

**DETEKSI TINGKAT KEMIRIPAN ABSTRAK TUGAS AKHIR  
MAHASISWA MENGGUNAKAN ALGORITME *WINNOWER*  
BERBASIS *N-GRAM* DAN *JACCARD SIMILARITY* PADA UNIVERSITAS  
BUDI LUHUR**

**TUGAS AKHIR**



**Oleh:**

**WAHYU DESENA**

**NIM : 1711502821**

**PROGRAM STUDI TEKNIK INFORMATIKA  
FAKULTAS TEKNOLOGI INFORMASI  
UNIVERSITAS BUDI LUHUR**

**JAKARTA**

**2021**

**DETEKSI TINGKAT KEMIRIPAN ABSTRAK TUGAS AKHIR  
MAHASISWA MENGGUNAKAN ALGORITME *WINNOWER*  
BERBASIS *N-GRAM* DAN *JACCARD SIMILARITY* PADA UNIVERSITAS  
BUDI LUHUR**

**Diajukan untuk memenuhi salah satu persyaratan  
memperoleh gelar Sarjana Komputer (S.Kom)**

**TUGAS AKHIR**



**Oleh:**

**WAHYU DESENA**

**NIM : 1711502821**

**PROGRAM STUDI TEKNIK INFORMATIKA  
FAKULTAS TEKNOLOGI INFORMASI  
UNIVERSITAS BUDI LUHUR**

**JAKARTA**

**2021**



**TEKNIK INFORMATIKA  
FAKULTAS TEKNOLOGI INFORMASI  
UNIVERSITAS BUDI LUHUR**

---

**PERSETUJUAN TUGAS AKHIR**

Nama : WAHYU DESENA  
Nomor Induk Mahasiswa : 1711502821  
Program Studi : Teknik Informatika  
Bidang Peminatan : *Programming Expert*  
Jenjang Studi : Strata 1  
Judul : Deteksi Tingkat Kemiripan Abstrak Tugas Akhir  
Mahasiswa Menggunakan Algoritme *Winnowing*  
berbasis *N-Gram* dan *Jaccard Similarity* pada  
Universitas Budi Luhur

Disetujui untuk dipertahankan dalam sidang Tugas Akhir periode semester Gasal  
tahun ajaran 2020/2021

Jakarta, 08 Februari 2021  
Dosen Pembimbing

( Dr. Achmad Solichin, S.Kom, M.T.I.)

## ABSTRAK

### DETEKSI TINGKAT KEMIRIPAN ABSTRAK TUGAS AKHIR MAHASISWA MENGGUNAKAN ALGORITME WINNOWER BERBASIS *N-GRAM* DAN *JACCARD SIMILARITY* PADA UNIVERSITAS BUDI LUHUR

Oleh : Wahyu Desena (1711502821)

Dokumen Skripsi merupakan dokumen yang merepresentasikan penelitian yang dilakukan oleh mahasiswa jenjang strata satu. Untuk menghasilkan skripsi yang berkualitas dibutuhkan penelitian yang kompeten, salah satu faktornya adalah orisinalitas, dalam arti tidak plagiat. Untuk mengantisipasi plagiarisme dibutuhkan suatu cara yang dapat mendeteksi tingkat *similarity* kata, diantaranya menggunakan algoritme *winnower*. Algoritme *winnower* digunakan untuk melakukan proses pengecekan kesamaan kata (*document fingerprinting*) untuk mengidentifikasi tingkat *similarity*. Algoritme ini dapat diproses dengan cara mencari nilai *hash* dari setiap *string* yang sudah dipotong pada tahapan *n-gram* berdasarkan nilai *k-gram* yang digunakan. Proses selanjutnya yaitu pembentukan *window* berdasarkan nilai *w-gram* yang digunakan. Setelah pembentukan *window*, kemudian dicari nilai *fingerprint* dari *window* tersebut (nilai terkecil dari *window* yang dibentuk). Setelah nilai *fingerprint* didapatkan, maka proses selanjutnya adalah mencari nilai *similarity* menggunakan metode *jaccard similarity*, metode ini digunakan untuk membandingkan dua sampel yaitu dokumen yang satu dengan yang lainnya. Hasil dari perhitungan *jaccard similarity* ditampilkan dalam bentuk persentase. Berdasarkan hasil pengujian, nilai *k-gram* dan *w-gram* mempengaruhi persentase *similarity*, penggunaan *k-gram* dan *w-gram* yang tepat sangat diperlukan. Nilai *k-gram* dan *w-gram* yang terlalu kecil mengakibatkan nilai persentase *similarity* yang besar, begitupun sebaliknya. Pada tahap pengujian dengan nilai *k-gram*=3 dan *w-gram*=4 menghasilkan nilai *similarity* terbesar yaitu dengan persentase 26.50%. Serta pengujian pada dokumen yang sama menghasilkan nilai *similarity* dengan persentase 100%, dengan demikian disimpulkan bahwa metode *jaccard similarity* dapat mendeteksi tingkat *similarity* dengan hasil yang cukup baik dan tepat. Diharapkan dengan adanya sistem pendeteksi *similarity* ini, dapat mengatasi tindakan plagiarisme pada dokumen abstrak skripsi mahasiswa Budi Luhur.

**Kata Kunci :** *Similarity, N-gram, Winnower, Jaccard Similarity, Abstrak*

Xiii + 72 halaman; 44 gambar; 19 tabel

## SURAT PERNYATAAN TIDAK PLAGIAT DAN PERSETUJUAN PUBLIKASI

Saya yang bertanda tangan di bawah ini:

Nama : Wahyu Desena  
NIM : 1711502821  
Program Studi : Teknik Informatika  
Bidang Peminatan : *Programming Expert*  
Jenjang Studi : Strata 1  
Fakultas : Teknologi Informasi

menyatakan bahwa TUGAS AKHIR yang berjudul:

DETEKSI TINGKAT KEMIRIPAN ABSTRAK TUGAS AKHIR  
MAHASISWA MENGGUNAKAN ALGORITME WINNOWER BERBASIS  
N-GRAM DAN JACCARD SIMILARITY PADA UNIVERSITAS  
BUDI LUHUR

Merupakan:

1. Karya tulis saya sebagai laporan tugas akhir yang asli dan belum pernah diajukan untuk mendapatkan gelar akademik apapun, baik di Universitas Budi Luhur maupun di perguruan tinggi lainnya.
2. Karya tulis ini bukan saduran / terjemahan, dan murni gagasan, rumusan dan pelaksanaan penelitian / implementasi saya sendiri, tanpa bantuan pihak lain, kecuali arahan pembimbing akademik dan pembimbing di organisasi tempat riset.
3. Dalam karya tulis ini tidak terdapat karya atau pendapat yang telah ditulis atau dipublikasikan orang lain, kecuali secara tertulis dengan dicantumkan sebagai acuan dalam naskah dengan disebutkan nama pengarang dan dicantumkan dalam daftar pustaka.
4. Saya menyerahkan hak milik atas karya tulis ini kepada Universitas Budi Luhur, dan oleh karenanya Universitas Budi Luhur berhak melakukan pengelolaan atas karya tulis ini sesuai dengan norma hukum dan etika yang berlaku.

Pernyataan ini saya buat dengan sesungguhnya dan apabila di kemudian hari terdapat penyimpangan dan ketidakbenaran dalam pernyataan ini, maka saya bersedia menerima sanksi akademik berupa pencabutan gelar yang telah diperoleh berdasarkan karya tulis ini, serta sanksi lainnya sesuai dengan norma di Universitas Budi Luhur dan Undang-Undang yang berlaku.

Jakarta, 08 Februari 2021



Wahyu Desena

KETERANGAN TIDAK PLAGIAT

## KATA PENGANTAR

Puji serta syukur Alhamdulillah, penulis panjatkan kehadiran Allah Azza Wa Jalla yang telah melimpahkan rahmat dan karunia-Nya, sehingga pada akhirnya penulis dapat menyelesaikan Tugas Akhir ini dengan baik. Adapun Tugas Akhir ini disusun untuk memenuhi persyaratan dalam menyelesaikan tingkat Pendidikan Strata 1 (S1) pada Program Studi Teknik Informatika, Fakultas Teknologi Informasi Universitas Budi Luhur dengan judul tugas akhir yang penulis ambil yaitu **“Deteksi Tingkat Kemiripan Abstrak Tugas Akhir Mahasiswa Menggunakan Algoritme Winnowing Berbasis N-Gram Dan Jaccard Similarity Pada Universitas Budi Luhur”**.

Penulis berharap laporan ini dapat memberikan manfaat kepada para pembaca umumnya dan kepada mahasiswa khususnya. Selain itu, diharapkan laporan ini juga dapat menjadi bahan perbandingan dalam melakukan karya penelitian selanjutnya. terselesaikannya penelitian ini tidak lepas dari bantuan berbagai pihak, rasa terimakasih yang mendalam ingin penulis sampaikan kepada mereka yang telah berjasa dalam membantu penyusunan tugas akhir ini, yaitu kepada:

1. Allah Azza Wa Jalla, atas segala petunjuk, kemudahan, serta nikmat-Nya yang diberikan sehingga penulis dapat menyelesaikan penyusunan tugas akhir ini dengan baik.
2. Segenap keluarga penulis, khususnya orang tua tercinta Ayah & Ibu, serta saudari khususnya Kakak, yang telah membantu memberikan dukungan baik moral maupun material, dan selalu memberikan doa restu.
3. Bapak Dr. Ir. Wendi Usino, M.Sc., M.M., selaku Rektor Universitas Budi Luhur.
4. Bapak Dr. Deni Mahdiana, M.M., M.Kom, selaku Dekan Fakultas Teknologi Informasi Universitas Budi Luhur.
5. Bapak Gunawan Pria Utama, M.Kom, selaku Dosen Penasehat Akademik.
6. Bapak Dr. Indra, S.Kom., M.T.I., selaku Kaprodi Teknik Informatika Fakultas Teknologi Informasi Universitas Budi Luhur.
7. Bapak Dr. Achmad Solichin, S.Kom, M.T.I., selaku Dosen Pembimbing Tugas Akhir yang telah membantu penulis dalam memberikan saran-saran dalam penyelesaian laporan tugas akhir ini.
8. Ibu Painem, M.Kom., selaku Kepala Laboratorium ICT Universitas Budi Luhur yang selalu memberikan arahan dan ilmu selama penulis mengabdikan di LAB ICT.
9. Teman-teman LDK Al-Azzam Universitas Budi Luhur khususnya periode 2020 yang telah memberikan dukungan dalam penyelesaian laporan tugas akhir ini.
10. Rekan-rekan Asisten Laboratorium ICT Terpadu Universitas Budi Luhur khususnya angkatan 2017, sebagai rekan kerja selama 3 tahun di LAB ICT.
11. Fatahillah Al Mahfudz, Dony Ramadhan Saputra, Mus Priandi, dan Gabriel Yoda Gustiegan sebagai sahabat maupun teman yang telah memberikan

bantuan, arahan, do'a, serta saran-saran dalam penyelesaian laporan tugas akhir ini.

Jakarta Selatan, 08 Februari 2021

Penulis

## DAFTAR TABEL

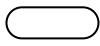

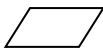
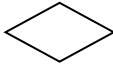
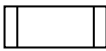



Tabel 2.4 Studi Literatur .....	10
Tabel 3.1 Sampel Data Abstrak .....	15
Tabel 4.1 Dokumen Abstrak .....	21
Tabel 4.2 Case Folding .....	23
Tabel 4.3 <i>Cleaning</i> .....	24
Tabel 4.4 Mengganti <i>slang word</i> .....	25
Tabel 4.5 Menghapus <i>stop word</i> .....	26
Tabel 4.6 Menghilangkan spasi antar karakter .....	26
Tabel 4.7 Pembentukan <i>N-Gram</i> .....	27
Tabel 4.8 Kode ASCII .....	28
Tabel 4.9 Perhitungan <i>Rolling Hash</i> .....	29
Tabel 4.10 Perhitungan <i>Hash Dataset</i> .....	29
Tabel 4.11 Perhitungan <i>Hash Data Tes</i> .....	31
Tabel 4.12 Hasil <i>Fingerprint</i> .....	33
Tabel 4.13 Uji Coba Nilai <i>k-gram</i> dan <i>w-gram</i> Sama .....	55
Tabel 4.14 Uji Coba Nilai <i>k-gram</i> dan <i>w-gram</i> Berbeda .....	55
Tabel 4.15 Uji Coba <i>Dataset</i> dijadikan Data Tes 1 .....	56
Tabel 4.16 Uji Coba <i>Dataset</i> dijadikan Data Tes 2 .....	57
Tabel 4.17 Uji Coba <i>Dataset</i> dijadikan Data Tes 3 .....	57



## DAFTAR GAMBAR

Gambar 2.1 Konsep Algoritme <i>Winnowing</i> (Sunardi et al., 2018).....	9
Gambar 3.1 Tahapan Metode.....	16
Gambar 4.1 <i>Flowchart</i> Keseluruhan Sistem.....	35
Gambar 4.2 <i>Flowchart</i> Data Dokumen.....	36
Gambar 4.3 <i>Flowchart Preprocessing Dataset</i> .....	37
Gambar 4.4 <i>Flowchart Preprocessing Data Tes</i> .....	38
Gambar 4.5 <i>Flowchart Case folding</i> .....	38
Gambar 4.6 <i>Flowchart Cleaning</i> .....	39
Gambar 4.7 <i>Flowchart Slang Word</i> .....	40
Gambar 4.8 <i>Flowchart Stop Word</i> .....	41
Gambar 4.9 <i>Flowchart Hapus Spasi</i> .....	41
Gambar 4.10 <i>Flowchart N-Gram</i> .....	42
Gambar 4.11 <i>Flowchart Algoritme Winnowing</i> .....	43
Gambar 4.12 <i>Flowchart Rolling Hash</i> .....	44
Gambar 4.13 <i>Flowchart Window</i> .....	45
Gambar 4.14 <i>Flowchart Fingerprint</i> .....	46
Gambar 4.15 <i>Flowchart Jaccard Similarity</i> .....	47
Gambar 4.16 Tampilan Layar <i>Home</i> .....	59
Gambar 4.17 Tampilan Layar <i>Document Data</i> .....	59
Gambar 4.18 Tampilan Layar <i>Document Data (Lanjutan)</i> .....	60
Gambar 4.19 Tampilan Layar <i>Preprocessing</i> .....	60
Gambar 4.20 Tampilan Layar Detail <i>Preprocessing</i> .....	61
Gambar 4.21 Tampilan Layar Data Awal.....	61
Gambar 4.22 Tampilan Layar <i>Case Folding</i> .....	62
Gambar 4.23 Tampilan Layar <i>Cleaning</i> .....	62
Gambar 4.24 Tampilan Layar <i>Slang Word</i> .....	62
Gambar 4.25 Tampilan Layar <i>Stop Word</i> .....	63
Gambar 4.26 Tampilan Layar Menghapus Spasi.....	63
Gambar 4.27 Tampilan Layar <i>Menu Slang word</i> .....	64
Gambar 4.28 Tampilan Layar <i>Menu Add Slang word</i> .....	64
Gambar 4.29 Tampilan Layar <i>Menu Edit Slang word</i> .....	65
Gambar 4.30 Tampilan Layar Menu Stop word .....	65
Gambar 4.32 Tampilan Layar <i>Menu Add Stop word</i> .....	66
Gambar 4.33 Tampilan Layar <i>Menu Edit Stop word</i> .....	66
Gambar 4.34 Tampilan Layar <i>Menu Scan Document</i> .....	67
Gambar 4.35 Tampilan Layar <i>Menu Scan Document (Lanjutan)</i> .....	67
Gambar 4.36 Tampilan Layar <i>Document Data Tes</i> .....	68
Gambar 4.37 Tampilan Layar Data Hasil .....	68
Gambar 4.38 Tampilan Layar <i>N-Gram</i> .....	68
Gambar 4.39 Tampilan Layar <i>Rolling Hash</i> .....	69
Gambar 4.40 Tampilan Layar <i>Window</i> .....	69
Gambar 4.41 Tampilan Layar <i>Fingerprint</i> .....	70
Gambar 4.42 Tampilan Layar Nilai <i>Similarity</i> .....	70

## DAFTAR SIMBOL

Simbol	Deskripsi
	<b>Terminator (Terminal)</b> Menggambarkan awal atau akhir sebuah aliran data.
	<b>Connector (Penghubung)</b> Menggambarkan arah proses..
	<b>Data (Input/ Output)</b> Menggambarkan masukan atau keluaran yang dihasilkan.
	<b>Decision (Kondisi)</b> Menggambarkan suatu kondisi yang harus dipilih oleh program.
	<b>Predefined Process</b> Menggambarkan proses – proses yang masih dapat dijabarkan dalam algoritme.
	<b>Process</b> Menggambarkan sebuah proses.
	<b>Off Page Reference</b> Menggambarkan penghubung dengan <i>flowchart</i> yang ada di halaman lain.
	<b>On Page Reference</b> Menggambarkan penghubung dalam satu halaman yang sama.

## DAFTAR ISI

<b>ABSTRAK .....</b>	<b>iv</b>
<b>KETERANGAN TIDAK PLAGIAT .....</b>	<b>v</b>
<b>KATA PENGANTAR.....</b>	<b>vi</b>
<b>DAFTAR TABEL .....</b>	<b>viii</b>
<b>DAFTAR GAMBAR.....</b>	<b>ix</b>
<b>DAFTAR SIMBOL.....</b>	<b>x</b>
<b>DAFTAR ISI .....</b>	<b>xi</b>
<b>BAB I PENDAHULUAN.....</b>	<b>1</b>
1.1. Latar Belakang .....	1
1.2. Perumusan Masalah.....	2
1.3. Batasan Masalah .....	2
1.4. Tujuan .....	3
1.5. Manfaat .....	3
1.6. Sistematika Penulisan .....	3
<b>BAB II LANDASAN TEORI.....</b>	<b>5</b>
2.1. <i>Data Mining</i> .....	5
2.2. <i>Text Mining</i> .....	5
2.3. Plagiarisme .....	5
2.4. <i>Preprocessing</i> .....	6
2.4.1. <i>Case folding</i> .....	6
2.4.2. Menghapus karakter kecuali a sampai z .....	6
2.4.3. Mengganti <i>slang word</i> .....	7
2.4.4. Menghapus <i>stop word</i> .....	7
2.5. <i>N-Gram</i> .....	7
2.6. <i>Jaccard Similarity</i> .....	7
2.7. <i>Algoritme Winnowing</i> .....	8
2.8. Studi Literatur .....	10
<b>BAB III METODOLOGI PENELITIAN .....</b>	<b>15</b>
3.1. Data Penelitian .....	15
3.2. Penerapan Metode .....	16
3.2.1. Data Dokumen .....	17
3.2.2. <i>Preprocessing</i> .....	17
3.2.3. <i>N-Gram</i> .....	18

3.2.4. Algoritme <i>Winnowing</i> .....	18
<b>3.3. Rancangan Pengujian.....</b>	<b>19</b>
<b>BAB IV HASIL DAN PEMBAHASAN .....</b>	<b>20</b>
<b>4.1. Lingkungan Percobaan .....</b>	<b>20</b>
4.1.1. Spesifikasi Perangkat Keras.....	20
4.1.2. Spesifikasi Perangkat Lunak.....	20
<b>4.2. Implementasi Metode dan Langkah Pengujian .....</b>	<b>20</b>
4.2.1. Data Dokumen .....	20
4.2.2. Tahapan <i>Preprocessing</i> .....	22
4.2.3. Pembentukan <i>N-Gram</i> .....	27
4.2.4. Algoritme <i>Winnowing</i> .....	27
4.2.5. <i>Jaccard Similarity</i> .....	33
<b>4.3. Flowchart Tahapan Metode .....</b>	<b>34</b>
4.3.1. <i>Flowchart</i> Keseluruhan Sistem.....	34
4.3.2. <i>Flowchart</i> Data Dokumen .....	36
4.3.3. <i>Flowchart Preprocessing Dataset</i> .....	36
4.3.4. <i>Flowchart Preprocessing Data Tes</i> .....	37
4.3.5. <i>Flowchart Case folding</i> .....	38
4.3.6. <i>Flowchart Cleaning</i> .....	39
4.3.7. <i>Flowchart Slang word</i> .....	39
4.3.8. <i>Flowchart Stop word</i> .....	40
4.3.9. <i>Flowchart Hapus Spasi</i> .....	41
4.3.10. <i>Flowchart N-Gram</i> .....	42
4.3.11. <i>Flowchart Algoritme Winnowing</i> .....	42
4.3.12. <i>Flowchart Rolling Hash</i> .....	43
4.3.13. <i>Flowchart Window</i> .....	44
4.3.14. <i>Flowchart Fingerprint</i> .....	45
4.3.15. <i>Flowchart Jaccard Similarity</i> .....	46
<b>4.4. Algoritme Tahapan Metode .....</b>	<b>48</b>
4.4.1. Algoritme Keseluruhan Sistem.....	48
4.4.2. Algoritme Data Dokumen.....	48
4.4.3. Algoritme <i>Preprocessing Dataset</i> .....	48
4.4.4. Algoritme <i>Preprocessing Data Tes</i> .....	49
4.4.5. Algoritme <i>Case folding</i> .....	49

4.4.6. Algoritme <i>Cleaning</i> .....	49
4.4.7. Algoritme <i>Slang word</i> .....	50
4.4.8. Algoritme <i>Stop word</i> .....	50
4.4.9. Algoritme Hapus Spasi .....	51
4.4.10. Algoritme <i>N-Gram</i> .....	51
4.4.11. Algoritme <i>Winnowing</i> .....	51
4.4.12. Algoritme <i>Rolling Hash</i> .....	52
4.4.13. Algoritme <i>Window</i> .....	52
4.4.14. Algoritme <i>Fingerprint</i> .....	53
4.4.15. Algoritme <i>Jaccard Similarity</i> .....	54
<b>4.5. Pengujian .....</b>	<b>54</b>
4.5.1. Contoh Pengujian.....	54
4.5.2. Penjelasan Hasil Pengujian .....	58
<b>4.6. Tampilan Layar Aplikasi .....</b>	<b>58</b>
4.6.1. Tampilan Layar <i>Home</i> .....	58
4.6.2. Tampilan Layar <i>Document Data</i> .....	59
4.6.3. Tampilan Layar <i>Document Dataset</i> .....	60
4.6.4. Tampilan Layar <i>Slang word</i> .....	63
4.6.5. Tampilan Layar <i>Stop word</i> .....	65
4.6.6. Tampilan Layar <i>Similarity Check</i> .....	66
<b>BAB V PENUTUP .....</b>	<b>71</b>
<b>5.1. Kesimpulan.....</b>	<b>71</b>
<b>5.2. Saran .....</b>	<b>71</b>
<b>DAFTAR PUSTAKA.....</b>	<b>72</b>

# BAB I

## PENDAHULUAN

### 1.1. Latar Belakang

Plagiarisme merupakan tindakan kejahatan yang sudah tidak asing lagi dalam dunia akademik, khususnya di Indonesia. Plagiarisme merupakan perbuatan yang tidak baik disebabkan mengambil karya seperti ide, gagasan, dan pikiran orang lain tanpa izin, tanpa menyatakan karya yang diambilnya merupakan karya orang lain atau tidak mencantumkan sumbernya, dan mengakui bahwa karya itu miliknya. Tindakan tersebut sama seperti tindakan mencuri (Sunardi, Yudhana, & Mukaromah, 2018). Tindakan plagiarisme ini semakin mudah dengan semakin pesatnya perkembangan dunia teknologi, serta semakin mudahnya mendapatkan suatu informasi. Hal ini menyebabkan banyak orang-orang yang mengambil karya orang lain dari internet, tanpa mencantumkan sumbernya.

Dalam dunia pendidikan, khususnya tingkat perguruan tinggi, melakukan tindakan plagiat bukanlah suatu hal yang baru. Namun sudah terjadi semenjak teknologi belum secanggih ini. Pelajar maupun mahasiswa ketika mengerjakan tugas-tugas sekolah maupun kuliah sangat jarang mencantumkan sumbernya atau referensi yang digunakan dalam pengerjaan tugas, yang menyebabkan banyaknya kesamaan tugas siswa maupun mahasiswa satu sama lain. Pelajar maupun mahasiswa banyak melakukan manipulasi pembuatan tugas seperti karya ilmiah, makalah, kuliah kerja praktek, maupun skripsi yang menyebabkan kreativitas seseorang tidak tersalurkan dengan baik.

Semakin maraknya tindakan plagiarisme, salah satunya pembuatan skripsi di kampus Budi Luhur, sehingga untuk mengatasi tindakan plagiarisme ini perlu dilakukan pencegahan awal untuk mengurangi praktik plagiarisme. Pendeteksian dapat dilakukan secara manual, namun tidak efektif sebab dibutuhkan waktu yang cukup lama, karena banyaknya dokumen skripsi yang perlu diperiksa nilai *similarity*nya dengan dokumen skripsi yang sudah dipublikasi ke perpustakaan Budi Luhur. Cara menangani hal tersebut yaitu dengan membuat sebuah sistem yang dapat mendeteksi nilai *similarity* dari dokumen skripsi mahasiswa Budi Luhur. Sistem ini di fokuskan untuk pendeteksian awal, sehingga dokumen yang diperiksa dari sistem ini adalah dokumen abstrak, di mana data dokumen abstrak dari skripsi mahasiswa Budi Luhur yang sudah dipublikasi ke perpustakaan dijadikan sebagai *dataset* dari penelitian ini.

Beberapa penelitian sebelumnya telah membuat pendeteksian *similarity* menggunakan beberapa algoritme yang mempunyai fungsi sama sebagai dokumen *fingerprint*, tetapi permasalahan yang sering terjadi pada hasil atau tingkat keakuratan (Sunardi et al., 2018). Penggunaan model *Bayesian* juga telah digunakan, namun tidak sepenuhnya dapat mendeteksi *similarity*. Pada penelitian yang dilakukan oleh (Rahmatulloh, Kurniati, Darmawan, Asyikin, & Witarsyah, 2019) dalam pendeteksian *similarity* dokumen berbahasa

indonesia, penelitian ini menggunakan kinerja algoritme *fingerprint* dan *winnowing*, hasil tertinggi pada analisis tes akurasi algoritme *fingerprint* sebesar 92,8% dengan nilai *threshold* 0,1 dan nilai *n-gram* berada di 3, algoritme *winnowing* sebesar 91,8%. hasil yang berbeda ditunjukkan pada tingkat relevansi akurasi dengan topik, hasil akurasi dari algoritme *winnowing* mendapatkan korelasi yang lebih kuat 37,1% dibandingkan *fingerprint* 33,6%.

Pada penelitian ini sistem yang akan dibuat menggunakan metode *N-gram*, *Jaccard Similarity*, dan algoritme *Winnowing*. *N-gram* merupakan algoritme yang akan digunakan untuk mengambil potongan huruf sejumlah *n* dan mempunyai pengaruh yang tinggi terhadap hasil *similarity*. Algoritme yang akan digunakan untuk mengetahui tingkat kemiripan teks atau dokumen yaitu *Jaccard Similarity* atau sering disebut *Jaccard Coefficient*. *Jaccard Similarity* merupakan algoritme yang berfungsi untuk membandingkan antar dua dokumen dengan menghitung *similarity* atau perbedaan dari dokumen tersebut (Sunardi et al., 2018). Untuk menjalankan *n-gram* dan *jaccard similarity* diperlukan algoritme yang berfungsi sebagai dokumen *fingerprint* dan algoritme yang akan digunakan untuk mendukung hal tersebut yaitu algoritme *winnowing*. Algoritme *winnowing* merupakan algoritme yang dapat digunakan untuk membantu mendeteksi kemiripan kata atau dokumen yang akan digunakan sebagai suatu cara untuk mendeteksi adanya suatu tindak plagiarisme. Diharapkan dengan adanya sistem pendeteksi *similarity* menggunakan metode *n-gram* dan *Jaccard similarity* terhadap algoritme *winnowing* dapat memberikan keakuratan yang tepat dalam mengatasi tindakan suatu plagiarisme yang terjadi di dunia akademik, tepatnya di kampus Budi Luhur.

## 1.2. Perumusan Masalah

Berdasarkan latar belakang yang telah disebutkan di atas maka didapatkan rumusan masalah yaitu :

- a. Bagaimana menerapkan metode *N-gram*, *Jaccard Similarity*, dan algoritme *Winnowing* untuk pemeriksaan kesamaan abstrak pada dokumen skripsi mahasiswa Budi Luhur?
- b. Bagaimana mengukur tingkat persentase kesamaan dalam penerapan metode *N-gram*, *Jaccard Similarity*, dan algoritme *Winnowing*?

## 1.3. Batasan Masalah

Adapun batasan atau ruang lingkup masalah antara lain sebagai berikut :

- a. *Platform* yang digunakan hanya berbasis *web*.
- b. Deteksi *similarity* hanya dilakukan pada dokumen abstrak.
- c. Fitur *import* hanya dapat mengenali file masukan berupa *.pdf*.
- d. Hanya dapat mengenali tulisan di dalam file masukan.
- e. Pemeriksaan dokumen hanya dilakukan untuk mendapatkan persentase *similarity* dari dokumen yang diuji, tidak untuk mendapatkan *output* berupa keterangan plagiat atau tidak plagiat.

#### 1.4. Tujuan

Tujuan yang ingin dicapai dalam penelitian ini adalah sebagai berikut :

- a. Menerapkan metode *n-gram* untuk mengambil potongan karakter huruf pada dokumen yang akan dicocokkan.
- b. Menerapkan metode *jaccard similarity* untuk menghitung berapa persen kesamaan pada sebuah dokumen yang diproses menggunakan algoritme *winnowing*.
- c. Mengimplementasikan rancangan model yang dibuat ke dalam aplikasi berbasis web.

#### 1.5. Manfaat

Adapun manfaat dari penelitian ini adalah sebagai berikut :

- a. Manfaat bagi penulis :
  - Dapat membantu institusi ataupun dunia ilmu pengetahuan dalam mencegah tindakan plagiarisme.
  - Dapat mengetahui apakah penelitian yang akan dibuat sudah pernah diteliti sebelumnya.
  - Dapat mengetahui nilai *similarity* dari sebuah dokumen abstrak mahasiswa Budi Luhur.
- b. Manfaat bagi institusi universitas Budi Luhur :
  - Pendeteksi awal tindakan plagiarisme pada penelitian mahasiswa.
  - Dapat meningkatkan kualitas penelitian mahasiswa.
  - Menghindari sanksi dari pihak dikti, karena penelitian yang mengandung unsur plagiat.
  - Mendukung program pemerintah untuk mencegah plagiarisme dalam dunia pendidikan.
- c. Manfaat bagi dunia ilmu pengetahuan :
  - Memperkecil terjadinya tindakan plagiat di dunia pendidikan.
  - Dapat menanggulangi tindakan pencurian karya orang lain.
  - Memberikan pengetahuan tentang perancangan sistem pendeteksi tingkat *similarity* pada sebuah dokumen abstrak.

#### 1.6. Sistematika Penulisan

Sistematika penulisan penelitian ini disusun untuk memberikan gambaran umum tentang penelitian yang dijalankan. Sistematika penulisan tugas akhir ini adalah sebagai berikut:

##### **BAB I : PENDAHULUAN**

Bagian ini berisi tentang latar belakang, rumusan masalah, batasan masalah, manfaat dan tujuan penelitian, dan juga membahas mengenai sistematika penulisan.

##### **BAB II : LANDASAN TEORI**

Bagian ini berisi tentang algoritme dan metode yang akan dibahas, serta teori-teori yang berkaitan dengan penelitian ini, yaitu pengertian dan pemahaman plagiarisme, *text mining*, data



*mining, preprocessing, n-gram, Jaccard Similarity, Algoritme Winnowing*, serta studi literatur.

### **BAB III : METODOLOGI PENELITIAN**

Bagian ini berisi tentang sumber data penelitian, penerapan atau tahapan metode yang digunakan. Bab ini juga berisi tentang rancangan pengujian dari ekstraksi informasi yang didapat.

### **BAB IV : HASIL DAN PEMBAHASAN**

Bagian ini berisi mengenai lingkungan percobaan sistem yang dibuat, implementasi metode, *flowchart* tahapan metode, dan uraian algoritme pada proses, serta analisa pengujian sistem yang telah dibangun apakah data hasil pengelompokan yang didapat sudah sesuai dan relevan.

### **BAB V : PENUTUP**

Bagian ini berisi tentang kesimpulan yang dapat ditarik dari penelitian dan saran untuk pengembangan lebih lanjut mengenai topik terkait dalam penelitian berikutnya.

## **BAB II**

### **LANDASAN TEORI**

#### **2.1. Data Mining**

Data *mining* merupakan serangkaian proses untuk menggali informasi dengan melakukan analisa data untuk menemukan suatu pola dari kumpulan data tersebut. Data mining mampu menganalisa data yang besar menjadi ekstraksi berupa pola yang mempunyai arti bagi pendukung keputusan (Gunadi & Sensuse, 2012). Data mining juga bisa disebut knowledge discovery adalah proses pengambilan pola pada data yang akan diproses lalu output tersebut berupa informasi yang sangat penting. Proses yang dilakukan untuk mengekstrak pengetahuan dalam data mining adalah pengenalan pola, clustering, asosiasi, prediksi dan klasifikasi (Fitri, Nurjanah, & Astuti, 2018). Data *mining* memiliki variasi untuk menemukan pola dari ekstraksi sebuah kumpulan sekumpulan data tekstual yang disebut dengan *text mining*. *Text mining* memiliki fokus pada pengolahan data berupa kata atau teks.

#### **2.2. Text Mining**

Menurut Hearst *text mining* diartikan sebagai penemuan informasi yang baru dan tidak diketahui sebelumnya oleh komputer, dengan secara otomatis mengekstrak informasi dari sumber-sumber yang berbeda (Februariyanti, 2012). Kunci dari proses ini adalah menggabungkan informasi yang berhasil diekstraksi dari berbagai sumber. Sedangkan menurut Harlian *text mining* memiliki definisi menambang data yang berupa teks dimana sumber data biasanya didapatkan dari dokumen, dan tujuannya adalah mencari kata-kata yang dapat mewakili isi dari dokumen sehingga dapat dilakukan analisa keterhubungan antar dokumen.

*Text mining* merupakan bagian dari *data mining*, yang mana digunakan untuk mendapatkan informasi dari sebuah data atau dokumen berupa sekumpulan teks yang memiliki format yang terstruktur maupun tidak terstruktur dengan jumlah yang besar. Dalam *text mining* memiliki tugas khusus yaitu klasifikasi dan klasterisasi. Sedangkan dalam penerapannya, *text mining* berfungsi untuk mencari pola dalam teks, menganalisis teks agar bisa menghasilkan keluaran berupa informasi yang bermanfaat pada tujuan tertentu. Dikarenakan data yang diproses pada *text mining* merupakan sebuah teks yang tidak terstruktur, maka diperlukan pemilihan teks sebelum dilakukan proses selanjutnya, pada tahap ini dikenal dengan prapemrosesan (*preprocessing*).

#### **2.3. Plagiarisme**

Plagiarisme atau plagiat adalah penjiplakan atau pengambilan karangan, pendapat orang lain dan menjadikannya seolah-olah karangan sendiri (Sunyoto & Informatika, 2013). Pendekatan deteksi plagiat terbagi menjadi *intrinsic* dan *external*. Pendekatan *external* terbagi lagi menjadi tiga, yaitu perbandingan teks lengkap, kesamaan kata kunci dan *fingerprinting*. Perbandingan teks lengkap diterapkan untuk membandingkan semua isi dokumen, kesamaan kata kunci bekerja dengan cara mengekstrak dan membandingkan kata kunci antara dokumen, dan *fingerprinting* untuk mendeteksi kemiripan antar dokumen dengan prinsip *hashing*.

Berdasarkan penelitian ini mengkategorikan praktek plagiat berdasarkan cara yang digunakan, diantaranya :

- *Copy Paste plagiarism*, menyalin setiap kata tanpa perubahan.
- *Disguised plagiarism*, tergolong ke dalam praktek menutupi bagian yang disalin, teridentifikasi ke dalam empat teknik, yaitu *shake paste*, *expansive plagiarism*, *contractive plagiarism*, dan *mosaic plagiarism*.
- *Technical disguise*, teknik meringkas untuk menyembunyikan konten plagiat dari deteksi otomatis dengan memanfaatkan kelemahan dari metode analisis teks dasar, misal dengan mengganti huruf dengan simbol huruf asing.
- *Undue paraphrasing*, sengaja menuliskan ulang pemikiran asing dengan pemilihan kata dan gaya plagiator dengan menyembunyikan sumber asli.
- *Translated plagiarism*, mengkonversi konten dari satu bahasa ke bahasa lain.
- *Idea plagiarism*, menggunakan ide asing tanpa menyatakan sumber.

*Self plagiarism*, penggunaan sebagian atau keseluruhan tulisan pribadi yang tidak dibenarkan secara ilmiah.

## **2.4. Preprocessing**

Tahapan *preprocessing* atau praproses merupakan bagian yang sangat penting dalam menyiapkan data, hal ini dikarenakan struktur data yang dihasilkan pada tahap pengumpulan tidak beraturan, sehingga menyebabkan proses menjadi tidak berjalan dengan baik.

Merujuk pada penelitian sebelumnya yang dilakukan oleh Himalatha dalam jurnal (Filcha & Hayaty, 2019) maka pada penelitian ini akan dibahas beberapa tahapan *preprocessing* teks antara lain, *case folding*, menghapus karakter selain a-z, menghapus teks dengan 1 karakter, mengganti *slang word*, menghapus *stop word*.

### **2.4.1. Case folding**

Pada proses ini bertujuan untuk mengubah semua karakter huruf menjadi huruf kecil (*lowercase*), hal ini dilakukan untuk menyamakan arti dari suatu kata yang sama, apabila penulisan besar kecilnya huruf tidak sama.

### **2.4.2. Menghapus karakter kecuali a sampai z**

Pada proses ini dilakukan penghapusan untuk seluruh karakter berupa simbol dan angka, atau menyisakan hanya karakter angka, termasuk menghapus hashtag (#) dan mention (@), hal ini dilakukan karena simbol dan angka dianggap tidak terlalu penting, tetapi jika ini diperlukan, maka proses ini dihilangkan.

#### 2.4.3. Mengganti *slang word*

Teks yang tidak terstruktur membuat sebuah teks terkadang tidak sesuai dengan ejaan bahasa Indonesia yang baku (EYD) pada konteks ini, kata yang tidak baku disebut dengan *slang word*, untuk mendapatkan informasi dari teks agar maksimal, kata-kata tidak baku, baik kata gaul, singkatan atau yang lain sebanyak mungkin ditampung ke dalam kamus *slang word*, untuk kemudian dilakukan *replace* supaya menjadi kata dengan bahasa Indonesia yang baku sesuai EYD.

#### 2.4.4. Menghapus *stop word*

*Stop word* merupakan salah satu kata yang diabaikan dalam pemrosesan, Ringkasnya *stop word* adalah kata hubung atau kata sambung dalam sebuah kalimat, seperti “di”, “pada”, “karena”, “sebuah”, “oleh”, dll. Sebelum melakukan proses penghapusan *stop word*, kumpulkan daftar atau kamus *stop word* yang diberi nama *stoplist*. Kemudian lakukan perbandingan antara sebuah teks dengan *stoplist*. Jika terdapat kata-kata yang terdapat dalam *stoplist*, maka kata tersebut dihilangkan. Untuk *stoplist* dalam bahasa Indonesia, datanya bersumber dari (Tala, 2003).

### 2.5. *N-Gram*

Dalam jurnal (Sunardi et al., 2018) menjelaskan bahwa, *n-gram* merupakan metode yang dilakukan dengan mengambil rangkaian *substring* dari *string* sejumlah *n* (rangkai token sepanjang *n*). Metode *n-gram* sering digunakan pada teknik analisis statistik dan juga bahasa. *N-gram* paling banyak digunakan dalam teks mining (pengolahan kata) dan pengolahan bahasa. Dalam mendeteksi plagiarisme, *n-gram* digunakan untuk mengambil potongan-potongan karakter huruf atau pemisahan *string* sepanjang *n* dari sebuah kata atau dokumen secara berkelanjutan (kontinuitas) hingga bergeser sesuai dengan *offset* yang diberikan atau akhir dari suatu kata atau dokumen.

*N-gram* dalam deteksi plagiarisme sangat mempengaruhi tingkat akurasi atau tingkat *similarity* dari sebuah dokumen yang dibandingkan, maka teknik *n-gram* dipadukan dengan pendekatan statistika untuk memperoleh *similarity* dari antar dokumen seperti *Simple Matching*, *Cosine Similarity*, *Jaccard Similarity* dan *Dice Coefficient*. Ada 2 (dua) teknik *n-gram* yaitu membagi *string* menjadi suatu set *substring* dengan panjang *n* (*overlapping n-gram*) dan mengecek untuk membentuk *substring* yang memiliki struktur yang sama.

### 2.6. *Jaccard Similarity*

Dalam jurnal (Sunardi et al., 2018) menjelaskan bahwa, *Jaccard Similarity* dan *Jaccard Coefficient* merupakan algoritme 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. *Jaccard similarity* dapat dirumuskan sebagai berikut:

$$\text{Similarity (X,Y)} = \frac{|x \cap y|}{|x \cup y|} \dots (2.1)$$

Dimana:

$X = \text{Dataset}$

$Y = \text{Data tes}$

Rumus 1 merupakan dari rumus *Jaccard Similarity* atau *Jaccard Coefficient* yang digunakan untuk mencari persamaan dan perbedaan pada dua sampel. sebagai contoh diketahui X “Magister Teknik Informatika Yogyakarta”, dan Y “Magister Teknik Sipil Yogyakarta”. Maka akan menghasilkan nilainya:

$$\text{Similarity (X,Y)} = \frac{|\text{Magister Teknik Yogyakarta}|}{|\text{Magister Teknik Informatika Sipil Yogyakarta}|} \dots (2.2)$$

$$\text{Similarity (X,Y)} = \frac{|3|}{|5|} = 0,6 \times 100\% = 60\%$$

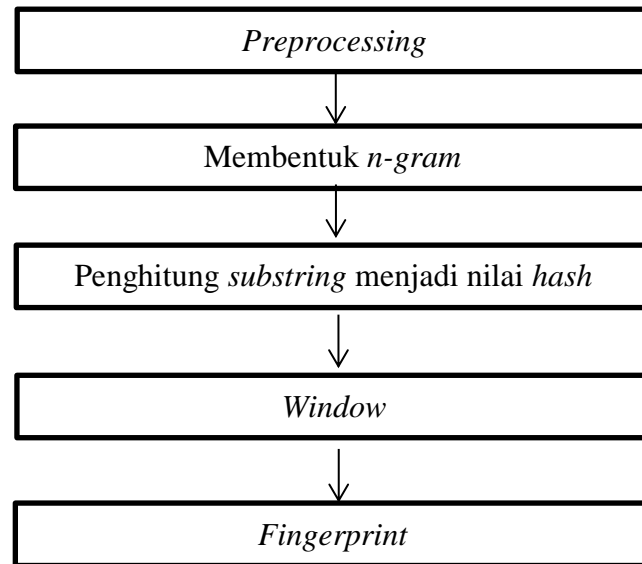
Dari dua contoh sampel di atas setelah dihitung kesamaannya menggunakan *Jaccard similarity*, bahwa kedua sampel tersebut memiliki kesamaan atau kemiripan sebesar 60%.

Ukuran tingkat kesamaan:

- a. 0% : dua dokumen tidak memiliki kesamaan
- b. <15% : memiliki sedikit kesamaan
- c. 15-50% : termasuk dalam kategori plagiarisme sedang
- d. 50% : mendeteksi adanya plagiarisme
- e. 100% : dokumen tersebut plagiarism

## 2.7. Algoritme *Winnowing*

Dalam jurnal (Sunardi et al., 2018) menjelaskan bahwa, Algoritme *winnowing* merupakan salah satu algoritme yang berfungsi sebagai dokumen *fingerprint* atau algoritme yang digunakan untuk mendeteksi tindakan plagiarisme dengan menggunakan teknik *hashing*. Input dari algoritme *winnowing* berupa dokumen teks, dan akan menghasilkan keluaran berupa kumpulan nilai hash yang terbentuk dari perhitungan ASCII pada setiap karakter. Dan nilai-nilai *hash* yang akan digunakan sebagai *fingerprint* untuk mendeteksi adanya suatu tindakan plagiarisme. Gambar 2.1 merupakan konsep algoritme *winnowing*.



**Gambar 2.1 Konsep Algoritme Winnowing**(Sunardi et al., 2018)

Konsep dari algoritme *winnowing* yang ditunjukkan pada gambar 2.1 yaitu menghapus karakter yang tidak relevan, membentuk *n-gram* dengan panjang *n*, menghitung nilai *hash*, membentuk nilai *window*, dan memilih nilai *hash* sebagai dokumen *fingerprint*. Nilai *hash* didapatkan dari rumus *Rolling hash*. *Rolling hash* adalah metode *hashing* yang digunakan untuk mencari nilai *hash* dari rangkaian *grams* yang telah terbentuk dan memberikan kemampuan untuk menghitung nilai tanpa mengulangi seluruh *string*. Nilai *hash* merupakan nilai numerik yang dibentuk dari kode ASCII[15]. Berikut rumus *rolling hash*

$$H(C1..C1) = C1.b^{(1-1)} + C2.b^{(1-2)} + \dots + C(1-1).b + C1...(2.3)$$

$$H(C2 \dots C1 + 1) = (H(C1 \dots C1) - C1*b^{(1-1)}) * b + c^{(1-1)} \dots(2.4)$$

Dimana:

- H (C1..C1) = nilai hash
- C1 = nilai ASCII karakter ke -1 pada string
- l = panjang string
- b = nilai basis hash

Perhitungan awal pada rangkaian *n-gram* paling awal dihitung menggunakan rumus nomor 2, dan rangkaian *n-gram* berikutnya sampai rangkaian *gram* terakhir dihitung menggunakan rumus nomor 3, sehingga proses akan jauh lebih cepat karena tidak menghitung lagi dari awal. Algoritme *winnowing* merupakan ekstensi dari algoritme *Rabin-karp*, proses yang dilakukan hampir sama hanya pada algoritme *winnowing* ditambahkan dengan konsep *window*.

## 2.8. Studi Literatur

Berdasarkan landasan teori yang telah dijelaskan, terdapat penelitian yang sudah ada sebelumnya, yang dirangkum dalam Tabel 2.4 berikut :

**Tabel 2.4 Studi Literatur**

No	Judul	Tujuan Penelitian	Metode	Hasil Penelitian
1	Rancang Bangun Aplikasi Pengecekan Kemiripan Judul Skripsi Dengan Metode <i>Cosine Similarity</i> (Abdullah & Aribowo, 2018)	Membantu mahasiswa atau dosen untuk melakukan pengecekan judul skripsi terhadap skripsi – skripsi sebelumnya, saat pengajuan judul skripsi, guna menghindari kesamaan skripsi	Metode Cosine Similarity	Sistem diuji dengan 2 metode yaitu <i>Black box Test</i> dan <i>Usability Test</i> . Pengujian dengan metode <i>Black box Test</i> dihasilkan bahwa 100% sistem sudah berjalan dengan baik, layak digunakan dan sesuai kebutuhan. Sedangkan pengujian dengan metode <i>Usability Test</i> 0 % responden menyatakan aplikasi tidak diterima, 20 % responden memberikan penilaian marginal dan 80 % responden menyatakan sistem aplikasi bisa diterima.
2	Rancang bangun aplikasi pendeteksi kesamaan pada dokumen teks menggunakan algoritma <i>Enhanced confix stripping</i> dan algoritma <i>winnowing</i> (Khidfi & Sari, 2018)	Membuat sistem untuk menentukan nilai similaritas antara 2 dokumen teks sehingga dosen dapat melihat tingkat similaritas antara tugas mahasiswa dan dapat mengurangi tingkat plagiarisme yang terjadi antara mahasiswa.	Algoritma Enhanced Confix Stripping Stemmer dan Algoritma Winnowing	Untuk mendapatkan hasil pada penelitian ini, peneliti melakukan pengujian dengan menentukan nilai <i>gram</i> dan <i>window</i> pada perhitungan algoritma <i>Winnowing</i> , tujuannya memudahkan user dalam menentukan nilai <i>similarity</i> yang akurat. Dari hasil pengujian 5 pasang bab 1 tugas akhir mahasiswa yang berkategori sama menghasilkan nilai <i>similarity</i> sekitar 45-20% .

No	Judul	Tujuan Penelitian	Metode	Hasil Penelitian
3	Pengukuran Kemiripan Dokumen Teks Bahasa Indonesia Menggunakan Metode <i>Cosine Similarity</i> (Ariantini, Lumenta, & Jacobus, 2016)	Membuat sistem pengecekan kemiripan dokumen teks berbahasa Indonesia	Cosine similarity	Untuk mendapatkan hasil pada penelitian ini dilakukan pengujian menggunakan data dummy. Data dummy adalah data yang digunakan untuk membandingkan dokumen 1 dengan dokumen 2. Nilai actual dari dokumen satu menjadi acuan nilai actual diperoleh dari memeriksa tugas secara manual kemudian dibandingkan dengan dokumen 2 prediksi dari sistem dengan nilai 100% karena di dalam dokumen tersebut memang sama persis. Kemudian dokumen satu dengan dokumen tiga nilai actualnya 66% dan nilai prediksi sebagian isi dokumen diambil dari dokumen satu sehingga nilai yang diperoleh adalah 75%. Nilai kesalahan yang diperoleh rata-rata yaitu 7% sehingga masih banyak kesamaan kata atau kesalahan, dengan demikian sistem yang dibangun sudah bisa digunakan untuk mendeteksi kemiripan dokumen.
4	Penerapan Metode <i>Cosine Similarity</i> Untuk Pengecekan Kemiripan Jawaban Ujian Siswa (Fataruba, 2018)	Untuk membantu pengajar memberikan nilai yang objektif dengan menggunakan metode cosine similarity pada sistem agar dapat melakukan penilaian jawaban essay dengan membandingkan kunci jawaban pengajar dengan jawaban peserta didik. Dan juga dapat membandingkan tingkat kemiripan antara siswa satu dengan siswa yang lain.	Cosine similarity	similarity telah berjalan dengan baik untuk ujian essay Biologi Hasil uji coba menunjukkan kesesuaian nilai sistem dengan nilai yang diberikan oleh pengajar tingkat akurasi 80% menggunakan Confusion Matrix.



No	Judul	Tujuan Penelitian	Metode	Hasil Penelitian
5	Pendeteksian tingkat similaritas dokumen berbasis web menggunakan algoritma <i>winnowing</i> (Ulfa, Mustikasari, & Bastian, 2016)	untuk membangun sebuah aplikasi berbasis website menggunakan algoritma Winnowing untuk mencari kesamaan pada dua dokumen teks yang diuji.	Algoritma Winnowing	Pada hasil eksperimen pengukuran kemiripan ini disimpulkan bahwa semakin kecil tingkat persentase kesamaan dokumen teks yang diuji, maka tingkat kemiripan dokumen kecil dan tidak termasuk plagiat, tetapi jika hasil dari pengujian pada dua dokumen semakin besar, maka dokumen tersebut mempunyai tingkat kemiripan yang tinggi dan tindakan tersebut dianggap plagiat. Penelitian ini juga menambahkan perbandingan nilai k-gram, basis (bilangan prima), nilai window, keterangan persentase, dan kategori plagiarisme.
6	Pendeteksian Plagiarisme Menggunakan Algoritma <i>Rabin-Karp</i> dengan Metode <i>Rolling Hash</i> (Priambodo, 2018)	Untuk mengatasi plagiat yang biasanya dilakukan terhadap konten digital adalah melakukan copy-paste, quote, dan revisi terhadap dokumen asli.	Algoritma Rabin-Karp dengan metode Rolling Hash.	Hasil pengujian 30 dokumen teks menghasilkan tingkat akurasi yang terbesar yaitu 47.58%. Hasil persentase tersebut termasuk dalam kategori tingkat plagiat 15-50%, berarti menandakan dokumen tersebut termasuk plagiat tingkat sedang. Sedangkan tingkat akurasi yang terkecil yaitu 19.28%, berarti menandakan dokumen tersebut termasuk plagiat tingkat sedang. Berdasarkan analisis proses pendeteksian tingkat plagiarisme menggunakan algoritma rabin-karp dengan metode rolling hash bisa membaca karakter berupa huruf, simbol seperti titik (.), koma (,), dan lain-lain.
7	Pendeteksian Dokumen Plagiarisme dengan Menggunakan Metode <i>Weight Tree</i> (Nurdin, Rizal, & Rizwan, 2019)	Untuk membangun sebuah sistem pendeteksian kemiripan dari dua dokumen teks yang berbeda untuk jenis dokumen teks berbahasa indonesia dengan format file dokumen yaitu: doc, docx, pdf, rtf. Tahapan	Weight Tree	Hasil pengujian sistem dapat dikategorikan sebagai sistem pendeteksian atau pengetesan kemiripan dokumen. Nilai rata-rata persentase kemiripan dalam pengujian sistem ini adalah 71,60%. Sistem yang dibangun ini berhasil dengan tingkat keakuratan mencapai 90%. Algoritma <i>Weight Tree</i> yang diterapkan pada sistem ini terbukti mampu mengidentifikasi dengan baik kemiripan dokumen plagiarisme.

No	Judul	Tujuan Penelitian	Metode	Hasil Penelitian
8	Pemodelan Penilaian <i>Essay</i> Otomatis Secara <i>Realtime</i> Menggunakan Kombinasi <i>Text Stemming</i> Dan <i>Cosine Similarity</i> (Rinartha, 2017)	Membangun sebuah sistem untuk melakukan penilaian essay otomatis secara realtime.	Cosine similarity	Pengajar akan memasukkan data soal dan jawaban beserta kuncinya ke dalam sistem pada waktu tertentu. Kemudian data diproses di sisi client untuk memperoleh kata-kata kunci dalam jawaban yang kemudian tersimpan di dalam database. Ketika peserta didik menjawab pertanyaan yang ada pada sistem, sistem akan memproses jawaban hingga menemukan kata-kata kunci yang digunakan dalam jawaban tersebut, kemudian dibandingkan dengan jawaban yang sudah ada didalam sistem. Hasil ujian akan ditampilkan pada sisi pengajar untuk mengetahui nilai yang didapatkan oleh peserta didik.
9	Implementasi Deteksi Plagiarisme Menggunakan Metode <i>N-Gram</i> Dan <i>Jaccard Similarity</i> Terhadap Algoritma <i>Winnowing</i> (Sunardi et al., 2018)	Membangun sebuah sistem untuk mengecek nilai kemiripan antara dokumen x dengan dokumen y	N-gram dan Jaccard similarity dengan Algoritme winnowing	Hasil penelitian yang dilakukan dapat dipahami bahwa n-gram sangat mempengaruhi hasil dari similarity, penggunaan n-gram yang tepat sangat diperlukan. Tingkat kemiripan pada tahap pengujian sebuah dokumen mencapai 100%, sehingga metode Jaccard Similarity memiliki prospek untuk digunakan dalam deteksi plagiat.
10	Implementasi Algoritma <i>Rabin-Karp</i> untuk Pendeteksi Plagiarisme pada Dokumen Tugas Mahasiswa (Filcha & Hayaty, 2019)	Membangun sebuah sistem untuk melakukan pemeriksaan plagiarisme pada dokumen tugas antar mahasiswa dengan cepat dan tepat.	Rabin-Karp	Hasil dari metode ini dihitung menggunakan <i>dice coefficient</i> . Perhitungan akurasi dengan melakukan 20 perbandingan antara sistem pendeteksi plagiarisme dan <i>software</i> Plagiarisme Checker X menggunakan <i>confusion matrix</i> menghasilkan tingkat keakuratan sebesar 90%.
11	Deteksi plagiarisme dokumen bahasa indonesia dengan algoritma <i>jaro-winkler Distance</i> (Anggara, 2016)	Untuk membandingkan kesamaan antar dokumen teks berbahasa Indonesia, sehingga dapat ditentukan sebuah dokumen tersebut plagiat atau tidak.	Jaro-Winkler distance.	Pengujian terhadap aplikasi menggunakan data abstrak jurnal skripsi. Dari hasil analisis dokumen uji 1 memiliki kesamaan tertinggi dengan dokumen nomor 55 dengan nilai 86,267%, dokumen 2 memiliki kesamaan tertinggi dengan dokumen 66 dengan nilai 95,922% dan dokumen 3 memiliki kesamaan tertinggi dengan dokumen 23 dengan nilai 98.361 %.

No	Judul	Tujuan Penelitian	Metode	Hasil Penelitian
12	Aplikasi pendeteksi plagiat terhadap karya tulis berbasis web menggunakan natural language processing dan algoritma knuth-morris-pratt (Alamanda, Suhery, & Brianorman, 2016)	membuat aplikasi berbasis web yang dapat mendeteksi plagiat terhadap karya tulis menggunakan Natural Language Processing	Algoritma Knuth–Morris-Pratt.	Dari hasil pengujian yang telah dilakukan, rata-rata persentase kemiripan yang dihasilkan dari pendeteksian tanpa menggunakan proses <i>TF-IDF</i> sebesar 45,98%, nilai tersebut lebih rendah dibandingkan dengan pendeteksian menggunakan proses <i>TF-IDF</i> , <i>Tokenizing</i> , <i>Filtering</i> dan <i>Stemming</i> yang menghasilkan persentase kemiripan 41,09%. Sedangkan tanpa menggunakan proses <i>Stemming</i> yaitu 40,58% serta tanpa menggunakan proses <i>TF-IDF</i> dan <i>Stemming</i> sebesar 40,54%
13	<i>Mining disinformation and fake news: Concepts, methods, and recent advancements</i> (Shu, Wang, Lee, & Liu, 2020)	Membahas metode pengecekan similarity dokumen	<i>String Based</i>	<i>String Based</i> ini dibagi menjadi 2, yaitu <i>Character Based</i> dan <i>Term Based</i> . Dari 2 pembagian ini ada beberapa metode yang bisa digunakan, sesuai dengan kebutuhan. Untuk <i>Character Based</i> pembagiannya ada beberapa metode diantaranya, <i>Longest Common SubString (LCS)</i> , <i>Longest Common SubString (LCS)</i> , <i>Jaro</i> , <i>Jaro–Winkler</i> , <i>Needleman-Wunsch</i> , <i>Smith-Waterman</i> , <i>N-gram</i> . Untuk <i>Term Based</i> pembagiannya ada, <i>Block Distance</i> , <i>Cosine similarity</i> , <i>Dice’s coefficient</i> , <i>Euclidean distance</i> , <i>Jaccard similarity</i> , <i>Matching Coefficient</i> , <i>Overlap coefficient</i> .
14	<i>Measurement of Text Similarity: A Survey</i> (Wang & Dong, 2020)	Untuk menganalisis keuntungan dan kerugian dari suatu metode yang digunakan dalam pendeteksian similarity		Memberikan referensi bagi penelitian dan aplikasi terkait, metode pengukuran kemiripan teks dijelaskan melalui dua aspek yaitu jarak teks dan representasi teks. Jarak teks dapat dibagi menjadi jarak panjang, jarak distribusi, dan jarak semantik; Representasi teks dibagi menjadi teks berbasis string, berbasis korpus, teks semantik tunggal, teks multi-semantik, dan representasi berbasis struktur grafik.

## BAB III METODOLOGI PENELITIAN

### 3.1. Data Penelitian

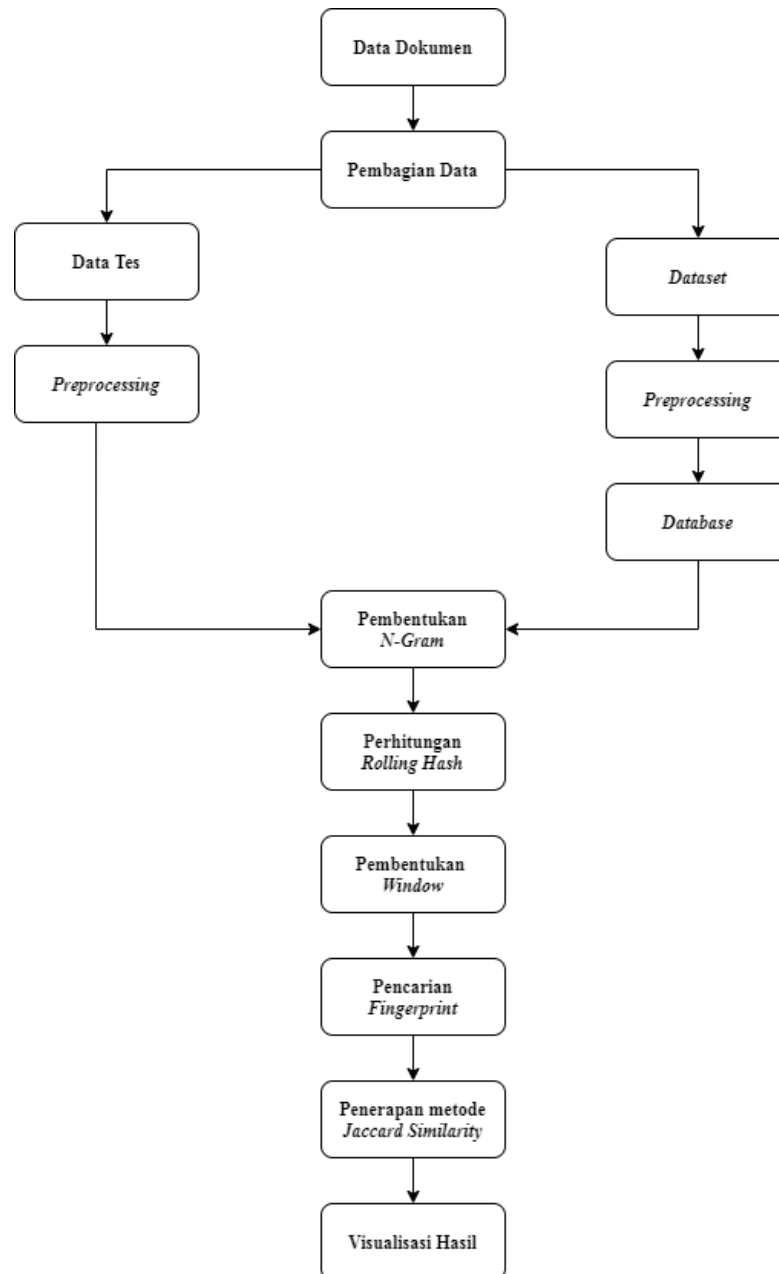
*Dataset* atau sumber data yang digunakan dalam penelitian ini bersumber dari data dokumen abstrak skripsi mahasiswa Budi Luhur, data ini berbentuk *.pdf*. *Dataset* yang digunakan dalam penelitian ini berjumlah 200 data, pada dokumen abstrak mahasiswa Fakultas Teknologi informasi tahun 2015. *Dataset* ini didapatkan dari perpustakaan Universitas Budi Luhur. Data tersebut kemudian diolah melalui tahap *preprocessing* sehingga data tersebut bisa digunakan sebagai *dataset*. Berikut adalah contoh *record* dari data dokumen abstrak hasil *preprocessing* dapat dilihat pada Tabel 3.1 berikut :

**Tabel 3.1 Sampel Data Abstrak**

Id	Nim	Uploaddate	File_Size	Content
1	1511500025	29-12-20	367036	implementasi algoritme multinomial naive bayes classifier tf ltf stemming nazief adriani klasifikasi sentimen alam interview twitter rausal valino adjir humaika resources consulting konsultan human resources department hrd management depan percaya naung.....
2	1511500082	29-12-20	23007	implementasi algoritme depth first search maze based level generator prototype game d top down shooter preposterous defiance bas mobile fadel achmad assegaf video game hiburan suka dewasa video game hilang jenuh jalan aktivitas hari latihan konsentrasi fokus....
3	1511500132	29-12-20	41297	implementasi aman file algoritme advanced encryption standard aes bit bas web smk pgri ciledug imbar yuli hartoko dikarenakan kembang komputer aman komputer simpan tukar sifat rahasia smk pgri ciledug instansi gerak tangerang file soal uji file sifat rahasia.....
4	1511500157	29-12-20	177537	implementasi augmented reality metode marker based tracking briefing terima calon karyawan magang suwiwi deddy rivaldy teliti tuju kembang aplikasi augmented reality briefing media kertas sewiwi briefing isi tulis jarang calon karyawan magang bingung baya....
5	1511500199	29-12-20	99221	aplikasi aman file algoritme advanced encryption standard aes bas web toko khayangan outdoor yayat khayangan outdoor gerak jual alat daki buku ketik ulang komputer file rentan curi rusak kam guna mesti simpan sadap susup tanggung curi pindah simpan komput....

### 3.2. Penerapan Metode

Untuk membangun sistem pendeteksi plagiarisme menggunakan metode *n-gram* dan *jaccard similarity* terhadap algoritme *winnowing*, terdapat beberapa tahapan yang menjadi rancangan utama, rancangan ini sebagai gambaran proses tahapan awal hingga akhir sistem berjalan, yang terdapat pada Gambar 3.1 berikut.



**Gambar 3.1 Tahapan Metode**

Pada Gambar 3.1 merupakan proses pengolahan data dokumen abstrak skripsi mahasiswa Budi Luhur, pada tahap ini dokumen abstrak digunakan

sebagai data tes dan *dataset*. Pada proses *dataset*, data yang sudah di *upload* akan diproses melalui tahap *preprocessing*, begitupun dokumen yang dijadikan sebagai data tes. Kemudian dokumen yang digunakan sebagai *dataset* akan disimpan ke dalam *database*, sedangkan dokumen data tes tidak disimpan ke dalam *database*. Dokumen abstrak yang digunakan berupa dokumen *.pdf*, ukuran < 1mb, file masukan berupa tulisan. Selanjutnya dokumen yang sudah melalui tahap *preprocessing*, akan dilakukan pembentukan *n-gram*, kemudian perhitungan *rolling hash*, setelah *hash*nya didapatkan, maka dilakukan pembentukan *window*, lalu dicarilah *fingerprint* dari *window* yang sudah dibentuk sebelumnya. Proses selanjutnya yaitu pencarian nilai *similarity* dari sebuah dokumen abstrak melalui metode *jaccard similarity*. Selanjutnya *user* bisa melihat visualisasi hasil dari pengecekan nilai *similarity* dokumen abstrak mahasiswa Budi Luhur.

#### 3.2.1. Data Dokumen

Pada tahap ini dilakukan proses pengumpulan data dokumen abstrak dari skripsi mahasiswa Budi Luhur. Dimana data dokumen ini ada yang dijadikan sebagai *dataset* dan ada juga yang menjadi data tes. *Dataset* ini berasal dari dokumen abstrak skripsi mahasiswa yang sudah di publish di perpustakaan Budi Luhur. Sedangkan data tes berasal dari dokumen abstrak mahasiswa yang akan menjalankan skripsi, sehingga mahasiswa bisa melakukan pengecekan *similarity* di dokumen abstraknya.

#### 3.2.2. Preprocessing

Pada tahapan *preprocessing* ini, dilakukan beberapa proses untuk mendapatkan *dataset* yang bersih, sehingga proses pengujian tingkat *similarity* lebih tepat dan akurat. Proses *preprocessing* yang digunakan pada penelitian ini diantaranya yaitu :

##### a. Case Folding

*Case Folding* merupakan proses penyetaraan kata yang mengandung huruf besar untuk diubah menjadi huruf kecil, misalnya Tujuan menjadi tujuan, Adapun menjadi adapun, dan seterusnya.

##### b. Cleaning

*Cleaning* merupakan proses untuk menghapus karakter selain a sampai z atau karakter selain huruf, karakter tersebut dihilangkan, seperti contoh 8266, / (garis miring), dan seterusnya sehingga hanya menyisakan karakter huruf saja.

##### c. Mengganti Slang word

*Slang word* merupakan proses mengganti kata tidak baku seperti “analisa” dan “rejek”, kata yang tidak baku tersebut biasanya berupa singkatan atau bahasa yang kekinian, untuk itu supaya kata-kata dalam teks setara dengan EYD, maka kata tersebut digantikan dengan kata baku yang seharusnya yaitu

menjadi “analisis” dan “rezeki”, pergantian kata-kata ini berdasarkan kamus yang terdapat dalam library *slang word*.

d. Menghapus *Stop word*

Dalam penelitian Tala, (2003), terdapat kumpulan kata-kata yang termasuk ke dalam *stoplist* yaitu kata umum yang dianggap tidak terlalu memiliki makna yang penting dan kemunculan kata ini sangat tinggi frekuensinya, seperti pada contoh teks di bawah ini yaitu pada kata “untuk”, “dan”, “bagi”, kata tersebut dihilangkan karena masuk ke dalam daftar *stop word*.

e. Menghapus Spasi

Pada proses ini dilakukan penghapusan spasi dari setiap kata, untuk melakukan proses selanjutnya, yaitu pada proses pencarian tingkat *similarity* sebuah dokumen.

### 3.2.3. N-Gram

*N-gram* adalah *substring* penggabungan karakter sejumlah  $k$  pada teks dokumen. Dalam menentukan hasil deteksi plagiarisme dengan menggunakan metode *n-gram*, dimana dokumen atau sekumpulan kata akan diproses dan akan dibentuk sebanyak *n-gram* atau memisahkan string sepanjang  $n$  yang akan dihitung pergeserannya secara terus menerus ke depan sejumlah nilai  $n$  sampai akhir dokumen. Sebagai contoh *n-gram* dari kalimat **N = Sortir berbasis Arduino**, dengan nilai  $N=3$  maka (sor ort rti tir irb rbe ber erb rba bas asi sis isa sar ardu dui uin ino).

### 3.2.4. Algoritme Winnowing

*Winnowing* adalah algoritme yang digunakan untuk melakukan proses pengecekan kesamaan kata (*document fingerprinting*) untuk mengidentifikasi tingkat *similarity*.

a. *Rolling Hash*

Pada tahapan ini dilakukan perhitungan *rolling hash* untuk mencari nilai *hash* dari setiap *string* yang sudah dipotong pada tahapan *n-gram*. Setiap *string* diubah menjadi ASCII lalu dihitung dengan rumus *hash* dibawah:

$$H(c_1 \dots c_k) = c_1 * b^{(k-1)} + c_2 * b^{(k-2)} + \dots + c_k * b^{(k-k)} \dots (3.1)$$

$c_1$  = nilai ascii dari huruf pertama dari satu buah *k-gram*

$b$  = bilangan prima

$k$  = nilai *k-gram*

Setelah hasil dari *hash* pertama didapatkan selanjutnya digunakan rumus *rolling hash* untuk menghitung nilai *hash* ke dua dan seterusnya tanpa menghitung ASCII yang sudah di hitung sebelumnya.

$$H(c_2 \dots c_k + 1) = (H(c_1 \dots c_k) - c_1 * b^{(k-1)}) * b + c^{(k+1)} \dots (3.2)$$

Hasil dari perhitungan sebelumnya dikurangi hasil *hash* dari huruf pertama *gram* sebelumnya lalu dikali dengan hasil *hash* dari perhitungan ASCII terakhir dari *gram* sekarang.

b. Pembentukan *Window*

Setelah nilai *hash* ditemukan semua maka langkah selanjutnya adalah pembentukan *window*. Nilai dari *rolling hash* akan dikelompokkan berdasarkan nilai *wgram*. Pada kasus ini nilai *window* yang dipergunakan adalah  $w=3$ .

c. Nilai *Fingerprint*

Proses selanjutnya setelah pembentukan *window* adalah mencari nilai *fingerprint*. Nilai *fingerprint* ditentukan berdasarkan nilai terkecil dari setiap *window* yang ada.

d. *Jaccard Similarity*

Pada tahapan ini untuk mencari nilai *similarity* pada sebuah dokumen dengan rumus *Jaccard similarity* dengan melihat *irisan* dan *union* dari *fingerprint* antar dua dokumen. Berikut adalah rumus *jaccard similarity* :

$$\text{Similarity (X,Y)} = \frac{|x \cap y|}{|x \cup y|} \times 100\% \quad \dots\dots\dots(3.3)$$

X = *Dataset*

Y = *Data Tes*

Dimana *irisan* dari dokumen X dan Y dibagi dengan *union* dari dokumen X dan Y kemudian di kali 100% maka menghasilkan persentase *similarity* dari dokumen X dan Y.

### 3.3. Rancangan Pengujian

Pengujian dilakukan untuk mengetahui tingkat *similarity* dari sebuah dokumen. Pada penelitian ini digunakan metode pengujian *Black box Testing* dalam melakukan proses pengujian program. *Black Box Testing* merupakan pengujian yang dapat dilakukan dengan melakukan pengamatan, pada hasil eksekusi melalui beberapa data tes dan memeriksa fungsional yang terdapat pada perangkat lunak. Pada langkah pengujian ini akan dijelaskan mengenai proses yang berjalan pada sistem yang dibangun, mulai dari *upload* dokumen, menjadikan dokumen sebagai *dataset* dan data tes sampai menampilkan hasil perhitungan tingkat *similarity*.



## BAB IV HASIL DAN PEMBAHASAN

### 4.1. Lingkungan Percobaan

Agar sistem yang telah dikembangkan dapat berjalan dengan semestinya, dibutuhkanlah perangkat dengan spesifikasi tertentu, adapun dalam penelitian ini menggunakan spesifikasi perangkat diantaranya:

#### 4.1.1. Spesifikasi Perangkat Keras

Perangkat keras yang mendukung aplikasi ini berjalan dengan baik sebagai berikut:

- a. *Processor* : Intel(R) Pentium(R) CPU 2020M @ 2.40GHz
- b. *RAM* : 6 GB DDR3
- c. *Harddisk* : 500 GB
- d. *VGA* : AMD HD8570 2GB dedicated

#### 4.1.2. Spesifikasi Perangkat Lunak

Perangkat lunak yang mendukung aplikasi ini berjalan dengan baik sebagai berikut:

- a. Sistem Operasi : Windows 7 Ultimate 64-bit
- b. IDE : Visual Studio Code v.1.38.1
- c. DBMS : Mysql Database
- d. Web Server : Apache (XAMPP v3.2.2, PHP versi 7.3.0)
- e. Browser : Google Chrome

### 4.2. Implementasi Metode dan Langkah Pengujian

#### 4.2.1. Data Dokumen

Data dokumen merupakan tahap untuk pengumpulan *dataset* dan proses pengujian data tes. Data yang digunakan berupa dokumen abstrak skripsi mahasiswa Budi Luhur dalam bentuk *pdf* yang kemudian di *convert* ke dalam bentuk teks. Data dokumen didapatkan dari perpustakaan Universitas Budi Luhur. Data dokumen yang sudah berbentuk teks akan dilakukan tahap *preprocessing* pembentukan *n-gram*, *rolling hash*, pembentukan *window* dan pencarian *fingerprint*. Sehingga diketahui tingkat *similarity* dari sebuah dokumen. Pada Tabel 4.1 merupakan contoh dokumen abstrak yang sebelumnya berupa file *pdf* kemudian di *convert* menjadi teks. Berikut bisa dilihat di Tabel 4.1 berikut.

**Tabel 4.1 Dokumen Abstrak**

No	Data Dokumen
1	<p>IMPLEMENTASI ALGORITME MULTINOMIAL NAÏVE BAYES CLASSIFIER DAN TF-LTF SERTA STEMMING NAZIEF &amp; ADRIANI UNTUK MENGLASIFIKASI SENTIMEN MASYARAKAT TENTANG PENGALAMAN INTERVIEW DARI DATA TWITTER Oleh : Rausal Valino Adjir (1511500025) Humaika Resources Consulting merupakan sebuah konsultan Human Resources Departement (HRD) dan Management terdepan dan terpercaya yang berada di bawah naungan PT Humanika Amanah Indonesia. Konsultasi yang diberikan berfokus pada Recruitment, Assessment, Training, Development, dan Organization Development. Humanika Consulting memudahkan para perusahaan untuk mencari dan mempersiapkan tenaga ahli yang dibutuhkan sesuai dengan kebutuhan dan standar masing-masing perusahaan. Permasalahan yang muncul adalah timbulnya sebuah pertanyaan seperti "Seberapa banyak masyarakat yang mengalami kesulitan dalam interview?". Oleh karena itu dibutuhkan sebuah sistem yang dapat mengklasifikasikan sentimen masyarakat tentang pengalaman interview mereka. Dalam penelitian ini, data bersumber dari twitter yang nantinya akan diolah terlebih dahulu. Data yang diklasifikasikan menjadi 3 kategori yaitu Positif, Negatif dan Netral. Dalam sistem ini, data akan diolah terlebih dahulu melalui tahap prapemrosesan atau Preprocessing yang dimana terdapat beberapa tahap didalamnya. Adapun tahap Preprocessing yang digunakan adalah Case Folding, Cleansing, Tokenizing, Stopword Removal dan Stemming. Algoritme Nazief &amp; Adriani digunakan dalam proses stemming untuk membuang imbuhan kata menjadi kata dasar pada Bahasa Indonesia/ Kemudian data tersebut diberikan bobot dengan metode pembobotan kata TF-LTF yang nantinya akan mengoptimalkan perhitungan probabilitas pada Algoritme Multinomial Naïve Bayes Classifier. Setelah melalui semua tahapan tadi, data kemudian diklasifikasikan dengan algoritme Vmap Naïve Bayes Classifier. Dengan adanya sistem klasifikasi ini diharapkan dapat membantu Humanika Resources Consulting dalam meningkatkan mutu interview berdasarkan pengalaman interview dari twitter. Kata Kunci : Klasifikasi Sentimen, Preprocessing, Nazief &amp; Adriani, TF-LTF, Multinomial Naïve Bayes Classifier, Vmap Naïve Bayes Classifier. Xviii + 92 halaman : 60 gambar, 24 tabel, 2 lampiran Perpustakaan Universitas Budi Luhur</p>

No	Data Dokumen
2	Implementasi Teknologi Augmented Reality Menggunakan Metode Marker Based Tracking Sebagai Media Briefing Penerimaan Calon Karyawan Magang pada PT. Sewiwi Indonesia Oleh: Deddy Rivaldy (1511500157) Penelitian ini bertujuan untuk mengembangkan aplikasi Augmented Reality pada bidang briefing yang medianya berupa kertas pada PT. Sewiwi Indonesia. Melalui media briefing berupa buku yang hanya berisi tulisan tidak jarang calon karyawan magang masih merasa bingung untuk membayangkan gambaran atau bentuk topologi yang akan dijadikan test perusahaan dan juga mudah rusak jika tetap menggunakan media kertas. Sehingga terkadang calon karyawan magang harus bertanya untuk mendapatkan informasi ataupun gambaran yang lebih jelas. Karena semakin berkembangnya zaman dan teknologi maka ada jalan keluar untuk mengatasi permasalahan tersebut yaitu dengan menggunakan aplikasi Augmented Reality (AR). Aplikasi ini nantinya akan berisi tentang Augmented Reality (AR) topologi jaringan yang akan dijadikan media briefing. Aplikasi ini dibangun dengan menggunakan bahasa pemograman C# dan juga menggunakan metode Marker Based Tracking yang nanti nya ada sebuah topologi jaringan berisi gambar sebagai salah satu marker penanda dan smartphone untuk memindai marker tersebut. Hasil dari penelitian ini menghasilkan aplikasi Augmented Reality (AR) yang dapat menyajikan sebuah gambar 3D dari Topologi Jaringan. Kata kunci: Augmented Reality, Android, Marker Based Tracking XIII+52halaman; 34 Gambar; 12 Tabel; 6 Lampiran Perpustakaan Universitas Budi Luhur

#### 4.2.2. Tahapan *Preprocessing*

Setelah data dokumen didapat, maka dokumen akan memasuki tahap *preprocessing* pada dokumen yang sudah menjadi teks, proses yang dilakukan yaitu, *case folding*, menghapus karakter selain a-z, mengganti *slang word*, menghapus *stop word*, Menghilangkan spasi antar karakter.

##### a. *Case Folding*

Pada Tabel 4.2 merupakan proses untuk merubah isi teks dokumen menjadi huruf kecil, melalui proses *case folding*, sehingga isi teks menjadi huruf kecil semua. Berikut contohnya.

**Tabel 4.2 Case Folding**

Penjelasan	Data Dokumen
Sebelum	IMPLEMENTASI DATA MINING UNTUK MEMPREDIKSI PENERIMAAN PERMOHONAN CALON NASABAH ASURANSI KESEHATAN DENGAN MENGGUNAKAN ALGORITMA NAÏVE BAYES BERBASIS WEB PADA PT. ASURANSI SINARMAS Oleh : Lady Caesar Sevika Detale (1511503581) Kesehatan merupakan bagian yang amat penting bagi kehidupan manusia. PT. Asuransi Sinarmas adalah salah satu asuransi yang menyediakan produk asuransi kesehatan. Banyaknya masyarakat yang mendaftar dan mengajukan permohonan asuransi kesehatan, membuat pihak asuransi harus lebih memperhatikan mengenai penerimaan permohonan asuransi kesehatan. Hal tersebut dikarenakan pemberian jaminan kesehatan memiliki resiko yang cukup tinggi. Di sisi lain, banyaknya nasabah yang sering terlambat dan kurang peduli membayar premi, membuat pihak asuransi mengalami kerugian semakin besar. Maka dari itu, upaya untuk meminimalisir adanya permasalahan tersebut, pihak asuransi harus lebih memperhatikan lagi terhadap penerimaan permohonan calon nasabah.
Sesudah	judul : aplikasi augmented reality sspol(solar system planet and orbital lines). sekolah dasar negri cipadu 3 adalah sebuah sekolah yang berada di tangerang selatan. tata surya adalah suatu pelajaran ilmu pengetahuan alam di sdn cipadu 3, media pembelajaran yang digunakan untuk mempelajari planet dan tata surya masih menggunakan globe, presentasi, dan buku. dalam menentukan aplikasi dibutuhkan beberapa tahap yaitu mengumpulkan data dan informasi melalui catatan kuliah, analisis, perancangan aplikasi dan implementasi aplikasi. dengan menggunakan media pembelajaran menggunakan augmented reality dengan media android dan dengan animasi 3d diharapkan siswa dan siswi jadi menarik untuk belajar, mengerti bentuk dari planet tata surya dan bisa mempelajari planet tata surya di rumah. . menggunakan video siswa-siswi juga dapat melihat sambil mendengarkan video yang disampaikan. augmented reality adalah sebuah teknologi masa kini yang menggabungkan antara dunia maya dan dunia nyata secara realtime.

**b. Cleaning**

Pada Tabel 4.3 merupakan proses untuk menghapus karakter selain a-z. Sehingga isi teks yang bukan huruf a-z akan dihapus, seperti angka, simbol, dan seterusnya. Berikut contohnya.

**Tabel 4.3 Cleaning**

Penjelasan	Data Dokumen
Sebelum	implementasi data mining untuk memprediksi penerimaan permohonan calon nasabah asuransi kesehatan dengan menggunakan algoritma naïve bayes berbasis web pada pt. asuransi sinarmas oleh : lady caesar sevika detale (1511503581) kesehatan merupakan bagian yang amat penting bagi kehidupan manusia. pt. asuransi sinarmas adalah salah satu asuransi yang menyediakan produk asuransi kesehatan. banyaknya masyarakat yang mendaftar dan mengajukan permohonan asuransi kesehatan, membuat pihak asuransi harus lebih memperhatikan mengenai penerimaan permohonan asuransi kesehatan. hal tersebut dikarenakan pemberian jaminan kesehatan memiliki resiko yang cukup tinggi. di sisi lain, banyaknya nasabah yang sering terlambat dan kurang peduli membayar premi, membuat pihak asuransi mengalami kerugian semakin besar. maka dari itu, upaya untuk meminimalisir adanya permasalahan tersebut, pihak asuransi harus lebih memperhatikan lagi terhadap penerimaan permohonan calon nasabah.
Sesudah	implementasi data mining untuk memprediksi penerimaan permohonan calon nasabah asuransi kesehatan dengan menggunakan algoritma naïve bayes berbasis web pada pt asuransi sinarmas oleh lady caesar sevika detale kesehatan merupakan bagian yang amat penting bagi kehidupan manusia pt asuransi sinarmas adalah salah satu asuransi yang menyediakan produk asuransi kesehatan banyaknya masyarakat yang mendaftar dan mengajukan permohonan asuransi kesehatan membuat pihak asuransi harus lebih memperhatikan mengenai penerimaan permohonan asuransi kesehatan hal tersebut dikarenakan pemberian jaminan kesehatan memiliki resiko yang cukup tinggi di sisi lain banyaknya nasabah yang sering terlambat dan kurang peduli membayar premi membuat pihak asuransi mengalami kerugian semakin besar maka dari itu upaya untuk meminimalisir adanya permasalahan tersebut pihak asuransi harus lebih memperhatikan lagi terhadap penerimaan permohonan calon nasabah

c. Mengganti *slang word*

Pada Tabel 4.4 merupakan proses untuk mengganti kata yang tidak baku menjadi kata baku, seperti kata analisa diubah menjadi analisis. Berikut contohnya.

**Tabel 4.4 Mengganti *slang word***

Penjelasan	Data Dokumen
Sebelum	implementasi data mining untuk memprediksi penerimaan permohonan calon nasabah asuransi kesehatan dengan menggunakan algoritma na ve bayes berbasis web pada pt asuransi sinarmas oleh lady caesar sevika detale kesehatan merupakan bagian yang amat penting bagi kehidupan manusia pt asuransi sinarmas adalah salah satu asuransi yang menyediakan produk asuransi kesehatan banyaknya masyarakat yang mendaftar dan mengajukan permohonan asuransi kesehatan membuat pihak asuransi harus lebih memperhatikan mengenai penerimaan permohonan asuransi kesehatan hal tersebut dikarenakan pemberian jaminan kesehatan memiliki resiko yang cukup tinggi di sisi lain banyaknya nasabah yang sering terlambat dan kurang peduli membayar premi membuat pihak asuransi mengalami kerugian semakin besar maka dari itu upaya untuk meminimalisir adanya permasalahan tersebut pihak asuransi harus lebih memperhatikan lagi terhadap penerimaan permohonan calon nasabah
Sesudah	implementasi data mining untuk memprediksi penerimaan permohonan calon nasabah asuransi kesehatan dengan menggunakan algoritma naive bayes berbasis web pada pt asuransi sinarmas oleh lady caesar sevika detale kesehatan merupakan bagian yang amat penting bagi kehidupan manusia pt asuransi sinarmas adalah salah satu asuransi yang menyediakan produk asuransi kesehatan banyaknya masyarakat yang mendaftar dan mengajukan permohonan asuransi kesehatan membuat pihak asuransi harus lebih memperhatikan mengenai penerimaan permohonan asuransi kesehatan hal tersebut dikarenakan pemberian jaminan kesehatan memiliki risiko yang cukup tinggi di sisi lain banyaknya nasabah yang sering terlambat dan kurang peduli membayar premi membuat pihak asuransi mengalami kerugian semakin besar maka dari itu upaya untuk meminimalisir adanya permasalahan tersebut pihak asuransi harus lebih memperhatikan lagi terhadap penerimaan permohonan calon nasabah

d. Menghapus *stop word*

Pada Tabel 4.5 merupakan proses untuk menghapus kata yang diabaikan dalam tahap *preprocessing*, seperti kata dan, untuk, bagi, dan seterusnya. Berikut contohnya.

**Tabel 4.5 Menghapus *stop word***

Penjelasan	Data Dokumen
Sebelum	implementasi data mining untuk memprediksi penerimaan permohonan calon nasabah asuransi kesehatan dengan menggunakan algoritma naive bayes berbasis web pada pt asuransi sinarmas oleh lady caesar sevika detale kesehatan merupakan bagian yang amat penting bagi kehidupan manusia pt asuransi sinarmas adalah salah satu asuransi yang menyediakan produk asuransi kesehatan banyaknya masyarakat yang mendaftar dan mengajukan permohonan asuransi kesehatan membuat pihak asuransi harus lebih memperhatikan mengenai penerimaan permohonan asuransi kesehatan hal tersebut dikarenakan pemberian jaminan kesehatan memiliki risiko yang cukup tinggi di sisi lain banyaknya nasabah yang sering terlambat dan kurang peduli membayar premi membuat pihak asuransi mengalami kerugian semakin besar maka dari itu upaya untuk meminimalisir adanya permasalahan tersebut pihak asuransi harus lebih memperhatikan lagi terhadap penerimaan permohonan calon nasabah
Sesudah	implementasi data mining memprediksi penerimaan permohonan calon nasabah asuransi kesehatan algoritma naive bayes berbasis web asuransi sinarmas lady caesar sevika detale kesehatan asuransi sinarmas asuransi asuransi kesehatan banyaknya mendaftar mengajukan permohonan asuransi kesehatan asuransi memperhatikan penerimaan permohonan asuransi kesehatan pemberian jaminan kesehatan risiko sisi banyaknya nasabah terlambat peduli membayar premi asuransi kerugian meminimalisir permasalahan asuransi memperhatikan terhadap penerimaan permohonan calon nasabah

e. Menghilangkan spasi antar karakter

Pada Tabel 4.6 merupakan proses untuk menghilangkan spasi antar karakter. Berikut contohnya.

**Tabel 4.6 Menghilangkan spasi antar karakter**

Penjelasan	Data Dokumen
Sebelum	implementasi data mining memprediksi penerimaan permohonan calon nasabah asuransi kesehatan algoritma naive bayes berbasis web asuransi sinarmas lady caesar sevika detale kesehatan asuransi sinarmas asuransi asuransi kesehatan banyaknya mendaftar mengajukan permohonan asuransi kesehatan asuransi memperhatikan penerimaan permohonan asuransi kesehatan pemberian jaminan kesehatan risiko sisi banyaknya nasabah terlambat peduli membayar premi asuransi kerugian meminimalisir permasalahan asuransi memperhatikan terhadap penerimaan permohonan calon nasabah

Penjelasan	Data Dokumen
Sesudah	implementasidataminingmemprediksipenerimaanpermohonan calon nasabah asuransi kesehatan algoritma naive bayes berbasis web asuransi inar mas l ad y caesar se vik ad etale kesehatan asuransi sinar mas asuransi s uransi kesehatan banyak nyamendaftarmengajukanpermohonanasuran sikehatan asuransi memperhatikan penerimaan permohonan asuransi kesehatan pemberian jaminan kesehatan risiko sibanyaknyanasabahte r lambat pedulimembayar premi asuransi kerugian meminimalisir perma salahan asuransi memperhatikan terhadap penerimaan permohonan calo n nasabah

#### 4.2.3. Pembentukan *N-Gram*

Setelah dokumen melalui tahap *preprocessing*, kemudian masuk ke tahap pembentukan *n-gram* yaitu, proses memecahkan *string* teks yang dikelompokkan berdasarkan nilai *n-gram* yang akan dihitung pergeserannya secara terus menerus ke depan sejumlah nilai *n* sampai akhir dokumen. Contoh tahapan pembentukan *k-gram* dengan nilai *N=3*. Berikut bisa dilihat di Tabel 4.7 berikut.

**Tabel 4.7 Pembentukan *N-Gram***

Keterangan	Hasil <i>Preprocessing</i>	<i>N-Gram</i>
<i>Dataset</i>	implementasiaugmentedrealitymetodemarkerbasedtrackingberbasisandroidpromosi smkyadikakarangtengah	imp mpl ple lem eme men ent nta tas asi sia iau aug ugm gme men ent nte ted edr dre rea eai ali lit ity tym yme met eto tod ode dem ema mar ark rke ker erb rba bas ase sed edt dtr tra rak aki kin ing ngb gbe ber erb rba bas asi sia isa san and ndr dro roi oid idp dpr pro rom omo mos osi sis ism smk mky kya yad adi dik ika kak aka kar ara ran ang ngt gte ten eng nga gah
Data Tes	implementasiteknologiaugmentedrealitypembelajaranmetodemarkerbasedtrackingberbasismobileandroidpau danandakreoselatan	imp mpl ple lem eme men ent nta tas asi sit ite tek ekn kno noi oio iog ogi gia iau aug ugm gme men ent nte ted edr dre rea eal ali lit ity typ ype pem emb mbe bel ela laj aja jar ara ran anm nme met eto tod ode dem ema mar ark rke ker erb rba bas ase sed edt dtr tra rac ack cki kin ing ngb gbe ber erb rba bas asi sis ism smo mob obi bil ile lea ean and ndr dro roi oid idp dpa pau aud uda dan ana nan and nda dak akr kre reo eos ose sel ela lat ata tan

#### 4.2.4. Algoritme *Winnowing*

Algoritme *winnowing* merupakan salah satu algoritme yang berfungsi sebagai dokumen *fingerprint* atau algoritme yang digunakan



untuk mendeteksi tindakan plagiarisme dengan menggunakan teknik *hashing*. *Input* dari algoritme *winnowing* berupa dokumen teks, dan akan menghasilkan keluaran berupa kumpulan nilai *hash* yang terbentuk dari perhitungan ASCII pada setiap karakter. Berikut tahapannya.

a. Perhitungan Rolling Hash

Setelah pembentukan *n-gram* maka proses selanjutnya yaitu, perhitungan *rolling hash* untuk mencari nilai *hash* dari setiap *string* yang sudah dipotong pada tahap *n-gram*. Setiap *string* yang sudah dipotong diubah menjadi ASCII lalu dihitung dengan rumus *hash*. Misal kita mengambil dari potongan kalimat *dataset* dan data tes yang ada. Rumus *hash*:

$$H(c_1 \dots c_k) = c_1 * b^{(k-1)} + c_2 * b^{(k-2)} + \dots + c_k * b^{(k-k)} \dots (4.1)$$

$c_1$  = nilai ascii dari huruf pertama dari satu buah *k-gram*

$b$  = bilangan prima

$k$  = nilai *k-gram*

**Tabel 4.8 Kode ASCII**

Karakter	Nilai Unicode	Nilai ANSI ASCII	Keterangan
	(heksadesimal)	(desimal)	
a	61	97	Huruf latin b kecil
b	62	98	Huruf latin b kecil
c	63	99	Huruf latin c kecil
d	64	100	Huruf latin d kecil
e	65	101	Huruf latin e kecil
f	66	102	Huruf latin f kecil
g	67	103	Huruf latin g kecil
h	68	104	Huruf latin h kecil
i	69	105	Huruf latin i kecil
j	006A	106	Huruf latin j kecil
k	006B	107	Huruf latin k kecil
l	006C	108	Huruf latin l kecil
m	006D	109	Huruf latin m kecil
n	006E	110	Huruf latin n kecil
o	006F	111	Huruf latin o kecil
p	70	112	Huruf latin p kecil
q	71	113	Huruf latin q kecil
r	72	114	Huruf latin r kecil
s	73	115	Huruf latin s kecil
t	74	116	Huruf latin t kecil

Karakter	Nilai Unicode	Nilai ANSI ASCII	Keterangan
	(heksadesimal)	(desimal)	
u	75	117	Huruf latin u kecil
v	76	118	Huruf latin v kecil
w	77	119	Huruf latin w kecil
x	78	120	Huruf latin x kecil
y	79	121	Huruf latin y kecil
z	007A	122	Huruf latin z kecil

Berikut contoh perhitungan *rolling hash* dengan nilai  $k=3$ , bisa dilihat di Tabel 4.9 berikut.

**Tabel 4.9 Perhitungan *Rolling Hash***

Keterangan	Data Contoh	<i>N-Gram</i>
<i>Dataset</i>	implementasipengamana nfile	imp mpl ple lem eme men ent nta tas asi sip ipe pen eng nga gam ama man ana nan anf nfi fil ile
Data Tes	aplikasipengamananfile	apl pli lik ika kas asi sip ipe pen eng nga gam ama man ana nan anf nfi fil ile

Pada Tabel 4.10 dan Tabel 4.11 akan menjelaskan contoh perhitungan dari potongan kata, untuk mencari nilai *hash* pada *dataset* dan data tes. Berikut penjabaran perhitungan nilai *hash* pada *dataset*, bisa dilihat di Tabel 4.10 berikut.

**Tabel 4.10 Perhitungan *Hash Dataset***

Kalimat ( $c_1 \dots c_k$ )	Desimal ASCII	Penjabaran $c_1 * b^{(k-1)} + c_2 * b^{(k-2)} + \dots + c_k * b^{(k-k)}$	Hasil
imp	i=105 m=109 p=112	$105 * 7^{(3-1)} + 109 * 7^{(3-2)} + 112 * 7^{(3-3)}$	6020
mpl	m=109 p=112 l=108	$109 * 7^{(3-1)} + 112 * 7^{(3-2)} + 108 * 7^{(3-3)}$	6233
ple	p=112 l=108 e=101	$112 * 7^{(3-1)} + 108 * 7^{(3-2)} + 101 * 7^{(3-3)}$	6345
lem	l=108 e=101 m=109	$108 * 7^{(3-1)} + 101 * 7^{(3-2)} + 109 * 7^{(3-3)}$	6108
eme	e=101 m=109 e=101	$101 * 7^{(3-1)} + 109 * 7^{(3-2)} + 101 * 7^{(3-3)}$	5813
men	m=109 e=101 n=110	$109 * 7^{(3-1)} + 101 * 7^{(3-2)} + 110 * 7^{(3-3)}$	6158

Kalimat (c1...ck)	Desimal ASCII	Penjabaran $c1 * b^{(k-1)} + c2 * b^{(k-2)} + \dots + ck * b^{(k-k)}$	Hasil
ent	e=101 n=110 t=116	$101 * 7^{(3-1)} + 110 * 7^{(3-2)} + 116 * 7^{(3-3)}$	5835
nta	n=110 t=116 a=97	$110 * 7^{(3-1)} + 116 * 7^{(3-2)} + 97 * 7^{(3-3)}$	6299
tas	t=116 a=97 s=115	$116 * 7^{(3-1)} + 97 * 7^{(3-2)} + 115 * 7^{(3-3)}$	6478
asi	a=97 s=115 i=105	$97 * 7^{(3-1)} + 115 * 7^{(3-2)} + 105 * 7^{(3-3)}$	5663
sip	s=115 i=105 p=112	$115 * 7^{(3-1)} + 105 * 7^{(3-2)} + 112 * 7^{(3-3)}$	6482
ipe	i=105 p=112 e=101	$105 * 7^{(3-1)} + 112 * 7^{(3-2)} + 101 * 7^{(3-3)}$	6030
pen	p=112 e=101 n=110	$112 * 7^{(3-1)} + 101 * 7^{(3-2)} + 110 * 7^{(3-3)}$	6305
eng	e=101 n=110 g=103	$101 * 7^{(3-1)} + 110 * 7^{(3-2)} + 103 * 7^{(3-3)}$	5822
nga	n=110 g=103 a=97	$110 * 7^{(3-1)} + 103 * 7^{(3-2)} + 97 * 7^{(3-3)}$	6208
gam	g=103 a=97 m=109	$103 * 7^{(3-1)} + 97 * 7^{(3-2)} + 109 * 7^{(3-3)}$	5835
ama	a=97 m=109 a=97	$97 * 7^{(3-1)} + 109 * 7^{(3-2)} + 97 * 7^{(3-3)}$	5613
man	m=109 a=97 n=110	$109 * 7^{(3-1)} + 97 * 7^{(3-2)} + 110 * 7^{(3-3)}$	6130
ana	a=97 n=110 a=97	$97 * 7^{(3-1)} + 110 * 7^{(3-2)} + 97 * 7^{(3-3)}$	5620
nan	n=110 a=97 n=110	$110 * 7^{(3-1)} + 97 * 7^{(3-2)} + 110 * 7^{(3-3)}$	6179
anf	a=97 n=110 f=102	$97 * 7^{(3-1)} + 110 * 7^{(3-2)} + 102 * 7^{(3-3)}$	5625
nfi	n=110 f=102 i=105	$110 * 7^{(3-1)} + 102 * 7^{(3-2)} + 105 * 7^{(3-3)}$	6209
fil	f=102 i=105 l=108	$102 * 7^{(3-1)} + 105 * 7^{(3-2)} + 108 * 7^{(3-3)}$	5841

Kalimat (c1...ck)	Desimal ASCII	Penjabaran $c1 * b^{(k-1)} + c2 * b^{(k-2)} + \dots + ck * b^{(k-k)}$	Hasil
ile	i=105 l=108 e=101	$105 * 7^{(3-1)} + 108 * 7^{(3-2)} + 101 * 7^{(3-3)}$	6002

Setelah dilakukan perhitungan diperoleh hasil dari *rolling hash* “6020 6233 6345 6108 5813 6158 5835 6299 6478 5663 6482 6030 6305 5822 6208 5835 5613 6130 5620 6179 5625 6209 5841 6002”.

Berikut penjabaran perhitungan nilai *hash* pada data tes, bisa dilihat di Tabel 4.11 berikut.

**Tabel 4.11 Perhitungan Hash Data Tes**

Kalimat (c1...ck)	Desimal ASCII	Penjabaran $c1 * b^{(k-1)} + c2 * b^{(k-2)} + \dots + ck * b^{(k-k)}$	Hasil
apl	a=97 p=112 l=108	$97 * 7^{(3-1)} + 112 * 7^{(3-2)} + 108 * 7^{(3-3)}$	5645
pli	p=112 l=108 i=105	$112 * 7^{(3-1)} + 108 * 7^{(3-2)} + 105 * 7^{(3-3)}$	6349
lik	l=108 i=105 k=107	$108 * 7^{(3-1)} + 105 * 7^{(3-2)} + 107 * 7^{(3-3)}$	6134
ika	i=105 k=107 a=97	$105 * 7^{(3-1)} + 107 * 7^{(3-2)} + 97 * 7^{(3-3)}$	5991
kas	k=107 a=97 s=115	$107 * 7^{(3-1)} + 97 * 7^{(3-2)} + 115 * 7^{(3-3)}$	6037
asi	a=97 s=115 i=105	$97 * 7^{(3-1)} + 115 * 7^{(3-2)} + 105 * 7^{(3-3)}$	5663
sip	s=115 i=105 p=112	$115 * 7^{(3-1)} + 105 * 7^{(3-2)} + 112 * 7^{(3-3)}$	6482
ipe	i=105 p=112 e=101	$105 * 7^{(3-1)} + 112 * 7^{(3-2)} + 101 * 7^{(3-3)}$	6030
pen	p=112 e=101 n=110	$112 * 7^{(3-1)} + 101 * 7^{(3-2)} + 110 * 7^{(3-3)}$	6305
eng	e=101 n=110 g=103	$101 * 7^{(3-1)} + 110 * 7^{(3-2)} + 103 * 7^{(3-3)}$	5822
nga	n=110 g=103 a=97	$110 * 7^{(3-1)} + 103 * 7^{(3-2)} + 97 * 7^{(3-3)}$	6208
gam	g=103 a=97 m=109	$103 * 7^{(3-1)} + 97 * 7^{(3-2)} + 109 * 7^{(3-3)}$	5835

Kalimat (c1...ck)	Desimal ASCII	Penjabaran $c1 * b^{(k-1)} + c2 * b^{(k-2)} + \dots + ck * b^{(k-k)}$	Hasil
ama	a=97 m=109 a=97	$97*7^{(3-1)} + 109*7^{(3-2)} + 97*7^{(3-3)}$	5613
man	m=109 a=97 n=110	$109*7^{(3-1)} + 97*7^{(3-2)} + 110*7^{(3-3)}$	6130
ana	a=97 n=110 a=97	$97*7^{(3-1)} + 110*7^{(3-2)} + 97*7^{(3-3)}$	5620
nan	n=110 a=97 n=110	$110*7^{(3-1)} + 97*7^{(3-2)} + 110*7^{(3-3)}$	6179
anf	a=97 n=110 f=102	$97*7^{(3-1)} + 110*7^{(3-2)} + 102*7^{(3-3)}$	5625
nfi	n=110 f=102 i=105	$110*7^{(3-1)} + 102*7^{(3-2)} + 105*7^{(3-3)}$	6209
fil	f=102 i=105 l=108	$102*7^{(3-1)} + 105*7^{(3-2)} + 108*7^{(3-3)}$	5841
ile	i=105 l=108 e=101	$105*7^{(3-1)} + 108*7^{(3-2)} + 101*7^{(3-3)}$	6002

Setelah dilakukan perhitungan diperoleh hasil dari *rolling hash* “5645 6349 6134 5991 6037 5663 6482 6030 6305 5822 6208 5835 5613 6130 5620 6179 5625 6209 5841 6002”.

b. Pembentukan *Window*

Setelah didapatkan hasil perhitungan *rolling hash*, maka tahap selanjutnya adalah pembentukan *window* dari nilai *hash* yang didapat. Kemudian nilai *hash* dikelompokkan sebanyak nilai *w-gram* atau *w=4*. Berikut contoh pembentukan *window*.

1) *Dataset*

Berikut pembentukan *window* pada *dataset* dengan *w=4*.

Nilai *hash* :

6020 | 6233 | 6345 | 6108 | 5813 | 6158 | 5835 | 6299 | 6478 | 5663  
 | 6482 | 6030 | 6305 | 5822 | 6208 | 5835 | 5613 | 6130 | 5620 |  
 6179 | 5625 | 6209 | 5841 | 6002

*Window* (w=4) :

**6020** | 6233 | 6345 | 6108  
**5813** | 6158 | 5835 | 6299  
 6478 | **5663** | 6482 | 6030  
 6305 | **5822** | 6208 | 5835  
**5613** | 6130 | 5620 | 6179

**5625** | 6209 | 5841 | 6002

2) Data Tes

Berikut pembentukan *window* pada data tes dengan  $w=4$ .

Nilai *hash* :

5645 | 6349 | 6134 | 5991 | 6037 | 5663 | 6482 | 6030 | 6305 | 5822  
| 6208 | 5835 | 5613 | 6130 | 5620 | 6179 | 5625 | 6209 | 5841 |  
6002

*Window* ( $w=4$ ) :

**5645** | 6349 | 6134 | 5991  
6037 | **5663** | 6482 | 6030  
6305 | **5822** | 6208 | 5835  
**5613** | 6130 | 5620 | 6179  
**5625** | 6209 | 5841 | 6002

c. Pencarian Fingerprint

Setelah pembentukan *window* selesai, maka tahap selanjutnya adalah pencarian *fingerprint* dari setiap *window* yang ada. Nilai *fingerprint* ditentukan berdasarkan nilai terkecil dari setiap *window* yang ada. Berikut hasil dari *fingerprint* dari contoh sebelumnya.

1) *Dataset*

Berikut hasil pencarian *fingerprint* dari *dataset*:

6020 | 5813 | 5663 | 5822 | 5613 | 5625

2) Data Tes

Berikut hasil pencarian *fingerprint* dari data tes:

5645 | 5663 | 5822 | 5613 | 5625

4.2.5. Jaccard Similarity

Setelah *fingerprint* dari dokumen ditemukan langkah selanjutnya adalah perhitungan *similarity* dari *fingerprint* yang ada menggunakan *jaccard Similarity*. Berikut hasil setiap *fingerprint* ada pada Tabel 4.12 berikut.

**Tabel 4.12 Hasil Fingerprint**

Keterangan	Contoh Kalimat	Fingerprint
<i>Dataset</i>	implementasipengaman anfile	6020   5813   5663   5822   5613   5625
Data Tes	aplikasipengamananfile	5645   5663   5822   5613   5625

Berikut adalah rumus *jaccard similarity* :

$$\text{Similarity (X,Y)} = \frac{|x \cap y|}{|x \cup y|} \times 100\%$$

$$X = \{6020,5813,5663,5822,5613,5625\}$$

$$Y = \{5645,5663,5822,5613,5625\}$$

$$X \cap Y = \{5663,5822,5613,5625\}$$

$$X \cup Y = \{6020,5813,5663,5822,5613,5625,5645\}$$

$$(X,Y) = \frac{|x \cap y|}{|x \cup y|} \times 100\%$$

$$(X,Y) = \frac{4}{7} \times 100\% = 0,571 \times 100\% = \mathbf{57,1\%}$$

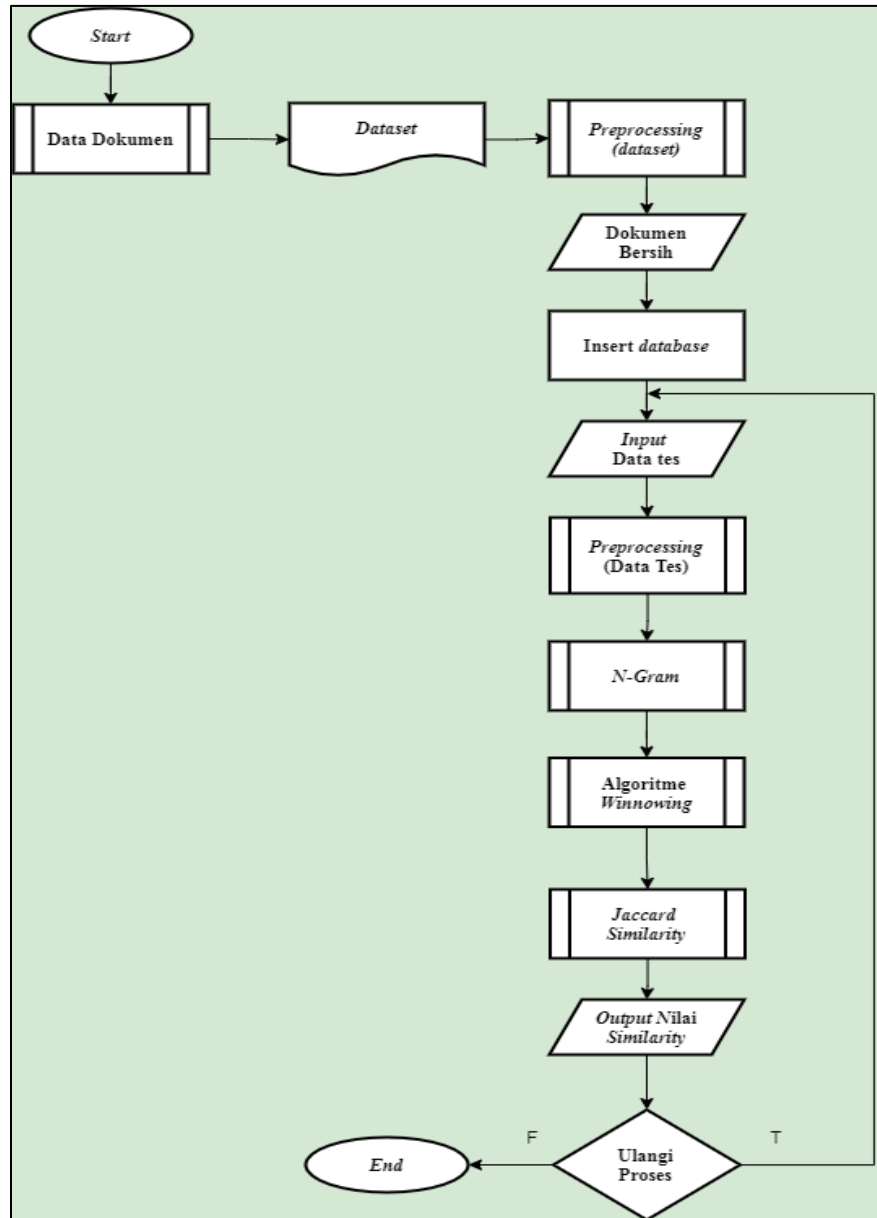
Sehingga tingkat *similarity* dari dokumen adalah 57,1%

#### **4.3. Flowchart Tahapan Metode**

*Flowchart* adalah suatu bagan atau simbol-simbol yang menggambarkan alur kerja atau urutan proses pada suatu program. Berikut adalah penjabaran *flowchart* pada tahapan metode yang digunakan:

##### **4.3.1. Flowchart Keseluruhan Sistem**

Pada *flowchart* ini menjelaskan tahapan mengenai berjalannya sistem, mulai dari pengumpulan data dokumen, hingga mendapatkan hasil tingkat *similarity* dari sebuah dokumen.



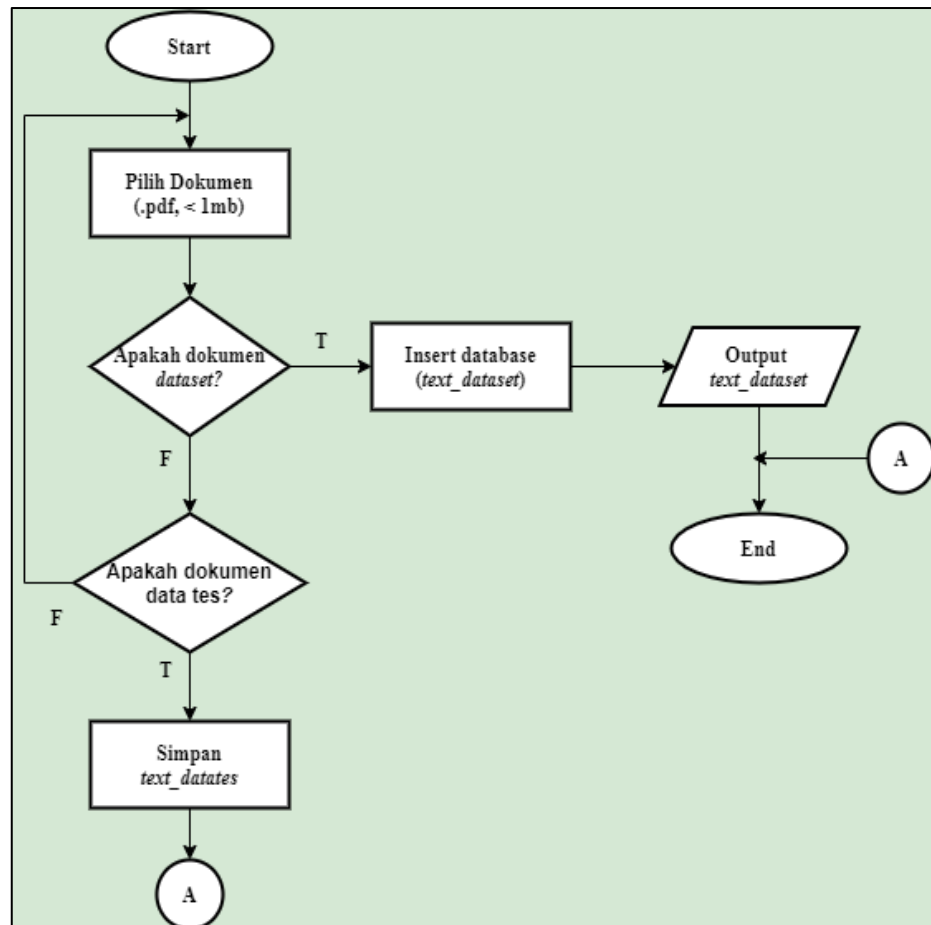
**Gambar 4.1 Flowchart Keseluruhan Sistem**

Pada gambar 4.1 menjelaskan proses keseluruhan sistem yang dibuat dengan tahap awal yaitu proses pengumpulan data dokumen, kemudian sebelum masuk ke tahapan *n-gram*, dokumen yang merupakan *dataset* atau data tes ini dilakukan tahapan pembersihan data atau *preprocessing*, setelah menjadi dokumen bersih, selanjutnya dilakukan proses tahapan utama *n-gram*, pembentukan *hash* melalui algoritme *winnowing*, perhitungan *similarity* dengan *jaccard similarity* hingga menghasilkan output, berupa tingkat *similarity*. Proses yang sama akan berulang jika dimasukkan dokumen untuk dijadikan *dataset* atau data tes.



#### 4.3.2. Flowchart Data Dokumen

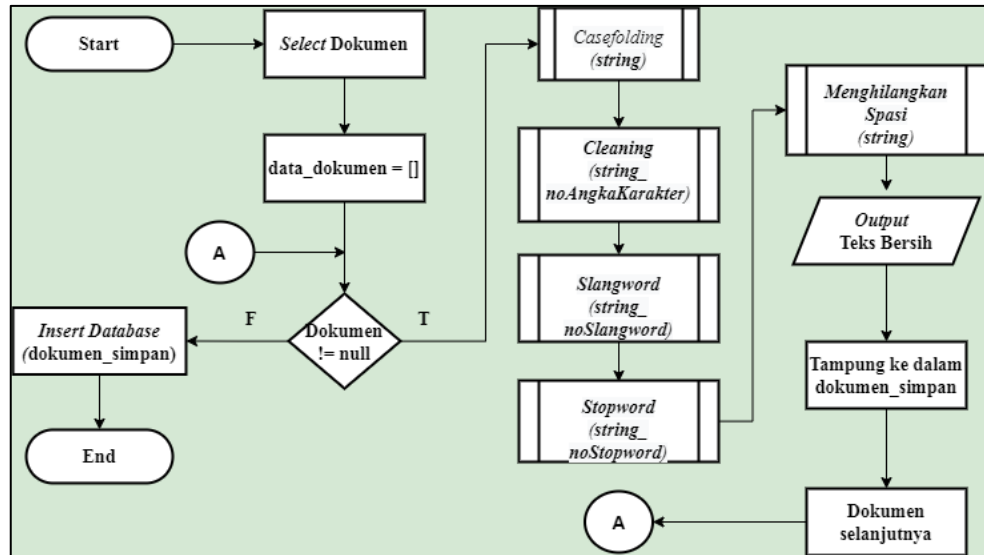
Pada *flowchart* ini, menjelaskan tahapan proses pengumpulan data dokumen *dataset* atau data tes. Dokumen yang bisa diproses berupa file *.pdf* serta berukuran <1 mb, setelah itu dokumen yang merupakan *dataset* akan disimpan kedalam *database*, sedangkan dokumen data tes akan disimpan di sebuah array *text\_datates* untuk proses selanjutnya, yaitu pencarian nilai *similarity* dokumen. Proses yang sama akan berulang jika ingin memproses dokumen kembali.



**Gambar 4.2 Flowchart Data Dokumen**

#### 4.3.3. Flowchart Preprocessing Dataset

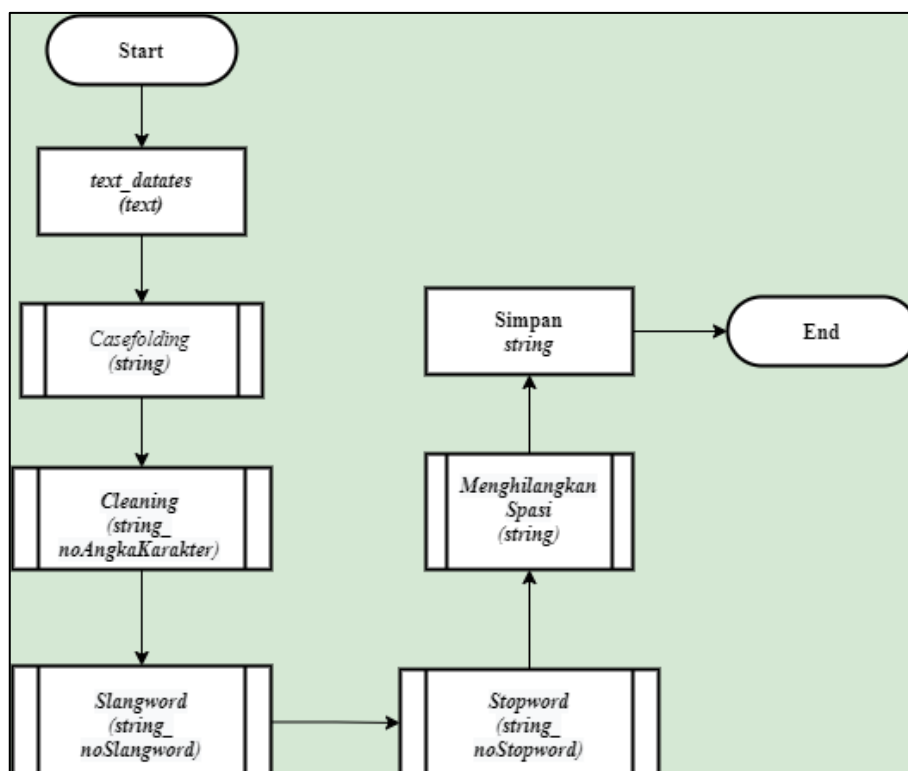
Pada *flowchart* ini, menjelaskan tahapan *preprocessing dataset*. Data Dokumen abstrak yang sudah disimpan di *database* yang merupakan *dataset* akan di *select*, kemudian disimpan di sebuah *array*, selanjutnya akan dilakukan proses *case folding*, *cleaning*, mengganti *slang word*, menghapus *stop word*, dan menghilangkan spasi antar kata, kemudian ditampung di sebuah *array* dan diinsert ke dalam *database*. Hasil dari tahapan ini berupa teks bersih, yang digunakan untuk proses pencarian nilai *similarity*.



**Gambar 4.3 Flowchart Preprocessing Dataset**

#### 4.3.4. Flowchart Preprocessing Data Tes

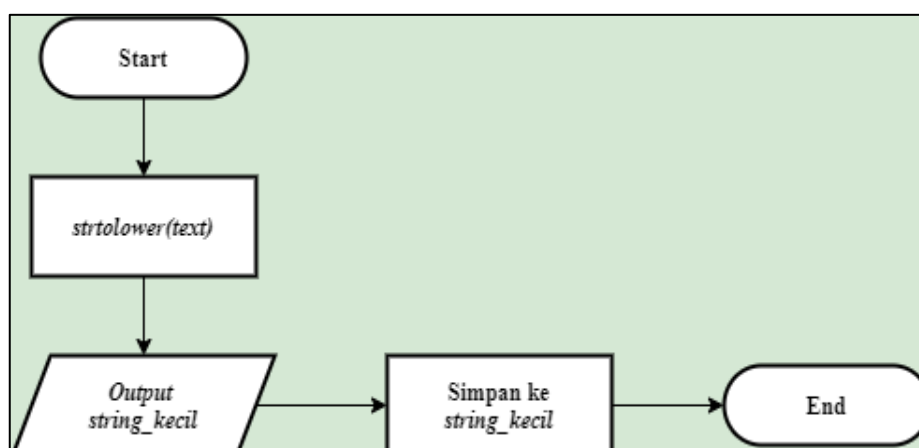
Pada flowchart ini, menjelaskan tahapan *preprocessing* data tes. Data Dokumen abstrak yang akan dijadikan data tes disimpan di sebuah *array text*, kemudian isi dari *array text* akan dilakukan proses *case folding*, *cleaning*, mengganti *slang word*, menghapus *stop word*, dan menghilangkan spasi antar kata, kemudian ditampung di sebuah *array string*. Selanjutnya digunakan untuk proses pencarian nilai *similarity*.



**Gambar 4.4 Flowchart Preprocessing Data Tes**

#### 4.3.5. Flowchart Case folding

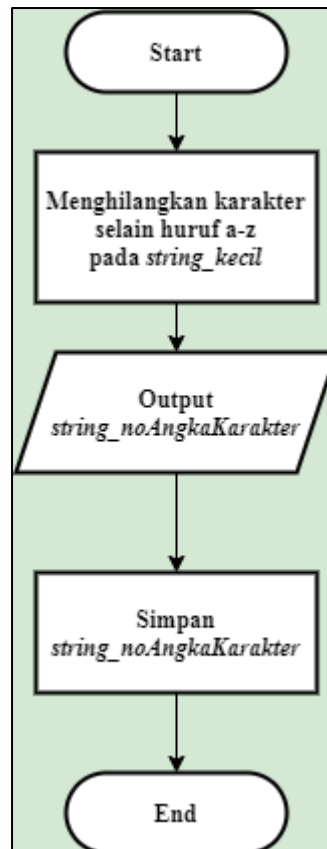
Pada *flowchart* ini, menjelaskan sub proses tahapan dari *preprocessing* yaitu *case folding*. Pada tahapan ini isi dokumen berupa teks akan dilakukan proses *case folding*, yaitu proses merubah isi teks menjadi huruf kecil, kemudian disimpan didalam *array string\_kecil*, yang akan digunakan untuk proses selanjutnya.



**Gambar 4.5 Flowchart Case folding**

#### 4.3.6. Flowchart Cleaning

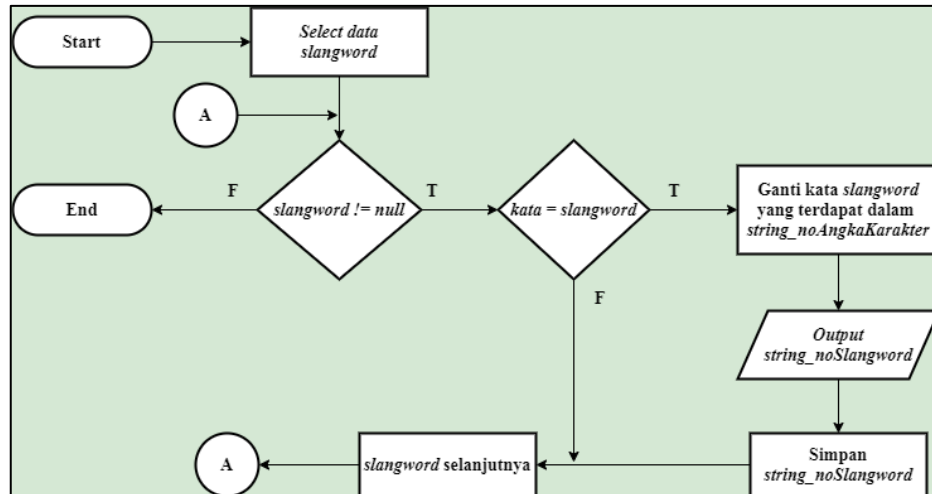
Pada *flowchart* ini, menjelaskan sub proses tahapan dari *preprocessing* yaitu *cleaning*. Pada tahapan ini isi teks yang telah diproses pada tahap *case folding* akan diproses lagi melalui proses *cleaning*, yaitu menghilangkan karakter selain huruf a-z yang ada di dalam array *string\_kecil*, hasil dari proses ini disimpan didalam array *string\_noAngkaKarakter*, yang akan digunakan untuk proses selanjutnya.



**Gambar 4.6 Flowchart Cleaning**

#### 4.3.7. Flowchart Slang word

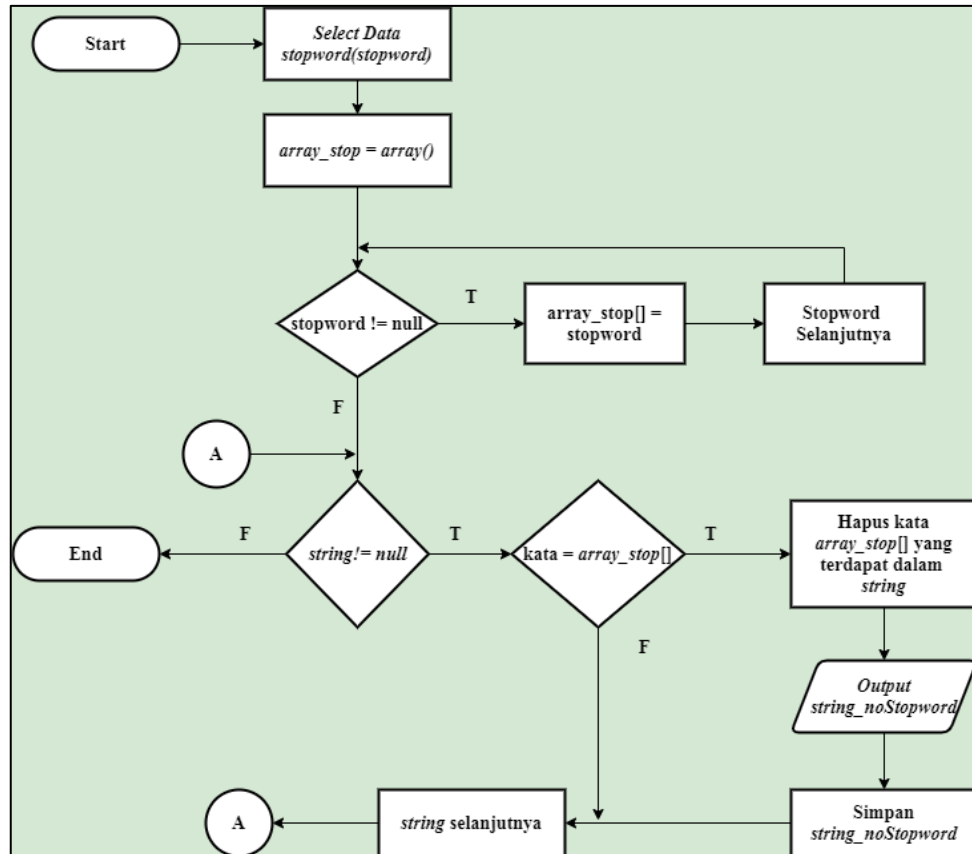
Pada *flowchart* ini, menjelaskan sub proses tahapan dari *preprocessing* yaitu *slang word*. Proses *slang word* diawali dengan *select* data *slang word* yang ada di *database*, kemudian akan diperiksa teks yang ada di array *string\_noAngkaKarakter*, apakah ada kata yang sama dengan data *slang word*, jika menemukan kata yang sama maka kata tersebut akan diganti dengan data yang ada di data *slang word*, selanjutnya teks yang sudah diproses disimpan didalam array *string\_noSlangword*, yang akan digunakan untuk proses selanjutnya.



**Gambar 4.7 Flowchart Slang Word**

#### 4.3.8. Flowchart Stop word

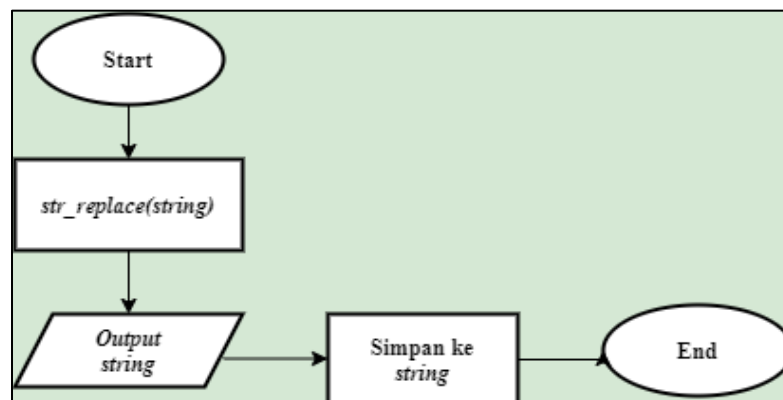
Pada flowchart ini, menjelaskan sub proses tahapan dari preprocessing yaitu stop word. Proses stop word diawali dengan select data stop word yang ada di database, kemudian akan diperiksa teks yang ada di array string\_noSlangword, apakah ada kata yang sama dengan data stop word, jika menemukan kata yang sama maka kata tersebut akan dihapus dan teks yang masih ada disimpan didalam array string\_noStopword, yang akan digunakan untuk proses selanjutnya.



**Gambar 4.8 Flowchart Stop Word**

#### 4.3.9. Flowchart Hapus Spasi

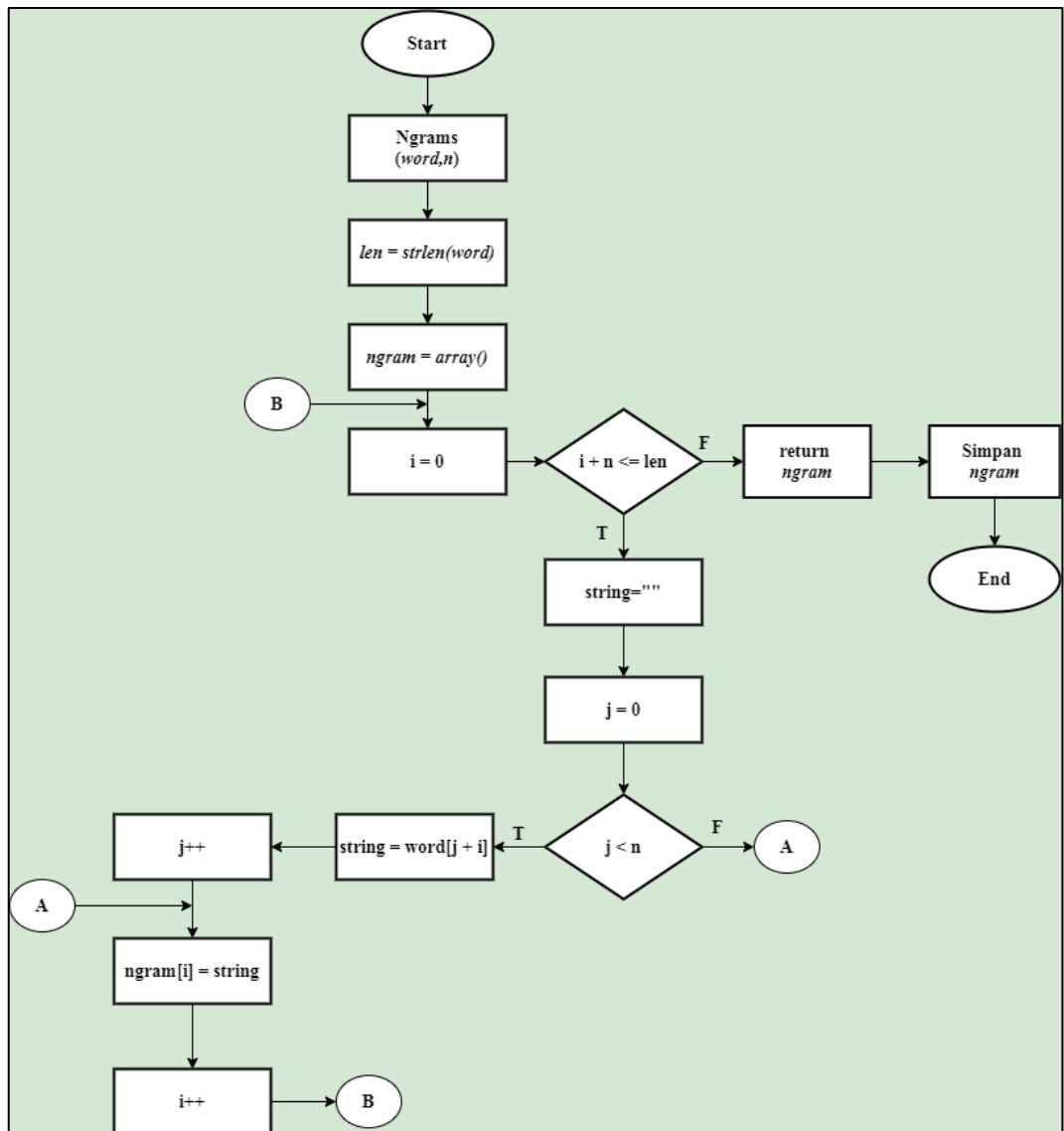
Pada *flowchart* ini, menjelaskan sub proses tahapan dari *preprocessing* yaitu hapus spasi. Pada proses ini teks yang sudah tersimpan di dalam *array string\_noStopword* akan dihapus spasi antar katanya, sehingga teks ini menjadi satu, sehingga menghasilkan teks bersih, kemudian disimpan kedalam *array string*, yang akan digunakan untuk proses selanjutnya.



**Gambar 4.9 Flowchart Hapus Spasi**

#### 4.3.10. Flowchart N-Gram

Pada *flowchart* ini, menjelaskan tahapan dari metode *n-gram*, pada tahap ini teks yang sudah di *preprocessing* yang menghasilkan teks bersih akan diproses melalui *n-gram*, yaitu dengan memisahkan *string* sepanjang jumlah *n* yang sudah ditentukan dan akan dihitung pergeserannya secara terus menerus ke depan sejumlah nilai *n* sampai akhir isi teks. Kemudian hasilnya akan disimpan didalam *ngram*, hasil *n-gram* ini akan dicari nilai hasnya pada proses selanjutnya.

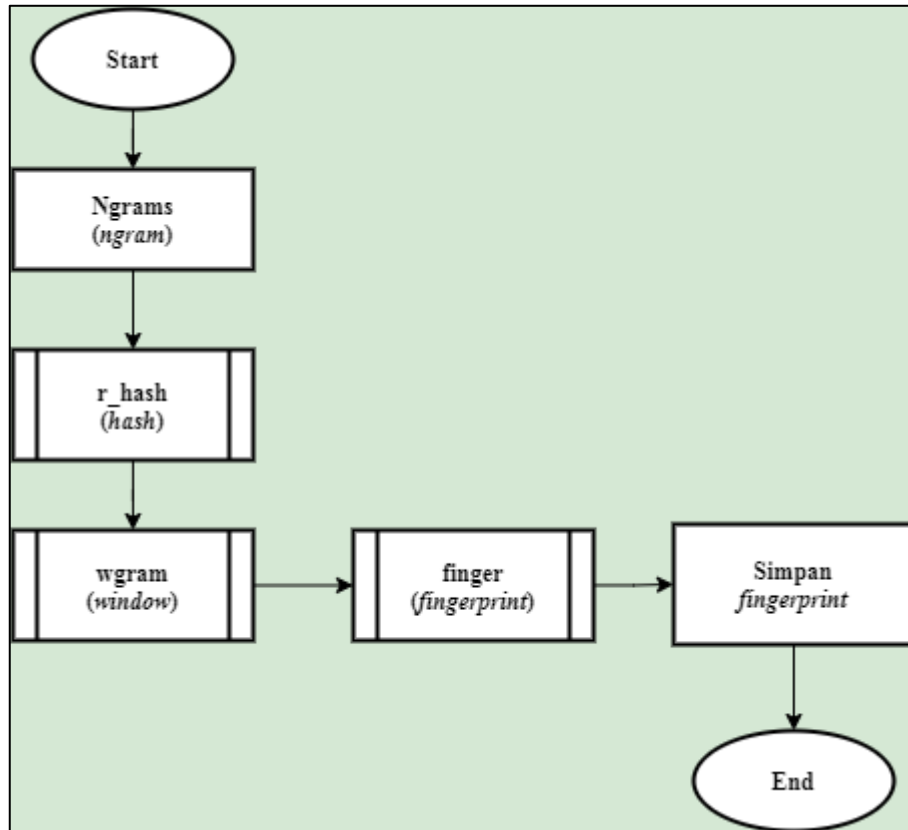


**Gambar 4.10 Flowchart N-Gram**

#### 4.3.11. Flowchart Algoritme Winnowing

Pada *flowchart* ini, menjelaskan tahapan dari algoritme *winnowing* untuk mencari nilai *hash* dari sebuah dokumen. Setelah dokumen yang diproses pada tahap *n-gram*, kemudian carilah nilai *hash* dari teks

tersebut, melalui beberapa sub proses *rolling hash*, pembentukan *window*, serta pencarian *fingerprint*. Setelah mendapatkan nilai *hash* dari dokumen yang diproses, maka setelah itu dapat dilakukan proses pencarian tingkat *similarity* dari sebuah dokumen abstrak.

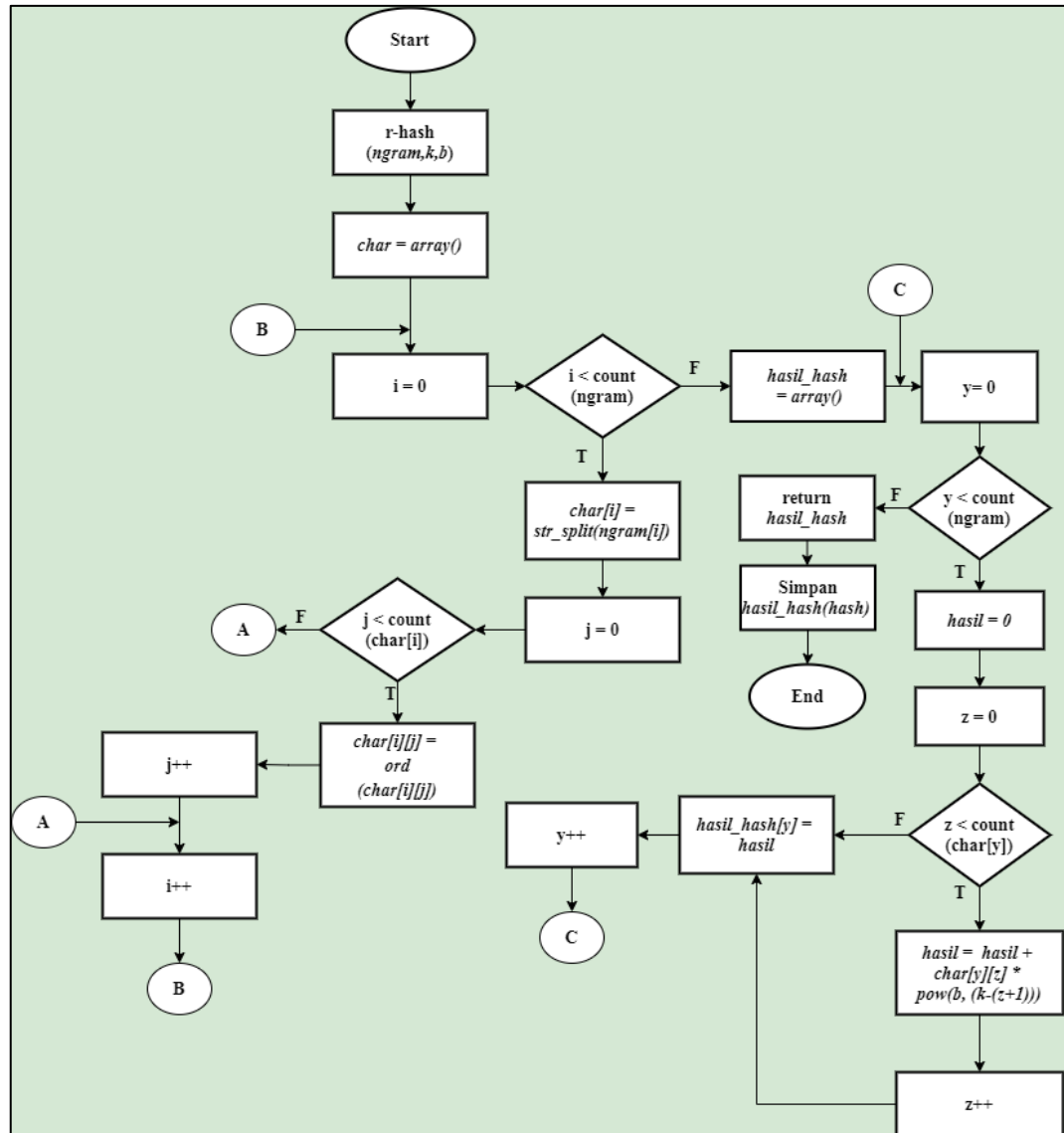


**Gambar 4.11 Flowchart Algoritme Winnowing**

#### 4.3.12. Flowchart Rolling Hash

Pada *flowchart* ini, menjelaskan tahapan dari *rolling hash* yang merupakan sub proses dari algoritme *winnowing*. Proses pertama yang dilakukan adalah memanggil isi teks pada *array ngram*, yang sudah diproses di tahap sebelumnya, maka proses selanjutnya mencari nilai ASCII dari setiap huruf yang dipecah, kemudian ditampung di *array char*, setelah mendapatkan nilai ASCII dari setiap huruf yang akan diproses, maka proses selanjutnya mencari nilai *hash* dari teks tersebut, kemudian nilai *hash* tadi akan dilakukan pembentukan *window* pada tahap selanjutnya.

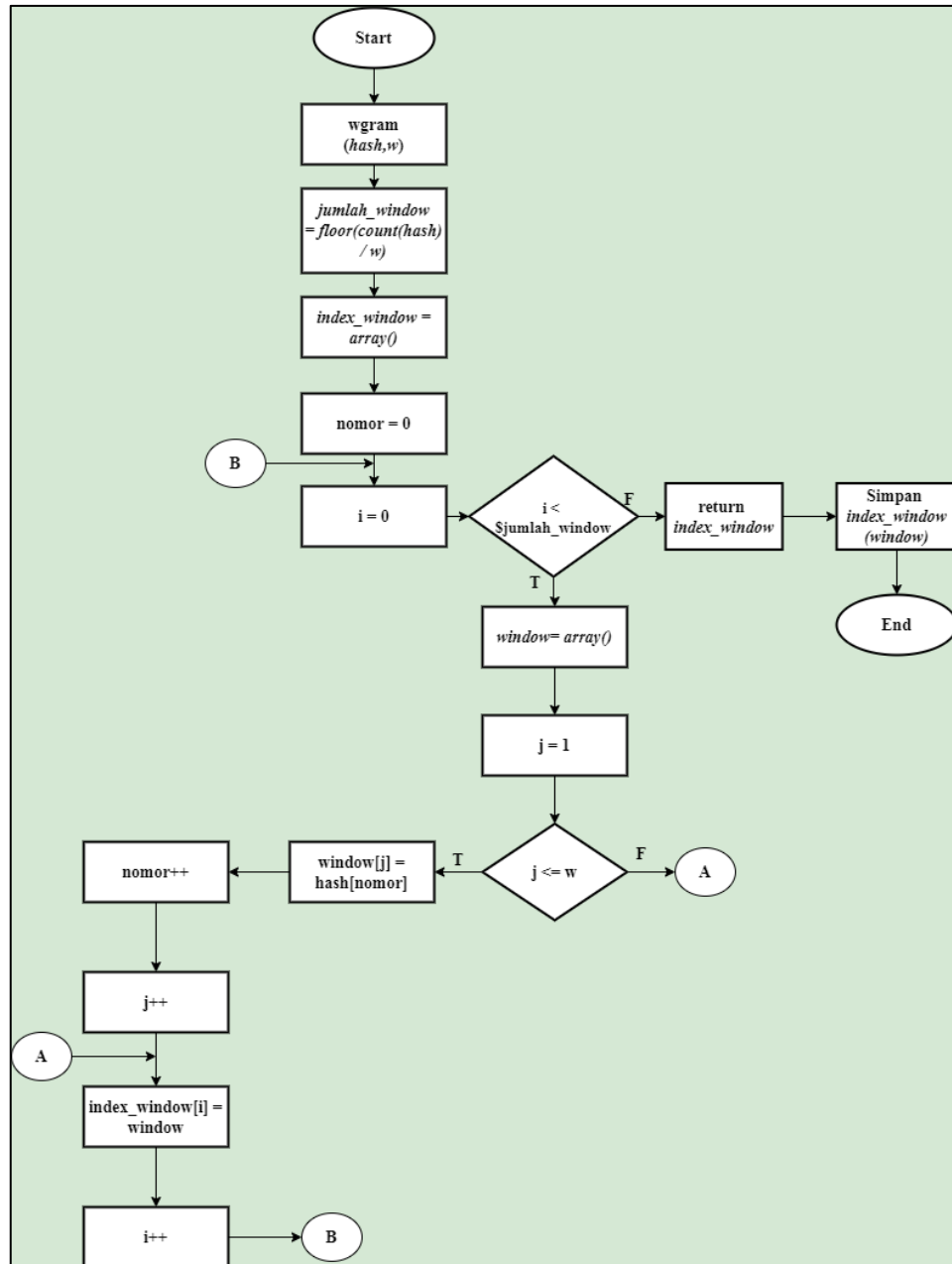




**Gambar 4.12 Flowchart Rolling Hash**

#### 4.3.13. Flowchart Window

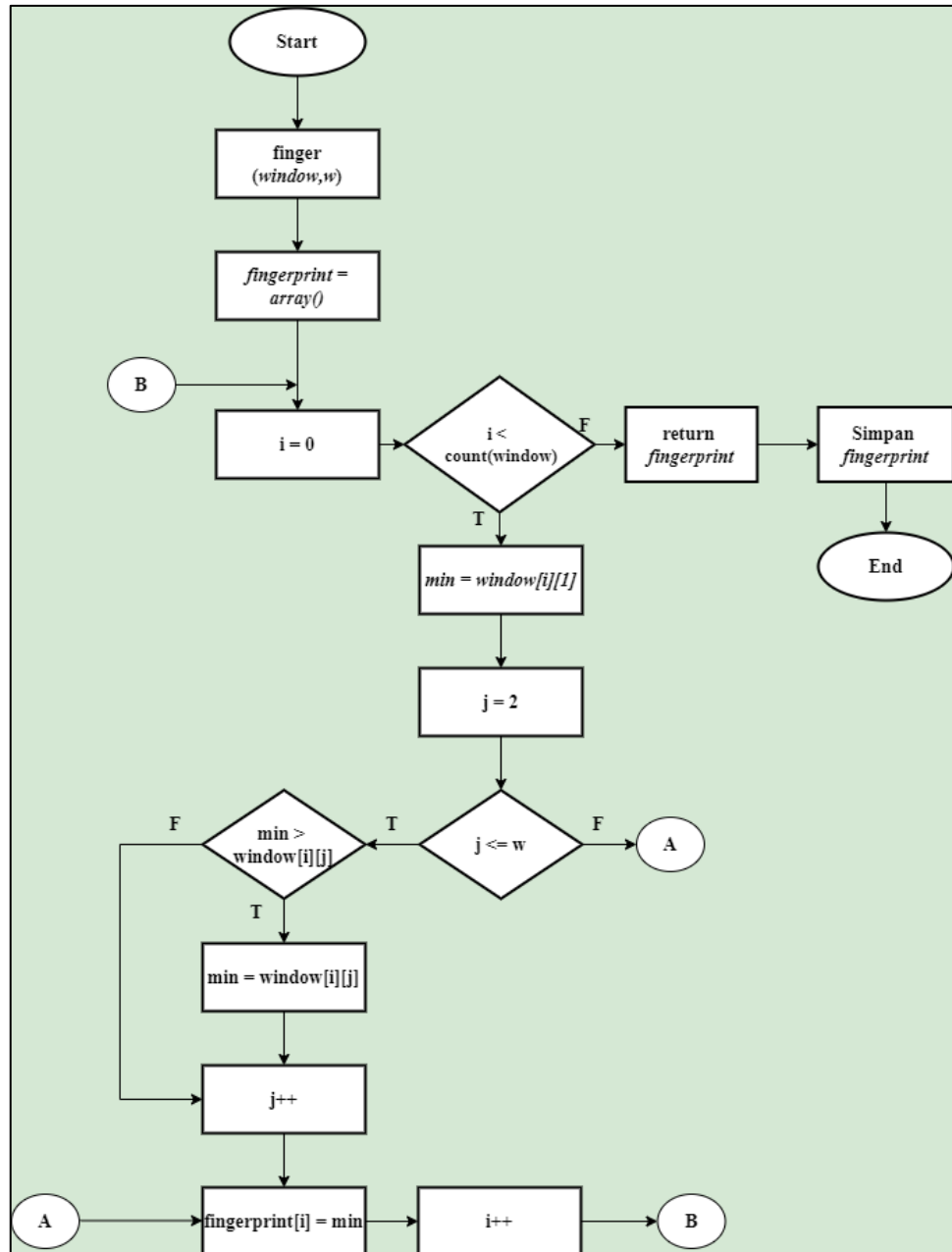
Pada *flowchart* ini, menjelaskan tahapan pembentukan *window* dari nilai *hash* yang sudah diperoleh di tahap sebelumnya, selanjutnya tentukan nilai *window* yang akan dibentuk, setelah itu nilai *hash* tadi akan dikelompokkan sesuai dengan *window* yang sudah ditentukan. Kemudian hasil dari *window* akan disimpan didalam *array window*, yang akan digunakan untuk proses selanjutnya.



**Gambar 4.13 Flowchart Window**

#### 4.3.14. Flowchart Fingerprint

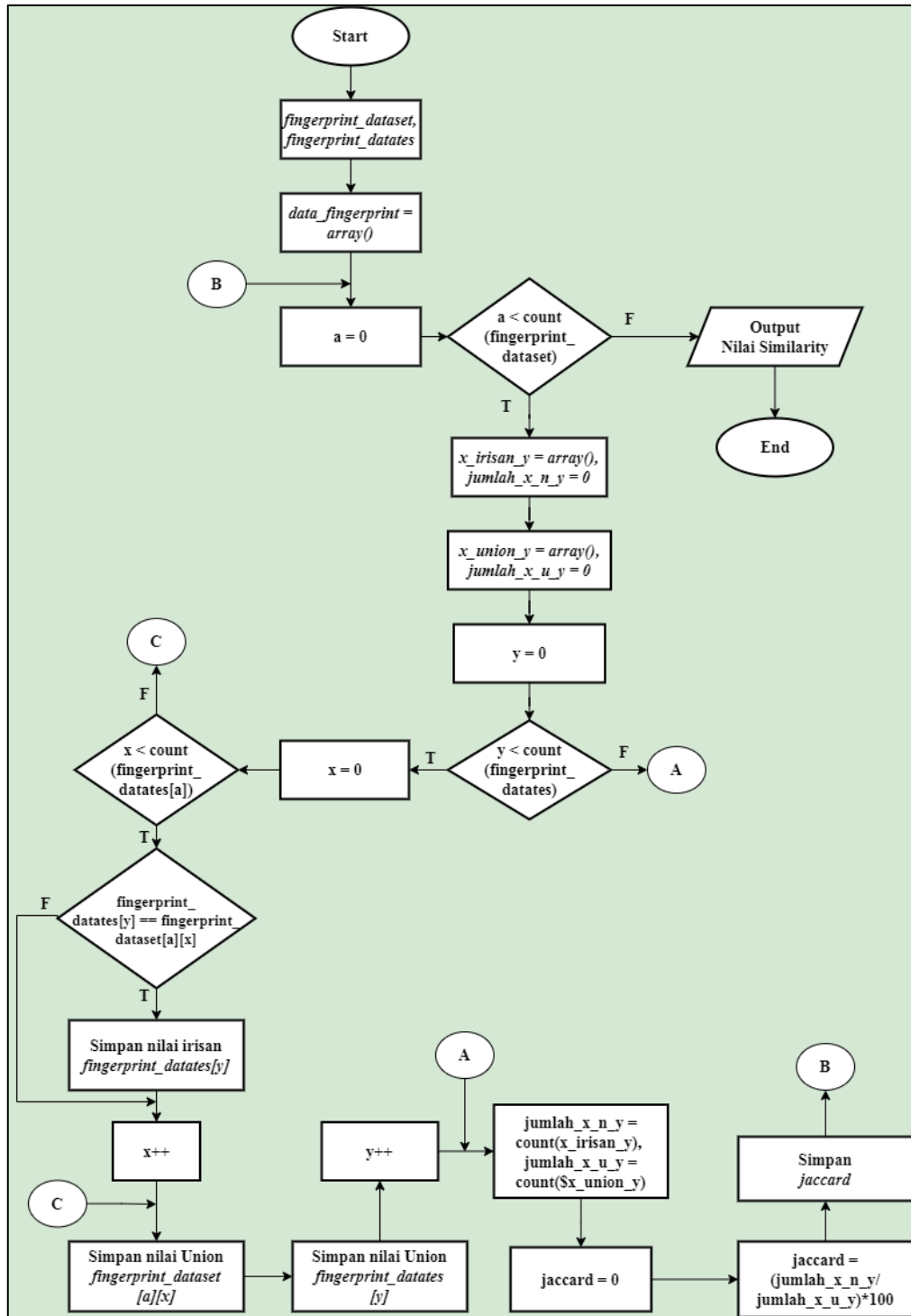
Pada *flowchart* ini, menjelaskan tahapan pencarian nilai *fingerprint* dari proses sebelumnya. Setelah dilakukan pembentukan *window*, maka tahap selanjutnya mencari nilai terkecil dari setiap *window* yang dibentuk, sehingga didapatkanlah nilai *fingerprint* dari setiap *window* yang telah dibentuk. Nilai *fingerprint* ini akan digunakan di tahap *jaccard similarity*, yaitu metode yang digunakan untuk mencari nilai *similarity* dari sebuah dokumen.



**Gambar 4.14** *Flowchart Fingerprint*

#### 4.3.15. *Flowchart Jaccard Similarity*

Pada *flowchart* ini, menjelaskan tahapan dari *jaccard similarity*. Tahapan ini merupakan tahapan untuk mencari persentase *similarity* dari sebuah dokumen. Setelah didapatkan nilai *fingerprint* pada tahap sebelumnya, maka pada proses selanjutnya adalah membandingkan nilai *fingerprint* dari dokumen *dataset* dengan dokumen data tes, sehingga dapat diperoleh persentase *similarity* dari dokumen yang telah diproses.



**Gambar 4.15 Flowchart Jaccard Similarity**

#### 4.4. Algoritme Tahapan Metode

Algoritme adalah urutan atau alur tahapan proses yang dijabarkan dalam bentuk tulisan, algoritme ini merupakan representasi dari flowchart yang telah dijelaskan sebelumnya.

##### 4.4.1. Algoritme Keseluruhan Sistem

Pada algoritme ini dijelaskan tentang proses sistem secara keseluruhan pada metode yang digunakan.

##### Algoritme 4.1 Keseluruhan Sistem

```
1. Start
2. Lakukan proses data dokumen
3. Baca dokumen dataset
4. Lakukan preprocessing dataset
5. Hasil dokumen bersih
6. Insert database
7. Input data tes
8. Lakukan preprocessing data tes
9. Lakukan n-gram
10. Lakukan algoritme winnowing
11. lakukan jaccard similarity
12. Output nilai similarity
13. if (ulangi proses)
14. kembali ke nomor 7
15. else
16. Endif
17. End
```

##### 4.4.2. Algoritme Data Dokumen

Pada algoritme ini dijelaskan tentang proses tahapan pengolahan dokumen yang akan diproses.

##### Algoritme 4.2 Data Dokumen

```
1. Start
2. Proses dokumen : .pdf,<1mb
3. if (dokumen dataset?)
4. insert database text_dataset
5. Output : dataset
6. else
7. if (dokumen data tes?)
8. Simpan text_datates
9. else
10. Endif
11. End
```

##### 4.4.3. Algoritme *Preprocessing Dataset*

Pada algoritme ini dijelaskan tentang proses tahapan pada *preprocessing dataset* secara keseluruhan.

### **Algoritme 4.3 Preprocessing Dataset**

```
1. Start
2. Select Dokumen
3. data_dokumen = []
4. if(Dokumen != null)
5.     Lakukan proses Casefolding (string)
6.     Lakukan proses Cleaning (noAngkaKarakter)
7.     Lakukan proses Slangword (string_noSlangword)
8.     Lakukan proses Stopword (string_nostopword)
9.     Lakukan proses Menghilangkan spasi(string)
10.    Output : Teks bersih
11.    Simpan string ke dalam list dokumen_simpan
12.    Dokumen selanjutnya
13.    Kembali ke nomor 4
14. else
15.     Simpan dokumen_simpan ke dalam database
16. Endif
17. End
```

#### **4.4.4. Algoritme Preprocessing Data Tes**

Pada algoritme ini dijelaskan tentang proses tahapan pada *preprocessing* data tes secara keseluruhan.

### **Algoritme 4.4 Preprocessing Data Tes**

```
1. Start
2. Proses text_datates
3. Lakukan proses Casefolding (string)
4. Lakukan proses Cleaning (noAngkaKarakter)
5. Lakukan proses Slangword (string_noSlangword)
6. Lakukan proses Stopword (string_nostopword)
7. Lakukan proses Menghilangkan spasi(string)
8. Simpan di variabel string
9. End
```

#### **4.4.5. Algoritme Case folding**

Pada algoritme ini dijelaskan tentang proses sub tahapan pada *preprocessing* yaitu *case folding*.

### **Algoritme 4.5 Case folding**

```
1. Start
2. Proses strtolower(text)
3. Output : string_kecil
4. Simpan string_kecil
5. End
```

#### **4.4.6. Algoritme Cleaning**

Pada algoritme ini dijelaskan tentang proses sub tahapan pada *preprocessing* yaitu *cleaning* dokumen.

#### **Algoritme 4.6 Cleaning**

```
1. Start
2. Menghilangkan karakter selain huruf a-z pada
   string_kecil
3. Output : string_noAngkaKarakter
4. Proses simpan string_noAngkaKarakter
5. End
```

#### **4.4.7. Algoritme Slang word**

Pada algoritme ini dijelaskan tentang proses sub tahapan pada *preprocessing* yaitu *slang word*.

#### **Algoritme 4.7 Slang word**

```
1. Start
2. Select data slangword dari database
3. if (slangword!=null)
4.     if (kata == slangword)
5.         Ubah dengan kata slangword yang terdapat
           dalam string_noAngkaKarakter menjadi kata
           asli
6.         Output string_noSlangword
7.         Simpan string_noSlangword
8.     else
9.         slangword selanjutnya
10.        Kembali ke nomor 3
11.    Endif
12. Endif
13. End
```

#### **4.4.8. Algoritme Stop word**

Pada algoritme ini dijelaskan tentang proses sub tahapan pada *preprocessing* yaitu *slang word*.

#### **Algoritme 4.8 Stop word**

```
1. Start
2. Select data stopword dari database
3. siapkan array_stop = array()
4. if (stopword != null)
5.     array_stop[] = stopword
6.     stopword selanjutnya
7.     kembali ke nomor 4
8. else
9.     if (string != null)
10.        if(kata = array_stop[])
11.        Hapus kata array_stop[] yang terdapat dalam
           string
12.        Otrput : string_noStopword
13.        Simpan string_noStopword
14.        string selanjutnya
```

```
15.     kembali ke nomor 9
16.     Endif
17. Endif
18. End
```

#### 4.4.9. Algoritme Hapus Spasi

Pada algoritme ini dijelaskan tentang proses sub tahapan pada *preprocessing* yaitu menghapus spasi pada sebuah teks dokumen.

##### Algoritme 4.9 Hapus Spasi

```
1. Start
2. str_replace(string)
3. Output : string
4. Simpan string
5. End
```

#### 4.4.10. Algoritme *N-Gram*

Pada algoritme ini dijelaskan tentang proses tahapan utama pada algoritme *n-gram*.

##### Algoritme 4.9 *N-Gram*

```
1. Start
2. Proses Ngrams(word,n)
3. len = strlen(word)
4. ngram = array()
5. i = 0
6. if (i + n <= len)
7.     string = ""
8.     j=0
9.     if(j < n)
10.         string = word[j+i]
11.         j++
12.     else
13.         ngram[i] = string
14.         i++
15.         kembali ke nomor 5
16.     Endif
17. else
18. return ngram
19. Simpan ngram
20. Endif
21. End
```

#### 4.4.11. Algoritme *Winnowing*

Pada algoritme ini dijelaskan tentang proses tahapan utama pada algoritme *winnowing* yaitu *rolling hash*, pembentukan *window* dan pencarian *fingerprint*.



#### Algoritme 4.11 *Winnowing*

```
1. Start
2. Proses Ngrams(ngram)
3. Lakukan proses r_hash (hash)
4. Lakukan proses wgram (window)
5. Lakukan prose finger (fingerprint)
6. Simpan fingerprint
7. End
```

#### 4.4.12. Algoritme *Rolling Hash*

Pada algoritme ini dijelaskan tentang proses sub tahapan pada algoritme *winnowing* yaitu *rolling hash*.

#### Algoritme 4.12 *Rolling Hash*

```
1. Start
2. Proses r_hash(ngram, k, b)
3. char = array()
4. i = 0
5. if (i < count(ngram))
6.     char[i] = str_split(ngram[i])
7.     j = 0
8.     if (j < count(char[i]))
9.         char[i][j] = ord(char[i][j])
10.        j++
11.    else
12.        i++
13.        kembali ke nomor 4
14. else
15.     hasil hash = array()
16.     y = 0
17.     if (y < count(ngram))
18.         hasil = 0
19.         z = 0
20.         if (z < count(char[y]))
21.             hasil = hasil + char[y][z] * pow(b, (k-
(z+1)))
22.             z++
23.         else
24.             hasil_hash[y] = hasil
25.             y++
26.             kembali ke nomor 16
27.         Endif
28.     else
29.         return hasil_hash
30.     Simpan hasil_hash(hash)
31. Endif
32. End
```

#### 4.4.13. Algoritme *Window*

Pada algoritme ini dijelaskan tentang proses sub tahapan pada algoritme *winnowing* yaitu pembentukan *window*.

### Algoritme 4.13 Window

```
1. Start
2. Proses wgram (hash,w)
3. jumlah_window = floor(count(hash)/w)
4. index_window = array()
5. nomor = 0
6. i = 0
7. if (i < jumlah_window)
8.     window = array()
9.     j = 1
10.    if (j <= w)
11.        window[j] = hash[nomor]
12.        nomor++
13.        j++
14.    else
15.        index_window[i] = window
16.        i++
17.        Kembali ke nomor 6
18.    Endif
19. else
20. return index_window
21. Simpan index_window(window)
22. Endif
23. End
```

#### 4.4.14. Algoritme Fingerprint

Pada algoritme ini dijelaskan tentang proses sub tahapan pada algoritme *winnowing* yaitu pencarian nilai *fingerprint*.

### Algoritme 4.14 Fingerprint

```
1. Start
2. Proses finger (window,w)
3. fingerprint = array()
4. i = 0
5. if (i < count(window))
6.     min = window[i][1]
7.     j = 2
8.     if (j <= w)
9.         if(min > window[i][j])
10.            min = window[i][j]
11.        else
12.            j++
13.    else
14.        fingerprint[i] = min
15.        i++
16.        Kembali ke nomor 4
17.    Endif
18. else
19. return fingerprint
20. Simpan fingerprint
21. Endif
22. End
```

#### 4.4.15. Algoritme Jaccard Similarity

Pada algoritme ini dijelaskan tentang proses tahapan metode *jaccard similarity*.

#### Algoritme 4.15 Jaccard Similarity

```
1. Start
2. fingerprint_dataset, fingerprint_datates
3. data_fingerprint = array()
4. a = 0
5. if (a < count(fingerprint_dataset)
6.     x_irisan_y = array(), jumlah_x_n_y = 0
7.     x_union_y = array(), jumlah_x_u_y = 0
8.     y = 0
9.     if (y < count(fingerprint_datates)
10.        x = 0
11.        if (x < count(fingerprint_datates[a])
12.            if (fingerprint_datates[y] ==
                fingerprint_dataset[a][x]
13.                Simpan nilai irisan
                    fingerprint_datates[y]
14.            else
15.                x++
16.            else
17.                Simpan nilai union fingerprint_dataset[a][x]
18.                Simpan nilai fingerprint_dataset[y]
19.                y++
20.        else
21.            jumlah_x_n_y = count(x_irisan_y), jumlah_x_u_y =
                count(x_union_y)
22.            jaccard = 0
23.            jaccard = (jumlah_x_n_y/jumlah_x_u_y)*100
24.            Simpan jaccard
25.            Kembali ke nomor 4
26.        Endif
27.    else
28.        Output : Nilai Similarity
29.    Endif
30. End
```

### 4.5. Pengujian

Pengujian merupakan salah satu hal yang perlu dilakukan dalam setiap pengembangan sistem untuk mengevaluasi, menganalisa dan mengetahui tingkat akurasi atau kesamaan hasil yang telah dicapai oleh sistem yang telah dirancang. Pengujian dilakukan dengan beberapa cara yaitu, pertama pengujian dengan nilai *k-gram* dan *w-gram* yang sama, kedua pengujian dengan nilai *k-gram* dan *w-gram* yang berbeda, dan terakhir pengujian dengan satu *dataset* sebagai data tes.

#### 4.5.1. Contoh Pengujian

- a. Pengujian dengan nilai *k-gram* dan *w-gram* yang sama

Pengujian menggunakan 13 dokumen abstrak yang semua data dokumen nya berbeda sebagai *dataset*, dan data tes yang digunakan berbeda dengan *dataset*. Pada pengujian ini digunakan nilai *k-gram* dan *w-gram* dari 2 sampai 5 dimana nilainya sama pada Abstrak\_1513500346. Hasil pengujian dapat dilihat pada Tabel 4.13.

**Tabel 4.13 Uji Coba Nilai *k-gram* dan *w-gram* Sama**

No	Nim	Similarity			
		k=2 w=2	k=3 w=3	k=4 w=4	k=5 w=5
1	1511500025	46.96%	21.78%	5.84%	2.08%
2	1511500082	49.55%	24.41%	8.20%	3.31%
3	1511500132	44.95%	20.62%	8.66%	3.32%
4	1511500157	46.53%	25.15%	6.95%	4.88%
5	1511500199	50.51%	21.39%	7.11%	3.19%
6	1511500207	45.54%	23.93%	8.06%	5.26%
7	1511500215	44.66%	19.23%	7.65%	3.64%
8	1511500231	50.96%	21.43%	8.02%	1.96%
9	1511500249	51.40%	20.51%	8.33%	3.74%
10	1511500264	40.21%	23.57%	7.95%	3.14%
11	1511500272	47.66%	20.53%	7.48%	3.59%
12	1511500298	49.49%	24.24%	8.84%	3.64%
13	1511500314	43.33%	22.47%	6.55%	2.36%
Persentase Similarity Terbesar		51.40%	25.15%	8.84%	5.26%

b. Pengujian dengan nilai *k-gram* dan *w-gram* yang berbeda

Pengujian menggunakan 13 dokumen abstrak yang semua data dokumen nya berbeda sebagai *dataset*, dan data tes yang digunakan berbeda dengan *dataset*. Pada pengujian ini digunakan nilai *k-gram* = 3 dan *w-gram* = 4, *k-gram* = 4 dan *w-gram* = 3, *k-gram* = 2 dan *w-gram* = 5, *k-gram* = 5 dan *w-gram* = 2, dimana nilainya berbeda pada Abstrak\_1513500346. Hasil pengujian dapat dilihat pada Tabel 4.14.

**Tabel 4.14 Uji Coba Nilai *k-gram* dan *w-gram* Berbeda**

No	Nim	Similarity			
		k=3 w=4	k=4 w=3	k=2 w=5	k=5 w=2
1	1511500025	20.59%	7.58%	35.71%	3.17%
2	1511500082	22.15%	8.59%	36.54%	3.87%
3	1511500132	19.08%	7.14%	36.54%	3.93%
4	1511500157	21.37%	8.13%	44.44%	4.42%
5	1511500199	17.39%	7.39%	38%	3.96%

No	Nim	Similarity			
		k=3 w=4	k=4 w=3	k=2 w=5	k=5 w=2
6	1511500207	23.62%	9.17%	40.43%	5.34%
7	1511500215	20.44%	6.58%	36.54%	3.97%
8	1511500231	21.64%	8.33%	35.29%	3.34%
9	1511500249	21.09%	8.39%	45.65%	4.20%
10	1511500264	26.50%	6.96%	41.86%	4.46%
11	1511500272	17.33%	6.88%	38.30%	3.66%
12	1511500298	23.26%	8.44%	44.44%	4.68%
13	1511500314	20.88%	6.92%	36.07%	4%
Persentase Similarity Terbesar		26.50%	9.17%	45.65%	5.34%

c. Pengujian dengan satu *dataset* sebagai data tes.

Pengujian menggunakan 13 dokumen abstrak yang semua data dokumen nya berbeda sebagai *dataset*, dan 1 dokumen data tes yang digunakan sama dengan *dataset*. Pada pengujian ini digunakan nilai *k-gram* dan *w-gram* dari 2 sampai 5 dimana nilainya berbeda pada Abstrak\_1511500025. Hasil pengujian dapat dilihat pada Tabel 4.15.

**Tabel 4.15 Uji Coba *Dataset* dijadikan Data Tes 1**

No	Nim	Similarity			
		k=2 w=2	k=3 w=3	k=4 w=4	k=5 w=5
1	1511500025	100%	100%	100%	100%
2	1511500082	69.17%	26.26%	10.30%	4.6%
3	1511500132	63.87%	27.49%	12.50%	5.46%
4	1511500157	62.28%	26.07%	8.70%	3.54%
5	1511500199	58.97%	24.89%	9.12%	5.56%
6	1511500207	64.29%	25.76%	7.89%	3.83%
7	1511500215	58.97%	23.17%	6.83%	2.34%
8	1511500231	63.03%	24.80%	7.52%	2.75%
9	1511500249	64.46%	27.89%	13.03%	3.3%
10	1511500264	53.10%	22.27%	10.27%	4.08%
11	1511500272	62.50%	31.25%	11.86%	4.64%
12	1511500298	62.28%	26.52%	9.59%	3.15%
13	1511500314	67.20%	30%	10.08%	7.01%
Persentase Similarity Terbesar		100%	100%	100%	100%

Pada pengujian ini digunakan nilai *k-gram* dan *w-gram* dari 2 sampai 5 dimana nilainya berbeda pada abstrak 1511500157\_Deddy Rivaldy. Hasil pengujian dapat dilihat pada Tabel 4.16.

**Tabel 4.16 Uji Coba Dataset dijadikan Data Tes 2**

No	Nim	Similarity			
		k=2 w=2	k=3 w=3	k=4 w=4	k=5 w=5
1	1511500025	62.28%	26.07%	8.70%	3.54%
2	1511500082	63.96%	29.41%	15.42%	5.9%
3	1511500132	58.18%	29.44%	13.88%	7.76%
4	1511500157	100%	100%	100%	100%
5	1511500199	60.19%	23.53%	11.16%	3.92%
6	1511500207	75.27%	44.81%	44.94%	27.74%
7	1511500215	55.66%	23.32%	7.66%	5%
8	1511500231	64.76%	37.50%	21.23%	12.44%
9	1511500249	61.82%	33.33%	18.03%	11.63%
10	1511500264	60%	27.22%	17.84%	15.92%
11	1511500272	67.31%	28.21%	12.17%	7.32%
12	1511500298	67.35%	36.97%	22.95%	13.17%
13	1511500314	62.07%	32.89%	16.30%	8.17%
Persentase Similarity Terbesar		100%	100%	100%	100%

Pada pengujian ini digunakan nilai *k-gram* dan *w-gram* dari 2 sampai 5 dimana nilainya berbeda pada Abstrak\_1511500298. Hasil pengujian dapat dilihat pada Tabel 4.17.

**Tabel 4.17 Uji Coba Dataset dijadikan Data Tes 3**

No	Nim	Similarity			
		k=2 w=2	k=3 w=3	k=4 w=4	k=5 w=5
1	1511500025	62.28%	26.52%	9.59%	3.15%
2	1511500082	66.97%	27.60%	13.78%	7.66%
3	1511500132	56.76%	29.38%	12.65%	5.38%
4	1511500157	67.35%	36.97%	22.95%	13.17%
5	1511500199	55.66%	24.73%	8.26%	6.03%
6	1511500207	68.04%	36.88%	21.51%	13.87%
7	1511500215	58.65%	21.88%	7.25%	2.73%
8	1511500231	61.68%	34.46%	17.59%	12.50%
9	1511500249	61.82%	34.04%	19.82%	12.21%
10	1511500264	56.70%	33.54%	18.78%	13.84%
11	1511500272	52.63%	28.13%	10.39%	6.31%

No	Nim	Similarity			
		k=2 w=2	k=3 w=3	k=4 w=4	k=5 w=5
12	1511500298	100%	100%	100%	100%
13	1511500314	62.07%	31.70%	15.22%	9.06%
Persentase Similarity Terbesar		100%	100%	100%	100%

#### 4.5.2. Penjelasan Hasil Pengujian

Hasil uji coba dari pencarian nilai *similarity* menggunakan metode *n-gram* dan *Jaccard Similarity* terhadap algoritme *winnowing* telah ditampilkan pada tabel 4.13 dengan menggunakan *k-gram* dan *w-gram* yang sama. Tabel 4.13 menjelaskan pengaruh dari penggunaan *k-gram* dan *w-gram* terhadap hasil *similarity*. Hasil pengujian terhadap 1 data tes dengan 13 dokumen *dataset* memiliki derajat kesamaan yang berbeda-beda. Semakin kecil nilai *k-gram* dan *w-gram* maka derajat kesamaan atau *similarity* terhadap suatu dokumen mempunyai derajat kesamaan yang tinggi, sedangkan jika semakin besar nilai *k-gram* dan *w-gram* maka derajat kesamaan suatu dokumen itu memiliki derajat yang rendah, jika nilai *k-gram* dan *w-gram* sama. Pada tabel 4.13 membuktikan *k-gram* dan *w-gram* dengan nilai 2 menghasilkan *similarity* 51.40%, sedangkan *k-gram* dan *w-gram* dengan nilai 5 menghasilkan *similarity* yang rendah yaitu 5.26%.

Penginputan nilai yang berbeda pada nilai *k-gram* dan *w-gram* dapat menghasilkan nilai *similarity* yang tinggi. Semakin kecil nilai *k-gram* dan *w-gram* maka akan semakin sering potongan suku kata akan dicocokkan dan sering ditemukan nilai yang sama, jika sebaliknya maka akan semakin jarang data itu dicocokkan atau di temukan nilai yang sama. Pada tabel 4.14 dengan nilai *k-gram* = 2 dan *w-gram* = 5 menghasilkan nilai *similarity* sebesar 45.65%. Pada tabel 4.17 dengan pengujian *dataset* dan data tes yang sama menghasilkan nilai *similarity* sebesar 100%. Dengan demikian, metode *Jaccard Similarity* memiliki kinerja yang baik untuk digunakan dalam pendeteksian nilai *similarity* terhadap suatu dokumen.

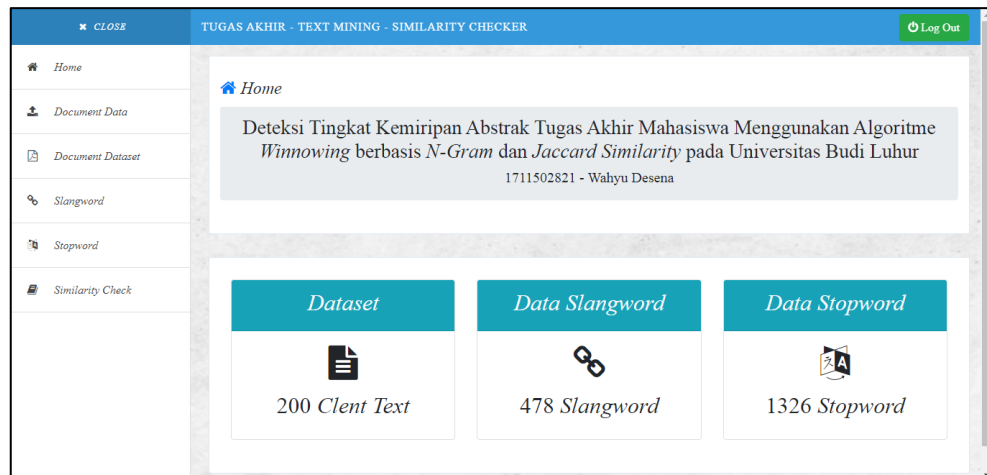
#### 4.6. Tampilan Layar Aplikasi

Setelah melewati beberapa tahap seperti implementasi metode, serta tahap pengujian sistem, pada tahap ini akan dijelaskan tampilan layar dari sistem yang telah dibuat.

##### 4.6.1. Tampilan Layar Home

Tampilan layar *home* adalah tampilan saat pertama kali masuk ke dalam sistem. Di *menu home* ini menjelaskan tentang sistem yang dibuat, jumlah *dataset*, *data slang word*, *data stop word*, serta terdapat judul dari penelitian yang dibuat. Sistem yang dibuat memiliki beberapa sub, diantaranya *document data*, *document dataset*, *slang*

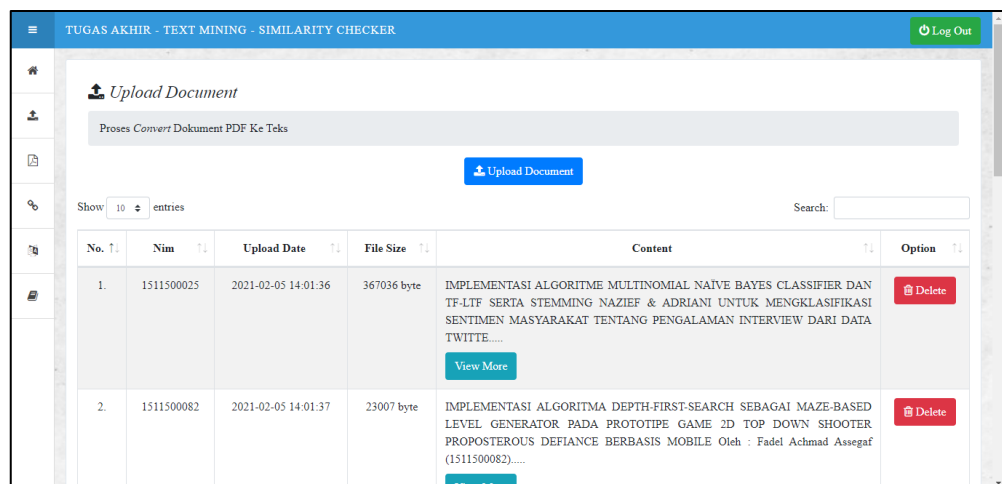
*word*, *stop word*, dan *similarity check*. Tampilan layar *home* bisa dilihat pada gambar 4.16.



**Gambar 4.16 Tampilan Layar *Home***

#### 4.6.2. Tampilan Layar *Document Data*

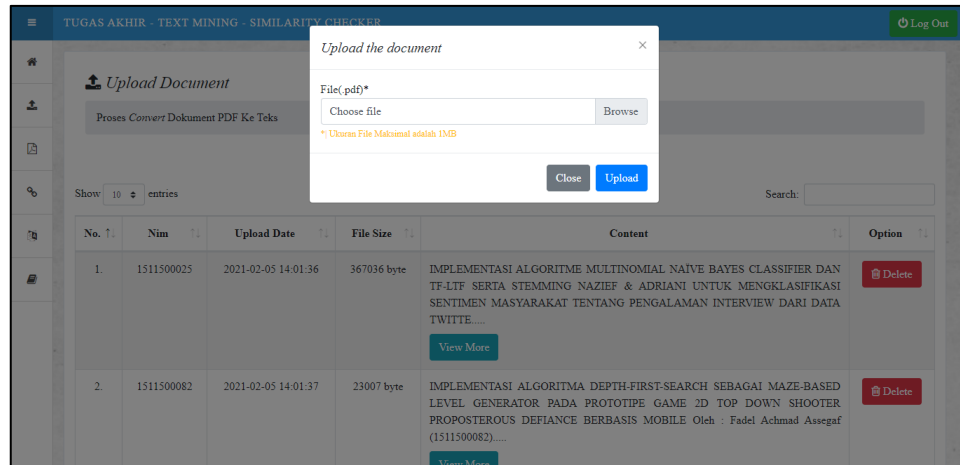
Tampilan layar *document data* terdapat proses untuk *upload* dokumen. Dimenu ini dokumen yang diupload berupa dokumen abstrak yang berbentuk *.pdf*, serta berukuran <10mb. Disini bisa dilihat juga dokumen yang sudah diupload. Tampilan layar *document data* bisa dilihat pada Gambar 4.17.



**Gambar 4.17 Tampilan Layar *Document Data***

Pada Gambar 4.18 merupakan tampilan layar untuk proses *upload* dokumen, pada proses ini *user* akan diarahkan ke *windows explorer* untuk memilih dokumen abstrak yang berbentuk *.pdf*, serta berukuran <1mb. Berikut tampilannya.





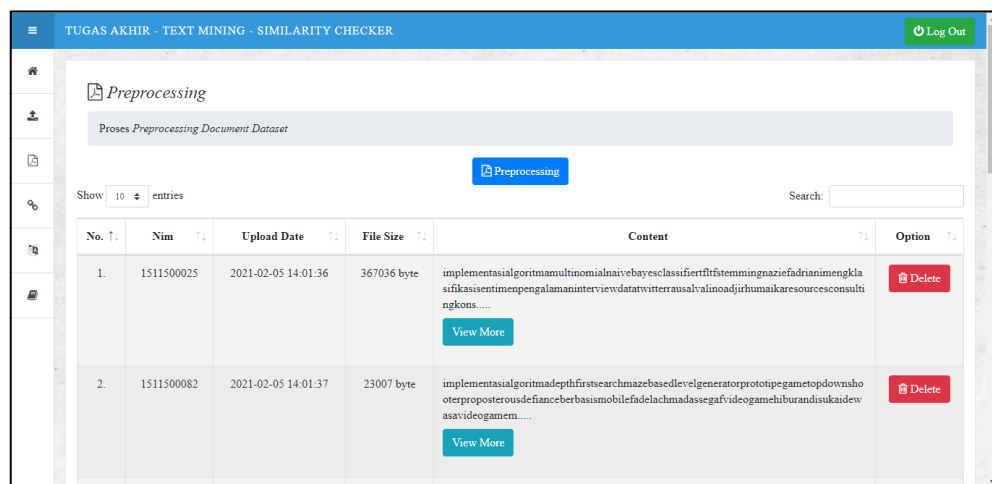
**Gambar 4.18 Tampilan Layar *Document Data* (Lanjutan)**

#### 4.6.3. Tampilan Layar *Document Dataset*

Tampilan layar *document dataset* terdapat proses *preprocessing* untuk menghasilkan kumpulan dokumen *dataset*. Berikut gambar tampilan layarnya.

##### a. Tampilan Layar *Preprocessing*

Pada Gambar 4.19 menjelaskan proses *preprocessing*, dari proses *case folding*, *cleaning*, mengganti *slang word*, menghapus *stop word*, dan penghapusan spasi antar kata.



**Gambar 4.19 Tampilan Layar *Preprocessing***

##### b. Tampilan Layar Detail *Preprocessing*

Pada Gambar 4.20 merupakan sub *menu* dari *document dataset*. Tampilan layar ini menjelaskan *detail* proses *preprocessing*, dan bisa dilihat lebih detail lagi pada gambar selanjutnya.

No.	Nim	Upload Date	File Size	Detail Content
1.	1511500025	2021-02-05 14:01:36	367036 byte	<a href="#">Detail</a>
2.	1511500082	2021-02-05 14:01:37	23007 byte	<a href="#">Detail</a>
3.	1511500132	2021-02-05 14:01:37	41297 byte	<a href="#">Detail</a>
4.	1511500157	2021-02-05 14:01:37	177537 byte	<a href="#">Detail</a>
5.	1511500199	2021-02-05 14:01:37	99221 byte	<a href="#">Detail</a>

**Gambar 4.20 Tampilan Layar Detail *Preprocessing***

- Tampilan Layar Data Awal  
Pada Gambar 4.21 merupakan bentuk data awal dari teks dokumen abstrak, yang akan diproses di tahap *preprocessing*.

Detail Content
<p><b>Data Awal</b></p> <p>IMPLEMENTASI ALGORITMA DEPTH-FIRST-SEARCH SEBAGAI MAZE-BASED LEVEL GENERATOR PADA PROTOTYPE GAME 2D TOP DOWN SHOOTER PROPOSTEROUS DEFIANCE BERBASIS MOBILE Oleh : Fadel Achmad Assegaf (1511500082) Video Game pada umumnya merupakan jenis hiburan yang disukai oleh hampir semua kalangan orang saat ini, mulai dari kalangan anak-anak hingga kalangan dewasa. Video game juga sering digunakan untuk menghilangkan kejenuhan dalam menjalankan aktivitas sehari-hari dan juga untuk melatih diri seseorang untuk lebih berkonsentrasi dan fokus pada video game tersebut. Pembuatan aset-aset level pada Video game biasanya dibuat secara manual dan desain dan alur level akan selalu sama ketika level tersebut dimuat lagi. Maka dari itu, pembuatan Video game ini yang dikembangkan melalui Unity Engine bersifat 2D, dengan genre top down shooter dengan judul game Proposterous Defiance menggunakan Algoritma Depth-First-Search dalam pembuatan level secara otomatis. Level nantinya dapat terbuat dengan arah jalur yang acak dari start hingga menuju finish. Algoritma ini digunakan untuk pembuatan labirin perfect sehingga tidak terdapat jalan yang berulang, selain itu tidak ada sel yang terisolasi pada labirin. Algoritma DFS merupakan algoritma yang digunakan untuk membuat sebuah labirin perfect. Pembuatan labirin pada game Proposterous Defiance yang selalu random atau acak dengan algoritma Depth-First-Search dapat memberikan kesan fresh setiap kali level dimainkan, karena pembuatan level dalam video game ini berbentuk labirin perfect yang mana selalu memiliki alur yang berbeda setiap kali level dimainkan. Metode pengembangan menggunakan Multimedia Development Life Cycle (MDLC), yaitu suatu tahap yang di dalamnya terdapat pengolahan konsep, perancangan, pengumpulan bahan, pembuatan, testing, dan yang terakhir distribusi. Dengan MDLC ini, video game yang dibuat akan mendapatkan hasil akhir yang diharapkan, akurat, serta optimal. Berdasarkan penelitian yang telah dijalankan, dengan menggunakan metode MDLC dapat disimpulkan penggunaan algoritma Depth-First-Search dalam pembuatan maze- based level-generator pada game 2D Proposterous Defiance telah terbukti dapat membuat alur jalan labirin yang selalu acak setiap kali level pada game dimuat. Kata Kunci : Depth-First-Search, Labirin, MDLC, Unity 3D, Video Game xv+70 halaman; 44 gambar; 2 tabel; 1 lampiran Perpustakaan Universitas Budi Luhur</p>

**Gambar 4.21 Tampilan Layar Data Awal**

- Tampilan Layar *Case Folding*  
Pada Gambar 4.22 merupakan tampilan layar dari proses *case folding*, yaitu proses merubah teks menjadi huruf kecil.

#### *Case Folding*

implementasi algoritma depth-first-search sebagai maze-based level generator pada prototipe game 2d top down shooter proposterous defiance berbasis mobile oleh : fadel achmad assegaf (1511500082) video game pada umumnya merupakan jenis hiburan yang disukai oleh hampir semua kalangan orang saat ini, mulai dari kalangan anak-anak hingga kalangan dewasa. video game juga sering digunakan untuk menghilangkan kejenuhan dalam menjalankan aktivitas sehari-hari dan juga untuk melatih diri seseorang untuk lebih berkonsentrasi dan fokus pada video game tersebut. pembuatan aset-aset level pada video game biasanya dibuat secara manual dan desain dan alur level akan selalu sama ketika level tersebut dimuat lagi. maka dari itu, pembuatan video game ini yang dikembangkan melalui unity engine bersifat 2d, dengan genre top down shooter dengan judul game proposterous defiance menggunakan algoritma depth-first-search dalam pembuatan level secara otomatis. level nantinya dapat terbuat dengan arah jalur yang acak dari start hingga menuju finish. algoritma ini digunakan untuk pembuatan labirin perfect sehingga tidak terdapat jalan yang berulang, selain itu tidak ada sel yang terisolasi pada labirin. algoritma dfs merupakan algoritma yang digunakan untuk membuat sebuah labirin perfect. pembuatan labirin pada game proposterous defiance yang selalu random atau acak dengan algoritma depth-first-search dapat memberikan kesan fresh setiap kali level dimainkan, karena pembuatan level dalam video game ini berbentuk labirin perfect yang mana selalu memiliki alur yang berbeda setiap kali level dimainkan. metode pengembangan menggunakan multimedia development life cycle (mdlc), yaitu suatu tahap yang di dalamnya terdapat pengolahan konsep, perancangan, pengumpulan bahan, pembuatan, testing, dan yang terakhir distribusi. dengan mdlc ini, video game yang dibuat akan mendapatkan hasil akhir yang diharapkan, akurat, serta optimal. berdasarkan penelitian yang telah dijalankan, dengan menggunakan metode mdlc dapat disimpulkan penggunaan algoritma depth-first-search dalam pembuatan maze-based level-generator pada game 2d proposterous defiance telah terbukti dapat membuat alur jalan labirin yang selalu acak setiap kali level pada game dimuat. kata kunci : depth-first-search, labirin, mdlc, unity 3d, video game xv+70 halaman; 44 gambar; 2 tabel; 1 lampiran perpustakaan universitas budi luhur

### **Gambar 4.22 Tampilan Layar *Case Folding***

- Tampilan Layar *Cleaning*  
Pada Gambar 4.23 merupakan tampilan layar dari proses *cleaning*, yaitu proses menghapus karakter selain huruf a-z.

#### *Cleaning*

implementasi algoritma depth first search sebagai maze based level generator pada prototipe game d top down shooter proposterous defiance berbasis mobile oleh fadel achmad assegaf video game pada umumnya merupakan jenis hiburan yang disukai oleh hampir semua kalangan orang saat ini mulai dari kalangan anak anak hingga kalangan dewasa video game juga sering digunakan untuk menghilangkan kejenuhan dalam menjalankan aktivitas sehari hari dan juga untuk melatih diri seseorang untuk lebih berkonsentrasi dan fokus pada video game tersebut pembuatan aset aset level pada video game biasanya dibuat secara manual dan desain dan alur level akan selalu sama ketika level tersebut dimuat lagi maka dari itu pembuatan video game ini yang dikembangkan melalui unity engine bersifat d dengan genre top down shooter dengan judul game proposterous defiance menggunakan algoritma depth first search dalam pembuatan level secara otomatis level nantinya dapat terbuat dengan arah jalur yang acak dari start hingga menuju finish algoritma ini digunakan untuk pembuatan labirin perfect sehingga tidak terdapat jalan yang berulang selain itu tidak ada sel yang terisolasi pada labirin algoritma dfs merupakan algoritma yang digunakan untuk membuat sebuah labirin perfect pembuatan labirin pada game proposterous defiance yang selalu random atau acak dengan algoritma depth first search dapat memberikan kesan fresh setiap kali level dimainkan karena pembuatan level dalam video game ini berbentuk labirin perfect yang mana selalu memiliki alur yang berbeda setiap kali level dimainkan metode pengembangan menggunakan multimedia development life cycle mdlc yaitu suatu tahap yang di dalamnya terdapat pengolahan konsep perancangan pengumpulan bahan pembuatan testing dan yang terakhir distribusi dengan mdlc ini video game yang dibuat akan mendapatkan hasil akhir yang diharapkan akurat serta optimal berdasarkan penelitian yang telah dijalankan dengan menggunakan metode mdlc dapat disimpulkan penggunaan algoritma depth first search dalam pembuatan maze based level generator pada game d proposterous defiance telah terbukti dapat membuat alur jalan labirin yang selalu acak setiap kali level pada game dimuat kata kunci depth first search labirin mdlc unity d video game xv halaman gambar tabel lampiran perpustakaan universitas budi luhur

### **Gambar 4.23 Tampilan Layar *Cleaning***

- Tampilan Layar *Slang Word*  
Pada Gambar 4.24 merupakan tampilan layar dari proses *slang word*, yaitu proses mengganti kata yang ada di data *slang word* menjadi kata asli.

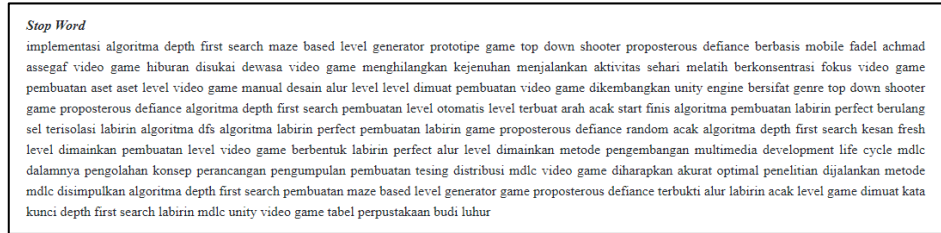
#### *Slang Word*

implementasi algoritma depth first search sebagai maze based level generator pada prototipe game d top down shooter proposterous defiance berbasis mobile oleh fadel achmad assegaf video game pada umumnya merupakan jenis hiburan yang disukai oleh hampir semua kalangan orang saat ini mulai dari kalangan anak anak hingga kalangan dewasa video game juga sering digunakan untuk menghilangkan kejenuhan dalam menjalankan aktivitas sehari hari dan juga untuk melatih diri seseorang untuk lebih berkonsentrasi dan fokus pada video game tersebut pembuatan aset aset level pada video game biasanya dibuat secara manual dan desain dan alur level akan selalu sama ketika level tersebut dimuat lagi maka dari itu pembuatan video game ini yang dikembangkan melalui unity engine bersifat d dengan genre top down shooter dengan judul game proposterous defiance menggunakan algoritma depth first search dalam pembuatan level secara otomatis level nantinya dapat terbuat dengan arah jalur yang acak dari start hingga menuju finis algoritma ini digunakan untuk pembuatan labirin perfect sehingga tidak terdapat jalan yang berulang selain itu tidak ada sel yang terisolasi pada labirin algoritma dfs merupakan algoritma yang digunakan untuk membuat sebuah labirin perfect pembuatan labirin pada game proposterous defiance yang selalu random atau acak dengan algoritma depth first search dapat memberikan kesan fresh setiap kali level dimainkan karena pembuatan level dalam video game ini berbentuk labirin perfect yang mana selalu memiliki alur yang berbeda setiap kali level dimainkan metode pengembangan menggunakan multimedia development life cycle mdlc yaitu suatu tahap yang di dalamnya terdapat pengolahan konsep perancangan pengumpulan bahan pembuatan tesing dan yang terakhir distribusi dengan mdlc ini video game yang dibuat akan mendapatkan hasil akhir yang diharapkan akurat serta optimal berdasarkan penelitian yang telah dijalankan dengan menggunakan metode mdlc dapat disimpulkan penggunaan algoritma depth first search dalam pembuatan maze based level generator pada game d proposterous defiance telah terbukti dapat membuat alur jalan labirin yang selalu acak setiap kali level pada game dimuat kata kunci depth first search labirin mdlc unity d video game xv halaman gambar tabel lampiran perpustakaan universitas budi luhur

### **Gambar 4.24 Tampilan Layar *Slang Word***

- **Tampilan Layar *Stop Word***

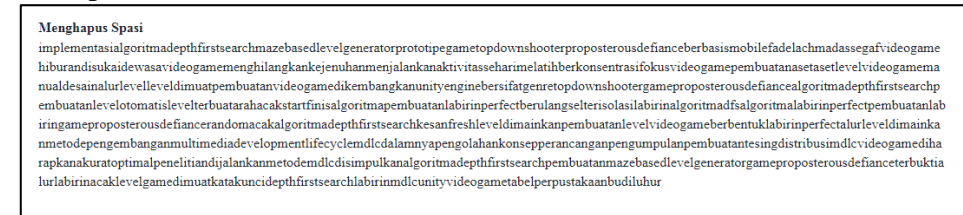
Pada Gambar 4.25 merupakan tampilan layar dari proses *stop word*, yaitu proses menghapus kata yang ada di data *stop word*.



**Gambar 4.25 Tampilan Layar *Stop Word***

- **Tampilan Layar Menghapus Spasi**

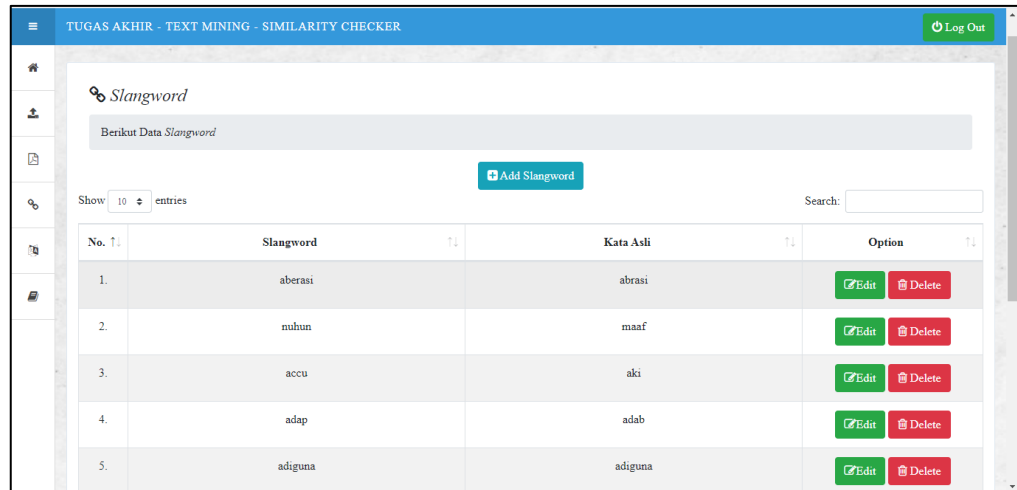
Pada Gambar 4.26 merupakan tampilan layar dari proses menghapus spasi, yaitu proses menghapus spasi antar kata di setiap teks.



**Gambar 4.26 Tampilan Layar Menghapus Spasi**

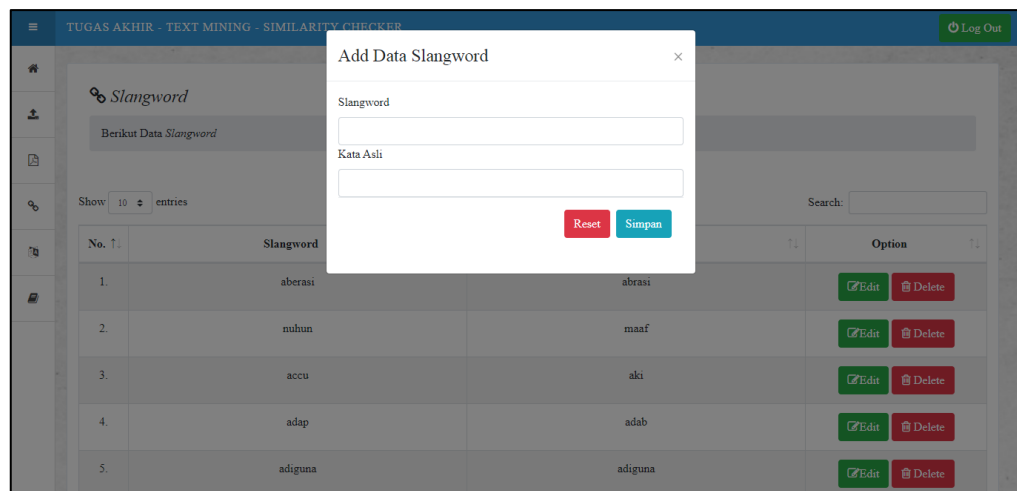
#### 4.6.4. Tampilan Layar *Slang word*

Tampilan layar pada Gambar 4.27, Gambar 4.28, dan Gambar 4.29 ini merupakan *menu slang word*, halaman ini berisi kumpulan data *slang word* yang digunakan untuk proses *preprocessing* nantinya, pada *menu* ini bisa dilakukan proses menambah, menghapus dan edit data *slang word*.



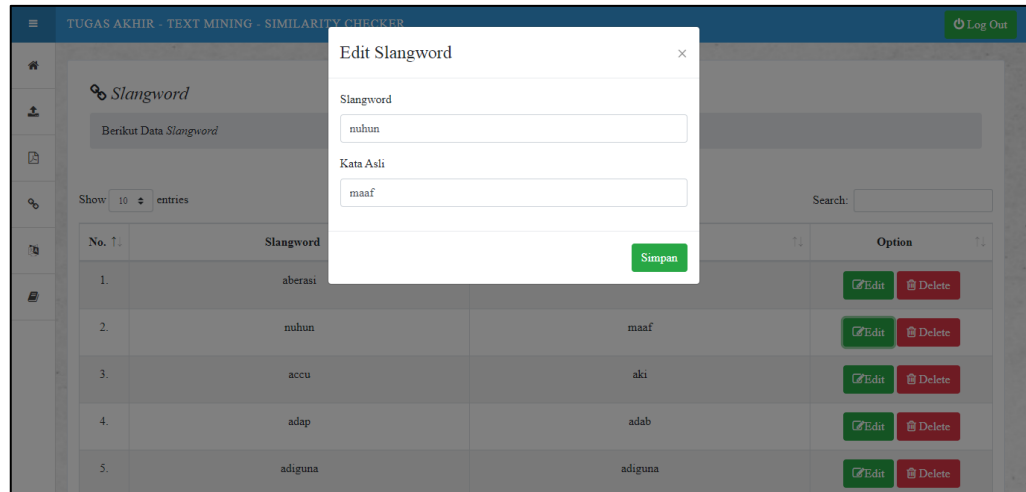
**Gambar 4.27 Tampilan Layar Menu Slang word**

Pada Gambar 4.28 merupakan tampilan layar untuk proses menambah data *slang word*, pada proses ini setelah *user* melakukan penambahan data *slang word* maka data tersebut akan disimpan kedalam *database* pada tabel *slang word*.



**Gambar 4.28 Tampilan Layar Menu Add Slang word**

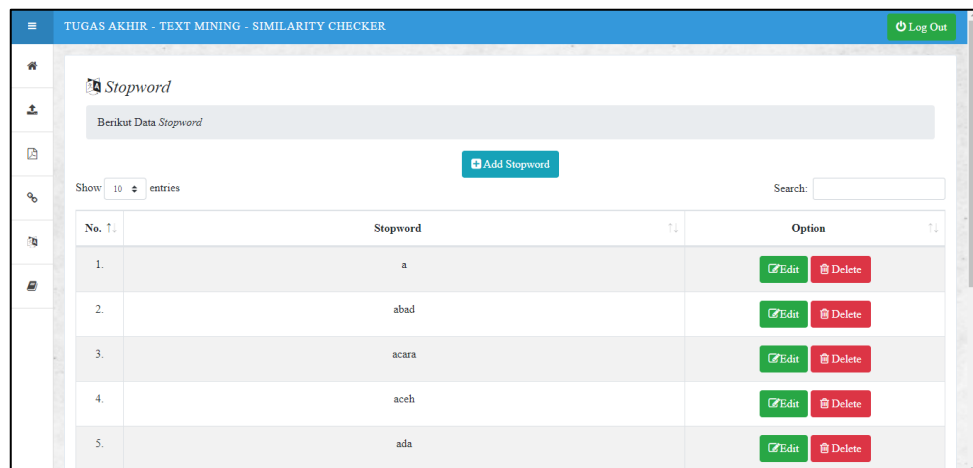
Pada Gambar 4.29 merupakan tampilan layar untuk proses mengedit data *slang word*, pada proses ini setelah *user* melakukan pengeditan data *slang word* maka data tersebut akan diupdate kedalam *database* pada tabel *slang word*.



**Gambar 4.29 Tampilan Layar Menu Edit Slang word**

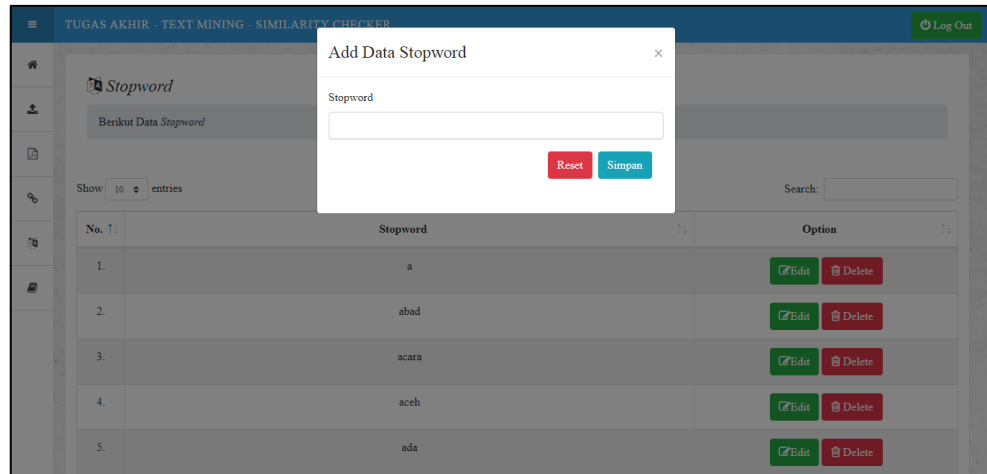
#### 4.6.5. Tampilan Layar Stop word

Tampilan layar pada Gambar 4.30, Gambar 4.31, dan Gambar 4.32 ini merupakan *menu stop word*, halaman ini berisi kumpulan data *stop word* yang digunakan untuk proses *preprocessing* nantinya, pada *menu* ini bisa dilakukan proses menambah, menghapus dan edit data *stop word*.



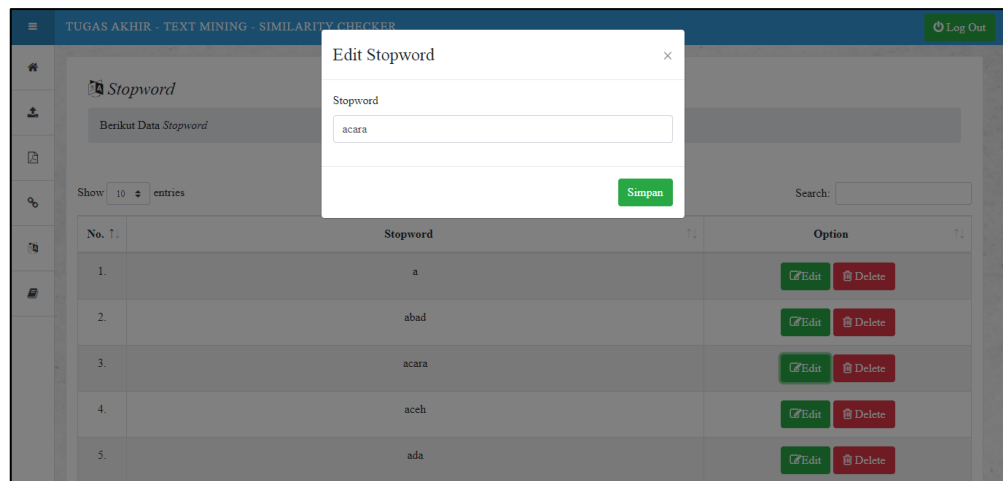
**Gambar 4.30 Tampilan Layar Menu Stop word**

Pada Gambar 4.31 merupakan tampilan layar untuk proses menambah data *stop word*, pada proses ini setelah *user* melakukan penambahan data *stop word* maka data tersebut akan disimpan kedalam *database* pada tabel *stop word*.



**Gambar 4.32 Tampilan Layar Menu Add Stop word**

Pada Gambar 4.33 merupakan tampilan layar untuk proses mengedit data *stop word*, pada proses ini setelah *user* melakukan pengeditan data *stop word* maka data tersebut akan diupdate kedalam database pada tabel *stop word*.



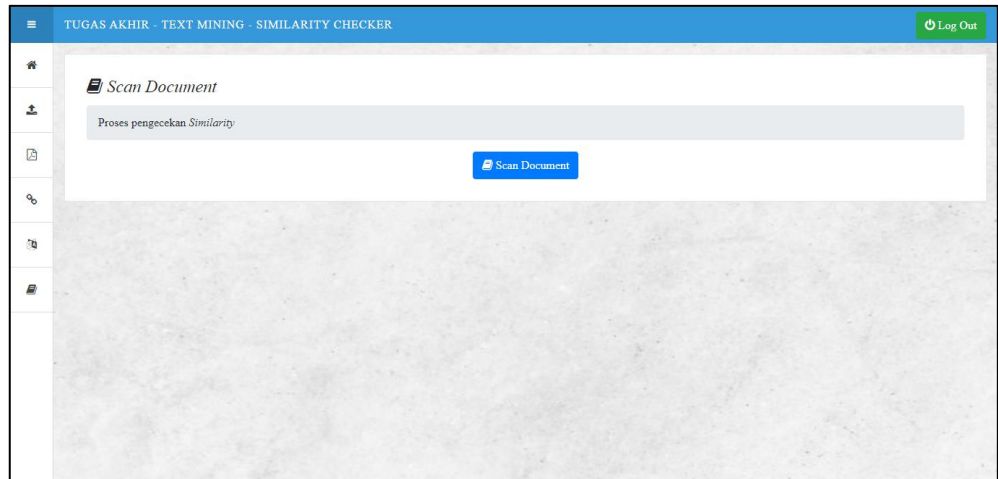
**Gambar 4.33 Tampilan Layar Menu Edit Stop word**

#### 4.6.6. Tampilan Layar *Similarity Check*

Pada tampilan layar *similarity check* ini menjelaskan proses untuk mengetahui nilai *similarity* dari sebuah dokumen abstrak.

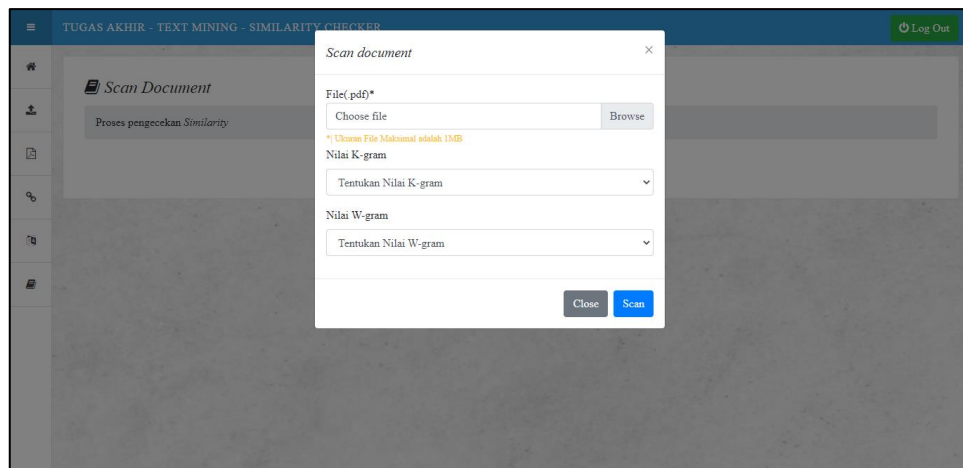
##### a. Tampilan Layar *Menu Scan Document*

Tampilan layar pada Gambar 4.34 merupakan proses *scan* dokumen abstrak untuk melihat nilai *similarity* dari dokumen tersebut.



**Gambar 4.34 Tampilan Layar Menu Scan Document**

Pada Gambar 4.35 merupakan tampilan layar untuk proses *scan* dokumen data tes, pada tahap ini *user* harus menentukan nilai *k-gram* dan nilai *w-gram* untuk proses pencarian nilai *similarity* dari dokumen yang telah di *scan*.

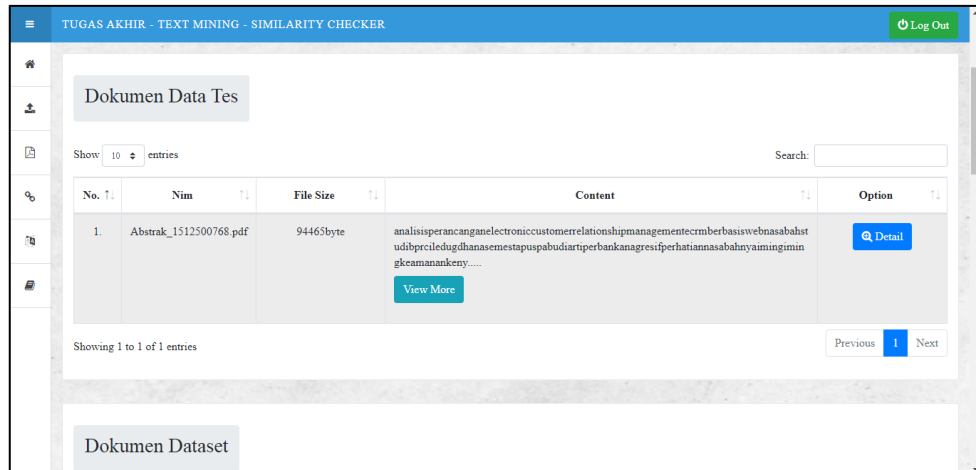


**Gambar 4.35 Tampilan Layar Menu Scan Document (Lanjutan)**

b. Tampilan Layar *Document Data Tes*

Tampilan layar pada Gambar 4.36 merupakan dokumen abstrak data tes yang di *scan* pada proses sebelumnya. Pada gambar ini bisa dilihat teks bersih, potongan *n-gram*, nilai *rolling hash*, pembentukan *window*, serta nilai *fingerprint* dari dokumen tersebut.

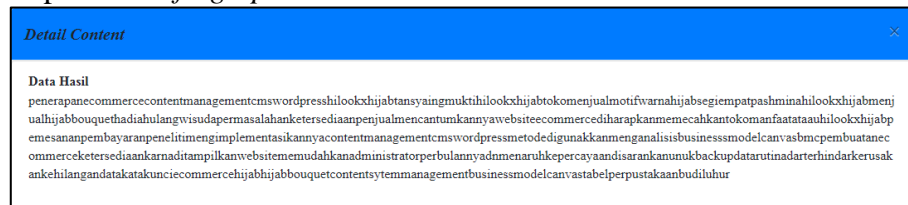




**Gambar 4.36 Tampilan Layar *Document Data Tes***

- Tampilan Layar Data Hasil

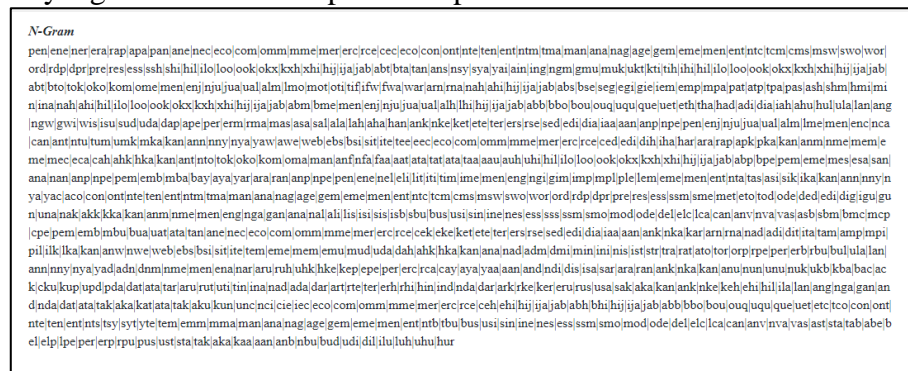
Pada Gambar 4.37 merupakan tampilan layar dari data tes yang sudah melalui tahap *preprocessing*, data ini akan diproses untuk pembentukan *n-gram*, *rolling hash*, *window*, dan pencarian *fingerprint*.



**Gambar 4.37 Tampilan Layar Data Hasil**

- Tampilan Layar *N-Gram*

Pada Gambar 4.38 merupakan tampilan layar dari *n-gram*, teks dari data hasil dipotong berdasarkan jumlah nilai *k-gram* yang telah ditentukan pada saat proses *scan* dokumen.



**Gambar 4.38 Tampilan Layar *N-Gram***

- Tampilan Layar *Rolling Hash*

Pada Gambar 4.39 merupakan tampilan layar dari *rolling hash*, pada gambar ini bisa dilihat nilai *hash* dari *n-gram* yang sudah dibentuk.

```
Rolling Hash
6305|5820|6211|5844|6377|5634|6277|5624|6196|5753|5737|6311|6205|6162|5846|6380|5657|5753|5738|6325|6303|6501|5835|6311|6544|6130|5620|6172|5575|586
3|5813|6158|5835|6301|6486|5729|6265|6579|6722|6337|6398|5798|6387|6408|5869|6544|6468|5939|6012|6180|6323|6308|6187|6713|5937|5984|5971|5555|5711|6
473|5638|6316|6579|6713|5598|6018|6220|5927|6267|6598|6160|6523|5978|5939|6012|6180|6323|6308|6187|6713|5937|5984|5971|5555|5725|6568|6299|6129|6303
|6158|5825|6249|6110|6520|5618|6166|6234|6356|6521|5978|5928|6624|5661|6453|6173|5586|5937|5984|5971|5554|5708|6445|5775|5883|5961|5824|6222|6283|56
77|6565|6282|5662|6472|5964|6186|6012|6173|5586|5939|6012|6180|6323|6308|6187|6713|5937|5984|5971|5548|5666|6158|5825|6249|6110|6520|5613|6125|5937
|5984|5971|5537|5599|5696|6371|6641|6457|6556|5865|6509|5875|5558|5732|5928|5598|6023|6586|6081|5626|6230|5985|6681|6067|6554|6530|5691|5638|6309|585
6|6446|6135|5655|6422|5606|6075|5578|5885|5630|6240|6066|5862|6505|5862|6492|6442|5754|5732|5921|5542|5635|6275|6305|5825|6249|6110|6520|5618|6156|6
158|5818|6180|5640|5639|6319|6612|6603|6187|6032|5633|6281|6334|6727|5687|6636|5750|5712|6486|6058|6492|5755|5753|5737|6311|6205|6162|5846|6380|5658
|5754|5739|5970|5889|5648|6377|5644|6334|6032|5632|6254|6157|5813|6147|5739|5634|5588|5942|6032|5639|6313|6568|6299|6129|6299|6130|5625|6201|5774|55
48|5662|6479|5662|6460|5549|5676|6566|5939|6012|6180|6323|6308|6187|6713|5937|5984|5971|5551|5687|6304|5813|6163|5851|6424|5620|6179|5635|6275|6304|
5810|6124|5602|5697|6722|5648|6375|5635|6275|6305|5820|6205|5810|6143|6062|6528|6009|6158|5822|6216|5891|6020|6233|6345|6108|5813|6158|5835|6275|6299|647
8|5663|6477|5991|6032|5633|6281|6334|6707|5557|5738|6325|6303|6501|5835|6311|6544|6130|5620|6172|5575|5863|5813|6158|5835|6301|6486|5729|6265|6579|6
722|6337|6398|5798|6387|6408|5869|6549|6499|6164|5872|6561|6240|5707|5754|5738|5983|5976|6600|6176|5609|6089|6032|5632|6254|6158|5822|6208|5836|5620
|6177|5614|6142|6055|6485|6048|6438|5736|6643|6480|6016|6212|5869|6555|6549|6509|6218|6240|5715|5804|6082|5640|5641|6313|6576|5656|6430|5664|6146|57
36|6304|5810|6144|5718|6528|5662|6473|5624|6196|5753|5737|6311|6205|6162|5846|6380|5665|5799|6066|5862|6505|5862|6492|6442|5754|5732|5921|5542|5630
|6236|6036|5661|6453|6169|5558|5751|6054|6472|5628|6230|6331|6008|6138|6032|5642|6324|6636|5750|5712|6486|6058|6500|5813|6157|5829|6260|6530|5683|558
8|5942|6032|5620|6169|5562|5768|6186|6020|6240|6066|6561|6579|6381|5676|6575|6349|6471|6309|5845|6389|5729|6586|6081|5633|6281|6334|6708|5563|5779|6
254|6158|5816|6183|5668|6509|6568|5946|6062|5834|6309|5846|6376|5651|5697|6705|5542|5623|6195|5750|6047|6428|5648|6375|5630|6236|6032|5640|6319|6620
|6316|6580|6026|5580|5553|5717|6174|6617|6285|5695|5662|6477|5668|6521|6650|6529|6012|6169|5550|5693|5667|6499|6505|5851|6419|5941|6015|6187|5693|56
58|6436|6064|5864|6520|6635|6421|5599|6032|5630|6240|6054|5782|5939|5998|6081|5626|6208|5836|5623|6187|5695|5662|6470|5599|6038|5662|6470|5619|6172
|6602|6188|5687|5951|5753|5737|6311|6205|6162|5846|6380|5662|5782|5937|5984|5971|5543|5635|5937|5984|5971|5537|5599|5696|6371|6641|6457|6556|5860|648
8|5738|6325|6303|6501|5835|6311|6610|6598|6842|6500|5821|6201|6130|5620|6172|5575|5863|5813|6158|5835|6300|6487|5736|6643|6480|6016|6212|5869|6549|6
509|6218|6240|5715|5804|6082|5640|5641|6313|6576|5674|6544|6461|5540|5617|5817|6177|6309|5859|6487|6422|6654|6544|6470|5599|6019|5542|5621|6193|5721
|6538|5743|6018|6215|6578|6029
```

**Gambar 4.39 Tampilan Layar *Rolling Hash***

- Tampilan Layar *Window*

Pada Gambar 4.40 merupakan tampilan layar dari *window*, pada gambar ini nilai *hash* yang sudah didapat pada proses sebelumnya akan dilakukan pembentukan *window* berdasarkan nilai yang telah ditentukan pada saat proses *scan* dokumen.

```
Window
6305|5820|6211|5844|6377|5634|6277|5624|6196|5753|5737|6311|6205|6162|5846|6380|5657|5753|5738|6325|6303|6501|5835|6311|6544|6130|5620|6172|5575|586
3|5813|6158|5835|6301|6486|5729|6265|6579|6722|6337|6398|5798|6387|6408|5869|6544|6468|5939|6012|6180|6323|6308|6187|6713|5937|5984|5971|5555|5711|6
473|5638|6316|6579|6713|5598|6018|6220|5927|6267|6598|6160|6523|5978|5939|6012|6180|6323|6308|6187|6713|5937|5984|5971|5555|5725|6568|6299|6129|6303
|6158|5825|6249|6110|6520|5618|6166|6234|6356|6521|5978|5928|6624|5661|6453|6173|5586|5937|5984|5971|5554|5708|6445|5775|5883|5961|5824|6222|6283|56
77|6565|6282|5662|6472|5964|6186|6012|6173|5586|5939|6012|6180|6323|6308|6187|6713|5937|5984|5971|5548|5666|6158|5825|6249|6110|6520|5613|6125|5937
|5984|5971|5537|5599|5696|6371|6641|6457|6556|5865|6509|5875|5558|5732|5928|5598|6023|6586|6081|5626|6230|5985|6681|6067|6554|6530|5691|5638|6309|585
6|6446|6135|5655|6422|5606|6075|5578|5885|5630|6240|6066|5862|6505|5862|6492|6442|5754|5732|5921|5542|5635|6275|6305|5825|6249|6110|6520|5618|6156|6
158|5818|6180|5640|5639|6319|6612|6603|6187|6032|5633|6281|6334|6727|5687|6636|5750|5712|6486|6058|6492|5755|5753|5737|6311|6205|6162|5846|6380|5658
|5754|5739|5970|5889|5648|6377|5644|6334|6032|5632|6254|6157|5813|6147|5739|5634|5588|5942|6032|5639|6313|6568|6299|6129|6299|6130|5625|6201|5774|55
48|5662|6479|5662|6460|5549|5676|6566|5939|6012|6180|6323|6308|6187|6713|5937|5984|5971|5551|5687|6304|5813|6163|5851|6424|5620|6179|5635|6275|6304|
5810|6124|5602|5697|6722|5648|6375|5635|6275|6305|5820|6205|5810|6143|6062|6528|6009|6158|5822|6216|5891|6020|6233|6345|6108|5813|6158|5835|6275|6299|647
8|5663|6477|5991|6032|5633|6281|6334|6707|5557|5738|6325|6303|6501|5835|6311|6544|6130|5620|6172|5575|5863|5813|6158|5835|6301|6486|5729|6265|6579|6
722|6337|6398|5798|6387|6408|5869|6549|6499|6164|5872|6561|6240|5707|5754|5738|5983|5976|6600|6176|5609|6089|6032|5632|6254|6158|5822|6208|5836|5620
|6177|5614|6142|6055|6485|6048|6438|5736|6643|6480|6016|6212|5869|6555|6549|6509|6218|6240|5715|5804|6082|5640|5641|6313|6576|5656|6430|5664|6146|57
36|6304|5810|6144|5718|6528|5662|6473|5624|6196|5753|5737|6311|6205|6162|5846|6380|5665|5799|6066|5862|6505|5862|6492|6442|5754|5732|5921|5542|5630
|6236|6036|5661|6453|6169|5558|5751|6054|6472|5628|6230|6331|6008|6138|6032|5642|6324|6636|5750|5712|6486|6058|6500|5813|6157|5829|6260|6530|5683|558
8|5942|6032|5620|6169|5562|5768|6186|6020|6240|6066|6561|6579|6381|5676|6575|6349|6471|6309|5845|6389|5729|6586|6081|5633|6281|6334|6708|5563|5779|6
254|6158|5816|6183|5668|6509|6568|5946|6062|5834|6309|5846|6376|5651|5697|6705|5542|5623|6195|5750|6047|6428|5648|6375|5630|6236|6032|5640|6319|6620
|6316|6580|6026|5580|5553|5717|6174|6617|6285|5695|5662|6477|5668|6521|6650|6529|6012|6169|5550|5693|5667|6499|6505|5851|6419|5941|6015|6187|5693|56
58|6436|6064|5864|6520|6635|6421|5599|6032|5630|6240|6054|5782|5939|5998|6081|5626|6208|5836|5623|6187|5695|5662|6470|5599|6038|5662|6470|5619|6172
|6602|6188|5687|5951|5753|5737|6311|6205|6162|5846|6380|5662|5782|5937|5984|5971|5543|5635|5937|5984|5971|5537|5599|5696|6371|6641|6457|6556|5860|648
8|5738|6325|6303|6501|5835|6311|6610|6598|6842|6500|5821|6201|6130|5620|6172|5575|5863|5813|6158|5835|6300|6487|5736|6643|6480|6016|6212|5869|6549|6
509|6218|6240|5715|5804|6082|5640|5641|6313|6576|5674|6544|6461|5540|5617|5817|6177|6309|5859|6487|6422|6654|6544|6470|5599|6019|5542|5621|6193|5721
|6538|5743|6018
```

**Gambar 4.40 Tampilan Layar *Window***

- Tampilan Layar *Fingerprint*

Pada Gambar 4.41 merupakan tampilan layar dari *fingerprint*, pada gambar ini dapat dilihat hasil dari proses pencarian nilai *finger* pada tahap sebelumnya, yaitu pencarian

nilai terkecil pada nilai *window* yang sudah dibentuk pada tahap sebelumnya.

**Fingerprint**  
5820|5624|5737|5846|5657|5835|5620|5575|5729|6265|5798|5869|6012|5937|5555|5638|5598|6160|5939|6187|5555|5725|5825|5618|5978|5661|5586|5554|5775|567  
7|5662|5586|5939|5937|5548|5825|5613|5537|5696|5865|5558|5626|5985|5638|5856|5606|5578|5862|5732|5542|5825|5618|5639|6187|5633|5687|5712|5737|5846|5  
658|5644|5632|5739|5588|5639|5625|5548|5549|5676|6180|5937|5551|5620|5635|5602|5635|5820|5810|5822|5891|5813|5663|5633|5557|5835|5620|5575|5835|5729  
|5798|5869|5872|5707|5609|5632|5822|5614|6048|5736|5869|5715|5640|5656|5664|5718|5624|5737|5665|5799|5754|5542|5661|5558|5628|5642|5712|5813|5829|55  
88|5562|6020|5676|6309|5729|5633|5563|5668|5946|5834|5542|5623|5630|5640|6026|5553|5662|5668|5550|5667|5851|5658|5864|5599|5782|5626|5623|5599|5619|  
5687|5737|5662|5782|5543|5537|6371|5738|5835|6500|5620|5575|5835|5736|5869|5715|5640|5674|5540|5859|5599|5542|5721

**Gambar 4.41 Tampilan Layar *Fingerprint***

c. Tampilan Layar Nilai *Similarity*

Tampilan layar pada Gambar 4.42 merupakan dokumen abstrak *dataset* yang diambil dari *database*. Pada gambar ini bisa dilihat teks bersih, potongan *n-gram*, nilai *rolling hash*, pembentukan *window*, serta nilai *fingerprint* dari dokumen *dataset*. Pada tampilan layar ini bisa melihat tingkat *similarity* dari dokumen abstrak yang *discan* pada proses sebelumnya dengan dokumen *dataset*.

TUGAS AKHIR - TEXT MINING - SIMILARITY CHECKER						
Dokumen Dataset						
Show 10 entries		Search:				
No.	Nim	File Size	Content	Similarity	Option	
1.	1512500339	102880 byte	analisisperancanganelectroniccustomerrrelationshipmanagemencrmberbasiswebpelaya nanbimbingbelajarstudibimbelprimagamasiscaagdinsaramadannimperkembangantek nologiinternetsatmen..... <a href="#">View More</a>	38.55 %	<a href="#">Detail</a>	
2.	1512503366	342467 byte	membangunelectroniccustomerrelationshipmanagemencrmupelayananloyalitasmahasis wakelaskaryawanbudiluhurayupurnamasariperkuliahanmahasiswakelaskaryawanketerb atasandimilikimahasiswa..... <a href="#">View More</a>	38.27 %	<a href="#">Detail</a>	
3.	1512500578	68014 byte	rancanganelectroniccustomerrrelationshipmanagemencrmberbasiswebpenjualandigital travelberkembangnyateknologimenerapkanteknologi kedalam bisnis menerapkanteknolo gisietametodecustomer..... <a href="#">View More</a>	36.99 %	<a href="#">Detail</a>	

**Gambar 4.42 Tampilan Layar Nilai *Similarity***

## **BAB V PENUTUP**

### **5.1. Kesimpulan**

Berdasarkan hasil evaluasi dari sistem deteksi tingkat *similarity* pada dokumen abstrak skripsi mahasiswa Budi Luhur, maka dapat disimpulkan bahwa:

- a. Sistem ini dapat mendeteksi tingkat *similarity* sebuah dokumen dengan sumber data dokumen abstrak mahasiswa Budi Luhur.
- b. Penggunaan metode *n-gram* dan *jaccard similarity* terhadap algoritme *winnowing* cukup baik dengan menghasilkan persentase sampai 100%.
- c. Nilai *k-gram* dan *w-gram* mempengaruhi persentase *similarity*, penggunaan *k-gram* dan *w-gram* yang tepat sangat diperlukan.
- d. Nilai *k-gram* dan *w-gram* yang terlalu kecil mengakibatkan nilai persentase *similarity* yang besar, begitupun sebaliknya.
- e. Metode *jaccard similarity* yang digunakan untuk menghasilkan tingkat *similarity* antar dokumen juga cukup baik karena koefisien ini sederhana dengan mencari item yang sama dari dua dokumen kemudian dibagi dengan total item kedua dari penggabungan dua dokumen.

### **5.2. Saran**

Adapun saran yang dapat peneliti berikan sebagai pengembangan lebih lanjut untuk sistem ini agar dapat berjalan dengan sempurna dengan fungsi yang lebih baik adalah sebagai berikut:

- a. Pengembangan sistem dapat dipadukan dengan metode lain, agar lebih baik hasil yang didapat.
- b. Dapat mengetahui kata atau kalimat mana yang memiliki *similarity*.
- c. Dapat melakukan pendeteksian *similarity* tidak hanya di bagian abstrak saja.

## DAFTAR PUSTAKA

- Abdullah, I., & Aribowo, E. (2018). Rancang Bangun Aplikasi Pengecekan Kemiripan Judul Skripsi Dengan Metode Cosine Similarity (Studi Kasus : Program Studi Teknik Informatika Uad). *JSTIE (Jurnal Sarjana Teknik Informatika) (E-Journal)*, 6(2), 43–52. <https://doi.org/10.12928/jstie.v6i2.15241>
- Alamanda, R., Suhery, C., & Brianorman, Y. (2016). Aplikasi pendeteksi plagiat terhadap karya tulis berbasis web menggunakan natural language processing dan algoritma knuth-morris-pratt. *Jurnal Coding, Sistem Komputer Untan*, 04(1).
- Anggara, Y. (2016). Deteksi plagiarisme dokumen bahasa indonesia dengan algoritma jaro-winkler Distance. *Program Studi Teknik Informatika Universitas Muhammadiyah Jember*.
- Ariantini, D. A. R., Lumenta, A. S. M., & Jacobus, A. (2016). Pengukuran Kemiripan Dokumen Teks Bahasa Indonesia Menggunakan Metode Cosine Similarity. *Jurnal Teknik Informatika*, 9(1), 1–8. <https://doi.org/10.35793/jti.9.1.2016.13752>
- Fataruba, F. (2018). Penerapan Metode Cosine Similarity Untuk Pengecekan Kemiripan Jawaban Ujian Siswa. *Jurnal Mahasiswa Teknik Informatika*, 2(2), 88–95.
- Februariyanti, H. (2012). Klasifikasi Dokumen Berita Teks Bahasa Indonesia menggunakan Ontologi. *Teknologi Informasi DINAMIK*, 17(1), 14–23. Retrieved from <http://www.unisbank.ac.id/ojs/index.php/fti1/article/view/1612/594>
- Filcha, A., & Hayaty, M. (2019). Implementasi Algoritma Rabin-Karp untuk Pendeteksi Plagiarisme pada Dokumen Tugas Mahasiswa. *JUITA : Jurnal Informatika*, 7(1), 25. <https://doi.org/10.30595/juita.v7i1.4063>
- Fitri, S., Nurjanah, N., & Astuti, W. (2018). Penerapan Data Mining Untuk Evaluasi Kinerja Akademik Mahasiswa (Studi Kasus: Umtas). *Simetris: Jurnal Teknik Mesin, Elektro Dan Ilmu Komputer*, 9(1), 633–640. <https://doi.org/10.24176/simet.v9i1.2002>
- Gunadi, G., & Sensuse, D. I. (2012). Penerapan Metode Data Mining Market Basket Analysis Terhadap Data Penjualan Produk Buku Dengan Menggunakan Algoritma Apriori Dan Frequent Pattern Growth ( Fp-Growth ) : *Telematika*, 4(1), 118–132.
- Khidfi, M. N., & Sari, J. Y. (2018). Rancang bangun aplikasi pendeteksian kesamaan pada dokumen teks menggunakan algoritma Enhanced confix stripping dan algoritma winnowing. *Jurusan Teknik Informatika, Fakultas*

- Teknik Universitas Halu Oleo, Kendari*, 4(2), 1–10.  
<https://doi.org/10.5281/zenodo.1407866>
- Nurdin, N., Rizal, R., & Rizwan, R. (2019). Pendeteksian Dokumen Plagiarisme dengan Menggunakan Metode Weight Tree. *Telematika*, 12(1), 31.  
<https://doi.org/10.35671/telematika.v12i1.775>
- Priambodo, J. (2018). Pendeteksian Plagiarisme Menggunakan Algoritma Rabin-Karp dengan Metode Rolling Hash. *Jurnal Informatika Universitas Pamulang*, 3(1), 39. <https://doi.org/10.32493/informatika.v3i1.1518>
- Rahmatulloh, A., Kurniati, N. I., Darmawan, I., Asyikin, A. Z., & Witarsyah, J. D. (2019). Comparison between the stemmer porter effect and nazief-adriani on the performance of winnowing algorithms for measuring plagiarism. *International Journal on Advanced Science, Engineering and Information Technology*, 9(4), 1124–1128. <https://doi.org/10.18517/ijaseit.9.4.8844>
- Rinartha, K. (2017). Pemodelan Penilaian Essay Otomatis Secara Realtime Menggunakan Kombinasi Text Stemming Dan Cosine Similarity. *Konferensi Nasional Sistem & Informatika*, 322–327.
- Shu, K., Wang, S., Lee, D., & Liu, H. (2020). Mining disinformation and fake news: Concepts, methods, and recent advancements. *International Journal of Computer Applications*, 68(13), 13–18. [https://doi.org/10.1007/978-3-030-42699-6\\_1](https://doi.org/10.1007/978-3-030-42699-6_1)
- 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>
- Sunyoto, A., & Informatika, T. (2013). Implementasi Algoritma Rabin Karp untuk Pendeteksian Plagiat Dokumen Teks Menggunakan Konsep Similarity, 23–28.
- Tala, F. Z. (2003). A Study of Stemming Effects on Information Retrieval in Bahasa Indonesia. *M.Sc. Thesis, Appendix D, pp*, 39–46.
- Ulfa, N. F., Mustikasari, M., & Bastian, I. (2016). Pendeteksian tingkat similaritas dokumen berbasis web menggunakan algoritma winnowing. *Konferensi Nasional Teknologi Informasi Dan Komunikasi (KNASTIK)*, (November), 194–203.
- Wang, J., & Dong, Y. (2020). Measurement of Text Similarity: A Survey. *Information*, 11(9), 421. <https://doi.org/10.3390/info11090421>