

DAFTAR ISI

HALAMAN PERNYATAAN ORISINALITAS	ii
HALAMAN PENGESAHAN.....	iii
KATA PENGANTAR	iv
ABSTRAK	vi
ABSTRACT.....	vii
DAFTAR ISI.....	viii
DAFTAR TABLE.....	xi
DAFTAR GAMBAR	xiii
BAB I PENDAHULUAN	1
1.1 Latar Belakang.....	1
1.2 Rumusan Masalah	3
1.3 Tujuan.....	4
1.4 Ruang Lingkup	4
1.5 Manfaat Penelitian.....	5
1.6 Tinjauan Pustaka	5
1.7 Kontribusi Penelitian.....	6
1.8 Sistematika Penulisan.....	6
BAB II LANDASAN TEORI	8
2.1 <i>Information Retrieval</i>	8
2.2 <i>Stemming</i>	9
2.2.1. <i>Stemming</i> Porter	9
2.2.2. <i>Stemming</i> Sastrawi	11
2.2.3. <i>Stemming</i> Idris.....	13
2.2.4. <i>Stemming</i> Arifin Setiono	15
2.3 Contoh Implementasi algoritma	18
2.3.1 Contoh Implementasi Algoritma Porter	18
2.3.2 Contoh Implementasi Algoritma Sastrawi	18
2.3.3 Contoh Implementasi Algoritma Idris	19
2.3.4 Contoh Implementasi Algoritma Arifin Setiono.....	20

2.3.5	Contoh Kata yang tidak berhasil di stemming oleh algoritma Porter	20
2.3.6	Contoh Kata yang tidak berhasil di stemming oleh algoritma Sastrawi	21
2.3.7	Contoh Kata yang tidak berhasil di stemming oleh algoritma Idris	22
2.3.8	Contoh Kata yang tidak berhasil di stemming oleh algoritma Arifin Setiono	23
2.3.9	Kata yang berhasil dan tidak dari seluruh algoritma.....	23
2.4	<i>PHP</i> (Fungsi Microtime).....	24
2.5	Algoritma Presisi	24
2.6	Rumus Perhitungan Waktu.....	25
BAB III METODE PENELITIAN.....		27
3.1	Metode Penelitian.....	27
3.1.1	Teknik penelitian.....	28
3.2	Teknik Pengumpulan Data	28
3.2.1	Sampel.....	29
3.3	Analisis Kebutuhan Sistem.....	29
3.3.1	Spesifikasi perangkat keras yang digunakan.....	29
3.3.2	Spesifikasi perangkat lunak yang digunakan	29
3.3.3	Dataset yang digunakan	30
3.4	Perancangan Umum.....	30
3.4.1	Gambaran Umum.....	30
3.4.2	Alur Kerja Sistem.....	31
3.4.3	Blok Diagram.....	32
3.4.4	<i>Usecase Diagram</i>	33
3.4.5	<i>Flowchart</i> (Diagram Alir) Sistem	36
3.5	Studi Kasus.....	38
3.5.1	Memasukan File Yang memiliki ekstension .pdf / .txt.	38
3.5.2	Fungsi Start <i>Microtime</i>	38
3.5.3	<i>Pre – Processing</i>	39
3.5.4	Proses Stemming.....	41
3.5.5	Memasang <i>End Microtime</i>	45
3.5.6	Perhitungan Waktu.....	46

3.5.7	Hasil Stemming.....	46
3.5.8	<i>Indexing</i>	47
3.5.9	Perhitungan Ketepatan	47
BAB IV IMPLEMENTASI DAN PENGUJIAN		49
4.1	Lingkungan Pengembang	49
4.1.1	Perangkat Keras	49
4.1.2	Perangkat lunak.....	49
4.2	Pengunaan Data.....	50
4.3	Implemntasi GUI.....	51
4.3.1	Fitur Login	51
4.3.2	Fitur Registrasi Akun	52
4.3.3	Fitur Profil Pengguna	53
4.3.4	Fitur Input Data Kata Berimbuhan.....	54
4.3.5	Fitur History Hasil Stemming	55
4.3.6	Fitur Detail History Hasil Stemming	55
4.4	Pengujian Sistem	56
4.4.1	Pengujian Proses Input Data	57
4.4.2	Pengujian <i>Pre-Processing</i>	59
4.4.3	Pengujian Proses <i>Stemming</i>	60
4.4.4	Pengujian Mengirim Data Ke <i>Database</i>	65
4.5	Pengujian Kinerja Algoritma.....	67
4.5.1	Kinerja Menyeluruh Dengan 1 File.....	67
4.5.2	Rekap Kinerja Algoritma Stemming.....	81
4.5.3	Validasi Algoritma Penelitian	83
4.6	Analisis	86
BAB V PENUTUP.....		89
5.1	Kesimpulan.....	89
5.2	Saran	90
DAFTAR PUSTAKA		91
Lampiran		93

DAFTAR TABLE

Tabel 1. 1 Tabel Penelitian.....	5
Tabel 2. 1 Aturan Awalan Algoritma Porter	10
Tabel 2. 2 Contoh Penggunaan Algoritma Porter	18
Tabel 2. 3 Contoh Implementasi Algoritma Sastrawi.....	19
Tabel 2. 4 Contoh Implementasi Algoritma IdrisContoh Implementasi Algoritma Idris	19
Tabel 2. 5 Contoh Implementasi Algoritma Arifin Setiono.....	20
Tabel 2. 6 Kata yang tidak berhasil di stemming oleh Porter	21
Tabel 2. 7 Kata yang tidak berhasil di stemming oleh Sastrawi	21
Tabel 2. 8 Kata yang tidak berhasil di stemming oleh Idris.....	22
Tabel 2. 9 Kata yang tidak berhasil di stemming oleh Arifin Setiono	23
Tabel 2. 10 Hasil Stemming Seluruh Algoritma.....	24
Tabel 2. 11 Contoh Studi Kasus Perhitungan Presisi.....	25
Tabel 2. 12 Hasil Perhitungan manual algoritma presisi	25
Tabel 3. 1 UC-01 Usecase Diagram.....	34
Tabel 3. 2 UC-02 Usecase Diagram.....	34
Tabel 3. 3 UC-03 Usecase Diagram.....	35
Tabel 3. 4 UC-04 Usecase Diagram.....	35
Tabel 3. 5 UC-05 Usecase Diagram.....	36
Tabel 3. 6 Hasil Case Folding.....	39
Tabel 3. 7 Hasil Tokenizing	40
Tabel 3. 8 Hasil Filtering	41
Tabel 3. 9 Tahapan Hapus Partikel Porter	42
Tabel 3. 10 Hasil Menghapus kata ganti Porter	42
Tabel 3. 11 Aturan awalan 1 Porter	42
Tabel 3. 12 Hasil Menghilangkan Awalan 1 Porter	43
Tabel 3. 13 Hasil Menghapus Akhiran Porter.....	43
Tabel 3. 14 Hasil Penghapusan Akhiran Porter	44
Tabel 3. 15 Tahapan Menghilangkan Kata Akhiran Sastrawi	45
Tabel 3. 16 Hasil Menghilangkan Awalan dan Akhiran Sastrawi	45
Tabel 3. 17 Hasil Akhir Tahapan Sastrawi	45
Tabel 3. 18 Hasil Kata Stemming Porter	46
Tabel 3. 19 Hasil Kata Stemming Sastrawi	47
Tabel 3. 20 Hasil Indexing Algoritma Porter	47
Tabel 3. 21 Hasil Indexing Algoritma Sastrawi.....	47
Tabel 3. 22 Perhitungan Ketepatan Studi Kasus Algoritma Porter.....	48
Tabel 3. 23 Perhitungan Ketepatan Studi Kasus Algoritma Sastrawi.....	48
Tabel 4. 1 Daftar Pengujian Sistem.....	56
Tabel 4. 2 Pengujian Sistem UJI-01.....	57
Tabel 4. 3 Pengujian Pre-Processing.....	59

Tabel 4. 4 Pengujian Proses Stemming Porter	60
Tabel 4. 5 Pengujian Proses Stemming Sastrawi	62
Tabel 4. 6 Pengujian Proses Stemming Idris	63
Tabel 4. 7 Pengujian Proses Stemming Arifin Setiono.....	64
Tabel 4. 8 Proses Mengirim data kedalam database	65
Tabel 4. 9 Hasil Stemming Algoritma Porter	67
Tabel 4. 10 Data Presisi Dan Kecepatan Algoritma Porter.....	70
Tabel 4. 11 Hasil Stemming Sastrawi	71
Tabel 4. 12 Nilai Presisi dan Kecepatan Sastrawi.....	73
Tabel 4. 13 Hasil Stemming Idris	74
Tabel 4. 14 Hasil Presisi dan Kecepatan Idris.....	77
Tabel 4. 15 Hasil Stemming Arifin Setiono.....	78
Tabel 4. 16 Total Presisi dan Kecepatan Arifin Setiono.....	81
Tabel 4. 17 Perbedaan Hasil Stemming Library & Algoritma Sastrawi Pada Penelitian.....	83

DAFTAR GAMBAR

Gambar 2. 1 Alur proses Information Retrieval (Sumber : https://ukdiss.com/intro/information-retrieval-model-system-design.php)	8
Gambar 2. 2 Flowchart Algoritma Porter	11
Gambar 2. 3 Flowchart Algoritma Sastrawi	13
Gambar 2. 4 Flowchart Algoritma Idris	15
Gambar 2. 5 Flowchart Algoritma Arifin Setiono	17
Gambar 3. 1 Blok Diagram Teknik Penelitian	28
Gambar 3. 2 Alur Kerja Sistem	31
Gambar 3. 3 Blok Diagram Sistem	32
Gambar 3. 4 Usecase Diagram	33
Gambar 3. 5 Flowchart Sistem	37
Gambar 3. 6 Fungsi Awal Microtime PHP	38
Gambar 3. 7 Case Folding PHP	39
Gambar 3. 8 Fungsi Tokenizing PHP	40
Gambar 3. 9 Contoh Filtering PHP	40
Gambar 4. 1 Data pengujian	51
Gambar 4. 2 GUI Login	52
Gambar 4. 3 GUI Registrasi	53
Gambar 4. 4 GUI Profil Pengguna	53
Gambar 4. 5 GUI Fitur Input Data	54
Gambar 4. 6 GUI History	55
Gambar 4. 7 GUI Detail History	56
Gambar 4. 8 Hasil Pengujian Kinerja Stemming Porter	67
Gambar 4. 9 Hasil Pengujian Kinerja Stemming Sastrawi	71
Gambar 4. 10 Hasil Pengujian Kinerja Stemming Idris	74
Gambar 4. 11 Hasil Pengujian Kinerja Stemming Arifin Setiono	78
Gambar 4. 12 Grafik rata - rata kecepatan dan ketepatan algoritma	82
Gambar 4. 13 Grafik rata - rata jumlah kata, kata berhasil dan gagal	82
Gambar 4. 14 Pengecekan Dictionary Library	84
Gambar 4. 15 Pengecekan Dictionary Tanpa Library	84
Gambar 4. 16 Hasil Evaluasi Rata - Rata Ketepatan dan Kecepatan	85
Gambar 4. 17 Evaluasi Rata - Rata Jumlah Kata, Kata Berhasil dan Gagal	86

BAB I

PENDAHULUAN

Didalam bab ini dipaparkan mengenai latar belakang, rumusan masalah, tujuan, ruang lingkup, dan tinjauan Pustaka.

1.1 Latar Belakang

Pencarian informasi, yang juga dikenal sebagai *Information Retrieval* (IR), adalah suatu proses dimana dokumen-dokumen teks dipisahkan dari sekumpulan dokumen yang tersedia berdasarkan tingkat relevansinya. *Information Retrieval* (IR) adalah ilmu pencarian informasi dari sejumlah data yang sangat banyak, yang mungkin telah hilang karena terlalu banyaknya data yang ada. Ilmu pencarian informasi ini diperkenalkan oleh Vannevar Bush pada tahun 1945, dan implementasinya mulai dikenalkan pada tahun 1950-an. Selama tahun 1990-an, banyak teknik dan metode pengambilan informasi yang dikembangkan dan digunakan. Tujuan utama dari sistem IR adalah untuk memenuhi kebutuhan informasi pengguna dengan mengambil semua dokumen yang mungkin relevan, dan meminimalkan pengambilan dokumen yang tidak relevan.

Sistem IR yang baik memungkinkan pengguna untuk dengan cepat dan akurat menentukan apakah isi dokumen yang diterima sesuai dengan kebutuhan mereka. Tujuan utamanya adalah menyusun dokumen-dokumen yang diperoleh secara terurut, dari dokumen yang paling relevan ke yang kurang relevan. Proses menyusun dokumen tersebut disebut perangkangan dokumen (Rezalina, 2017). Didalam proses *Information Retrieval* terdapat proses paling penting yang dinamakan *Stemming*.

Stemming merupakan proses dimana kata yang memiliki imbuhan dikembalikan ke dasarnya atau akar kata dengan menggunakan aturan tertentu. Cara proses *stemming* adalah dengan cara menghilangkan *prefix*, *infix* (penyisipan), *sufiks*, dan *konfiks* (kombinasi *prefix* dan *surfix*) dari kata imbuhan. *Stemming* merupakan bagian penting dalam *Information Retrieval* untuk pencarian dokumen pada web dan terjemahan. algoritma yang baik akan kembali

membubuhkan kata ke kata dasar dengan benar. Misalnya kata dasar diberikan awalan me- menjadikan kata tersebut menjadi “membaca” yang memberikan arti yang berbeda maka dari itu dibutuhkan proses *stemming* agar kata yang dicari memiliki arti yang sama (Mustikasari, Widaningrum, Arifin, & Putri, 2020).

Dalam pencarian kemiripan string/teks pada dokumen *stemming* merupakan peran paling penting karena kata – kata yang memiliki imbuhan biasanya memiliki arti yang berbeda sehingga Ketika sudah diubah menjadi kata dasar, kata – kata pada dokumen tersebut bisa dideteksi kemiripannya dengan menggunakan algoritma Rabin-Karp untuk proses mencari kemiripan pada dokumen teks Bahasa Indonesia (Yulianto & Nurhasanah, 2021).

Didalam konsep *stemming*, kata yang digunakan adalah kata yang berada dalam bentuk umum yang nantinya akan dimasukan kedalam index. Sehingga nantinya akan menghasilkan dokumen yang lebih relevan. Pada proses *stemming* juga terdapat beberapa algoritma yang memiliki kelebihan dan kekurangan pada masing – masing algoritmanya. Seperti kata “pembelajaran” pada algoritma porter hasil *stemming* nya menjadi “pembelajaran” yang bisa dibilang tidak terdeteksi *stemming* apapun pada algoritma tersebut, lalu ada juga kesalahan yang terjadi pada algoritma sastrawi yaitu kata “beberapa” yang diberikan proses *stemming* oleh algoritma sastrawi kata tersebut tetap menjadi kata “beberapa” dan tidak diubah menjadi kata dasar, lalu pada algoritma idris kesalahan terjadi pada kata “negeri” yang dilakukan *stemming* menjadi “neger” yang artinya terdapat kesalahan *overstemming*, serta pada Arifin setiono kesalahan yang sama terjadi pada kata “inginya” menjadi “ingi” yang bisa dibilang terjadi *overstemming*. Selain kesalahan yang terjadi pada algoritma tersebut, algoritma sastrawi, porter, idris, dan Arifin setiono ini memiliki fungsi untuk kalimat – kalimat atau kata yang beragam seperti sastrawi dan Arifin setiono diperuntukan untuk Bahasa Indonesia, porter diperuntukan untuk Bahasa inggris, idris digunakan untuk bahasa melayu.

Pada beberapa tinjauan pustaka yang telah dipelajari, ada beberapa hasil dari perbandingan yang terjadi pada algoritma sastrawi, porter, idris, dan Arifin

setiono. Pada laporan penelitian (Aria Hendrawan, Dr. Titin Winarti, & Henny Indriyawati., 2021) menyatakan algoritma sastrawi mencapai $\pm 95\%$ dan bisa dinyatakan algoritma tersebut yang terbaik untuk digunakan, namun ada beberapa juga yang menyatakan algoritma porter lebih baik digunakan untuk dokumen teks Bahasa Indonesia meskipun algoritma tersebut tidak diperuntukan untuk Bahasa Indonesia.

Dari beberapa beberapa tinjauan pustaka yang ditemukan terdapat beberapa perbedaan hasil yang dimiliki, seperti menurut (Rezalina, 2017) mengatakan bahwa hasil penelitiannya menunjukan algoritma Nazief & Adriani memiliki nilai akurasi terbesar dengan nilai 93.3%. Pada pencarian kata dasar untuk bahasa indonesia memiliki beberapa macam algoritma yang digunakan seperti Arifin & Setiono pada tahun 2002, Jelita Asian pada tahun 2005, Ahmad Yusoff pada tahun 1996, Vega pada tahun 2001, Nazief & Adriani pada tahun 1996, Idris 2001, Sastrawi pada tahun 2008, dan porter pada tahun 1980.

Dari beberapa algoritma yang telah diketahui maka dalam penelitian ini algoritma yang digunakan lebih dipersempit dengan memilih algoritma Porter, Sastrawi, Idris dan Arifin & Setiono dikarenakan algoritma yang dipilih tersebut termasuk algoritma yang memiliki nilai yang cukup tinggi, selain itu ada algoritma yang telah disempurnakan dan juga yang belum disempurnakan, seperti algoritma sastrawi telah disempurnakan dari algoritma nazief & adriani, serta adanya algoritma Porter yang bertujuan untuk *stemming* terhadap bahasa inggris dan juga Algoritma Idris yang bertujuan untuk *stemming* terhadap bahasa melayu sehingga perbandingan ini tidak hanya menggunakan algoritma yang diperuntukan untuk bahasa indonesia saja. Agar perbandingan ini menemukan algoritma terbaik dengan perhitungan parameter kecepatan dan ketepatan (Prasidhatama & Suryaningrum, Perbandingan Algoritma Nazief & Adriani Dengan Algoritma Idris Untuk Pencarian Kata Dasar, 2018).

1.2 Rumusan Masalah

Berdasarkan latar belakang masalah tersebut, dapat dirumuskan beberapa masalah sebagai berikut :

1. Bagaimana performansi algoritma porter dalam *stemming* teks berbahasa indonesia.
2. Bagaimana performansi algoritma sastrawi dalam *stemming* teks Bahasa Indonesia
3. Bagaimana performansi algoritma idris dalam *stemming* teks Bahasa Indonesia
4. Bagaimana performansi algoritma Arifin setiono dalam *stemming* teks Bahasa Indonesia
5. Bagaimana hasil perbandingan antara algoritma porter, sastrawi, idris, dan juga Arifin setiono dalam proses *stemming* teks berbahasa Indonesia

1.3 Tujuan

Dalam penelitian kali ini memiliki sebuah tujuan dan tujuan tersebut untuk mengetahui performansi dalam mengukur ketepatan dan kecepatan dari algoritma *stemming* porter, sastrawi, idris dan Arifin setiono pada dokumen teks Bahasa Indonesia

1.4 Ruang Lingkup

Pada Penelitian kali ini memiliki ruang lingkup yang dibuat sebagai batasan masalah yang akan dibahas agar penelitian tidak memiliki aspek yang terlalu banyak. Batasan masalah yang dimiliki penelitian ini adalah sebagai berikut :

1. Dokumen yang digunakan adalah dokumen berbahasa Indonesia yang dituangkan pada format .pdf atau .txt. dengan kapasitas kata minimal 50 kata dan maksimal 3000 kata
2. Parameter yang diteliti adalah kecepatan dan ketepatan pada penelitian ini dengan menggunakan empat algoritma yaitu porter, sastrawi, idris dan Arifin setiono
3. Kata dasar yang digunakan sebagai pembanding adalah kata dasar yang sesuai dengan kamus besar bahasa indonesia.

1.5 Manfaat Penelitian

Manfaat penelitian ini adalah dapat memahami tahapan proses *stemming* pada metode – metode yang digunakan saat penelitian terutama pada bagian *stemming* untuk dokumen teks Bahasa Indonesia. Mengetahui hasil dari *stemming* terbaik yang dibandingkan dari empat algoritma yang digunakan pada penelitian ini

1.6 Tinjauan Pustaka

Pada bagian tinjauan pustaka ini menampilkan beberapa literatur atau referensi yang digunakan atau yang memiliki kesamaan dengan penelitian saat ini dan tabel 1.1 adalah daftar beberapa referensi yang digunakan untuk menunjang penelitian ini .

Tabel 1. 1 Tabel Penelitian

No	Nama Peneliti	Judul Penelitian	Hasil
1	Oppie Rezalina	Perbandingan algoritma stemming nazief & adriani, porter dan Arifin setiono pada dokumen teks bahasa Indonesia	Hasil dari sistem ini algoritma nazief adriani memiliki nilai algoritma terbaik dengan nilai akurasi 93,93% dibandingkan arifin setiono 89,98%
2	Ledy Agusta	Perbandingan Algoritma Stemming Porter Dengan Algoritma Nazief & Adriani untuk stemming dokumen teks Bahasa Indonesia	Hasil dari penelitian ini mengevaluasi bahwa algoritma stemming nazief & adriani dibandingkan algoritma porter dengan nilai 97%
3	Manase Sahat H Simarangkir	Studi Perbandingan Algoritma – algoritma stemming untuk dokumen teks Bahasa Indonesia	Hasil dari sistem ini algoritma nazief adriani memiliki algoritma terbaik dengan nilai akurasi 97,93% dibandingkan dengan arifin setiono 92.09%, vega 63,48%, tala 78,27%
4	Diki Susandi, Usep Sholahudin	Pemanfaatan Vector Space Model pada Penerapan Algoritma Nazief Adriani, KNN dan Fungsi Similarity Cosine untuk Pembobotan IDF dan WIDF pada Prototipe Sistem Klasifikasi Teks Bahasa	Dari hasil penelitian bisa disimpulkan hasil yang didapatkan dari pengujian diukur dari precision dan recall tf-idf dan widf yang memiliki nilai 70,7% untuk tf-idf, dan 70.6% untuk widf sebagai

No	Nama Peneliti	Judul Penelitian	Hasil
		Indonesia	pengklasifikasian dokumen
5	Meisya Fitri	Perancangan Sistem Temu balik informasi dengan metode pembobotan kombinasi TF-IDF untuk pencarian dokumen berbahasa Indonesia	Pada hasil penelitian yang dilakukan sistem dapat mengumpulkan dokumen berita melalui proses crawling website dan memberikan bobot dengan mengimplementasikan metode pembobotan kata dengan metode kombinasi Tf-Idf. Sistem dapat melakukan pencarian dan menemukan informasi yang relevan berdasarkan hasil pengujian yang dilakukan pada 5 kata kunci

1.7 Kontribusi Penelitian

Pada bagian ini akan dijelaskan kontribusi penelitian yang dilakukan adalah untuk mengetahui perbedaan ketepatan dan kecepatan algoritma stemming Porter, Sastrawi, Idris, dan Arifin Setiono pada dokumen teks Bahasa Indonesia.

1.8 Sistematika Penulisan

Sistematika penulisan yang digunakan untuk memberikan gambaran isi dari laporan ini dijelaskan sebagai berikut :

BAB I PENDAHULUAN

Bab ini menjelaskan mengenai latar belakang masalah, rumusan masalah, ruang lingkup, tujuan, dan sistematika penulisan.

BAB II LANDASAN TEORI

Bab ini menjelaskan mengenai berbagai teori dasar yang digunakan dalam penelitian “PERBANDINGAN ALGORITMA STEMMING PORTER, SASTRAWI, IDRIS, DAN ARIFIN SETIONO PADA DOKUMEN TEKS BAHASA INDONESIA”.

BAB III METODE PENELITIAN

Pada bab ini dipaparkan metode yang digunakan dalam penelitian, uraian perancangan dari penelitian yang diusulkan. Pada bagian ini terdapat blok diagram yang menerangkan cara kerja dari penelitian ini.

BAB IV IMPLEMENTASI DAN PENGUJIAN

Pada bab ini akan disajikan hasil dari rancangan yang diajukan. Pada bagian ini diperlihatkan hasil pembangunan piranti lunak, berupa arsitektur dan juga model sistem seperti tampilan dan rincian dari pembangunan sistem. Pada sub-bab pengujian disajikan proses pencapaian penelitian berupa pengujian dari hasil implementasi yang dilakukan, penggunaan dari sistem yang telah selesai dibuat serta menampilkan hasil evaluasi terhadap pengujian yang telah dilakukan.

BAB V PENUTUP

Pada bab ini disajikan kesimpulan dari hasil penelitian yang telah dilakukan dan diuji

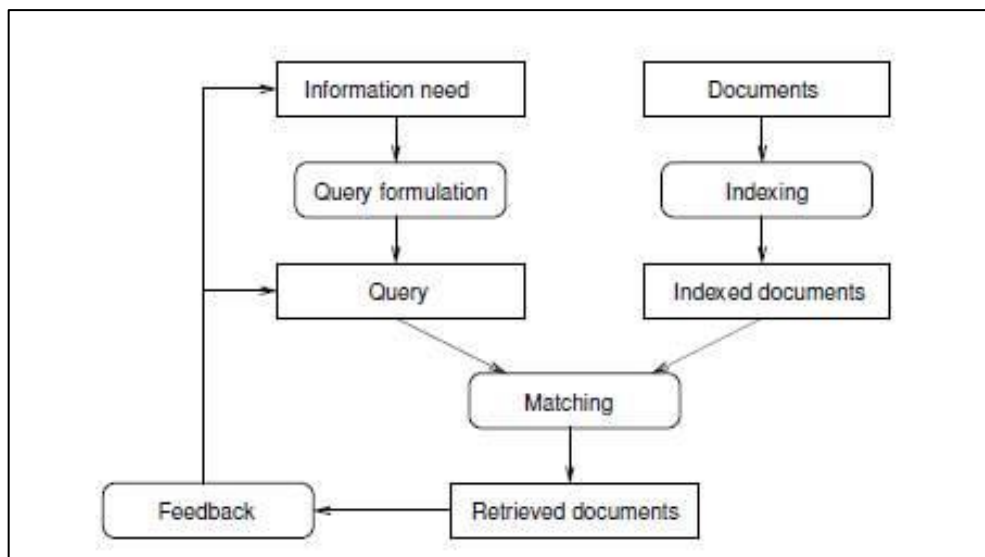
BAB II

LANDASAN TEORI

Untuk mendukung pembahasan dan penelitian yang dilakukan terdapat beberapa landasan teori sebagai acuan dan pendukung untuk menguatkan teori dan hasil penelitian yang dilakukan. Beberapa landasan teori yang didapatkan adalah sebagai berikut.

2.1 *Information Retrieval*

Sistem temu balik informasi (*Information Retrieval*) adalah salah satu penerapan teknologi komputer untuk perolehan, pengorganisasian, penyimpanan, pencarian dan pendistribusian informasi. Tujuan Sistem temu balik informasi (*information retrieval*) adalah untuk mencari informasi relevan terhadap kebutuhan pengguna, menggunakan search engine dengan memasukkan query untuk mendapatkan informasi yang diinginkan (Siregar, 2017). Alur proses dari sistem *information retrieval* akan ditampilkan pada gambar 2.1



Gambar 2. 1 Alur proses Information Retrieval (Sumber : <https://ukdiss.com/intro/information-retrieval-model-system-design.php>)

Information Retrieval ini dipopulerkan oleh Vannevar Bush pada tahun 1945. Namun cara dan proses implementasinya dikenalkan mulai pada tahun 1950- an. Pada ilmu ini sudah banyak teknik dan metode yang dikembangkan dan

dipakai. Tujuan IR sendiri adalah untuk memenuhi kebutuhan informasi pengguna dengan me-retrieve semua dokumen yang mungkin relevan, pada waktu yang sama meretrieve sesedikit mungkin dokumen yang tidak relevan.

Sistem IR yang baik memungkinkan pengguna menentukan secara cepat dan akurat apakah isi dari dokumen yang diterima memenuhi kebutuhannya. Tujuan yang harus dipenuhi adalah bagaimana menyusun dokumen yang telah didapatkan tersebut ditampilkan terurut dari dokumen yang memiliki tingkat relevansi tinggi ke tingkat relevansi yang lebih rendah. Penyusunan dokumen tersebut disebut sebagai perankingan dokumen.

2.2 Stemming

Algoritma *stemming* merupakan metode untuk meningkatkan performansi *Information Retrieval* dengan cara mentransformasi kata-kata di dalam suatu dokumen teks ke dalam bentuk kata dasarnya. Dengan melakukan proses *stemming*, maka efisiensi algoritma *Information Retrieval* dapat ditingkatkan karena menghilangkan kata-kata berimbuhan yang memiliki makna morfologi berbeda - beda namun memiliki interpretasi semantik yang sama (Sardjono, Cahyanti, Mujahidin, & Arianty, 2020)

Pada penerapan algoritma *stemming* untuk bahasa indonesia ini lebih kompleks karena terdapat berbagai macam variasi serta kombinasi imbuhan yang harus dihapus untuk mendapatkan kata dasar. Dalam penelitian ini untuk bahasa indonesia digunakan algoritma *stemming* porter, sastrawi, idris dan Arifin setiono yang memiliki penjelasan sebagai berikut

2.2.1. Stemming Porter

Algoritma *Porter* stemmer merupakan proses penentuan kata dasar melalui penghapusan afiks atau imbuhan, kata yang ambigu dan membingungkan dapat terjadi karena aturan morfologi yang tidak konsisten dalam tata bahasa Indonesia (Permana, 2017). Algoritma *Porter* untuk Bahasa Indonesia dikembangkan oleh Fadilah Z Tala yang mengadopsi Algoritma *Porter* yang seharusnya diperuntukan untuk Bahasa Inggris menjadi aturan –

aturan untuk Bahasa Indonesia, langkah atau tahapan algoritma *Porter* adalah sebagai berikut

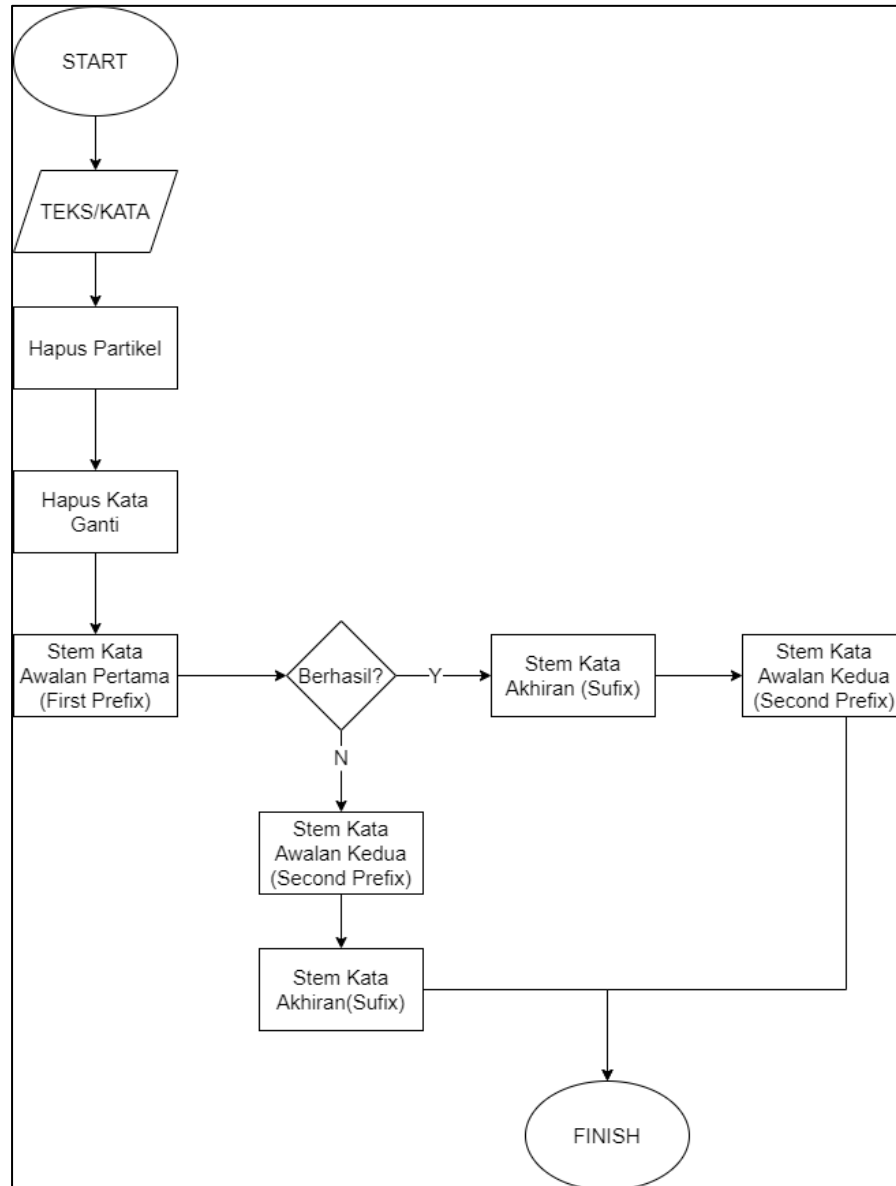
1. Menghapus partikel (“lah”, “kah”, “pun”).
2. Menghapus kata ganti (Possesive Pronoun), seperti –ku, -mu, -nya
3. Menghapus awalan pertama. Jika tidak ditemukan, maka lanjut ke langkah 4a, dan jika ada maka lanjut ke langkah 4b.
4.
 - a. Menghapus awalan kedua, dan dilanjutkan pada langkah ke
 - b. Menghapus akhiran, jika tidak ditemukan maka kata tersebut diasumsikan sebagai kata dasar (root word). Jika ditemukan maka lanjut ke langkah 5b.
5.
 - a. Menghapus akhiran dan kata akhir diasumsikan sebagai kata dasar (root word).
 - b. Menghapus awalan kedua dan kata akhir diasumsikan sebagai kata dasar (root word). Dalam sebuah kata, memungkinkan adanya dua awalan yang saling berurutan.

Adapun beberapa aturan urutan awalan yang diperbolehkan dalam algoritma porter pada tabel 2.1 dikarenakan tidak semua awalan kata bisa ditambahkan pada awalan lainnya dalam sebuah kata (Wahyudi, Susyanto, & Nugroho, 2017).

Tabel 2. 1 Aturan Awalan Algoritma *Porter*

Awalan 1	Awalan 2
Meng	Per
Di	Ber
Ter	
Ke	

Serta jika diterapkan dalam sebuah alur diagram, tahapan – tahapan yang dimiliki oleh algoritma *Porter* terlihat pada gambar 2.2 yang menerangkan *flowchart* (alur diagram) untuk algoritma *Porter*.



Gambar 2. 2 Flowchart Algoritma Porter

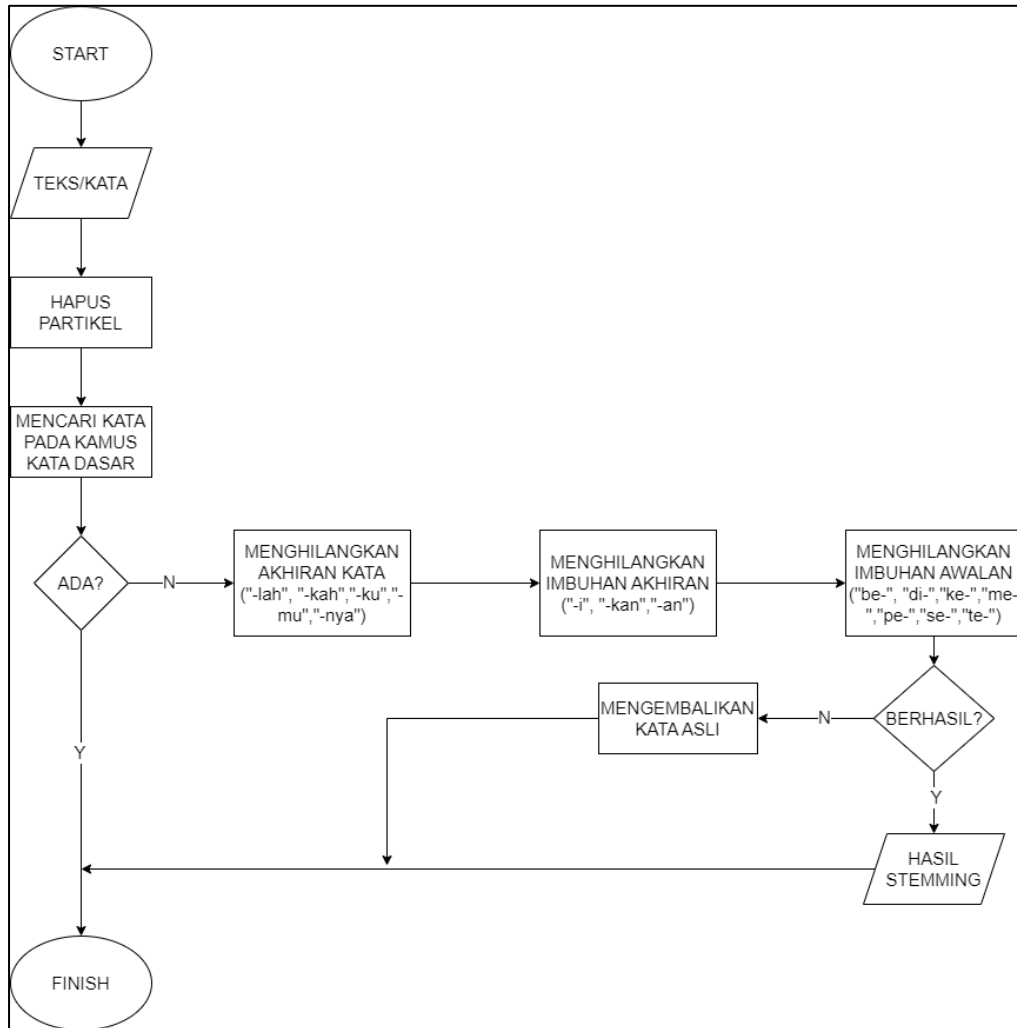
2.2.2. *Stemming Sastrawi*

Pustaka Sastrawi merupakan pustaka stemmer yang digunakan untuk mengatasi masalah pergantian kata dengan kata menjadi kata dasar. Sastrawi stemmer kemudian menerapkan algoritma berdasarkan Nazief dan Adriani dan disempurnakan dengan algoritma CS (Confix Stripping), Algoritma ECS (Enhanced Confix Stripping), serta lebih ditingkatkan dengan Modifikasi ECS (Rosid, Fitriani, Mulloh, & Gozali, 2020).

Diharapkan dengan menerapkan pustaka sastrawi ini pada proses preprocessing dokumen latar belakang / abstrak suatu jurnal, khususnya *stemming* dan stopword tahap penghilang, dapat membuat kata-kata penting serta menghilangkan yang tidak penting sehingga dapat mengukur ketepatan dan kecepatan pada proses *stemming* tersebut. Langkah – langkah dalam algoritma ini adalah sebagai berikut :

1. Melakukan pemeriksaan apakah kata yang akan di *stemming* ada dalam kamus kata dasar atau tidak. Jika ada, maka proses *stemming* berhenti pada langkah ini.
2. Jika tidak ada dalam kamus, artinya kata tersebut merupakan kata berimbuhan. Kemudian menghilangkan kata akhiran “-lah”, “-kah”, “-ku”, “-mu”, “-nya”, “-tah” atau “-pun”.
3. Menghilangkan kata imbuhan akhiran “-i”, “-kan”, “-an”, kemudian hapus kata imbuhan awalan “be-”, “di-”, “ke-”, “me-”, “pe-”, “se-”, dan “te-”.
4. Jika kata dasar yang dihasilkan dari langkah sebelumnya tidak terdapat di kamus, maka kata tersebut dicek apakah termasuk pada table keambiguan atau tidak.
5. Jika seluruh proses langkah 1-4 gagal dilakukan, maka algoritma mengembalikan kata aslinya.

Semua langkah – langkah yang dilakukan oleh proses *stemming* sastrawi dituangkan dalam *flowchart* pada gambar 2.3, agar penjelasan penggunaan algoritma terhadap kata atau kalimat berimbuhan lebih jelas dijelaskan oleh diagram alir atau *flowchart*



Gambar 2. 3 Flowchart Algoritma Sastrawi

2.2.3. *Stemming Idris*

Permatasari menjabarkan pada tahun 2016 mengenai *stemming* menggunakan algoritma Idris adalah sebagai berikut :

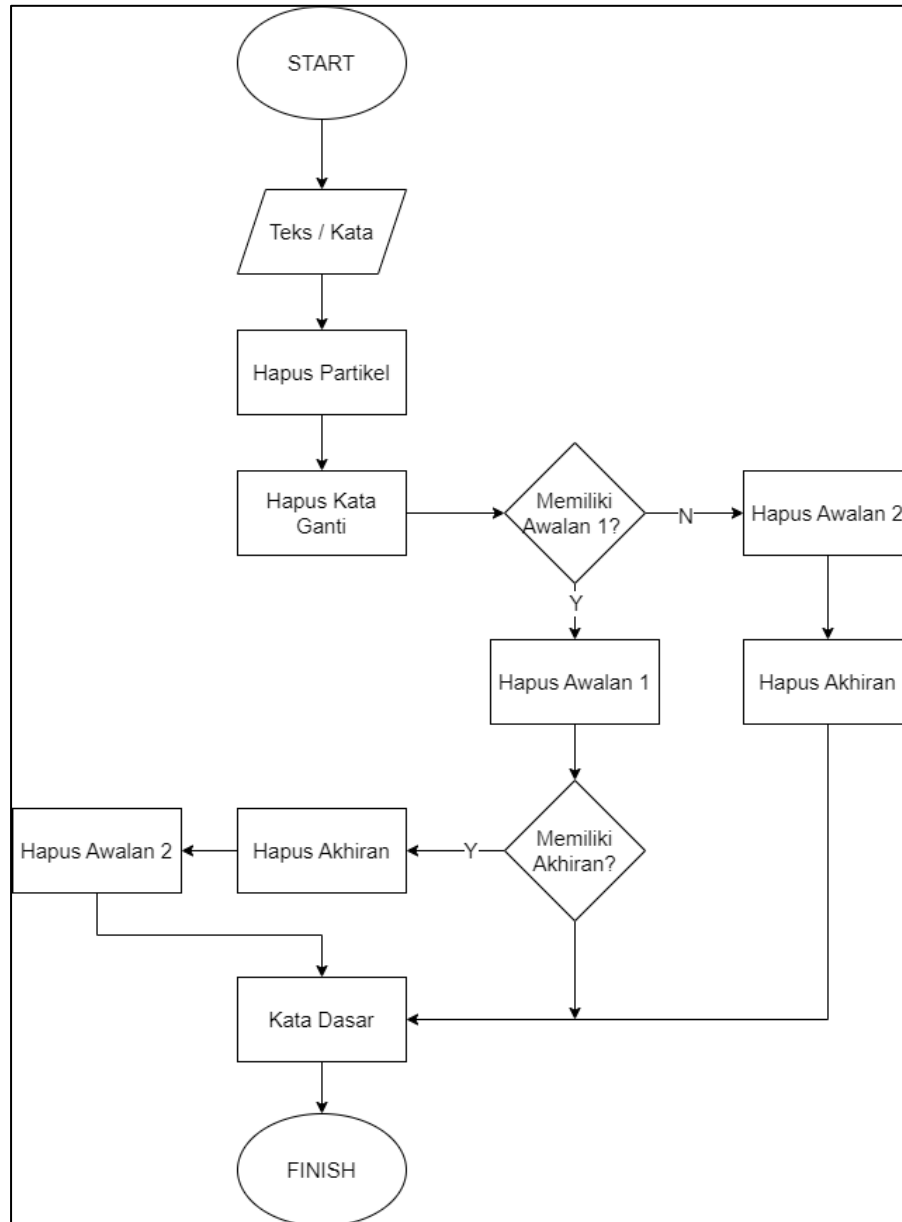
1. Algoritma awal yang dilakukan adalah sebagai berikut :

- a. Kata yang belum di-*stemming* dicari pada kamus umum atau Kamus Besar Bahasa Indonesia (KBBI). Jika kata itu langsung ditemukan, berarti kata tersebut adalah kata dasar. Kata tersebut dikembalikan dan algoritma dihentikan.
- b. Kata yang belum di-*stemming* dicari pada kamus lokal. Jika kata itu langsung ditemukan, berarti kata tersebut adalah kata dasar. Kata tersebut

dikembalikan dan algoritma dihentikan. Kamus lokal digunakan karena algoritma Idris dibuat untuk *stemming* bahasa melayu, yang dimana kamus lokal berisi kata-kata dasar bahasa melayu. Sedangkan *stemming* Bahasa Indonesia tidak memerlukan kamus lokal.

- c. Menghilangkan turunan awalan. Langkah ini terus dilakukan sampai tidak ada lagi turunan awalan. Jika tidak ada lagi, maka lanjutkan ke Langkah berikutnya. Untuk beberapa kombinasi imbuhan, dilakukan penghilangan awalan terlebih dahulu. Yaitu pada kombinasi imbuhan “ber-lah”, “ber-an”, “men-i”, “di-i”, “pe-i”, “ter-i”.
- d. Hilangkan ifleksi akhiran terlebih dahulu. Jika hal ini berhasil dan akhiran adalah partikel (“-lah”, ”-pun” dan ”-kah”), langkah ini dilakukan lagi untuk menghilangkan kata ganti akhiran posesif (“ku”, “mu” atau ”nya”).
- e. Hilangkan turunan akhiran. Langkah ini terus dilakukan sampai tidak ada lagi penurunan akhiran. Jika tidak ada lagi, maka lanjutkan ke Langkah berikutnya.
- f. Setelah setiap penghilangan imbuhan dilakukan, maka lakukan pengecekan menggunakan kamus. Jika kata ditemukan, maka algoritma berhenti dan tidak perlu dilakukan pengecekan terhadap imbuhan lainnya. Jika sampai selesai penghilangan imbuhan masih belum menemukan kata dasar, maka dilakukan recoding.
- g. Jika semua langkah sudah dilakukan termasuk recoding dan tidak juga ditemukan dalam kamus, maka algoritma ini akan menganggap kata semula sebagai kata dasar.

Semua langkah – langkah yang dilakukan oleh proses *stemming* idris dituangkan dalam *flowchart* pada gambar 2.4, agar penjelasan penggunaan algoritma terhadap kata atau kalimat berimbuhan lebih jelas dijelaskan oleh diagram alir atau *flowchart*.



Gambar 2. 4 Flowchart Algoritma Idris

2.2.4. *Stemming Arifin Setiono*

Algoritma Arifin dan Setiono ini mendahulukan pembacaan tiap kata dari data yang ada. Sehingga setiap tahap yang dilakukan dalam algoritma ini adalah sebagai berikut :

1. Pemeriksaan semua kemungkinan bentuk kata. Setiap kata diasumsikan memiliki 2 Awalan (prefiks) dan 3 Akhiran (sufiks). Jika dalam kata yang diperiksa tidak memiliki imbuhan sebanyak imbuhan seperti formula di atas,

maka imbuhan yang kosong atau tidak ada tersebut diberi tanda x untuk prefiks dan diberi tanda xx untuk sufiks.

2. Pemotongan dalam Algoritma ini dilakukan secara berurutan sebagai berikut

AW : AW (Awalan)

AK : AK (Akhiran)

KD : KD (Kata Dasar)

a. AW I, hasilnya disimpan pada pe1 (prefiks 1)

b. AW II, hasilnya disimpan pada pe2 (prefiks 2)

c. AK I, hasilnya disimpan pada su1 (sufiks 1)

d. AK II, hasilnya disimpan pada su2 (sufiks 2)

e. AK III, hasilnya disimpan pada su3 (sufiks 3)

Dalam setiap tahap pemotongan di atas selalu diikuti dengan pemeriksaan di dalam kamus. Hal ini untuk mengetahui apakah hasil pemotongan tersebut sudah ada dalam bentuk dasar. Apabila pemeriksaan ini berhasil maka proses dinyatakan selesai dan tidak perlu melanjutkan proses pemotongan imbuhan selanjutnya.

3. Akan tetapi, apabila sampai pada pemotongan AK III, belum ditemukan dalam kamus, maka akan dilakukan proses kombinasi. Kata dasar yang dihasilkan dikombinasikan dengan imbuhan-imbuhan nya dalam 12 konfigurasi berikut :

a. KD

b. KD + AK III

c. KD + AK III + AK II

d. KD + AK III + AK II + AK I

e. AW I + AW II + KD

f. AW I + AW II + KD + AK III

g. AW I + AW II + KD + AK III + AK II

h. AW I + AW II + KD + AK III + AK II + AK I

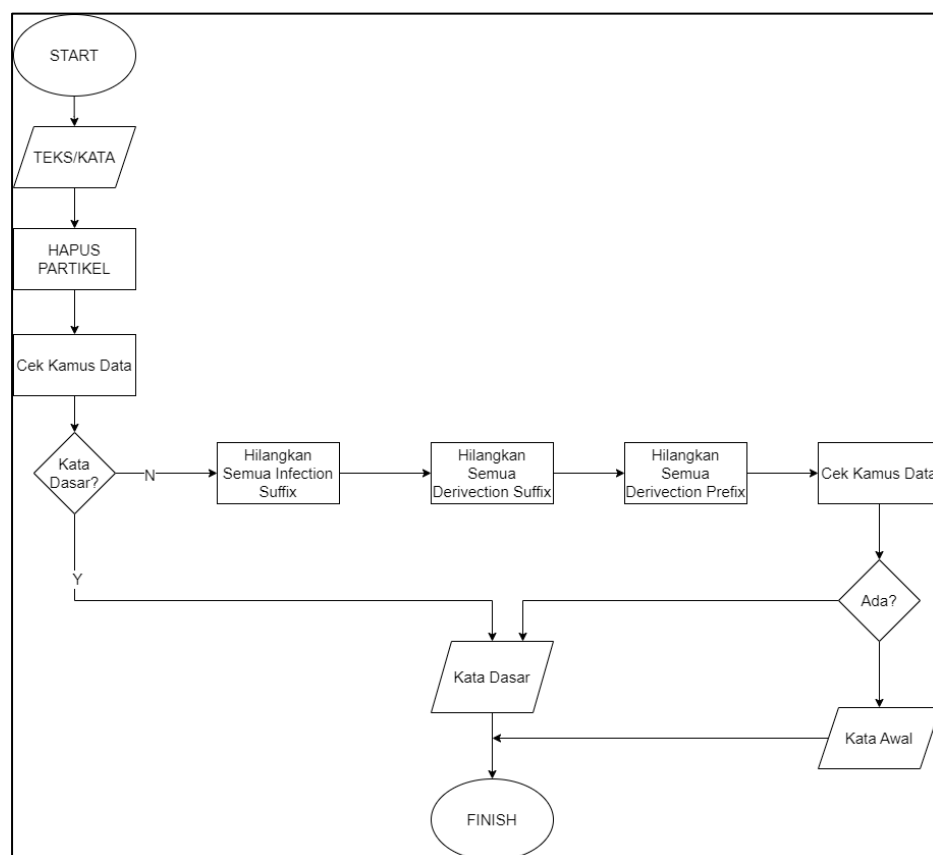
i. AW II + KD

j. AW II + KD + AK III

k. AW II + KD + AK III + AK II

1. AW II + KD + AK III + AK II + AK I

Kombinasi a, b, c, d, h, dan l sudah diperiksa pada tahap sebelumnya, karena kombinasi ini adalah hasil pemotongan bertahap tersebut. Dengan demikian, kombinasi yang masih perlu dilakukan tinggal yakni pada kombinasi-kombinasi yang belum dilakukan (e, f, g, i, j, dan k). Apabila dalam proses kombinasi yang dilakukan itu ada, maka pemeriksaan pada kombinasi lainnya sudah tidak diperlukan lagi. Pemeriksaan dalam 12 kombinasi ini sangat diperlukan karena fenomena *overstemming* pada algoritma pemotongan imbuhan. Kelemahan ini berakibat pada pemotongan bagian kata yang sebenarnya adalah milik kata dasar itu sendiri yang kebetulan mirip dengan salah satu jenis imbuhan yang ada. Dengan 12 kombinasi itu, pemotongan yang sudah terlanjur tersebut dapat dikembalikan sesuai posisinya. *Flowchart* atau alur diagram Algoritma Arifin Setiono ditampilkan pada gambar 2.5



Gambar 2. 5 Flowchart Algoritma Arifin Setiono

2.3 Contoh Implementasi algoritma

Pada bab ini menjelaskan contoh implementasi dari masing – masing algoritma *stemming* yang digunakan pada penelitian ini diantaranya sebagai berikut:

2.3.1 Contoh Implementasi Algoritma Porter

Dari penjelasan algoritma *stemming* porter pada subbab 2.2.1 mengenai proses atau tahapan algoritma *stemming* porter. Berikut ini contoh dari kata berimbuhan “merupakan”, “pencarian”, dan juga “ketepatan” jika diberikan algoritma porter maka hasilnya seperti tabel 2.2.

Tabel 2. 2 Contoh Penggunaan Algoritma Porter

No	Kata Berimbuhan	Hasil Stemming	Kata Dasar Seharusnya
1	Merupakan	Rupa	Rupa
2	Pencarian	Cari	Cari
3	Ketepatan	Tepat	tepat

Tabel 2.2 adalah hasil *stemming* menggunakan algoritma porter dari kata berimbuhan “merupakan”, “pencarian”, dan juga “ketepatan”. Penjelasan tahap implementasi untuk kata “merupakan” adalah menghilangkan partikel “-kah”, “-lah”, “-pun” karena kata berimbuhan yang dicari tidak ada dalam penghilangan partikel maka dilanjutkan ke tahap penghilang *posesive pronoun* seperti “-ku”, “-mu” dilanjutkan kembali ke tahap hapus awalan 1 dan seperti “me-“,,”meny-“,,”meng-“ jika ada dalam kondisi dan sudah masuk kedalam tahap hapus awalan satu maka kata akan keluar menjadi “rupakan” setelah itu di cek menggunakan kamus jika kata tersebut merupakan kata dasar maka akan dikembalikan, namun jika bukan maka akan dimasukan ke tahap penghilang akhiran “kan” ,”I”, “an” jika ada maka dihilangkan kata atau akhiran tersebut dan akan mengembalikan hasil kata menjadi “rupa” dan kata tersebut diasumsikan sebagai kata dasar.

2.3.2 Contoh Implementasi Algoritma Sastrawi

Pada pembahasan contoh implementasi ini melandaskan dari pembahasan sub bab 2.2.2 mengenai proses atau tahapan algoritma sastrawi. Berikut tabel 2.3

merupakan contoh kata berimbuhan “merupakan”, “pencarian”, dan “ketepatan” jika diberikan algoritma sastrawi

Tabel 2. 3 Contoh Implementasi Algoritma Sastrawi

No	Kata Berimbuhan	Hasil Stemming	Kata Dasar Seharusnya
1	Merupakan	Rupa	Rupa
2	Pencarian	Cari	Cari
3	Ketepatan	Tepat	tepat

Hasil dari contoh implementasi algoritma sastrawi untuk kata “merupakan”, “pencarian”, dan “ketepatan” menghasilkan data yang sama seperti algoritma porter namun dengan cara yang berbeda. Jika diterapkan algoritma sastrawi untuk kata “merupakan” adalah sebagai berikut langkah awal adalah melakukan pemeriksaan kata yang akan di *stemming* ke dalam kamus, karena kata “merupakan” bukanlah kata dasar maka dilanjutkan ketahap penghilang kata akhiran “-lah”, “-kah”, “-ku”, “-mu”, “-nya”, “-tah” atau “-pun”. Lanjut ke tahap hapus kata imbuhan akhiran “-i”, “-kan”, “-an”, jika ditemukan maka kata akan keluar menjadi “merupa”. Lalu masuk ketahap hapus kata imbuhan awalan “be-”, “di-”, “ke-”, “me-”, “pe-”, “se-”, dan “te-”. Maka kata berimbuhan tadi akan keluar menjadi “rupa” dan kata tersebut dikatakan sebagai kata dasar.

2.3.3 Contoh Implementasi Algoritma Idris

Dalam contoh implementasi algoritma idris dilakukan pemeriksaan *stemming* untuk kata berimbuhan “merupakan”, “pencarian”, “ketepatan” tabel 2.4 merupakan contoh implementasi algoritma idris.

Tabel 2. 4 Contoh Implementasi Algoritma Idris

No	Kata Berimbuhan	Hasil Stemming	Kata Dasar Seharusnya
1	Merupakan	Rupa	Rupa
2	Pencarian	Cari	Cari
3	Ketepatan	Tepat	tepat

Dari hasil memasukan kata berimbuhan kedalam algoritma idris didapatkan hasil *stemming* seperti tabel 2.4. sebagai penjelasan terdapat kata berimbuhan

yang gagal dirubah menjadi kata dasar seperti kata “merupakan” jika menggunakan algoritma idris. Hal tersebut gagal dikarenakan kata “merupakan” tidak termasuk dalam kombinasi imbuhan seperti “ber-lah”, “ber-an”, “men-i”, “di-i”, “pe-i”, “ter-i”.

2.3.4 Contoh Implementasi Algoritma Arifin Setiono

Untuk contoh implementasi yang terakhir adalah algoritma Arifin setiono yang akan dilakukan pemeriksaan kata berimbuhan “merupakan”, ”pencarian” dan “ketepatan” diubah menjadi kata dasar seperti tabel 2.5.

Tabel 2. 5 Contoh Implementasi Algoritma Arifin Setiono

No	Kata Berimbuhan	Hasil Stemming	Kata Dasar Seharusnya
1	Merupakan	Rupa	Rupa
2	Pencarian	Cari	Cari
3	Ketepatan	Tepat	tepat

Setelah dilakukan semua tahapan algoritma Arifin setiono kata berimbuhan “merupakan”, “pencarian”, dan “ketepatan” seperti di gambar 2.9 hasil pencarian kata dasarnya bisa dikatakan berhasil dilakukan dengan algoritma ini

2.3.5 Contoh Kata yang tidak berhasil di stemming oleh algoritma Porter

Berikut ini akan dijelaskan beberapa contoh kata yang tidak berhasil di *stemming* oleh algoritma porter yang akan dijelaskan dalam gambar 2.10. tujuannya ditampilkan contoh kata yang tidak berhasil di *stemming* oleh algoritma porter adalah agar penelitian ini dapat mengetahui perbedaan dari masing – masing algoritma, sehingga hasil *stemming* pun akan berbeda.

Serta dapat mengetahui alasan mengapa algoritma *stemming* ini harus dilakukan penelitian yang lebih mendalam lagi, dan mengapa setiap algoritma memiliki perbedaan serta ada beberapa algoritma yang akhirnya menjadi disempurnakan. Seperti algoritma porter ini sudah ada beberapa orang yang memperbaharui algoritma ini dan disempurnakan. Kata yang tidak berhasil di *stemming* oleh Algoritma Porter ditampilkan pada tabel 2.6

Tabel 2. 6 Kata yang tidak berhasil di stemming oleh Porter

Data Kata Yang Tidak Bisa Terstemming Oleh Algoritma Porter			
NO	Kata	Hasil	Seharusnya
1	Berupa	Upa	rupa
2	Dokumen – dokumen	Dokumendokumen	dokumen
3	Memiliki	Pilik	milik
4	Masing – masing	Masingmasing	masing
5	Mengetahui	Ketahu	tahu
6	Menyelesaikan	Lesai	selesai
7	Memberikan	I	Beri
8	Tindakan	Tinda	tindak
9	Pendidikan	Didi	Didik
10	Menentukan	Entu	Tentut

Pada tabel 2.6 menjelaskan tentang kata yang tidak berhasil di *stemming* oleh algoritma porter. Kata tersebut tidak berhasil di *stemming* dikarenakan algoritma yang digunakan kemungkinan tidak mendukung untuk kata tersebut

2.3.6 Contoh Kata yang tidak berhasil di stemming oleh algoritma Sastrawi

Berikut ini akan dijelaskan beberapa contoh kata yang tidak berhasil di *stemming* oleh algoritma Sastrawi yang akan dijelaskan dalam gambar 2.11. tujuannya ditampilkan contoh kata yang tidak berhasil di *stemming* oleh algoritma sastrawi adalah agar penelitian ini dapat mengetahui perbedaan dari masing – masing algoritma, sehingga hasil *stemming* pun akan berbeda.

Serta dapat mengetahui alasan mengapa algoritma *stemming* ini harus dilakukan penelitian yang lebih mendalam lagi, dan mengapa setiap algoritma memiliki perbedaan serta ada beberapa algoritma yang akhirnya menjadi disempurnakan. Kata yang tidak berhasil di *stemming* oleh Algoritma Sastrawi ditampilkan pada tabel 2.7

Tabel 2. 7 Kata yang tidak berhasil di stemming oleh Sastrawi

Data Kata Yang Tidak Bisa Terstemming Oleh Algoritma Porter			
NO	Kata	Hasil	Seharusnya
1	Memberikan	Berik	beri
2	Diajukan	Ketahu	tahu
3	Diberikan	Berik	beri
4	Memastikan	Mastik	pasti
5	Tangan	Tang	Tangan
6	Perubahan	Rubah	ubah
7	Memetakan	Meta	peta

Data Kata Yang Tidak Bisa Terstemming Oleh Algoritma Porter			
NO	Kata	Hasil	Seharusnya
8	Bali	Bal	Bali
9	Dikatakan	Katak	kata
10	Sesuai	Suai	Sesuai

Pada tabel 2.7 menjelaskan tentang kata yang tidak berhasil di *stemming* oleh algoritma Sastrawi. Kata tersebut tidak berhasil di *stemming* dikarenakan algoritma yang digunakan kemungkinan tidak mendukung untuk kata tersebut

2.3.7 Contoh Kata yang tidak berhasil di stemming oleh algoritma Idris

Berikut ini akan dijelaskan beberapa contoh kata yang tidak berhasil di *stemming* oleh algoritma Idris yang akan dijelaskan dalam tabel 2.8. tujuannya ditampilkan contoh kata yang tidak berhasil di *stemming* oleh algoritma idris adalah agar penelitian ini dapat mengetahui perbedaan dari masing – masing algoritma, sehingga hasil *stemming* pun akan berbeda.

Selain mengetahui perbedaan kesalahan dari masing – masing algoritma, kesalah pada *stemming* juga menjadi alasan yang kuat mengapa penelitian kali ini dilakukan sehingga dapat mengetahui algoritma mana yang bisa dibilang terbaik untuk dilakukan *stemming* teks Bahasa Indonesia. Kata yang tidak berhasil di *stemming* oleh Algoritma Idris ditampilkan pada tabel 2.8.

Tabel 2. 8 Kata yang tidak berhasil di stemming oleh Idris

Data Kata Yang Tidak Bisa Terstemming Oleh Algoritma Porter			
NO	Kata	Hasil	Seharusnya
1	Menyesatkan	Menyesatkan	sesat
2	Menyampaikan	Menyampaikan	sampai
3	Sebagai	Sebagai	Bagai
4	Kebenaran	Kebenaran	Benar
5	Bertujuan	Bertujuan	tuju
6	Mempengaruhi	Mempengaruhi	Pengaruh
7	Mengambil	Mengambil	Ambil
8	Mengetahui	Mengetahui	Tahu
9	Diperlukan	Diperlukan	perlu
10	Menggunakan	Menggunakan	Guna

Pada tabel 2.8 menjelaskan tentang kata yang tidak berhasil di *stemming* oleh algoritma Sastrawi. Kata tersebut tidak berhasil di *stemming* dikarenakan algoritma yang digunakan kemungkinan tidak mendukung untuk kata tersebut

2.3.8 Contoh Kata yang tidak berhasil di stemming oleh algoritma Arifin Setiono

Berikut ini akan dijelaskan beberapa contoh kata yang tidak berhasil di *stemming* oleh algoritma Arifin Setiono yang akan dijelaskan dalam tabel 2.9. tujuannya ditampilkan contoh kata yang tidak berhasil di *stemming* oleh algoritma Arifin setiono adalah agar penelitian ini dapat mengetahui perbedaan dari masing – masing algoritma, sehingga hasil *stemming* pun akan berbeda.

Jika dilihat dari hasil dan contoh – contoh kata yang tidak berhasil di *stemming* dan dibandingkan dengan masing – masing algoritma yang digunakan pada penelitian kali ini. Titik kesalah dari setiap kata berbeda dilihat dari algoritma dan aturan yang digunakan oleh algoritma itu sendiri. Kata yang tidak berhasil di *stemming* oleh Algoritma Arifin Setiono ditampilkan pada tabel 2.9

Tabel 2. 9 Kata yang tidak berhasil di stemming oleh Arifin Setiono

Data Kata Yang Tidak Bisa Terstemming Oleh Algoritma Porter			
NO	Kata	Hasil	Seharusnya
1	Anak – anak	Anakanak	Anak
2	Dikarenakan	Diakrenakan	Karena
3	Tokoh – tokoh	Tokohtokoh	Tokoh
4	Perolehan	Perolehan	Oleh
5	Menyampaikan	Menyampaikan	Sampai
6	Bertujuan	Bertujuan	Tuju
7	Penyaluran	Penyaluran	Salur
8	Seluruhnya	Seluruhnya	Seluruh
9	Mengenai	Mengenai	Kena
10	Dituangkan	Dituangkan	tuang

Pada tabel 2.9 menjelaskan tentang kata yang tidak berhasil di *stemming* oleh algoritma Arifin Setiono. Kata tersebut tidak berhasil di *stemming* dikarenakan algoritma yang digunakan kemungkinan tidak mendukung untuk kata tersebut.

2.3.9 Kata yang berhasil dan tidak dari seluruh algoritma

Pada bab ini dijelaskan beberapa data yang berhasil dan tidak untuk seluruh algoritma dan dituangkan pada taabel 2.10.

Tabel 2. 10 Hasil Stemming Seluruh Algoritma

No	Kata	Porter		Sastrawi		Idris		ArifinSetiono	
		Hasil	seharusnya	Hasil	seharusnya	Hasil	seharusnya	Hasil	seharusnya
1	Berupa	Upa	Rupa	Rupa	Rupa	Rupa	Rupa	Berupa	rupa
2	Dikenal	Kenal	Kenal	Kenal	Kenal	Kenal	Kenal	Dikenal	kenal
3	Merupakan	Rupa	Rupa	Rupa	Rupa	Merupakan	Rupa	Rupa	rupa
4	Memiliki	Pilik	Milik	Milik	Milik	Memiliki	Milik	Memiliki	milik
5	Berimbuhan	Imbuh	Imbuh	Imbuh	Imbuh	Berimbuhan	Imbuh	Imbuh	imbuh

2.4 PHP (Fungsi Microtime)

PHP merupakan bahasa pemrograman *script server-side* yang dirancang khusus untuk pengembangan *website*. Selain itu, *PHP* dapat berfungsi sebagai bahasa pemrograman umum. Dibuat pada tahun 1995 oleh Rasmus Lerdorf, *PHP* disebut sebagai bahasa pemrograman *server-side* karena diproses di komputer *server*. Ini berbeda dari bahasa pemrograman *client-side* seperti *JavaScript* yang diproses di *web browser (client)* (Suhartini, Sadali, & Putra, 2020).

Dalam Bahasa pemrograman *PHP* terdapat fungsi yang disebut sebagai *microtime*, dimana fungsi tersebut digunakan untuk menghitung waktu atau lamanya sebuah algoritma atau perintah *script* dieksekusi oleh program tersebut. *Microtime* tersebut digunakan pada penelitian ini guna untuk menghitung lamanya waktu dari masing – masing algoritma yang digunakan pada penelitian ini.

2.5 Algoritma Presisi

Dari langkah – langkah yang dijelaskan pada bagian teknik penelitian terdapat tingkat keakuratan algoritma yang dihitung menggunakan rumus 2.1.

$$Presisi = \frac{RW}{W} \times 100$$

(2. 1)

Dimana *W* adalah jumlah kata yang dilakukan proses stemming, dan *RW* adalah jumlah kata yang berhasil dilakukan stemming atau dinyatakan proses stemmingnya benar. Serta perhitungan ini akurasi ini akan dinyatakan dalam hitungan persen (%). Sebagai contoh penggunaan algoritma presisi ini terdapat

studi kasus yang memiliki 10 data kata dalam 1 dokumen yang berhasil di stemming menggunakan algoritma porter dalam hitungan manual seperti pada tabel 2.11

Tabel 2. 11 Contoh Studi Kasus Perhitungan Presisi

No	Kata Berimbuhan	Hasil	Jumlah	Status
1	Pencarian	Cari	2	Berhasil
2	Menentukan	Tentu	1	Berhasil
3	Berupa	Upa	1	Gagal
4	Dikenal	Kenal	1	Berhasil
5	Mengambil	Ambil	1	Berhasil
6	Memberikan	Berik	1	Gagal
7	Diberikan	Berik	1	Gagal
8	Merupakan	Rupa	1	Berhasil
9	Memisahkan	Pisah	1	Berhasil
10	Dianggap	Anggap	1	Berhasil

Pada tabel 2.11 menunjukan hasil dari perhitungan manual yang dilakukan oleh rumus presisi 2.1, sehingga menghasilkan data perhitungan seperti tabel 2.12 yang menunjukan hasil seluruh perhitungan algoritma presisi

Tabel 2. 12 Hasil Perhitungan manual algoritma presisi

Jumlah Kata Berhasil di <i>Stemming</i>	8
Jumlah Kata Gagal di <i>Stemming</i>	3
Jumlah Seluruh Kata Dalam Indeks	11
Presisi	$(8/11) \times 100 = 72,72\%$

Pada tabel 2.12 terdapat jumlah kata yang berhasil di stemming yaitu 8 kata dan total seluruh kata yang dilakukan stemming adalah 11 data dilihat dari indeks jumlah kata yang berhasil distemming.

2.6 Rumus Perhitungan Waktu

Pada penelitian kali ini digunakan perhitungan parameter waktu untuk menghitung kecepatan algoritma yang digunakan, rumus ini diimplementasikan menggunakan perhitungan waktu pada pemograman *PHP* yang memiliki fungsi *microtime* seperti yang dijelaskan pada bab 2.5 dengan cara memasang *microtime* pada awal algoritma dijalankan dan juga pada saat algoritma berakhir dijalankan. Rumus tersebut akan dijelaskan pada rumus 2.2 sebagai berikut.

$$Waktu = EndMicrotime - StartMicrotime$$

(2. 2)

Pada rumus 2.2 StartMicrotime digunakan saat awal algoritma dijalankan agar mengatur awal mula algoritma bekerja, sementara untuk EndMicrotime digunakan pada akhir algoritma dijalankan agar menghitung waktu terakhir yang ditempuh oleh algoritma tersebut. Setelah itu akan dilakukan pengurangan antara EndMicrotime dengan StartMicrotime agar hasilnya menjadi detik (PHP, 2001 - 2023).

BAB III

METODE PENELITIAN

Pada bab ini dijelaskan mengenai metode penelitian, teknik penelitian, studi kasus serta tahapan – tahapan yang digunakan pada penelitian

3.1 Metode Penelitian

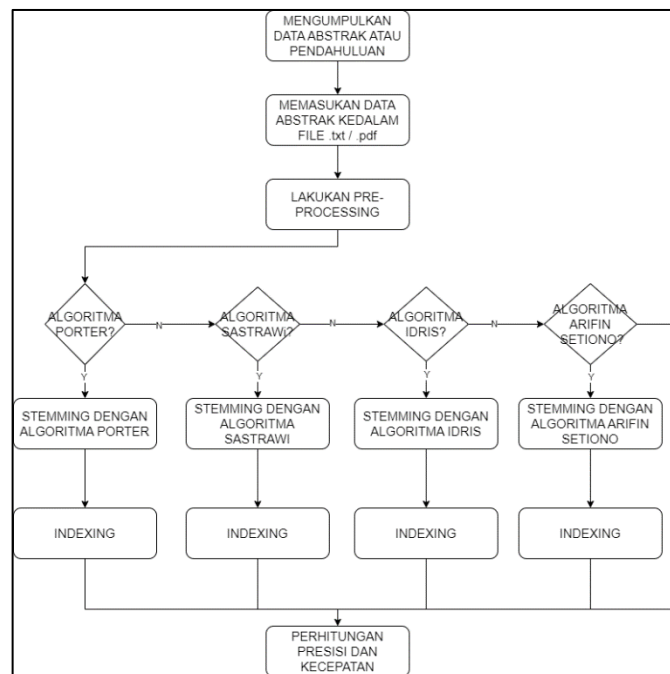
Pada penelitian ini menggunakan metode penelitian komparatif, dimana metode ini digunakan untuk mengetahui perbedaan parameter yang diteliti, tidak untuk kemampuan manipulasi dalam proses penelitiannya dengan tujuan agar data yang dihasilkan benar – benar objektif dan akurat. Dapat dikatakan bahwa metodekomparatif dilakukan selama mungkin sehingga hasil dari analisa pada perbedaan parameter terlihat jelas.

Tujuan penulis menggunakan metode ini adalah cocok dengan kasus yang dihadapi karena studi kasus ini membandingkan tentang ketepatan dan kecepatan *stemming* pada sebuah dokumen teks. Sehingga diharapkan memiliki luaran nilai ketepatan dan kecepatan dalam bentuk persen. Metode penelitian komparatif ini membandingkan 2 parameter yaitu ketepatan dan kecepatan *stemming* pada dokumen teks dengan menggunakan algoritma porter, sastrawi, idris dan Arifin setiono.

Desain penelitian yang digunakan dalam penelitian ini adalah perbandingan universal karena jenis perbandingan ini adalah jenis perbandingan yang digunakan untuk menetapkan bahwa setiap fenomena dari kejadian-kejadian mengikuti aturan yang sama. Jenis perbandingan universal ini akan menggunakan perbandingan untuk mengembangkan landasan teori, dengan tujuan untuk memberikan teori yang akan membantu menjelaskan kasus yang sedang diteliti dalam penelitian. Desain penelitian komparatif perbandingan universal ini dapat dilakukan dengan langkah – langkah yaitu merumuskan masalah, mengkaji teori, mengumpulkan data latar belakang / abstrak dalam suatu jurnal, mengumpulkan data kata dasar, menganalisa data dan membuat kesimpulan dan saran dari hasil penelitian. Selain itu ada juga teknik penelitian yang dilakukan teknik tersebut adalah sebagai berikut.

3.1.1 Teknik penelitian

Teknik penelitian atau cara penelitian yang dilakukan kali ini adalah dengan cara mengambil data dari setiap jurnal dan laporan tugas akhir yang akan digunakan dan format dokumen tersebut memiliki format data .pdf atau .txt agar saat pengujian sistem akan dibatasi akses penerimaan data file yang di upload. Setelah file – file yang digunakan terkumpul maka Langkah selanjutnya adalah memasukan file – file tersebut kedalam masing algoritma *stemming* yang digunakan, sehingga bisa dibanding perbandingan kali ini dilakukan pada masing – masing algoritma dan tidak ada kombinasi algoritma yang dilakukan dikarenakan tujuan dari penelitian ini ingin mencari salah satu algoritma *stemming* terbaik bukan dari hasil kombinasi. Setelah langkah pemasukan algoritma *stemming* dilakukan maka akan diuji dari masing – masing algoritma tersebut dengan perhitungan dan algoritma presisi serta kecepatan. Gambar 3.1 merupakan blok diagram yang dibuat untuk langkah teknik penelitian ini.



Gambar 3. 1 Blok Diagram Teknik Penelitian

3.2 Teknik Pengumpulan Data

Data yang digunakan untuk pengujian penelitian diambil dari jurnal dan laporan tugas akhir untuk dilakukan proses *stemming* menggunakan algoritma

porter, sastrawi, idris dan Arifin setiono. Latar belakang atau abstrak suatu jurnal dalam penelitian adalah subjek dari penelitian yang dilakukan dan sampel sebagai sebagian atau wakil dari jurnal tersebut sebagai informasi / data yang akan dijelaskan sebagai berikut:

3.2.1 Sampel

Sampel merupakan sebagian atau wakil dari populasi sebagai sumber informasi/data. Sampel yang akan diambil sebagai percobaan harus diperhatikan. Karena sampel akan digunakan untuk menguji aplikasi dari segi ketepatan dan kecepatan dalam proses *stemming* pada dokumen teks. Sampel pada penelitian kali ini biasanya berupa jurnal yang telah di publish dengan melihat isi latar belakang yang ada dalam jurnal tersebut. Maka dari itu pada penelitian.

ini diharapkan luaran yang didapat adalah hasil dari ketepatan serta kecepatan proses *stemming* menggunakan algoritma yang telah dibahas sebelumnya dan mengetahui performansi mana yang lebih baik dari algoritma yang dibandingkan tersebut.

3.3 Analisis Kebutuhan Sistem

Pada bab ini dijelaskan analisis untuk mengetahui perangkat yang dibutuhkan atau yang digunakan guna menyelesaikan penelitian yang dilakukan.

3.3.1 Spesifikasi perangkat keras yang digunakan

Pada saat membangun sistem untuk penelitian ini digunakan perangkat keras yang menunjang kebutuhan sistem ini yaitu berupa laptop dengan spesifikasi sebagai berikut :

- a. Processor Intel Core i5
- b. Ram 8gb
- c. SSD 512gb
- d. HDD 512gb

3.3.2 Spesifikasi perangkat lunak yang digunakan

Pada saat membangun sistem untuk penelitian ini digunakan perangkat lunak yang menunjang kebutuhan sistem dengan spesifikasi sebagai berikut:

- a. Sistem Operasi Windows 11
- b. Web Browser (Google Chrome)
- c. PHP8
- d. Visual Studio Code
- e. XAMPP
- f. GitHub Desktop

3.3.3 Dataset yang digunakan

Pada penelitian kali ini dataset digunakan untuk mendapatkan hasil perhitungan presisi dari setiap masing – masing algoritma *stemming* yang dilakukan. Dataset didapatkan dari hasil *preprocessing* dan stemming setiap algoritma dan dimasukkan kedalam index sehingga menjadi bentuk tabel yang nantinya data dalam tabel tersebut akan diolah sehingga menghasilkan nilai presisi. Seluruh data diambil dari abstrak, latar belakang, atau juga judul jurnal yang akan dimasukkan terlebih dahulu kedalam file dengan ekstensi .txt karena sistem yang dibuat akan hanya membaca file dengan ekstensi .txt

3.4 Perancangan Umum

Perancangan umum pada tahapan penelitian kali ini dibagi menjadi 3 bagian diantaranya adalah gambaran umum, gambaran sistem (input, proses, output), dan *flowchart*.

3.4.1 Gambaran Umum

Untuk terkait gambaran umum pada sistem atau aplikasi yang dibuat di penelitian kali ini akan dijelaskan secara garis besar fungsi dan interaksi yang dilakukan user kepada sistem sesuai aturan atau fungsionalitas yang dibuat. Dalam pembuatan algoritma stemming diperlukan data yang memiliki file ekstensi .txt dan memiliki kata imbuhan didalam file tersebut.

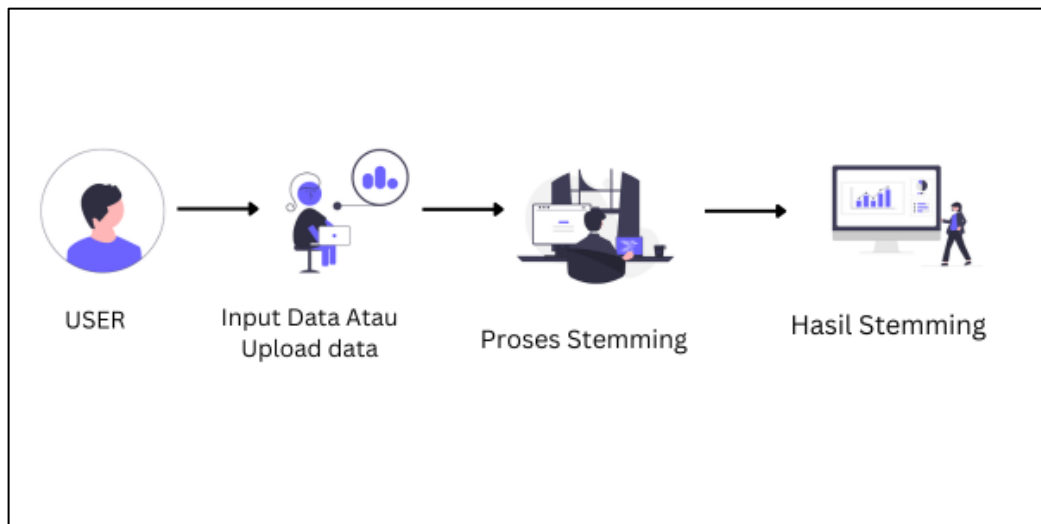
Selanjutnya, proses yang dilakukan adalah memilih algoritma stemming yang akan dilakukan dan upload file data yang memiliki ekstensi .txt kedalam sistem tersebut. Sehingga hasil kata asli dan hasil kata stemming akan keluar dan

seluruh hasil stemming akan dimasukkan kedalam index sehingga dijadikan dataset untuk perhitungan presisi nantinya.

Langkah terakhir yang dilakukan adalah submit data index kedalam database, fungsi database ini diperuntukan untuk history setiap algoritma yang digunakan serta nama file dan siapa yang melakukan percobaan tersebut.

3.4.2 Alur Kerja Sistem

Pada bab ini ilustrasi atau alur kerja sistem aplikasi yang dimulai dari proses masukan data oleh user sampai dengan keluaran hasil proses stemming yang diberikan sistem akan ditampilkan pada gambar 3.3.



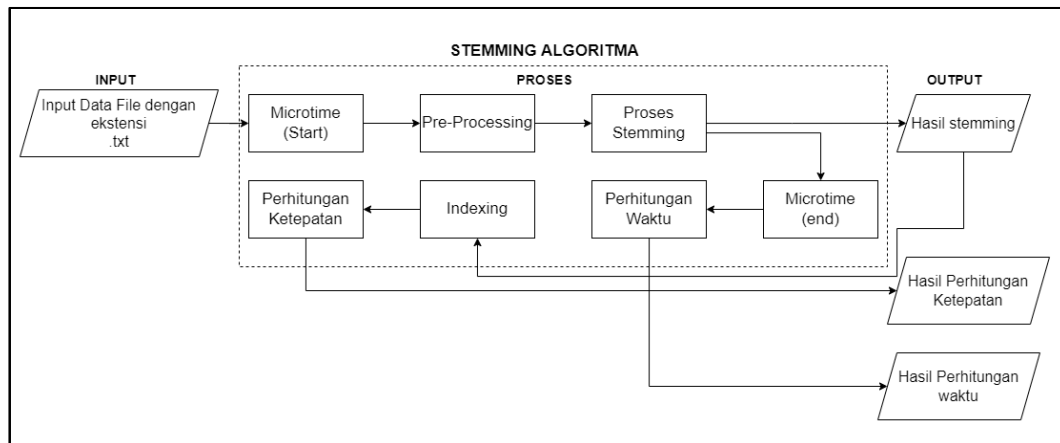
Gambar 3. 2 Alur Kerja Sistem

Alur kerja pada gambar 3.3 atau proses bisnis terdapat keterangan sebagai berikut :

- a. Upload file dokumen teks Bahasa Indonesia dengan ekstensi .pdf dan .txt
- b. Sistem akan melakukan langkah *preprocessing* serta melakukan proses stemming dengan algoritma yang digunakan pada penelitian ini.
- c. Setelah berhasil melakukan proses stemming sistem akan menampilkan data hasil stemming (*output*)
- d. Langkah terakhir setelah semuanya berhasil dilakukan adalah mengirim seluruh pengujian algoritma stemming ke database agar memiliki *history*

3.4.3 Blok Diagram

Pada bab ini akan dijelaskan Blok Diagram mengenai sistem yang telah dibuat pada penelitian ini. Blok Diagram tersebut akan dijelaskan pada gambar 3.4 agar lebih jelas dan terlihat alur dari sistem yang dibuat dalam penelitian ini.



Gambar 3. 3 Blok Diagram Sistem

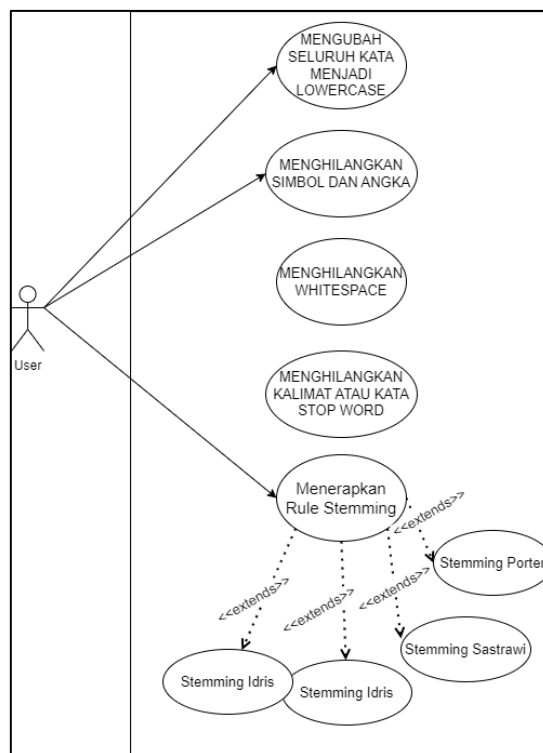
Blok diagram sistem pada gambar 3.4 memiliki keterangan sebagai berikut :

- Proses pertama adalah melakukan *input* data atau *upload* file berekstensi .txt kedalam sistem
- Setelah file berhasil di masukan lalu pasang fungsi *microtime php* dan inisialisasi sebagai *start micortime*
- Selanjutnya langkah *pre-processing* dilakukan oleh sistem setelah menerima data yang berhasil di *upload*
- Lalu sistem akan melakukan proses *stemming* dengan masing – masing algoritma yang digunakan dari hasil *pre-processing*
- Setelah berhasil dilakukan proses *stemming* sistem akan mengeluarkan hasil dari *stemming* tersebut.
- Setelah itu pasang end microtime setelah algoritma proses stemming.
- Dari hasil endmicrotime akan dilakukan langkah perhitungan waktu menggunakan rumus waktu seperti pada bab 2.6 dan akan ditampilkan hasil perhitungan waktu tersebut
- Hasil stemming yang telah dikeluarkan akan masuk kedalam indexing pengumpulan jumlah stemming

- i. Dari hasil index tersebut akan dimasukan kedalam rumus perhitungan ketepatan. Setelah berhasil dihitung akan ditampilkan hasil dari ketepatan berikut.
- j. Langkah terakhir adalah user diminta memasukan data ke dalam database dengan menekan tombol submit (jika ingin menyimpan *history* hasil dari proses stemming).

3.4.4 Usecase Diagram

Pada bab ini akan dijelaskan seluruh kinerja sistem terhadap penerapan algoritma stemming yang akan dituangkan dalam sebuah usecase diagram. Usecase tersebut akan dijelaskan lebih dalam lagi dengan konsep scenario usecase yang dibuat untuk setiap masing – masing langkah yang dilakukan. Penjelasan mengenai proses usecase diagram ditampilkan pada gambar 3.4



Gambar 3. 4 Usecase Diagram

Berdasarkan *usecase* pada gambar 3.4 semua penjelasan mengenai *usecase* yang dibuat tentang sistem aplikasi *stemming* dijelaskan semua pada tabel 3.1 – 3.5. berikut ini adalah penjelasan mengenai *usecase* tersebut.

Tabel 3. 1 UC-01 Usecase Diagram

UC-01		
Nama	Mengubah Seluruh Kata Menjadi <i>Lower Case</i>	
Tujuan	Agar seluruh kata menjadi satu jenis yaitu <i>lower case</i>	
Deskripsi	Sistem mengubah seluruh kata yang berada dalam file menjadi <i>lower case</i>	
Trigger	Langkah dimulai Ketika file berhasil di masukan kedalam sistem	
Flow Of Events	User	Sistem
	1. User memasukan file berkestensi .txt untuk proses <i>stemming</i>	2. Membaca dan mengubah seluruh kata dalam file menjadi <i>lower case</i> dengan fungsi <i>php</i>
Kondisi akhir	Seluruh kata menjadi <i>lower case</i>	

Tabel 3. 2 UC-02 Usecase Diagram

UC-02		
Nama	Menghilangkan simbol dan angka	
Tujuan	Agar dalam pengujian hanya menguji kata dalam file	
Deskripsi	Sistem dapat mengubah atau menghilangkan seluruh simbol dan angka	
Trigger	Langkah ini dimulai Ketika proses <i>lower case</i> sudah berhasil dilakukan	
Flow Of Events	Sistem	
	1. Sistem menerima hasil dari langkah <i>lower case</i> 2. Sistem mengubah dan menghilangkan simbol dan angka dari kata yang diterima pada langkah <i>lower case</i>	
Kondisi akhir	Sistem mengeluarkan hasil kata tanpa adanya simbol dan angka	

Tabel 3. 3 UC-03 Usecase Diagram

UC-03	
Nama	Menghilangkan <i>whitespace</i>
Tujuan	Agar seluruh kata yang akan diuji tidak terdapat <i>whitespace</i> yang terdeteksi oleh sistem
Deskripsi	Sistem dapat menghilangkan seluruh <i>whitespace</i> pada kata
Trigger	Dilakukan setelah proses penghilangan simbol dan angka
<i>Flow Of Events</i>	<p>Sistem</p> <ol style="list-style-type: none"> 1. Sistem menerima hasil dari langkah penghilang simbol dan angka 2. Sistem menghilangkan <i>whitespace</i> yang terjadi karna langkah sebelumnya
Kondisi akhir	Sistem mengeluarkan seluruh kata yang sudah bersih dari <i>whitespace</i>

Tabel 3. 4 UC-04 Usecase Diagram

UC-04	
Nama	Menghilangkan kata yang termasuk <i>stop word</i>
Tujuan	Seluruh data yang tidak memiliki makna akan dihilangkan
Deskripsi	Sistem dapat memotong atau menghilangkan kata yang tidak akan terdeteksi oleh kamus besar Bahasa Indonesia
Trigger	Langkah ini dilakukan Ketika sistem sudah menerima hasil dari langkah penghilang <i>whitespace</i>
<i>Flow Of Events</i>	<p>Sistem</p> <ol style="list-style-type: none"> 1. Sistem menerima hasil kata yang sudah bersih dari <i>whitespace</i> 2. Sistem menghilangkan kata yang termasuk <i>stopword</i> seperti (adalah, dan, dia, dll.)
Kondisi akhir	Sistem menghasilkan kata yang hanya memiliki makna menurut kamus besar Bahasa indonesia

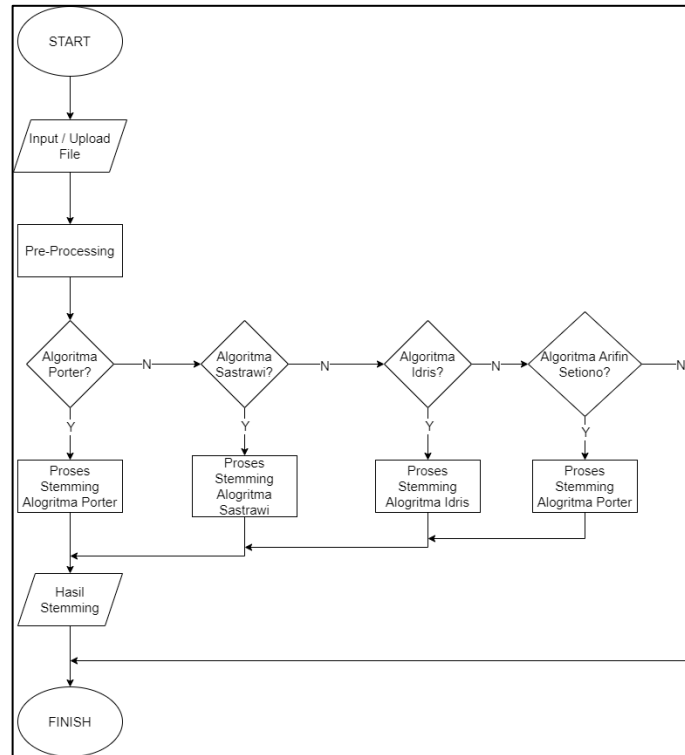
Tabel 3. 5 UC-05 *Usecase* Diagram

UC-05	
Nama	Menerapkan Aturan <i>Stemming</i>
Tujuan	Seluruh kata yang sudah dilakukan tahapan sebelumnya dilakukan pengujian menggunakan aturan <i>stemming</i>
Deskripsi	Sistem mengubah seluruh kata berimbuhan menjadi kata dasar
Trigger	Langkah ini dilakukan setelah seluruh langkah sebelumnya (<i>preprocessing</i>) berhasil dilakukan
Flow Of Events	Sistem
	<ol style="list-style-type: none"> 1. Sistem menerima data yang sudah berhasil dilakukan proses <i>pre-processing</i> 2. Sistem mengubah kata berimbuhan menjadi kata dasar menggunakan aturan <i>stemming</i>
Kondisi akhir	Sistem menghasilkan kata dasar yang cocok dengan kamus besar bahasa indonesia

Untuk setiap langkah proses stemming yang dilakukan mengikuti seluruh aturan algoritma stemming yang digunakan pada penelitian ini yaitu Porter, Sastrawi, Idris dan Arifin & Setiono.

3.4.5 *Flowchart* (Diagram Alir) Sistem

Pada bab ini proses atau alur dari kerja sistem aplikasi stemming yang dibuat akan dituangkan dalam diagram alir sistem pada gambar 3.5. agar lebih jelas kembali pemahaman tentang algoritma stemming yang dituangkan kedalam aplikasi itu seperti apa. Seluruh alur sistem yang dibuat pada diagram alir ini dibuat dengan menyesuaikan pada *usecase* diagram sebelumnya yang telah dibuat pada bab 3.4.4. semua alur mengikuti komunikasi antara user dan sistem yang telah dibuat.



Gambar 3. 5 Flowchart Sistem

Untuk tahapan diagram alir akan dijelaskan dalam keterangan proses berikut ini:

- Langkah awal adalah memasukan atau *upload* file kedalam sistem
- Lalu sistem akan melakukan proses *pre-processing*
- Jika memilih untuk di proses algoritma porter maka akan dilanjutkan ke tahap proses stemming algoritma proter. Jika tidak akan masuk ke tahap poin d
- Jika memilih proses algoritma Sastrawi, maka dilanjutkan ke tahap proses stemming algoritma sastrawi. Jika tidak maka akan masuk ke tahap poin e
- Jika memilih proses algoritma Idris, maka dilanjutkan ke tahap proses stemming algoritma idris. Jika tidak maka akan masuk ke tahap poin f
- Jika memilih proses algoritma Arifin Setiono, maka dilanjutkan ke tahap proses stemming algoritma Arifin setiono. Jika pada tahap ini juga tidak memilih maka proses tidak dilakukan

- g. Jika seluruh tahap poin c-f dilakukan maka dari setiap masing – masing algoritma yang dipilih akan menampilkan hasil stemmingnya

Dalam tahapan poin keterangan diagram alir sistem seluruhnya berdasarkan pilihan menu yang dipilih oleh user, Karna dalam sistem terdapat menu – menu untuk memilih algoritma mana yang akan dilakukan proses stemming untuk dokumen yang akan di *upload* oleh user .

3.5 Studi Kasus

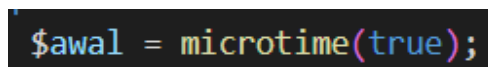
Pada penelitian ini telah dibuat studi kasus untuk contoh penggunaan atau Perbandingan algoritma stemming yang dijelaskan pada blok diagram bab 3.4.3. langkah – langkah yang dibuatkan adalah memasukan data menggunakan file yang memiliki extension .pdf dan .txt. berikut ini penjelasan setiap langkah yang dilakukan.

3.5.1 Memasukan File Yang memiliki ekstension .pdf / .txt.

Pada tahapan ini file yang dibuat dalam .pdf atau .txt dimasukan kedalam sistem bagian input dokumen yang berada dalam aplikasi stemming. Salah satu contoh yang digunakan disini menggunakan file stemming.txt untuk dilakukan uji coba pada studi kasus ini.

3.5.2 Fungsi Start *Microtime*

Fungsi *microtime* akan dipasang saat akan melakukan proses *pre-processing* tujuannya adalah agar fungsi ini dapat menginisialisasikan bahwa algoritma stemming siap dijalankan atau dieksekusi oleh sistem yang berbasis *PHP*, algoritma akan dimasukan sesuai denga aturan yang dimiliki dari masing – masing algoritma tersebut. Penggunaan fungsi tersebut akan dijelaskan pada gambar 3.6.



```
$awal = microtime(true);
```

Gambar 3. 6 Fungsi Awal Microtime PHP

Pada gambar 3.6 dijelaskan bahwa fungsi *StartMicrotime* disimpan dalam variabel *\$awal* yang nantinya akan dilakukan perhitungan waktu oleh rumus perhitungan waktu.

3.5.3 Pre – Processing

Pada tahapan pre – processing ini memiliki tahapan seperti *case folding*, *Tokenizing*, *Filtering*. Berikut penjelasan mengenai tahapan pre-processing

1. Case Folding

Pada tahapan *case folding* ini kata yang dilakukan stemming akan diubah menjadi *lower case* atau kata menjadi huruf tidak kapital berlaku untuk seluruh kata yang digunakan, langkah ini juga menghapus tanda baca, angka dan juga *whitespace* atau spasi Pembuatan tahapan ini dilakukan dengan fungsi seperti gambar 3.7

```
$lower = mb_convert_case($kata, MB_CASE_LOWER, "UTF-8");
$stringreplace = array('/', '\\', ',', '.', ':', ';', '\\', ' ', '{', '}', '(', ')', '[', ']', '^', '~', '!', '@', '%', '$', '^', '&', '*', '=', '?', '+', '-', '_', '0', '1', '2', '3', '4', '5', '6', '7', '8', '9');
$lower = str_replace($stringreplace, "", $lower);
$lower = str_replace(' ', '-', $lower);
$lower = preg_replace('/[A-Za-z0-9 - ? " ' ! ]/', ' ', $lower);
$lower = str_replace('-', ' ', $lower);
$lower = str_replace(' ', ' ', $lower);
$lower = str_replace(' ', ' ', $lower);
$lower = str_replace(' ', ' ', $lower);
```

Gambar 3. 7 Case Folding PHP

Dari hasil *case folding* kata yang digunakan dalam proses stemming pada dokumen stemming.txt adalah seperti pada tabel 3.6.

Tabel 3. 6 Hasil Case Folding

No	Kalimat Asli	Hasil <i>Case Folding</i>
1	Nazief & Adriani Memiliki Tingkat Keakurasian paling tinggi dengan 0.1% dibandingkan dengan algoritma Arifin Setiono	nazief adriani memiliki tingkat keakurasian paling tinggi dengan keakurasian dengan dibandingkan dengan algoritma arifin setiono

2. Tokenizing

Tahapan *tokenizing* dibuat agar masing – masing kata dari kalimat yang berada dalam file .pdf atau .txt dimasukan kedalam index agar dapat dibaca menjadi perkata dan disimpan dalam array yang berada

dalam fungsi *PHP*. Contoh penggunaan *tokenizing* adalah seperti gambar 3.8.

```
$explodestem = explode(' ', $preproses);
```

Gambar 3. 8 Fungsi Tokenizing PHP

Hasil *tokenizing* dalam pre-processing ditunjukkan pada tabel 3.7 berikut ini.

Tabel 3. 7 Hasil Tokenizing

No	Kata Hasil <i>Case Folding</i>	Hasil <i>Tokenizing</i>
1	nazief adriani memiliki tingkat keakurasian paling tinggi dengan keakurasian dengan dibandingkan dengan algoritma arifin setiono	[0] => nazief [1] => adriani [2] => memiliki [3] => tingkat [4] => keakurasian [5] => paling [6] => tinggi [7] => dengan [8] => keakurasian [9] => dengan [10] => dibandingkan [11] => dengan [12] => algoritma [13] => arifin [14] => setiono

3. *Filtering*

Tahapan *filtering* dibuat dengan menggunakan Bahasa program *PHP*, dimana tahapan ini adalah menghilangkan kata kurang penting (stopword). Contoh kata kurang penting adalah kata yang tidak memiliki makna seperti “yang”, “dan”, “di”, “dari”, “dengan”, dll. Contoh pembuatan proses *filtering* adalah seperti gambar 3.9.

```
include "../config/database.php";
$db = $koneksi->query("SELECT * FROM dictionary WHERE stopwords = 'Ya'");
$stopword = array();
while($dt = $db->fetch_array()){
    $stopword[] = $dt['word'];
}
```

Gambar 3. 9 Contoh Filtering PHP

Dari gambar 3.9 hasil dari tahapan *filtering* ditunjukkan pada tabel 3.8 berikut ini.

Tabel 3. 8 Hasil Filtering

No	Hasil Tokenizing	Hasil Filtering
1	[0] => nazief [1] => adriani [2] => memiliki [3] => tingkat [4] => keakurasian [5] => paling [6] => tinggi [7] => dengan [8] => keakurasian [9] => dengan [10] => dibandingkan [11] => dengan [12] => algoritma [13] => arifin [14] => setiono	[0] => nazief [1] => adriani [2] => memiliki [3] => tingkat [4] => keakurasian [5] => paling [6] => tinggi [7] => keakurasian [8] => dibandingkan [9] => algoritma [10] => arifin [11] => setiono

3.5.4 Proses Stemming

Pada dokumen studi kasus yang telah dilakukan *pre-processing* sebelumnya pada bab 3.5.3 berikut ini dijelaskan mengenai tahapan proses *stemming* yang dilakukan pada penelitian. Pada studi kasus kali ini proses *stemming* dibuatkan 3 contoh kata dari hasil *pre-processing*, kata tersebut adalah “memiliki”, “keakurasian”, “dibandingkan”. Sementara sebagai contoh algoritma yang digunakan pada studi kasus ini adalah algoritma *Sastrawi* dan *Porter*

1. Porter

Berikut ini akan dijelaskan tahapan – tahapan algoritma *Porter* untuk kalimat yang dijelaskan sebelumnya mengikuti aturan algoritma *porter* pada penjelasan bab 2.2.1. Berikut adalah tahapan – tahapan algoritma *Porter*

- Menghapus Partikel (“lah”, ”kah”, ”pun”)

Pada langkah ini kata “memiliki”, “keakurasian”, “dibandingkan” akan dilakukan proses hapus partikel dan hasilnya adalah seperti tabel 3.9

Tabel 3. 9 Tahapan Hapus Partikel Porter

No	Kata	Hasil Proses
1	memiliki	memiliki
2	keakurasian	keakurasian
3	dibandingkan	dibandingkan

Dikarenakan kata yang digunakan dalam proses *stemming* ini tidak memiliki kata yang ada dalam partikel maka hasil proses langkah ini akan mengembalikan kata awal sebagai hasil proses

- Menghapus kata ganti (*Possesive Pronoun*), seperti –ku, -mu, -nya
Tahapan berikut ini adalah menghapus kata ganti dari tahapan sebelumnya seperti tabel 3.10

Tabel 3. 10 Hasil Menghapus kata ganti Porter

No	Kata	Hasil Proses
1	memiliki	memiliki
2	keakurasian	keakurasian
3	dibandingkan	dibandingkan

- Menghapus awalan pertama. Jika tidak ditemukan, maka lanjut ke langkah 4a, dan jika ada maka lanjut ke langkah 4b.

Pada tahapan ini akan dihilangkan awalan pertama, dimana aturan untuk awalan pertama dijelaskan pada tabel 3.11 berikut ini.

Tabel 3. 11 Aturan awalan 1 Porter

No	Awalan	Kata Ganti	Catatan
1	“meng-”	“k”	Jika setelah awalan “meng” adalah huruf e dan u
2	“meny-”	“s”	-
3	“men-”	-	-
4	“mem-”	“p”	Jika setelah awalan “mem” adalah huruf a, i, u, e dan o
5	“me-”	-	-
6	“peng-”	“k”	Jika huruf setelah awalan “peng”

No	Awalan	Kata Ganti	Catatan
			adalah huruf e dan a
7	“peny-”	“s”	-
8	“pen-”	“t”	Jika setelah awalan “mem” adalah huruf a, i, u, e dan o
9	“pem-”	“p”	Jika setelah awalan “mem” adalah huruf a, i, u, e dan o
10	“di-”	-	-
11	“ter-”	-	-
12	“ke-”	-	-

Dari aturan awalan 1 yang dimiliki algoritma *porter* ini dihasilkan proses seperti tabel 3.12 berikut ini.

Tabel 3. 12 Hasil Menghilangkan Awalan 1 Porter

No	Kata	Hasil Proses
1	memiliki	piliki
2	keakurasian	akurasian
3	dibandingkan	bandingkan

- Menghapus akhiran, jika tidak ditemukan maka kata tersebut diasumsikan sebagai kata dasar (root word). Jika ditemukan maka lanjut ke langkah 5b.

Pada tahap ini masuk kedalam penghapusan akhiran dikarenakan awalan pertama pada studi kasus langkah sebelumnya ditemukan awalan 1 terhadap kata yang digunakan. Aturan yang digunakan pada penghapusan akhiran adalah menghapus kata akhiran “-kan”, “-i” dan “-an”. Maka hasil dari proses ini ditampilkan seperti pada tabel 3.13.

Tabel 3. 13 Hasil Menghapus Akhiran Porter

No	Kata	Hasil Proses
1	memiliki	pilik
2	keakurasian	akurasi

No	Kata	Hasil Proses
3	dibandingkan	banding

- Menghapus awalan kedua dan kata akhir diasumsikan sebagai kata dasar (root word). Dalam sebuah kata, memungkinkan adanya dua awalan yang saling berurutan serta cek kamus kata yang digunakan penelitian.

Pada tahapan ini dihilangkan awalan kedua dimana aturan dari penghilang awalan kedua adalah “ber-“, “bel-“, “be-“, “per-“, “pe-“, “pel-“ dan “se-“, sehingga menghasilkan kata seperti tabel 3.14.

Tabel 3. 14 Hasil Penghapusan Akhiran Porter

No	Kata	Hasil Proses
1	memiliki	pilik
2	keakurasian	akurasi
3	dibandingkan	banding

Tahapan ini merupakan tahapan akhir sehingga apapun hasil dari tahapan ini diasumsikan sebagai kata dasar dan diperiksa kembali kedalam kamus kata yang digunakan pada penelitian.

2. *Sastrawi*

Pada bagian ini akan dijelaskan tahapan untuk data studi kasus menggunakan algoritma *Sastrawi* seperti yang dijelaskan pada bab 2.2.2, tahapan tersebut adalah sebagai berikut.

- Melakukan pemeriksaan apakah kata yang akan di stemming ada dalam kamus kata dasar atau tidak. Jika ada, maka proses stemming berhenti pada langkah ini. Karena kata “memiliki”, “keakurasian” dan “dibandingkan” tidak ada dalam kamus besar bahasa indonesia sebagai kata dasar maka langkahnya masuk kedalam tahapan selanjutnya.
- Mehilangkan kata akhiran

Pada tahapan ini kata berimbuhan akan dilakukan penghilangan kata akhiran seperti “-lah”, “-kah”, “-ku”, “-mu”, “-nya”, “-tah” atau “-pun”, sehingga menghasilkan kata seperti tabel 3.15.

Tabel 3. 15 Tahapan Menghilangkan Kata Akhiran Sastrawi

No	Kata	Hasil Proses
1	memiliki	memiliki
2	keakurasian	keakurasian
3	dibandingkan	dibandingkan

- Menghilangkan kata imbuhan akhiran, kemudian hapus kata imbuhan awalan.

Pada tahapan ini akan dihilangkan kata imbuhan akhiran “-i”, “-kan”, “-an”. Kemudian akan dihilangkan kata imbuhan awalan seperti “be-”, “di-”, “ke-”, “me-”, “pe-”, “se-”, dan “te-”, sehingga menghasilkan hasil seperti tabel 3.16.

Tabel 3. 16 Hasil Menghilangkan Awalan dan Akhiran Sastrawi

No	Kata	Hasil Proses
1	memiliki	milik
2	keakurasian	akurasi
3	dibandingkan	banding

- Periksa kamus yang digunakan penelitian

Kata yang dihasilkan dari tahapan terakhir akan diperiksa kembali kedalam kamus, sehingga menghasilkan data seperti tabel 3.17 .

Tabel 3. 17 Hasil Akhir Tahapan Sastrawi

No	Kata	Hasil Proses	Status
1	memiliki	milik	ada
2	keakurasian	akurasi	ada
3	dibandingkan	banding	ada

3.5.5 Memasang *End Microtime*

Setelah proses stemming berhasil dilakukan pada bab 3.5.4 maka diasumsikan proses algoritma sudah dilakukan dengan baik oleh program,

sehingga pada tahapan ini diasumsikan waktu yang dilakukan oleh program dalam menjalankan program telah selesai. Maka fungsi *EndMicrotime* pada *PHP* akan dipasangkan ditahap ini agar perhitungan waktu bisa dilakukan dalam tahap selanjutnya .

3.5.6 Perhitungan Waktu

Pada tahapan ini diasumsikan fungsi *EndMicrotime* telah dipasangkan maka selanjutnya adalah menghitung jumlah waktu yang dilakukan, dimana prosesnya adalah dengan mengurangi hasil *EndMicrotime* dengan *StartMicrotime*, sehingga akan menghasilkan waktu dalam satuan detik.

Waktu dalam satuan detik tersebut diasumsikan sebagai hasil waktu eksekusi program yang dijalankan selama algoritma *stemming* tersebut dijalankan. Bisa disebut bahwa waktu yang keluar merupakan waktu yang dimiliki algoritma tersebut. Waktu tersebut ditampilkan didalam program sebagai jumlah waktu yang dimiliki algoritma *stemming*

3.5.7 Hasil Stemming

Dari tahapan bab 3.5.4 proses *stemming* yang dijelaskan pada blok diagram sebelumnya telah selesai dilakukan maka tahapan ini juga merupakan lanjutan dari tahapan bab 3.5.4. selanjutnya yang dilakukan pada program penelitian kali ini adalah menampilkan hasil dari setiap kata setelah dilakukan proses *stemming*.

1. Hasil *Stemming* Algoritma Porter

Berikut ini adalah hasil data kata yang telah diberikan proses *stemming* oleh algoritma *Porter* dijelaskan pada tabel 3.18.

Tabel 3. 18 Hasil Kata Stemming Porter

No	Kata	Hasil Proses	Status
1	memiliki	pilik	Gagal Stemming
2	keakurasian	akurasi	Berhasil Stemming
3	dibandingkan	banding	Berhasil Stemming

Kesalahan yang terjadi diakibatkan oleh kata ganti yang dibuat dalam aturan saat dihilangkan awalan 1 ditahapan algoritma *Porter*.

2. Hasil Stemming Algoritma Sastrawi

Berikut ini adalah hasil data kata yang telah diberikan proses *stemming* oleh algoritma *Sastrawi* dijelaskan pada tabel 3.19.

Tabel 3. 19 Hasil Kata Stemming Sastrawi

No	Kata	Hasil Proses	Status
1	memiliki	milik	Berhasil Stemming
2	keakurasian	akurasi	Berhasil Stemming
3	dibandingkan	banding	Berhasil Stemming

3.5.8 Indexing

Dari hasil stemming yang telah ditampilkan pada bab 3.5.7 lalu dimasukan kedalam *indexing* tujuannya adalah agar kata tersebut bisa dihitung sehingga nantinya hasil perhitungan tersebut dimasukan kedalam proses algoritma presisi seperti pada bab 2.5. contoh cara memasukan data kata kedalam *indexing* seperti pada tabel 3.20 untuk hasil *indexing* algoritma *Porter* dan tabel 3.21 untuk hasil *indexing* algoritma *Sastrawi*.

Tabel 3. 20 Hasil Indexing Algoritma *Porter*

No	Kata	Hasil Proses	Jumlah	Status
1	memiliki	pilik	1	Gagal Stemming
2	keakurasian	akurasi	1	Berhasil Stemming
3	dibandingkan	banding	1	Berhasil Stemming

Tabel 3. 21 Hasil Indexing Algoritma Sastrawi

No	Kata	Hasil Proses	Jumlah	Status
1	memiliki	milik	1	Berhasil Stemming
2	keakurasian	akurasi	1	Berhasil Stemming
3	dibandingkan	banding	1	Berhasil Stemming

3.5.9 Perhitungan Ketepatan

Setelah langkah *indexing* dilakukan maka selanjutnya seluruh hasil tersebut dapat dihitung dengan menggunakan algoritma presisi seperti bab 2.5

dimana seluruh hasil kata yang berhasil dilakukan proses stemming dibagi dengan seluruh jumlah kata dan dikalikan dengan nilai 100, sehingga hasil yang dikeluarkan oleh algoritma ini menjadi satuan persen. Berikut tabel 3.22 merupakan hasil perhitungan algoritma presisi terhadap data studi kasus dengan algoritma *Porter* dan 3.23 merupakan hasil perhitungan algoritma presisi terhadap data studi kasus dengan algoritma *Sastrawi*.

Tabel 3. 22 Perhitungan Ketepatan Studi Kasus Algoritma Porter

No	Kata	Hasil Proses	Jumlah	Status
1	memiliki	pilik	1	Gagal Stemming
2	keakurasian	akurasi	1	Berhasil Stemming
3	dibandingkan	banding	1	Berhasil Stemming
Jumlah benar				2
Jumlah Salah				1
Jumlah Seluruh Kata				3
Jumlah Presisi				66,67%

Tabel 3. 23 Perhitungan Ketepatan Studi Kasus Algoritma Sastrawi

No	Kata	Hasil Proses	Jumlah	Status
1	memiliki	milik	1	Berhasil Stemming
2	keakurasian	akurasi	1	Berhasil Stemming
3	dibandingkan	banding	1	Berhasil Stemming
Jumlah benar				3
Jumlah Salah				0
Jumlah Seluruh Kata				3
Jumlah Presisi				100%