian (*relevance* *notation*) dokumen terhadap *query* (Bunyamin, 2005).

## 2.5 Typographical Error

*Typographical* *error* merupakan kesalahan yang terjadi pada saat proses mengetik teks dan dapat mengubah arti dari suatu kata bahkan arti dari suatu kalimat. Istilah ini mencakup kesalahan karena kegagalan mekanis atau slip tangan atau jari, dan juga timbul akibat ketidaktahuan penulis seperti kesalahan ejaan. *Typographical* *error* dapat disebabkan oleh jari menekan dua tombol *keyboard* yang berdekatan secara bersamaan.

*Typographical* *error* ini bervariasi mulai dari kesalahan ketik biasa sampai kesalahan dalam tatanan bahasa yang digunakan atau bahkan pengertian dari kata tersebut. Kesalahan-kesalahan tersebut dikategorikan ke dalam 2 jenis yaitu *nonword error* dan *real word error*. *Non-word error* adalah *error* yang tidak terdapat makna di dalamnya sedangkan pada *real-word error*, kata yang tertulis bernilai benar atau bisa disebut mempunyai arti dalam kamus namun tidak dimaksudkan dalam kalimat tersebut maupun mempunyai arti yang berbeda dan bahkan kalimat tersebut memiliki tata bahasa yang salah (Fahma, 2018).

## 2.6 Metode N-gram

Ide penggunaan N-gram telah diterapkan untuk berbagai masalah seperti prediksi kata, koreksi ejaan, pengenalan suara, koreksi kata terjemahan dan pencarian *string*. Salah satu keuntungan dari metode N-gram ini adalah bahwa bahasa bersifat independen. Dalam koreksi ejaan, N-gram merupakan urutan sebanyak N huruf dalam sebuah kata atau *string*. N-gram dapat digunakan untuk menghitung kesamaan antara dua *string* dengan cara menghitung jumlah N-gram yang sama. Semakin banyak jumlah N-gram yang sama antara 2 kalimat yang ada maka semakin mirip (Fahma dkk, 2018).

N-gram dapat dibedakan berdasarkan berapa jumlah huruf yang dipergunakan dalam pemisahan huruf untuk setiap kata antara lain dengan nilai n = 1, 2, 3. n = 1 biasanya disebut dengan Uni-gram, n = 2 disebut dengan Bi-gram, dan n = 3 disebut dengan Tri-gram. Misalnya dalam kata “TYPO” akan didapatkan N gram sebagai berikut (Indriani dkk, 2018):

Uni-gram: T, Y, P, O

Bi-gram: \_T, TY, YP, PO, O\_

Tri-gram: \_TY, TYP, YPO, PO\_, O\_\_

Pada penelitian Hamzah, (2010) membandingkan antara waktu deteksi dan akurasi dari unigram, bigram, dan trigram dan menarik kesimpulan bahwa akurasi akan semakin tinggi jika n semakin tinggi, akan tetapi waktu deteksi juga akan semakin lama. Unigram akan cenderung cepat dalam waktu deteksi dan akurasi cenderung lebih rendah. Bigram cenderung memiliki akurasi tinggi namun waktu deteksi yang lebih lama dari unigram dan trigram memliki waktu deteksi yang lebih lama dari bigram dan memiliki akurasi yang tinggi. Sehingga pada penelitian ini digunakan Bi-gram sebagai nilai tengah. Berikut adalah contoh penggunaan Bigram:

###### Tabel 2.2 Contoh pengunaan Bi-gram

|  |  |  |
| --- | --- | --- |
| No | Kata | Bi-gram |
| 1 | Desa | \_d, de, es, sa, a\_ |
| 2 | Ini | \_i, in, ni, i\_ |
| 3 | penggunaan | \_p, pe, en, ng, gg, gu, un, na, aa, an, n\_ |
| 4 | Sosial | \_s, so, os, si, ia, al, l\_ |
| 5 | Media | \_m, me, ed, di, ia, a\_ |
| 6 | Sebagai | \_s, se, eb, ba, ag, ga, ai, i\_ |
| 7 | Pencari | \_p, pe, en, nc, ca, ar, ri, i\_ |

###### Lanjutan Tabel 2.2 Contoh pengunaan Bi-gram

|  |  |  |
| --- | --- | --- |
| No | Kata | Bi-gram |
| 8 | Informasi | \_i, in, nf, fr, ro, om, ma, as, si, i\_ |
| 9 | Semakin | \_s, se, em, ma, ak, ki, in, n\_ |
| 10 | meningkat | \_m, me, en, ni, in, ng, gk, ka, at, t\_ |

Kata yang terdapat pada *database* yang akan dijadikan sebagai pembanding juga mengalami proses pemotongan karakter dengan Bigram.

## 2.7 Jaccard Similarity

*Jaccard* *Similarity* atau *Jaccard Coefficient* merupakan algoritma yang fungsinya untuk membandingkan dua sampel yaitu dokumen yang satu dengan yang lainnya berdasarkan kata yang dimilikinya. *Jaccard* *similarity* biasanya digunakan untuk membandingkan dokumen dan menghitung nilai kemiripan (*similarity*) dari dua buah objek atau dokumen (Sunardi dkk, 2018). Rumus *Jaccard* *similarity* dapat dilihat pada persamaan 2.1.

*Similarty (X,Y)* = |𝑋 ∩𝑌| (2.1)

|𝑋∪𝑌|

Dimana:

1. = Dokumen 1
2. = Dokumen 2

Rumus 2.1 merupakan dari rumus *Jaccard* *Similarity* atau *Jaccard* *Coefficient* yang digunakan untuk mencari persamaan dan perbedaan pada dua sampel. Pada Tabel 2.3 terdapat kata yang mengalami *typo* dan kata dalam Kamus Besar Bahasa Indonesia.

###### Tabel 2.3 Contoh kata perhitungan *Jaccard* *Similarity*

|  |  |  |  |
| --- | --- | --- | --- |
| No | Kata *Preprocessing* |  | Kata dalam *Database* |
| 1. | deasa | 1 | dewasa |
| 2 | devisa |
| 3 | delusi |

###### Lanjutan Tabel 2.3 Contoh kata perhitungan *Jaccard* *Similarity*

|  |  |  |  |
| --- | --- | --- | --- |
| No | Kata *Preprocessing* |  | Kata dalam *Database* |
| 2. | mdia | 4 | melia |
| 5 | media |
| 6 | medan |
| 3. | infromasi | 7 | informasi |
| 8 | informan |
| 9 | info |

Kata-kata tersebut sudah mengalami proses Bi-gram dan langsung masuk pada perhitungan *Jaccard Similarity.* Berikut adalah perhitungan *Jaccard*

*Similarity*:

A1 = {\_d, \_de, \_ea, \_as, \_sa, \_a}

B1 = {\_d, \_de, \_ew, \_wa, \_as, \_sa, \_a} *Similarty (A1,B1)* = |𝐴1 ∩𝐵1|

|𝐴1∪𝐵1|

A1∩ B1 = {\_d, \_de, \_as, \_sa, \_a} = 5

A1∪ B1 = {\_d, \_de, \_ea, \_ew, \_as, \_wa, \_sa, \_a} = 8

*Similarty (A1,B1)* = 

*Similarty (A1,B1)* = 0,625

Contoh di atas adalah salah satu perhitungan yang dilakukan oleh *Jaccard Similarity.* Untuk hasil perhitungan keseluruhan dapat dilihat pada Tabel 2.4.

###### Tabel 2.4 Hasil perhitungan Bi-gram dan *Jaccard* *Similarity*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Kata *Preprocessing* |  | Kata dalam *Database* | Hasil perhitungan Bi-gram dan *Jaccard* *Similarity* |
| 1. | deasa | 1 | dewasa | 0,625 |
| 2 | devisa | 0,444 |
| 3 | delusi | 0,182 |
| 2. | mdia | 4 | melia | 0,375 |
| 5 | media | 0,571 |
| 6 | medan | 0,1 |
| 3. | infromasi | 7 | informasi | 0,538 |
| 8 | informan | 0,267 |
| 9 | info | 0,250 |

Dari hasil perhitungan di atas dapat dilihat bahwa kata “deasa” terhadap kata

“dewasa” memiliki tingkat kesamaan sebesar 0,625 yang memiliki nilai tertinggi.

Begitu juga dengan kata “mdia” terhadap kata “media” dan kata “infromasi” terhadap kata “informasi” yang memiliki nilai kesamaan tertinggi.

## 2.8 Unified Modeling Language (UML)

*Unified Modeling Language* (UML) adalah salah satu standar Bahasa yang banyak digunakan didunia industri untuk mendefinisikan *requirement*, membuat analisis dan desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek.

UML muncul karena adanya kebutuhan pemodelan visual untuk menspesifikasikan, menggambarkan, membangun, dan dokumentasi dari sistem perangkat lunak. UML merupakan Bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung. UML hanya berfungsi untuk melakukan pemodelan. Jadi penggunaan UML tidak terbatas pada metodologi tertentu, meskipun pada kenyataannya UML paling banyak digunakan pada metodologi berorientasi objek (Salahuddin, 2015).

#### 2.8.1 Diagram-diagram UML

UML memiliki diagram-diagram tertentu dan terstandar untuk memodelkan perangkat lunak terstruktur. Berikut adalah diagram-diagram dari UML.

1. *Use* *case diagram*

*Use case* diagram merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use* *case* mendeskripsikan sebuah interaksi anatar satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar, *use* *case* digunakan untuk mengetahui fungi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu. Berikut adalah simbol-simbol yang ada pada diagram *use* *case*:

Tabel 2.5 Simbol use case diagram

|  |  |  |
| --- | --- | --- |
| **Simbol** | **Nama** | **Deskripsi** |
|  | *Use Case* | Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama *use case*. |
|  | Aktor | Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri |
|  | Asosiasi | Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor. |
|  | Ekstensi (*extend*) | Relasi *use case* tambahan ke sebuah *use case*  dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu. |
|  | Generalisasi | Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari yang lainnya. |
|  | Menggunakan  (*include*) | 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. *Activity diagram*

Diagram aktivitas atau *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. Berikut adalah simbol-simbol yang ada pada *activity diagram*:

###### Tabel 2.6 Simbol *activity* *diagram*

|  |  |  |
| --- | --- | --- |
| **Simbol** | **Nama** | **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 digunakan untuk mendefinisikan jika ada pilihan aktivitas lebih dari satu. |
|  | Penggabungan  (*join*) | Asosiasi penggabungan digunakan untuk mendefinisikan beberapa aktivitas yang digabungkan menjadi satu. |
|  | Status Akhir | Status akhir yang dilakukan sistem. Sebuah diagram aktivitas memiliki sebuah status akhir. |
|  | *Swimlane* | Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi. |

3. *Class diagram*

Diagram kelas atau *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.

Susunan struktur kelas yang baik pada diagram kelas sebaiknya memiliki jenis-jenis kelas berikut : a. Main *Class*

*Class* yang memiliki fungsi awal dieksekusi ketika sistem dijalankan.

1. *Class* yang menangani tampilan sistem (*view*)

*Class* yang mendefinisikan dan mengatur tampilan ke pemakai.

1. *Class* yang diambil dari pendefinisian *use case* (*controller*)

*Class* yang menangani fungsi – fungsi yang harus ada diambil dari pendefinisian *use case*, kelas ini biasanya disebut dengan kelas proses yang menangani proses bisnis pada perangkat lunak.

1. *Class* yang diambil dari pendefinisian data (model)

*Class* yang digunakan untuk memegang atau membungkus data menjadi sebuah kesatuan yang diambil maupun akan disimpan ke basis data. Adapun simbol – simbol yang digunakan dalam *Class* *diagram* adalah sebagai berikut :

###### Tabel 2.7 Simbol *class* *diagram*

|  |  |  |
| --- | --- | --- |
| **Simbol** | **Nama** | **Deskripsi** |
|  | Class | Kelas pada stuktur sistem. |
|  | Antarmuka  (*interface*) | Sama dengan konsep antarmuka dalam pemrograman berorientasi objek. |
|  | Asosiasi | Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multipicity*. |
|  | Asosiasi Berarah  (*directed association*) | Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*. |
|  | Generalisasi | Relasi antar kelas dengan makna generalisasi–spesialisasi (umum khusus). |
|  | Kebergantungan (*dependency*) | Relasi antar kelas dengan makna kebergantungan antar kelas. |
|  | Agregasi  (*aggreggation*) | Relasi antar kelas dengan makna semua bagian (*whole part*). |

4. *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 menggambarkan *sequence* *diagram* maka harus diketahui objek-objek yang terlibat dalam sebuah *use* *case* beserta metode-metode yang dimiliki kelas yang diinstansi menjadi objek itu. Berikut adalah simbol-simbol yang ada pada *sequence* *diagram*:

Tabel 2.8 Simbol sequence diagram

|  |  |  |
| --- | --- | --- |
| **Simbol** | **Nama** | **Deskripsi** |
|  | Garis Hidup  (*lifeline*) | Menyatakan kehidupan suatu objek. |
|  | Objek | Menyatakan objek yang berinteraksi. |
|  | Waktu Aktif | Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya. |
|  | Pesan Tipe *Call* | Menyatakan suatu objek memanggil operasi atau metode yang ada pada objek lain atau dirinya sendiri. |
|  | Pesan Tipe *Send* | Menyatakan bahwa suatu objek mengirimkan data/ masukan/ informasi ke objek lainnya, arah panah mengarah pada objek yang dikirimi. |

## 2.9 Flowchart Diagram

*Flowchart* adalah penyajian yang sistematis tentang proses dan logika dari kegiatan penanganan informasi atau penggambaran secara grafik dari langkahlangkah dan urut-urutan prosedur dari suatu program. *Flowchart* menolong analis dan *programmer* untuk memecahkan masalah kedalam segmen-segmen yang lebih kecil dan menolong dalam menganalisis alternatif-alternatif lain dalam pengoperasian. Berikut adalah simbol-simbol dalam *flowchart* *diagram* (Murdock, 2018):

Tabel 2.9 Simbol flowchart diagram

|  |  |  |
| --- | --- | --- |
| **Simbol** | **Nama** | **Deskripsi** |
|  | *Terminator* | Permulaan/akhir program |
|  | Garis alir  (*Flow* *line*) | Arah alir program |

###### Lanjutan Tabel 2.6 Simbol *flowchart diagram*

|  |  |  |
| --- | --- | --- |
| **Simbol** | **Nama** | **Deskripsi** |
|  | *Preparation* | Proses inisialisasi/pemberian harga awal |
|  | *Process* | Proses perhitungan/ proses pengolahan data |
|  | *Input/output data* | Proses input/output data, parameter, informasi |
|  | *Predefined*  *Process* | Permulaan sub program/proses menjalankan sub program |

## 2.10 Metode Pengembangan Sistem

Metode pengembangan sistem yang digunakan pada penelitian ini adalah *Rational Unified Process* (RUP). Rosa dan Shalahuddin (2011) dalam (Fitria & Widowati, 2017) 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.

1. Fase *transition*

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

## 2.11 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 2.10 Matriks *recall* dan *precision*

|  |  |  |
| --- | --- | --- |
|  | ***True*** | ***False*** |
| ***Positive*** | TP (*True Positive*) | FP (*False Positive*) |
| ***Negative*** | FN (*False Negative*) | TN (*True Negative*) |

Lalu, berdasarkan tabel tersebut, rumus *recall* – *precision* pun dapat dilihat pada persamaan 2.2 dan persamaan 2.3.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *Precision* = [ 𝑇𝑃 ] 𝑥100  𝑇𝑃+𝐹𝑃 |  |  |  |  |  |  | (2.2) |
| *Recall* = [ 𝑇𝑃 ] 𝑥100 |  |  |  |  |  |  | (2.3) |

𝑇𝑃+𝐹𝑁

Keterangan:

TP = Data dengan nilai sebenarnya positif dan nilai prediksi positif

FP = Data dengan nilai sebenarnya negatif dan nilai prediksi positif

FN = Data dengan nilai sebenarnya positif dan nilai prediksi negatif

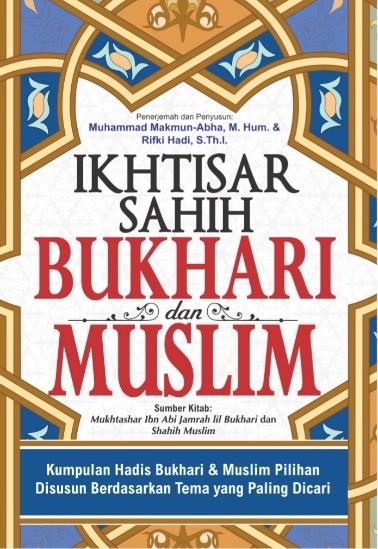
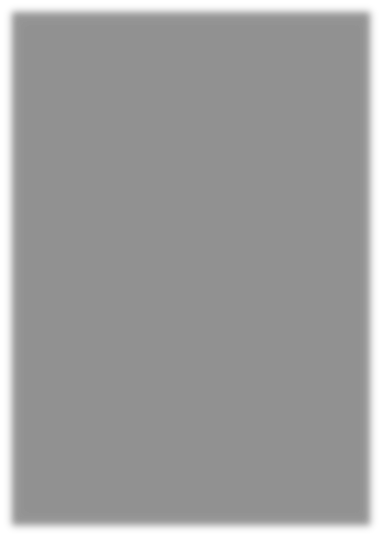
TN = Data dengan nilai sebenarnya negatif dan nilai prediksi negatif

# 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 N-gram yang akan diterapkan pada penelitian ini. Sedangkan data hadis yang digunakan yaitu hadis riwayat imam al-Bukhari yang diambil dari buku yang berjudul “Ikhtisar sahih Bukhari dan Muslim” di perpustakaan *online* (iPusnas) sebanyak 281 hadis dan 31 judul bab. Buku yang digunakan dapat dilihat pada gambar 3.1



###### Gambar 3.1 Buku Hadis Riwayat Imam al-Bukhari dan Muslim

### 3.2 Metode Pengembangan Sistem

Metode pengembangan sistem yang digunakan pada penelitian ini yaitu *Unified Process*, yang mendefinisikan tahap pelaksanaan, kegiatan peran pelaksana, hasil kerja dan prinsip yang harus diikuti.

25

###### Tabel 3.1 Fase Pengembangan *Software*

|  |  |
| --- | --- |
| ***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 Laboratorium *Computer Science and Artificial Intelligence*, Jurusan Teknik Informatika, Fakultas Teknik, 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 dibawah ini :

27

**Tabel 3.2** **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*** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

### 3.4 Analisis Sistem

Analisis sistem merupakan suatu tahapan yang bertujuan untuk mengetahui dan mengamati apa saja yang terlibat dalam suatu 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 fungsoinal dan nonfungsional yang meliputi kebutuhan perangkat keras dan perangkat lunak yang akan digunakan.

##### 3.4.1 Analisis Kebutuhan Non-Fungsional

Analisis kebutuhan non-fungsional adalah sebuah langkah dimana pembangun aplikasi menganalisis sumber daya yang dibutuhkan 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. Kebutuhan perangkat keras yaitu kebutuhan perangkat atau komponen yang dibutuhkan pada sistem dan perangkat lunak yaitu kebutuhan perangkat lunak untuk membantu agar komponen perangkat keras dapat berfungsi dan dapat dijalankan pada sistem.

1. Kebutuhan Perangkat Keras

Untuk menerapkan rancangan yang telah dijelaskan sebelumnya, dibutuhkan beberapa perangkat keras sebagai sarana untuk mengimplementasikan aplikasi yang dibangun. Berikut ini spesifikasi perangkat keras dapat dilihat pada Tabel 4.1.

###### Tabel 4.1 Spesifikasi Perangkat Keras

|  |  |  |
| --- | --- | --- |
| **No** | **Nama Perangkat** | **Spesifikasi** |
| 1. | *Notebook* | *Asus A456U* |
| 2. | *Processor* | *Intel Core i5-6198DU, up to 2,8 GHz* |
| 3. | *Monitor* | *Monitor 14 inch* |

28

###### Lanjutan Tabel 4.1 Spesifikasi Perangkat Keras

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Nama Perangkat** |  | **Spesifikasi** |
| 4. | *Memory* | *RAM 4 GB* |  |
| 5. | *Harddisk* | *1 TB* |  |

2. Kebutuhan Perangkat Lunak

Perangkat lunak yang digunakan pada pembangunan aplikasi glosarium teknologi informasi. Adapun rincian kebutuhan perangkat lunak dapat dilihat pada Tabel 4.2.

###### Tabel 4.2 Spesifikasi Perangkat Lunak

|  |  |  |
| --- | --- | --- |
| **No** | **Nama Perangkat** | **Spesifikasi** |
| 1. | *Operating System* | *Windows 10 Profesional 64 bit* |
| 2. | *Python* | *Versi* 3.8 |
| 3. | *Xampp* | *Versi 3.2.2* |
| 4. | *Browser* | *Google chrome* |
| 5. | *Gui Python* |  |

##### 3.4.2 Analisis Kebutuhan Fungsional

Analisis 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 pembuatan *flowchart* sistem, *flowchart* metode, *use case diagram*, *activity diagram, class diagram* serta *sequence diagram.*

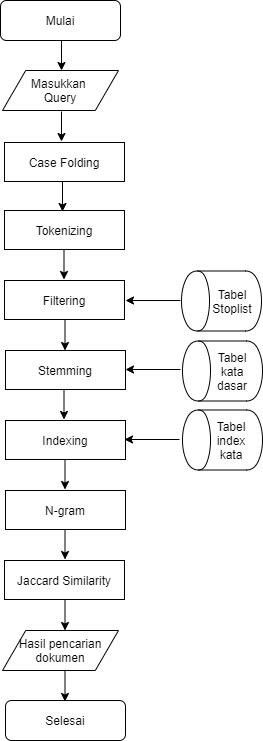
###### A. Perancangan *Flowchart*

*Flowchart* adalah simbol-simbol pekerjaan yang menunjukkan bagan aliran proses yang saling terhubung. Perancangan *flowchart* yang akan dibangun adalah *flowchart* sistem, *flowchart case folding*, *flowchart tokenizing*, *flowchart filtering*, *flowchart stemming, flowchart indexing,* dan *flowchart* metode.

1. Flowchart Diagram Sistem

Adapun alur kerja *flowchart* diagram sistem adalah sebagai berikut :

1. *User* memulai sistem
2. *User* memasukkan kata kunci atau *query*.
3. Kata kunci yang dimasukkan melalui tahap proses *case* *folding*, *tokenizing*, *filtering,* *stemming* dan *indexing*.
4. Kemudian kata kunci yang dimasukkan melalui tahap N-gram.
5. Lalu masuk pada tahap *Jaccard* *similarty*.
6. Kemudian sistem akan mencari dokumen-dokumen yang berhubungan dengan kata kunci yang dimasukkan dan menampilkan dokumen tersebut.



Mulai

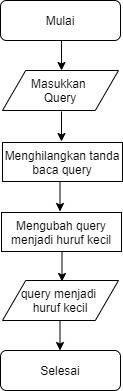
Selesai

Gambar 4.1 Flowchart Diagram Sistem

2. Flowchart Diagram Case Folding

Adapun alur kerja *flowchart diagram case folding* adalah sebagai berikut:

1. Kata kunci atau *query* akan dihilangkan semua tanda bacanya.
2. Kemudian semua kata kunci diubah menjadi huruf kecil atau *lower* *case*.



Mulai

Selesai

Gambar 4.2 flowchart Case Folding

3. Flowchart Diagram Tokenizing

Adapun alur kerja dari *flowchart diagram tokenizing* adalah sebagai berikut:

1. Masukkan kata kunci pada sistem temu pencarian.
2. Memisahkan setiap kata yang bergabung.
3. Memisahkan kalimat berdasarkan tiap kata yang menyusunnya.
4. Kalimat terpisah menjadi kata atau token.



Mulai

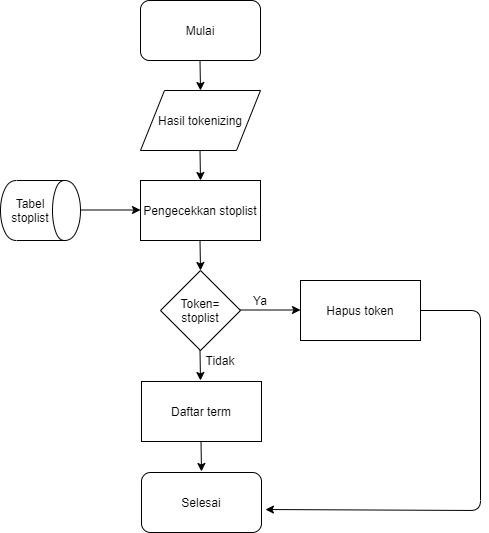
Selesai

Gambar 4.3 flowchart Tokenizing

4. Flowchart Diagram Filtering

Adapun alur kerja dari *flowchart diagram filtering* adalah sebagai berikut:

1. Hasil dari *tokenizing* yang berupa kalimat yang telah dipecah menjadi kata atau token akan dicek apakah kata tersebut masuk dalam *stoplist* atau tidak.
2. Token akan dicek di *database* *stoplist*.
3. Jika masuk dalam *stoplist* maka kata tersebut akan dihapus dan tidak diproses.

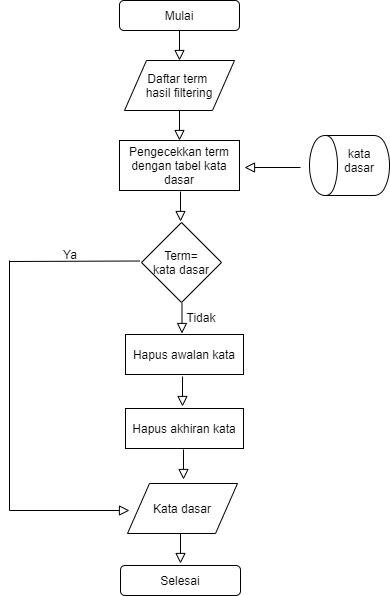


Gambar 4.4 Flowchart diagram filtering

5. Flowchart Diagram Stemming

Adapun alur kerja dari *flowchart diagram stemming* adalah sebagai berikut:

1. Daftar *term* yang telah di filtering akan dicek apakah *term* tersebut merupakan kata dasar atau tidak.
2. Jika tidak akan dihapus awalan serta akhiran imbuhan pada *term* tersebut.
3. Jika merupakan kata dasar maka tidak melalui tahap *stemming* dan langsung diproses pada sistem.

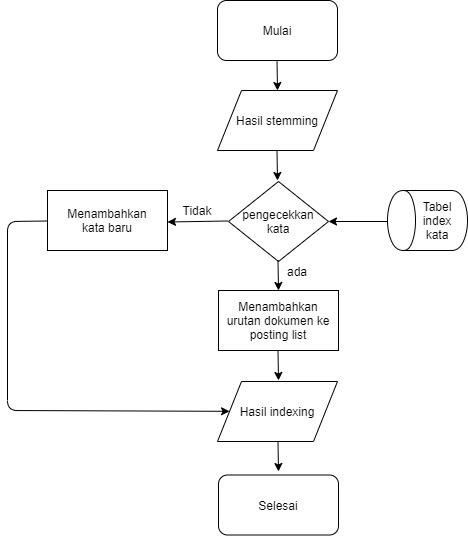


Gambar 4.5 Flowchart diagram stemming

6. Flowchart Diagram Indexing

Adapun alur kerja dari *flowchart diagram indexing* adalah sebagai berikut :

1. Kata dasar hasil proses *stemming* akan dicek pada tabel index kata yang ada di *database*.
2. Jika kata dasar ada pada tabel index kata maka sistem akan menambahkan urutan dokumen ke *posting* *list*.
3. Jika tidak maka sistem akan menambahkan kata baru pada tabel index kata.

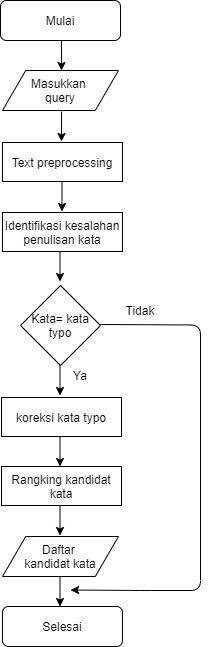


Gambar 4.6 Flowchart diagram indexing

7. Flowchart N-gram dan Jaccard Similarity

Adapun alur kerja dari *flowchart* metode adalah sebagai berikut :

1. Masukkan kata kunci pada aplikasi pencarian.
2. Kata kunci yang dimasukkan akan mengalami proses *preprocessing*.
3. Kemudian kata kunci akan diidentifikasi kesalahan penulisan katanya.
4. Jika salah, maka akan mengalami pengoreksian kata. Lalu setelah di koreksi kata tersebut akan dirangking. Kemudian muncul daftar kandidat kata yang kemungkinan adalah kata yang dimaksud.
5. Jika benar, maka kata kunci tersebut tidak melalui tahapan pengoreksian kata.



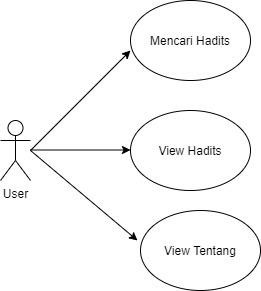
**Gambar 4.7 *Flowchart diagram* N-gram dan *Jaccard Similarity***

B. 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, Class Diagram* dan *Sequence Diagram.*

1. Use case diagram

*Use Case Diagram* adalah sebuah diagram yang dapat merepresentasikan interaksi yang terjadi antara *user* dengan sistem. *Use Case Diagram* ini mendeskripsikan siapa saja yang menggunakan sistem dan bagaimana cara mereka berinteraksi dengan sistem. *Use* *case* *diagram* dari sistem yang akan dibangun dapat dilihat pada gambar 4.8.



Gambar 4.8 Use case diagram

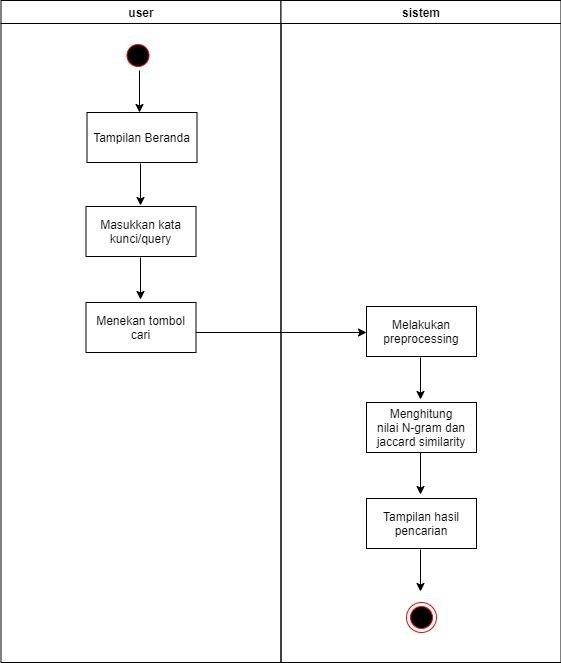
2. Activity diagram

*Activity diagram* menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity* *diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi.

Berikut ini adalah *activity diagram* yang akan menggambarkan alir aktivitas sistem.

a. Activity diagram Mencari Hadis

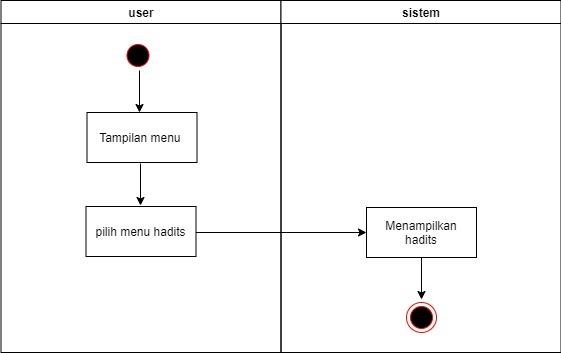
Gambar 4.9 tersebut merupakan diagram aktivitas yang menunjukkan aktivitas *user* ketika Mencari hadis. Ketika *user* ingin mencari hadis, *user* hanya perlu menuliskan *query* atau kata kunci pada tombol pencarian lalu sistem akan menampilkan hasil dari pencarian yang sesuai dengan *query* yang dituliskan.



###### Gambar 4.9 *Activity* *diagram* cari hadis

b. Activity diagram hadis

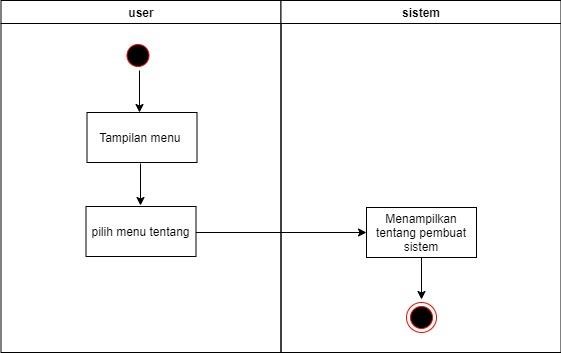
Gambar 4.10 merupakan *Activity Diagram* yang menunjukkan aktivitas *user* ketika memilih menu hadis, lalu sistem akan menampilkan hadis yang berisikan 400 hadis riwayat imam Bukhari.



###### Gambar 4.10 *Activity* *diagram* hadis

c. Activity diagram tentang

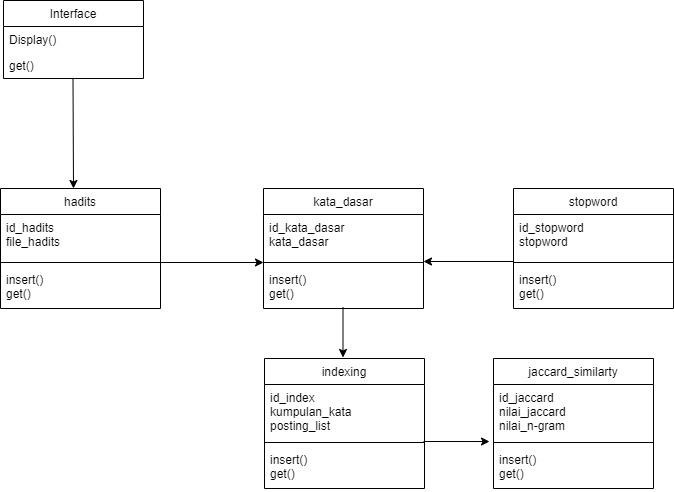
Gambar 4.11 merupakan *Activity Diagram* yang menunjukkan aktivitas *user* ketika memilih menu tentang, lalu sistem akan menampilkan menu tentang yang berisikan tampilan tentang buku hadis yang dijadikan data.

.

###### Gambar 4.11 *Activity* *diagram* tentang

3. Class diagram

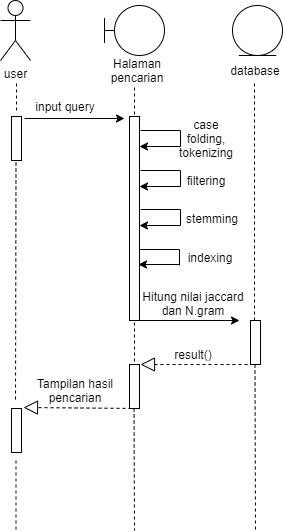
*Class diagram* merupakan diagram yang selalu ada dipemodelan sistem berorientasi objek. *Class diagram* menunjukkan hubungan antar *class* dalam sistem yang sedang dibangun dan bagaimana mereka saling berkolaborasi untuk mencapai suatu tujuan. Berikut ini adalah *class diagram* sistem temu pencarian hadis riwayat imam Bukhari.



Gambar 4.12 Class diagram

4. Sequence diagram

*Sequence Diagram* menggambarkan interaksi antar objek di dalam dan di sekitar sistem yang digambarkan terhadap waktu. Berikut ini adalah *Sequence diagram* yang akan menggambarkan interaksi antar objek dan sistem pada aplikasi sistem temu pencarian hadis riwayat Bukhari.



Gambar 4.13 Sequence diagram

### 4.2 Perancangan Antar Muka (*Interface*)

Perancangan *user interface* adalah tahapan pembuatan antarmuka yang akan digunakan pada pembangunan sistem temu pencarian hadis riwayat imam Bukhari yang dibagi menjadi tiga bagian yaitu perancangan *form* beranda, *form* pencarian hadis imam Bukhari, dan *form* tentang.

###### *a. Form* Beranda

Pada *form* Beranda ini terdapat form pencarian hadis yang menerapkan metode N-gram dan *jaccard similarity*. Hasil pencariannya berupa dokumen hadis yang urutan keluarnya berdasarkan nilai *jaccard similarity* yang tertinggi.

**Gambar**

**4**

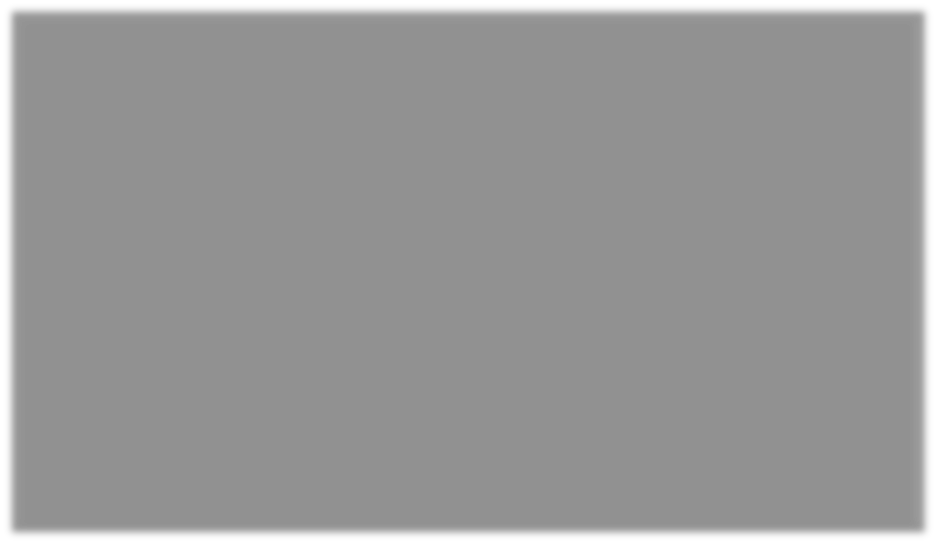
**.**

**14**

***Form***

**cari**

**hadis**



b. Form hadis

Pada saat *user* memilih *form* hadis, sistem akan menampilkan hadis bukhari, kemudian ketika *user* sudah memilih hadis, sistem akan menampilkan hadis secara lengkap.

**Gambar**

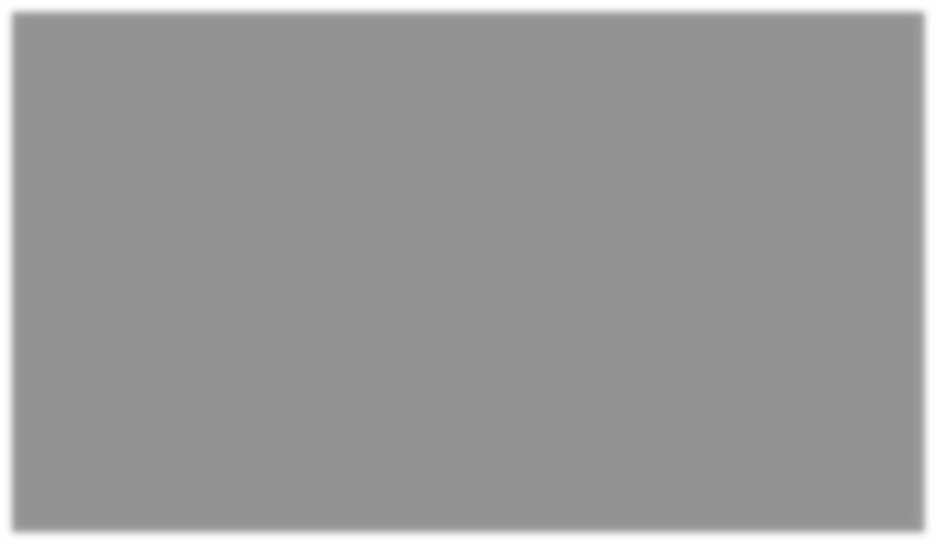
**4**

**.**

**15**

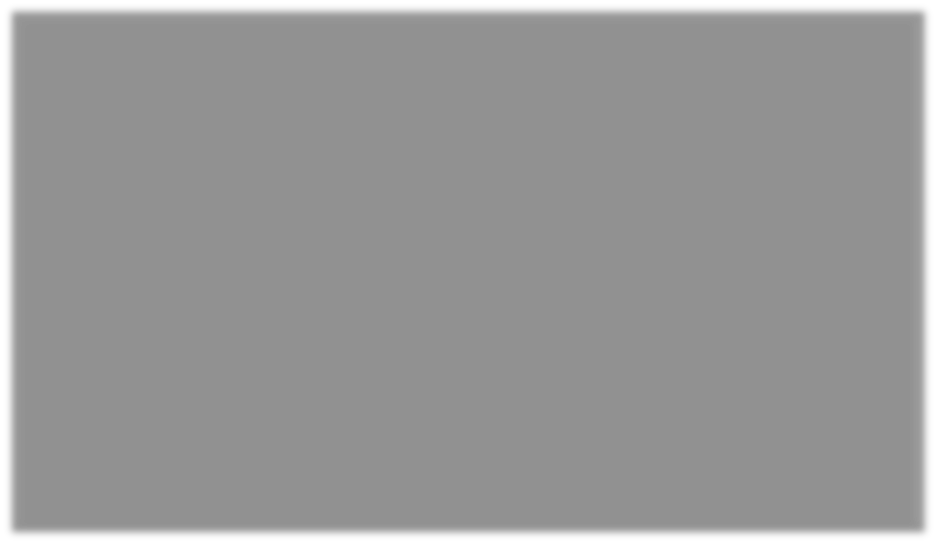
***Form***

**hadis**



c. Form tentang

Pada saat *user* memilih *form* tentang, sistem akan menampilkan *profil* pembuat aplikasi. Pada *form* pembuat berisi foto pembuat dan biodata pembuat.



**Gambar 4.16 *Form* tentang**

**DAFTAR PUSTAKA**

Abdurrahman, M. (2012). Imam Al-Bukhari Dan Lafal Al-Qur’an. *Kalimah*,

*11*(1), 120. https://doi.org/10.21111/klm.v11i1.487

Ahmed, F., De Luca, E. W., & Nürnberger, A. (2009). Revised N-Gram based Automatic Spelling Correction Tool to Improve Retrieval Effectiveness. *Polibits*, *40*(40), 39–48. https://doi.org/10.17562/pb-40-6

Baskoro, S. Y., Ridok, A., & Furqon, M. T. (2015). *Berdasarkan Kasus Menggunakan Metode Cosine Similarity Dan Latent Semantic Indexing ( Lsi )*. *2*(2), 83–88.

Bintana, R. R. (2012). *Penerapan Model Okapi Bm25 Pada Sistem Temu Kembali Informasi*. 273–279.

Bunyamin, H. (2005). Algoritma Umum Pencarian Informasi Dalam Sistem Temu Kembali Informasi Berbasis Metode Vektorisasi Kata dan Dokumen. *Jurnal Informatika*, *1*(2), 85–92.

Fahma, A. I., Cholissodin, I., & Perdana, R. S. (2018). Identifikasi Kesalahan

Penulisan Kata ( Typographical Error ) pada Dokumen Berbahasa Indonesia

Menggunakan Metode N-gram dan Levenshtein Distance. *Jurnal*

*Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *2*(1), 53–62.

Fitria, A., & Widowati, H. (2017). Implementasi metode rational unified process dalam pengembangan sistem administrasi kependudukan. *Jurnal Teknologi Rekayasa*, *22*, 27–36.

Hamzah, A. (2010). No Title. *Deteksi Bahasa Untuk Dokumen Text Bahasa Indonesia*.

Indraloka, D. S., & Santosa, B. (2017). Penerapan Text Mining untuk Melakukan

Clustering Data Tweet Shopee Indonesia. *Jurnal Sains Dan Seni ITS*, *6*(2),

6–11. https://doi.org/10.12962/j23373520.v6i2.24419

Indriani, A., Muhammad, Suprianto, & Hadriansa. (2018). Implementasi Jaccard Index Dan N-Gram Pada Rekayasa Aplikasi Koreksi Kata Berbahasa Indonesia. *Sebatik 1410-3737*, 95–101.

Lestari, N. P. (2016). Uji Recall and Precision Sistem Temu Kembali. *Universitas Airlangga*.

Lisangan, E. A. (2015). *Implementasi n-gram Technique dalam Deteksi Plagiarisme pada Tugas IMPLEMENTASI n-GRAM TECHNIQUE DALAM DETEKSI*. *May*.

Masrukhin. (2016). *metode Bukhari dalam al-jami’ al-sahih*. *2*(2), 279–291.

Melita, R. (2018). Penerapan Metode Term Frequency Inverse Document Frequency (Tf-Idf) Dan Cosine Similarity Pada Sistem Temu Kembali Informasi Untuk Mengetahui Syarah Hadits Berbasis Web (Studi Kasus: Hadits Shahih Bukhari-Muslim). *Jurnal Teknik Informatika*, *11*(2), 149–164. https://doi.org/10.15408/jti.v11i2.8623

Murdock, D. H. (2018). Flowcharts. *Auditor Essentials*, 235–239. https://doi.org/10.1201/9781315178141-51

ROSYID, K. (2016). *Kepemimpinan Menurut Hadits Nabi Saw*.

Salahuddin, R. A. . M. (2015). *Rekayasa Perangkat Lunak Terstruktur dan Beriorentasi Objek*.

Sri, C., Aditya, K., Rahmayanti, V., & Nastiti, S. (2019). *SISTEM TEMU*

*KEMBALI INFORMASI BUKU HADITS MENGGUNAKAN PEMBOBOTAN TERM FREQUENCY INVERSE DOCUMENT FREQUENCY DAN COSINE*.

*2016*, 124–131.

Sunardi, S., Yudhana, A., & Mukaromah, I. A. (2018). Implementasi Deteksi Plagiarisme Menggunakan Metode N-Gram Dan Jaccard Similarity Terhadap Algoritma Winnowing. *Transmisi*, *20*(3), 105. https://doi.org/10.14710/transmisi.20.3.105-110

Wardhana, W. S., Wirayuda, T. A. B., & Shaufiah. (2011). *Pengoreksian Ejaan Kata Menggunakan Metode N-Gram (Studi Kasus Dokumen Teks Berbahasa Indonesia)*. 0–6.

Zaman, B., Hariyanti, E., & Purwanti, E. (2015). Sistem Deteksi Bahasa pada Dokumen menggunakan N-Gram. *Multinetics*, *1*(2), 21. https://doi.org/10.32722/vol1.no2.2015.pp21-26

Zuhri, S. (2008). *Predikat Hadis Dari Segi Jumlah Riwayat Dan Sikap Para Ulama Terhadap Hadis Ahad*.

https://publikasiilmiah.ums.ac.id/xmlui/handle/11617/917