BAB III
METODOLOGI PENELITIAN

3.1 TujuanOperasional Penelitian

Tujuandaripenelitianiniadalahpemanfaatan algoritma *TermFrequency-InverseDocumentFrequency* dalam menciptakan sebuah aplikasi yang dapat mengelompokkan dokumen karya akhir mahasiswa kedalam kategori yang telah ditentukan berdasarkan abstrak bahasa Indonesia.

3.2 TempatdanWaktuPenelitian

Penelitiandilakukan dilaboraturiumkomputer MultimediaJurusanTeknikElektroUniversitasNegeri Jakarta sejakbulanJuni2014 hinggaJanuari 2015.

3.3 DefinisiKebutuhan(*Requirements Definition*)

Pemanfaatanalgoritma TF-IDF (*TermFrequency-InverseDocumentFrequency*) dalam penelitian ini memiliki fitur dan *output* yang dapat membantu *user* dalam mengklasifikasi dokumen karya akhir kedalam kategori tertentu berdasarkan abstrak bahasa Indonesia.

Pada tahap definisi kebutuhan ini, langkah yang dilakukan adalah menciptakan sebuah sistem untuk aplikasi yang dapat mengimplementasi algoritma TF-IDF untuk dapat mengklasifikasi dokumen karya akhir mahasiswa berdasarkan abstrak bahasa Indonesia, sehingga dapat digunakan untuk membatu mempermudah dalam mengklasifikasi dokumen karya akhir mahasiswa.

Tabel 3.1 Daftar kebutuhan fungsional

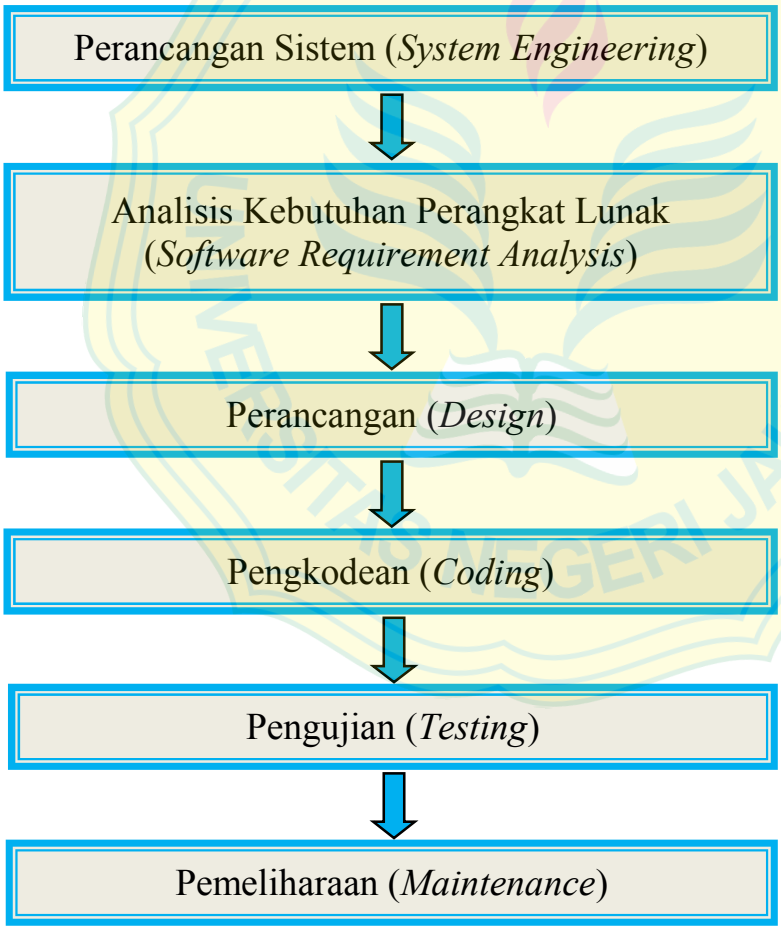
Aspek	Kebutuhan
Content	Suatu aplikasi perangkat lunak yang dapat mengklasifikasi kategori dokumen karya akhir kedalam kategori tertentu berdasarkan abstrak.
	Memberikan pengeluaran berupa Nilai perhitungan algoritma TF-IDF perkategori karya akhir.
User	*User* dapat mengetahui hasil algoritma TF-IDF berupa nilai sehingga dapat membantu *user* dalam mengklasifikasi dokumen karya akhir ke dalam kategori masing – masing.

3.4 MetodePenelitian

Metode yang digunakanpenulis dalampenelitianiniadalahmetode*Waterfall*. Metode pengembangan software ini bersifat sekuensial. Menurut Roger S. Pressman (1992:39), model *Waterfall* adalah metode klasik yang bersifat sistematis, berurutan dalam membangun software. Inti dari metode *waterfall* adalah pengerjaan dari suatu sistem dilakukan secara berurutan atau linear. Jadi jika tahap ke-1 belum dikerjakan, maka tidak akan bisa melanjutkan ke langkah 2, 3 dan seterusnya.

Aplikasi pada penelitian skripsi inidibuatdengan menggunakanbahasapemrogramanPHP, MySQL dan HTML. Pada tahap uji coba, dokumen yang digunakan adalah abstrak dari skripsi mahasiswa yang sudah lulus untuk menguji keberhasilan aplikasi dalam mengklasifikasi dokumen karya akhir. *Output*dari aplikasi iniadalah nilai persentase untuk masing-masing kategori karya akhir yang telah ditentukan.

Setelah disesuaikan dengan kebutuhan aplikasi, maka tahapan-tahapan dari metode *waterfall* yang diterapkan pada penelitian ini adalah:



Gambar 3.1 Tahapan yang Diterapkan

3.4.1 Perancangan Sistem

Tahap Perancangan sistem merupakan tahap awal yang menjelaskan sistem aplikasi secara garis besar. Perancangan sistem disini meliputi cara kerja sistem, ruang lingkup atau batasan sistem, serta tujuan yang dihasilkan sistem/*output*.

Cara kerja sistem pengklasifikasi dokumen karya akhir berdasarkan abstrak dengan menerapkan algoritma TF-IDF adalah:

1. *User* memasukkan dokumen abstrak yang ingin diklasifikasi.
2. Sistem mengolah dan memberikan *output* berupa kategori yang cocok bagi dokumen abstrak tersebut.

Sistem aplikasi ini menggunakan algoritma TF-IDF untuk melakukan pembobotankata abstrak karya akhir terhadap kumpulan kata yang terdapat dalam *database*setiap kategori. *Output* yang dihasilkan dari aplikasi ini berupa kategori yang sesuai menurut perhitungan TF-IDF.

3.4.2 Analisis Kebutuhan Perangkat Lunak

Analisis kebutuhan perangkat lunak merupakan tahap kedua dalam metode *waterfall* setelah perancangan sistem (*system engineering*), yang bertujuan untuk memahami kebutuhan dasar perangkat lunak mengenai ruang lingkup informasi yang diperlukan dalam menerapkan algoritma TF-IDFdalam sistem aplikasi pengklasifikasi dokumen karya akhir yang akan dibuat. Kebutuhan yang diperlukan untuk membangun aplikasi ini adalah:

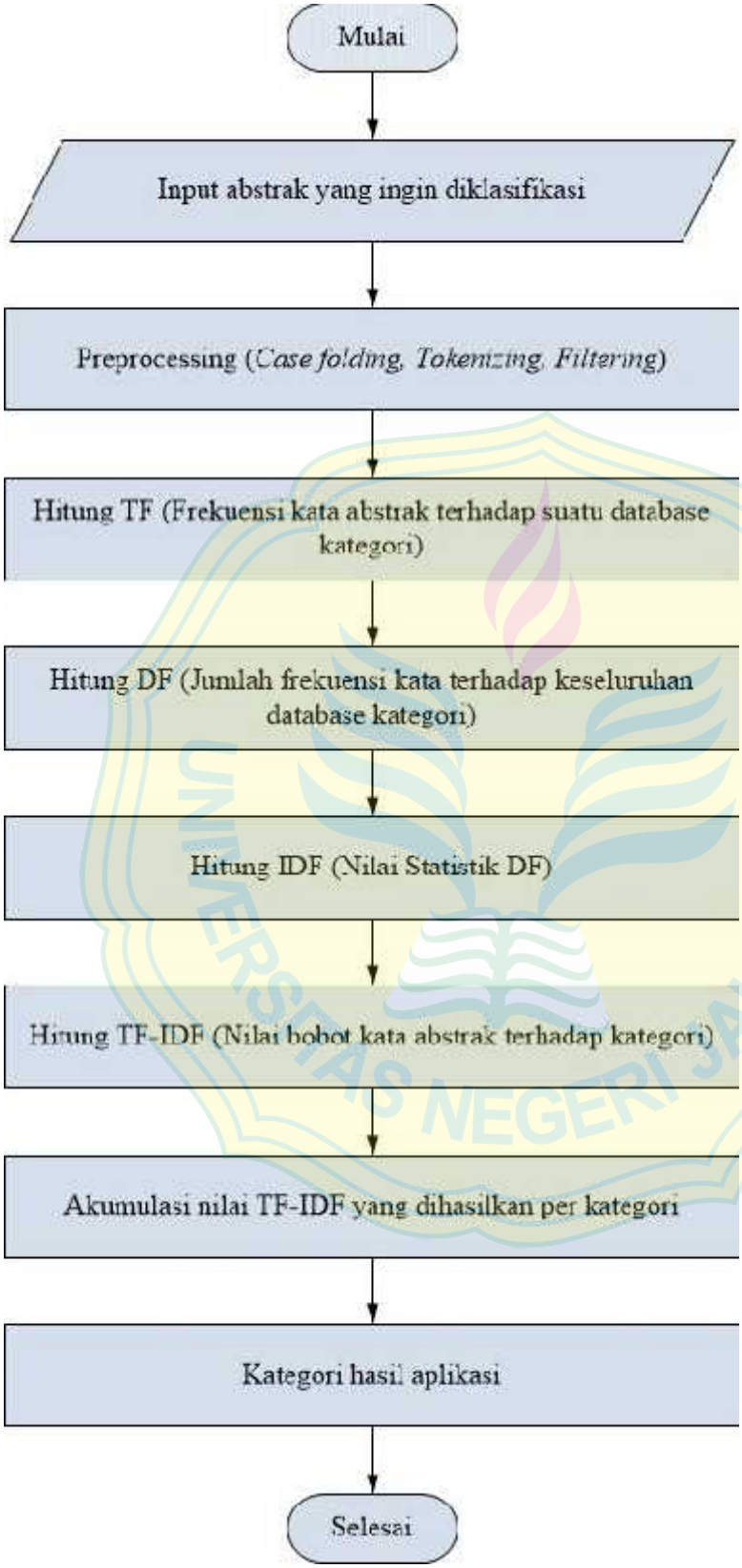
1. Perangkat lunak harus dapat melakukan *pre-processing* sebelum tahap perhitungan algoritma TF-IDF yaitu meliputi *case folding, tokenizing, filtering*.

2. Perangkat lunak harus dapat melakukan perhitungan algoritma TF-IDF yang sudah dimodifikasi seperti penjelasan pada bab II.
3. Perangkat lunak harus mampu menghasilkan dugaan kategori yang sesuai dengan abstrak yang di-*input.*

3.4.3 Perancangan

3.4.3.1 Perancangan Proses

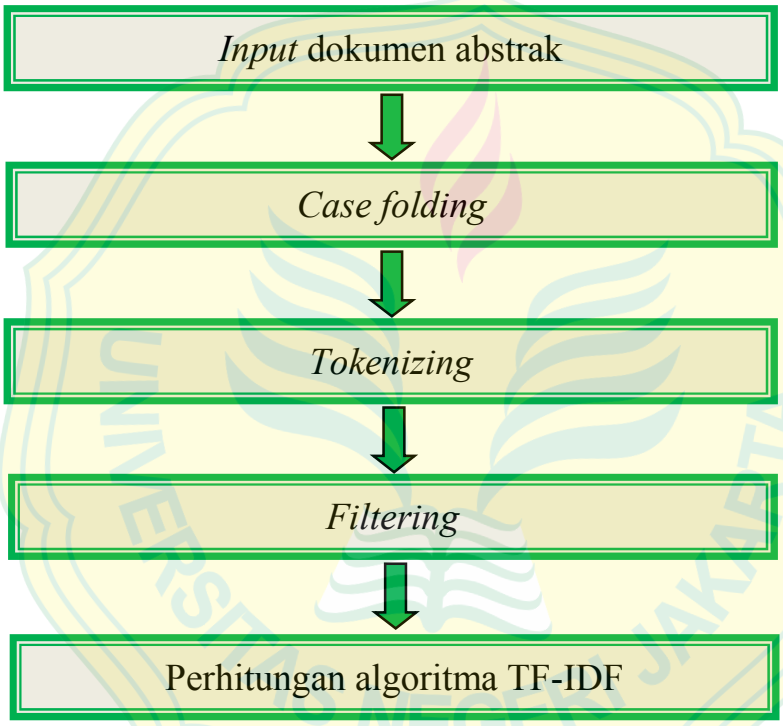
Penerapan algoritma TF (*TermFrequency*) – IDF (*InversedDocumentFrequency*) dalam aplikasi pengklasifikasi dokumen, terdiri dari 2 tahap utama, yaitu tahap *preprocessing* dan penghitungan. Tahap *preprocessing* yang dilakukan pada sistem aplikasi ini adalah *case folding, tokenizing,* dan *filtering.* Tahap *preprocessing* dilakukan untuk mengurangi *noise*yang terdapat dalam dokumen. Hasil yang diperoleh dari tahap *preprocessing* ini berupa kumpulan kata yang siap untuk dibandingkan dengan kumpulan kata kunci yang terdapat dalam *database*. Algoritma TF-IDF diterapkan pada saat dilakukan penghitungan frekuensi kata hasil dari *preprocessing* dengan kumpulan kata kunci dalam *database.* Selain digunakan untuk melakukan penghitungan frekuensi, algoritma TF-IDF ini juga digunakan untuk menghitung nilai TF-IDF tiap kata terhadap kata kunci dalam *database. Flowchart* proses aplikasi dapat dilihat pada gambar 3.2.


Mulai
Input abstrak yang ingin diklasifikasi
Preprocessing (Case folding, Tokenizing, Filtering)
Hitung TF (Frekuensi kata abstrak terhadap suatu database kategori)
Hitung DF (Jumlah frekuensi kata terhadap keseluruhan database kategori)
Hitung IDF (Nilai Statistik DF)
Hitung TF-IDF (Nilai bobot kata abstrak terhadap kategori)
Akumulasi nilai TF-IDF yang dihasilkan per kategori
Kategori hasil aplikasi
Selesai

Gambar 3.2Flowchart Aplikasi

1) *Preprocessing*

Seperti yang sudah dibahas pada bab sebelumnya, fungsi dari *preprocessing* pada aplikasi ini adalah untuk mendapatkan keyword yang nantinya akan dibandingkan dengan kata kunci yang terdapat didalam *database.* Alur *preprocessing* yang terdapat dalam sistem pengklasifikasi kategori karya akhir ini yaitu :



Gambar 3.3 Alur *Preprocessing*Sistem

Sesuai dengan gambar diatas, setelah *user* melakukan *input* abstrak, tahap *preprocessing* pertama yang dilakukan pada sistem ini adalah *case folding,* yaitu mengubah semua huruf yang terdapat didalam abstrak menjadi huruf kecil. Kemudian dilakukan tahap *tokenizing,* yaitu memecah sekumpulan kalimat atau paragraf menjadi kata per kata. Tahap terakhir pada *preprocessing* dalam sistem ini adalah tahap *filtering*, tahap ini membuang

kata yang kurang penting. Setelah semua tahap *preprocessing* dilakukan, maka perhitungan algoritma TF-IDF dijalankan.

1) *Case Folding*

Case folding merupakan tahap *preprocessing* yang mengubah semua huruf dalam dokumen menjadi huruf kecil. Proses ini dilakukan agar mencegah terjadinya perbedaan penggunaan huruf besar-kecil pada tiap kata yang akan di bandingkan dengan *database* yang terdapat dalam sistem aplikasi. Penggunaan tahap case folding ini membuat huruf besar-kecil yang terdapat dalam abstrak yang akan di*input* tidak berpengaruh. Contoh tahap *case*

Sensor arus hampir mendekati dengan alat ukur amperemeter. Program ladder yang dibuat melalui software Cx – Programmer dapat bekerja sesuai yang diinginkan setelah mendapat *input* dari sensor arus. Simulator rangkaian instalasi listrik satu fasa yang telah mengalami beban lebih dapat memindahkan sebagian bebannya menuju grup beban cadangan dengan bantuan relai setelah mendapatkan perintah dari PLC dan ketika keadaan normal maka kembali ke kondisi semula. Kesimpulan dari penelitian ini adalah sistem pembagi daya otomatis ini dapat bekerja setelah PLC mendapatkan *input* dari sensor arus ketika terjadi beban lebih kemudian untuk memindahkan sebagian bebannya menuju grup beban cadangan agar tidak terjadi pemutusan aliran listrik.

(Dokumen *input*)

sensor arus hampir mendekati dengan alat ukur amperemeter. program ladder yang dibuat melalui software cx – programmer dapat bekerja sesuai yang diinginkan setelah mendapat *input* dari sensor arus. simulator rangkaian instalasi listrik satu fasa yang telah mengalami beban lebih dapat memindahkan sebagian bebannya menuju grup beban cadangan dengan bantuan relai setelah mendapatkan perintah dari plc dan ketika keadaan normal maka kembali ke kondisi semula. kesimpulan dari penelitian ini adalah sistem pembagi daya otomatis ini dapat bekerja setelah plc mendapatkan *input* dari sensor arus ketika terjadi beban lebih kemudian untuk memindahkan sebagian bebannya menuju grup beban cadangan agar tidak terjadi pemutusan aliran listrik.

(Dokumen hasil *case folding*)

Gambar 3.4 Tahap *Case Folding*

2) *Tokenizing*

Tahap *tokenizing*yaitu tahap pemecahan kalimat yang ada di dalam abstrak menjadi kata. *Tokenizing* akan mengurai seluruh kalimat yang terdapat dalam abstrak menjadi kata-kata. Karakter titik dan spasi memiliki peranan yang penting dalam tahap ini. Karakter titik digunakan sebagai *delimiter* untuk memecah isi teks dalam abstrak menjadi kumpulan kalimat-kalimat. Sedangkan karakter spasi digunakan sebagai *delimiter* untuk memecah kalimat menjadi kumpulan kata kata.

sensor	simulator	dan	arus
arus	rangkaian	ketika	ketika
hampir	instalasi	keadaan	terjadi
mendekati	listrik	normal	beban
dengan	satu	maka	lebih
alat	fasa	kembali	kemudian
ukur	yang	ke	untuk
amperemeter	telah	kondisi	memindahkan
program	mengalami	semula	sebagian
ladder	beban	kesimpulan	bebannya
yang	lebih	dari	menuju
dibuat	dapat	penelitian	grup
melalui	memindahkan	ini	beban
software	sebagian	adalah	cadangan
cx	bebannya	sistem	agar
programmer	menuju	pembagi	tidak
dapat	grup	daya	terjadi
bekerja	beban	otomatis	pemutusan
sesuai	cadangan	ini	aliran
yang	dengan	dapat	listrik
diinginkan	bantuan	bekerja	
setelah	relai	setelah	
mendapat	setelah	plc	
input	mendapatkan	mendapatkan	
dari	perintah	*input*	
sensor	dari	dari	
arus	plc	sensor	

(Dokumen hasil *tokenizing*)

Gambar 3.5 Tahap *Tokenizing*

3) *Filtering*

Tahap *Filtering*adalah tahap penghapusan atau pembuangan kata kurang penting yang terdapat dalam abstrak yang di*input*. Tahap ini dapat menggunakan 2 algoritma, yaitu*stoplist*dan *wordlist. Stoplist*adalah proses pembuangan kata yang kurang penting, misalnya: atau, dan, di, yang, dan sebagainya. Sedangkan*wordlist*yaitu menyimpan kata-kata penting yang dibutuhkan dalam proses. Algoritma yang digunakan dalam tahap filtering pada aplikasi ini adalah algoritma *stoplist* (membuang kata yang kurang penting).

sensor	beban	bekerja
arus	memindahkan	plc
mendekati	sebagian	*input*
alat	bebannya	sensor
ukur	bebannya	arus
amperemeter	menuju	beban
program	grup	memindahkan
ladder	beban	sebagian
software	cadangan	bebannya
cx	bantuan	menuju
programmer	relai	grup
bekerja	perintah	beban
input	plc	cadangan
dari	keadaan	pemutusan
sensor	normal	aliran
arus	kondisi	listrik
simulator	semula	
rangkaian	kesimpulan	
instalasi	penelitian	
listrik	sistem	
satu	pembagi	
fasa	daya	
mengalami	otomatis	

(Dokumen hasil *filtering*)

Gambar 3.6 Tahap *Filtering*

2) Algoritma TF-IDF (*TermFrequency-InversedDocumentFrequency*)

Setelah proses *preprocessing*, tahap berikutnya adalah menghitung frekuensi kemunculan kata yang terdapat di dalam abstrak(selanjutnya disebut kata abstrak) terhadap kata yang sudah tersimpan di dalam *database* (selanjutnya disebut *database*) dengan metode TF(*TermFrequency*) yang terdapat dalam algoritma TF-IDF. Setelah ditemukan frekuensinya, maka pembobotan TF-IDF per kata bisa dilakukan.

Peran *database* dalam proses ini sangat penting dalam mengklasifikasi abstrak kedalam kategori yang sesuai. Masing-masing kategori mempunyai *database*yang berbeda.*Database* disini berupa kumpulan kata-kata yang sering dipakai dalam abstrak dengan kategori yang bersangkutan.Contoh Proses implementasi algoritma TF-IDF yaitu:
Misalnya terdapat dua kategori, yaitu pendidikan dan informatika. Dan contoh *database* pendidikan dan informatika adalah sebagai berikut:

Tabel 3.2 Contoh *Database*

No	Pendidikan	Informatika
1	pelajaran	aplikasi
2	pendidikan	database
3	silabus	informatika
4	belajar	java
5	kurikulum	software
6	kompetensi	php

7	siswa	mysql
8	interaktif	waterfall
9	ceramah	website
10	materi	server
11	soal	tfidf
12	prestasi	ip

Kemudian contoh kata abstrak yang di *input* adalah:

> Penelitian dilakukan dengan tujuan mengembangkan aplikasi pengklasifikasi dokumen karya akhir menggunakan metode TF-IDF di Jurusan Teknik Elektro Universitas Negeri Jakarta. Sistem diimplementasi menggunakan PHP dan

Gambar 3.7 Contoh Abstrak

1) Menghitung *TermFrequency*dan *DocumentFrequency*

   Proses perhitungan ini dilakukan setelah melalui proses *preprocessing*.

Tabel 3.3 Hasil *TermFrequency*

No	Kata	Pendidikan	Informatika
1	penelitian	0	0
2	tujuan	0	0
3	mengembangkan	0	0
4	aplikasi	0	1
5	pengklasifikasi	0	0
6	dokumen	0	0
7	karya	0	0
8	metode	0	0
9	tfidf	0	1
10	jurusan	0	0
11	teknik	0	0
12	elektro	0	0
13	universitas	0	0
14	negeri	0	0
15	jakarta	0	0
16	sistem	0	0
17	implementasi	0	0
18	php	0	1
19	mysql	0	1

Pada tabel diatas dapat diketahui bahwa kata abstrak yang termasuk dalam kategori informatika terdapat 4 kata. Pada kasus seperti ini, sebenarnya sudah dapat disimpulkan bahwa abstrak

diatas *term*asuk dalam kategori informatika. Berbeda halnya jika menggunakan abstrak yang kata-kata di dalamnya hampir ada di setiap kategori yang berbeda, selain itu kategori yang ditentukan lebih dari 2. Sehingga kemungkinan jumlah frekuensi bisa sama antar kategori.

Setelah mendapatkan TF, perhitungan DF dapat dikerjakan. DF adalah *DocumentFrequency*, maksudnya yaitu jumlah *frequency*kata dari keseluruhan kategori.

Tabel 3.4 Jumlah *DocumentFrequency*

No	Kata	DF
1	penelitian	0
2	tujuan	0
3	mengembangkan	0
4	aplikasi	1
5	pengklasifikasi	0
6	dokumen	0
7	karya	0
8	metode	0
9	tfidf	1
10	jurusan	0
11	teknik	0
12	elektro	0
13	universitas	0

14	negeri	0
15	jakarta	0
16	sistem	0
17	implementasi	0
18	php	1
19	mysql	1

2) Menghitung IDF (*InversedDocumentFrequency*)

Rumus dari IDF adalah:

$$IDF_t = \log(\frac{D}{DF_t})$$

Dimana D disini adalah jumlah kategori yang ada. Kategori yang ada pada contoh disini yaitu 2.

$$IDF = \log(\frac{2}{DF})$$

Tabel 3.5 Hasil *InversedDocumentFrequency*

No	Kata	DF	IDF
1	Penelitian	0	0
2	Tujuan	0	0
3	mengembangkan	0	0
4	Aplikasi	1	0.301
5	pengklasifikasi	0	0

6	Dokumen	0	0
7	Karya	0	0
8	Metode	0	0
9	Tfidf	1	0.301
10	Jurusan	0	0
11	Teknik	0	0
12	Elektro	0	0
13	Universitas	0	0
14	Negeri	0	0
15	Jakarta	0	0
16	Sistem	0	0
17	implementasi	0	0
18	Php	1	0.301
19	Mysql	1	0.301

3) Menghitung TF-IDF

$$W = TF * IDF$$

Rumus diatas adalah rumus untuk pembobotan TF-IDF. Pembobotan yang dilakukan nantinya akan di jumlah per masing-masing kategori. Sehingga setiap kategori memiliki jumlah bobot TF-IDF. Jumlah TF-IDF yang paling besar merupakan kategori yang dihasilkan.

Tabel 3.6 Hasil TF-IDF

No	Kata	TF-IDF Pendidikan	TF-IDF Informatika
1	penelitian	0	0
2	tujuan	0	0
3	mengembangkan	0	0
4	aplikasi	0	0.301
5	pengklasifikasi	0	0
6	dokumen	0	0
7	karya	0	0
8	metode	0	0
9	tfidf	0	0.301
10	jurusan	0	0
11	teknik	0	0
12	elektro	0	0
13	universitas	0	0
14	negeri	0	0
15	jakarta	0	0
16	sistem	0	0
17	implementasi	0	0
18	php	0	0.301
19	mysql	0	0.301

Dari tabel di atas dapat diketahui bahwa TF-IDF pendidikan berjumlah 0, sedangkan TF-IDF Informatika berjumlah 1,204. Jadi

disimpulkan bahwa berdasarkan metode TF-IDF, abstrak yang diinput merupakan kategori dari Informatika.

3.4.3.2 Perancangan Implementasi dan Uji Coba

1) Lingkungan Implementasi dan Uji Coba

Implementasi dan uji coba membutuhkan dukungan perangkat keras dan perangkat lunak. Adapun spesifikasi perangkat keras dan perangkat lunak yang digunakan dalam pembuatan serta implementasi pengklasifikasi dokumen karya akhir berda

sarkan abstrak adalah:

1) Perangkat Keras (Hardware)
    1) ProcessorIntel® Core™ i5-2400 CPU @ 3.10 GHz 3.09 GHz
    2) Memori RAM 4GB DDR3
    3) Harddisk 500GB
    4) LED monitor 14”
    5) Keyboard dan mouse
2) Perangkat Lunak (Software)
    1) Sistem operasi Windows 7 Ultimate 64-bit
    2) Netbeans IDE 8
    3) JDK 1.8
    4) MySQL *Database* 5.5.32
    5) XAMPP for Windows x64 1.8.2

2) Perancangan *Database*

Database pada penerapan metode TF-IDF ini berguna untuk mempermudah sistem dalam mengklasifikasi dokumen karya akhir. Sistem

akan membandingkan data yang di *input* oleh *user* dengan data yang ada di dalam *database*. Pada perancangan *database* ini, data yang digunakan untuk disimpan kedalam *database* berasal dari abstrak skripsi yang sudah di klasifikasi secara manual. Dalam pengambilan kata kunci yang menjadi acuan dalam penentuan kategori, kata kunci diambil berdasarkan 3 kata yang memiliki frekuensi tertinggi dalam abstrak yang akan diambil untuk dimasukkan ke dalam database. Pengambilan kata kunci dalam database kategori, dilakukan terlebih dahulu sebelum pengujian pada 200 dokumen abstrak. Selain itu, dokumen abstrak yang akan diuji harus berbeda dengan dokumen abstrak untuk pengambilan database kategori. Contoh dalam pengambilan database kategori yaitu:

PEMBUATAN SIMULASI DETEKTOR KEBOCORAN GAS LPG DENGAN SENSOR MQ-5 SEBAGAI INPUT PLC
Penelitianinibertujuanuntukmembuatsistempendeteksikebocoran gas LPG secaraotomatisdalambentuk model simulasidenganmenggunakan Programmable Logic Controller sebagaikontrol. Pelaksanaanpembuatanskripsidilakukan di Laboratorium Programmable Logic Controller (PLC), JurusanTeknikElektro, FakultasTeknik, UniversitasNegeri Jakarta. Penelitianinidilakukankarenatingginyapresentasekebakaranakibatkebocoran gas LPG 3 kg, disebabkankarenafaktorkebocoranpadatabung gas LPG 3 kg, selang yang bocorpada LPG 3 kg, dansil yang rusakpada LPG 3 kg. Hal inimenyebabkanseringterjadikebakaranakibatbocornyatabung gas LPG 3 kg. Untukmengetahuiadanyakebocoran gas LPG (PropanadanButana) makakitalakukanpenelitianuntukmembuatsimulasidetektorkebocoran gas LPG yang dirancanguntukmengantisipasiefeknegatif LPG sebagaibahanbakar. Cara mendeteksikebocoran gas LPG adalahsaat sensor MQ-5 sebagai input PLC tidakmendeteksiadanyakebocoran gas makasistemdalamkeadaan standby, tetapiketikaadanyakebocoranpada gas LPG 3 kg maka sensor MQ-5 akanmendeteksiadanyakebocoranpada gas tersebut, dansetelahkadar gas LPG (PropanadanButana) terdeteksimaka alarm, selenoid valve, danlampu gas fault akanbekerjasecarabersamaan.Kesimpulanalatpendeteksikebocoran gas LPG bekerjasecaraotomatisdengansangatbaik, haliniditunjukanketeikaalatmendeteksiadanyakebocorantabung gas

Gambar 3.8 Contoh Perancangan Database

Gambar 3.8 merupakan salah satu abstrak yang akan digunakan dalam pengambilan kata kunci dari kategori elektronika. Berdasarkan gambar tersebut, diperoleh hasil frekuensi yang diperlihatkan pada tabel 3.7 dibawah ini.

Tabel 3.7 Hasil Frekuensi Perancangan Database

Kata	Frekuensi
simulasi	3
bocor/kebocoran	12
gas	15
lpg	16
sensor	4
mq5	4
plc	4
penelitian	3
pendeteksi	3
tabung	3
propana	3
butana	3
kerja	3
alat	3

Pada tabel 3.7 dapat terlihat perbedaan warna pada tabel. Warna yang menunjukkan bahwa frekuensi kata yang terdapat didalam abstrak

lebih dari 10 kata dan kata tersebut akan dimasukkan ke dalam database kategori elektronika. Sedangkan yang berwarna dimasukkan ke dalam database karena merupakan 3 frekuensi tertinggi yang kurang dari 10. Jadi kata yang diambil untuk database kategori, adalah kata yang mempunyai frekuensi lebih dari 10 dan 3 frekuensi tertinggi di bawah 10.

Selain terdapat *database* untuk perbandingan data abstrak yang di *input*, didalam *database* ini juga terdapat *stopword*yang berguna untuk melakukan proses *filtering*. Pada *stopword* ini terdapat 1151 kumpulan kata yang tidak terpakai dalam proses pengklasifikasian, sehingga berguna untuk menghilangkan atau membuang kata yang kurang penting.

3) Bahan Pengujian

Uji coba terhadap aplikasi ini dilakukan untuk mengetahui kesalahan dan kekurangan pada sistem aplikasi, sehingga bisa diperbaiki sebelum digunakan oleh *user*. Untuk uji coba, data yang digunakan adalah 200 abstrak skripsi dari berbagai program studi yang berasal dari Jurusan Teknik Elektro Universitas Negeri Jakarta.

NOREG	NAMA	JUDUL
[illegible]	[illegible]	ANALISIS SISTEM [illegible] TEGANGAN MENENGAH [illegible] KV STUDY PADA APARTEMEN "X", JAKARTA TIMUR
5115077450	YUSUF ARDI	RANCANG BANGUN SIMULATOR INTERLOCK SISTEM DOUBLE BUSBAR BERBASIS PLC DAN SCADA (STUDI EKSPERIMEN PEMBUATAN SISTEM INTERLOCK PADA SEBUAH GARDU INDUK YANG DILAKUKAN DI LABORATORIUM PLC FAKULTAS TEKNIK UNIVERSITAS NEGERI JAKARTA)
5115077494	AGUS [illegible]	ANALISIS PENGARUH BATTERY CHARGE REGULATOR (BCR) TERHADAP KINERJA PEMBANGKIT LISTRIK TENAGA SURYA (STUDI KASUS PADA PANEL PEMBANGKIT LISTRIK TENAGA SURYA DI PUSPITEK SERPONG)
[illegible]	ARI CHANDRA PUTRA	ANALISIS POWER AND COS ? METER ELEKTRIK PADA BALLAST YANG ADA DI PASARAN (STUDI EKSPERIMEN DI LABORATORIUM TEKNIK INSTALASI LISTRIK TEKNIK ELEKTRO FT UNJ)
5115080255	ROTA MASA	KESEIMBANGAN BEBAN DAN KETIDAKSTABILAN DAYA PASCA PENGGUNAAN GEDUNG SERTIFIKASI GURU DAN PUSAT STUDI UNJ
[illegible]	ANTONY CHRISTOVEL [illegible]	UJI STANDARISASI KABEL [illegible] MM2 BEBERAPA MEREK KABEL BERDASARKAN PENGUJIAN LISTRIK DAN PENGUJIAN NON LISTRIK MENURUT SPLN 43-5-4 (STUDI PADA PT PLN PERSERO PUSAT PENELITIAN DAN PENGEMBANGAN KETENAGALISTRIKAN DUREN TIGA)
[illegible]	FAJAR R SUHADA	SIMULASI PENGALIH DAYA OTOMATIS BERBASIS PROGRAMMABLE LOGIC CONTROLLER (PLC)
[illegible]	[illegible]	[illegible]

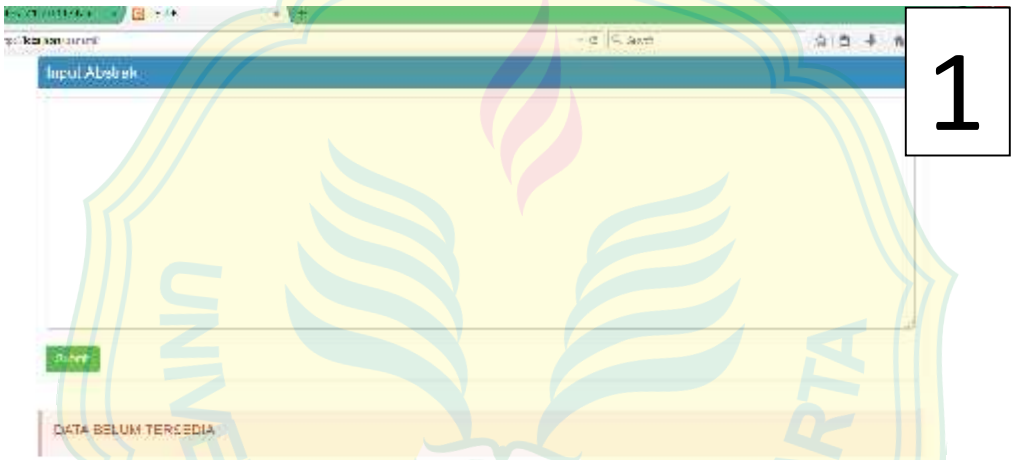
Gambar 3.9Kumpulan Abstrak Uji Coba

3.4.3.3 Perancangan *Input* dan *UserInterface* Aplikasi

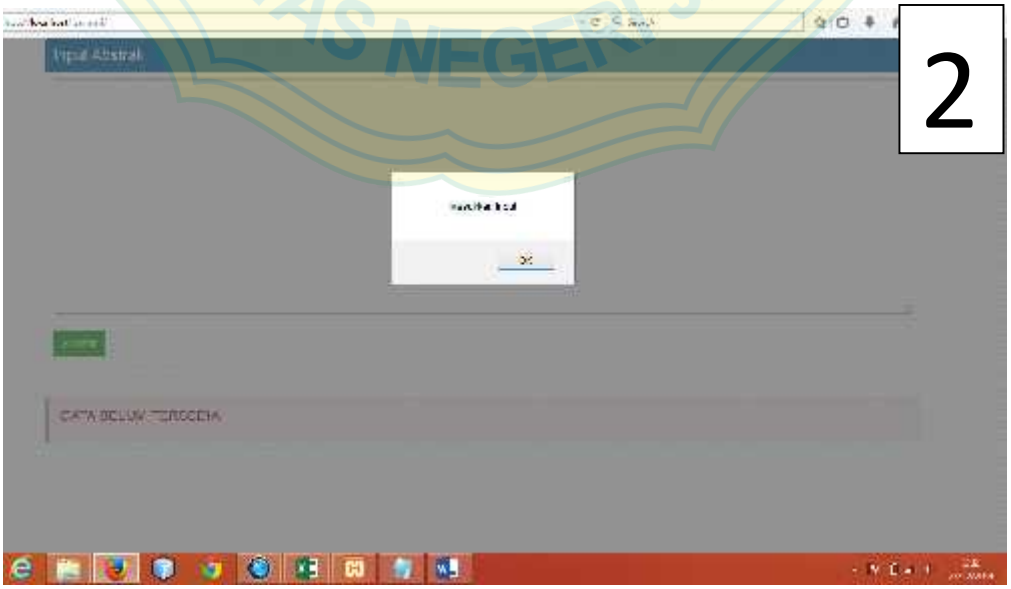
1) Perancangan *Input* pada Aplikasi

   Untuk meng-*input*abstrak yang akan di klasifikasi, *user* harus mengetik manual atau dengan teknik *copy-paste* di dalam *text field.*

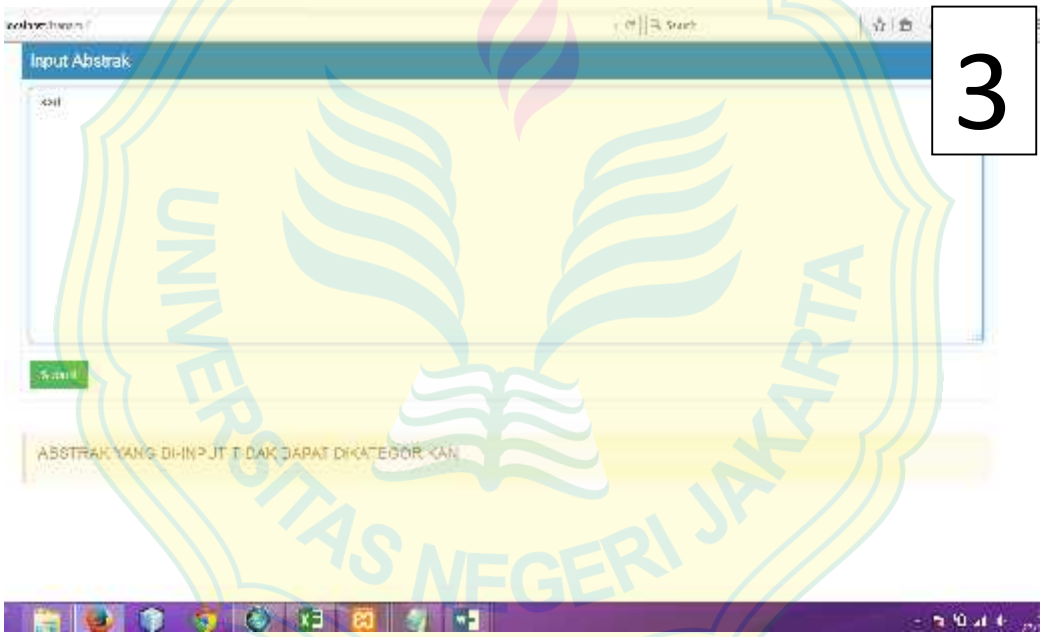
2) Perancangan *UserInterface* pada Aplikasi



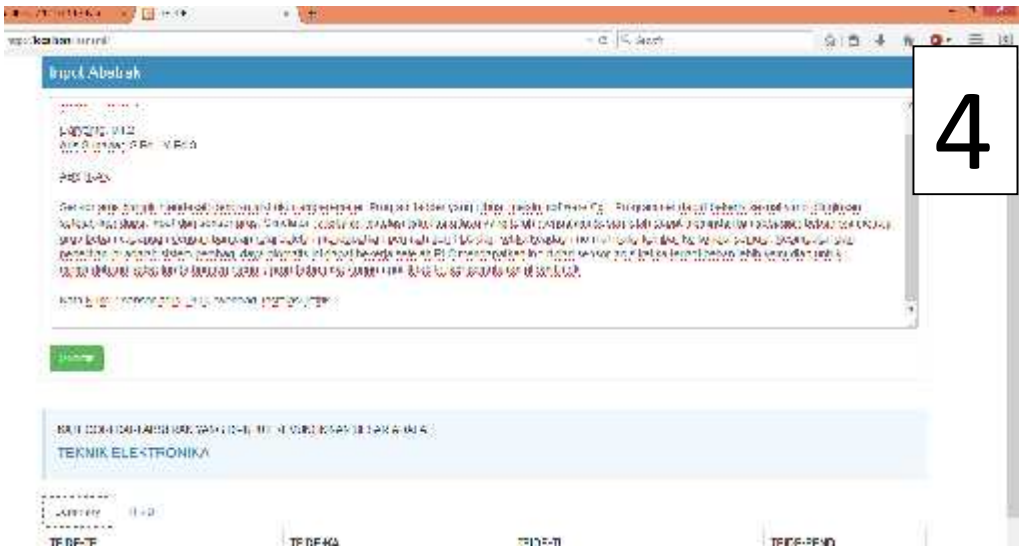
Gambar 3.10Tampilan Pertama Aplikasi



Gambar 3.11Tampilan Saat Tidak Ada *Input*



Gambar 3.12Tampilan Jika Tidak Terkategori



Gambar 3.13Tampilan Setelah *Input* Abstrak

Gambar 3.14Tampilan Tab Summary

Gambar 3.15Tampilan Tab TF-IDF

1) Gambar 1 merupakan tampilan awal saat baru masuk aplikasi pengklasifikasi karya akhir.

2) Gambar 2 menunjukkan saat *user* menekan submit atau enter pada keyboard tanpa ada kata yang di*input*.
3) Gambar 3 menunjukkan saat kata yang di*input* oleh *user* tidak ada didalam *database*, sehingga tidak dapat dikategorikan.
4) Gambar 4 adalah tampilan aplikasi saat abstrak sudah di*input* dan sudah selesai di proses.
5) Gambar 5 merupakan tab summary, yang merupakan ringkasan dari perhitungan TF-IDF. Ringkasan ini hanya menampilkan jumlah bobot TF-IDF perkategori.
6) Gambar 6 merupakan tab TF-IDF yang didalamnya terdapat tampilan proses perhitungan TF-IDF kata abstrak terhadap *database*.