BAB I
PENDAHULUAN

1.1 Latar Belakang

Teknologi Text To Speech (TTS) merupakan sebuah teknologi yang dapat membangkitkan sintesa ucapan dengan mengubah teks menjadi suara atau ucapan yang menyerupai ucapan manusia. TTS mengkonversi teks dalam format suatu bahasa menjadi ucapan sesuai dengan teks dalam bahasa yang digunakan. TTS sendiri dapat diimplementasikan ke dalam suatu bentuk aplikasi telekomunikasi yang dapat mempermudah pekerjaan manusia. Selain itu TTS juga dapat membantu orang-orang dengan kebutuhan khusus seperti tunanetra ataupun tunawicara.

TTS dapat diterapkan kedalam bentuk aplikasi website. Penerapan TTS pada sebuah website dapat dibantu dengan menggunakan aplikasi *responsive voice*. *Responsive voice* merupakan sebuah *tools* yang dapat digunakan untuk membangkitkan teks menjadi ucapan dalam berbagai ucapan bahasa. *Service* dari *responsive voice* dibagikan secara gratis sebagai solusi untuk sintesa ucapan sesuai dengan bahasa yang dibutuhkan, namun dalam penerapannya terdapat beberapa kekurangan seperti ucapan pada frasa yang tidak sesuai khususnya untuk ucapan kalimat masukan yang panjang, yang memiliki jumlah karakter lebih dari 100 karakter. Misalnya:

“Acara tahunan yang rutin digelar ini diselenggarakan sebagai bentuk rangkaian dari kegiatan ulang tahun Prodi Informatika.”

Kalimat tersebut seharusnya diucapkan sesuai dengan frasa yang semestinya. Seperti pada frasa “ulang tahun”, seharusnya diucapkan dalam satu-kesatuan tetapi terjadi jeda dalam frasa tersebut. Hal ini dikarenakan jumlah karakter dari awal kalimat sampai kata “tahun” adalah 103 karakter, sehingga *service* dari *responsive voice* akan memberi jeda berupa tanda koma secara otomatis setelah kata “ulang”.

Frasa adalah satuan gramatik yang terdiri atas satu kata atau lebih yang tidak melebihi batas fungsi tertentu, sehingga dalam hal ini frasa merupakan satu-kesatuan yang tidak bisa dipisah karena hal ini dapat mempengaruhi makna dari kalimat yang diucapkan. Pembentukan frasa dapat dilakukan dengan menggunakan metode *shallow parsing*. Metode *shallow parsing* merupakan metode yang mudah,

cepat dan andal, tidak memberikan analisis sintaksis penuh namun terbatas pada parsing konstituen sintaksis yang lebih kecil. Ketika kita membaca sebuah kalimat, kita membacanya sepotong demi sepotong yang membentuk kesatuan kelompok kata yang disebut frasa. Pada penentuan frasa, *shallow parsing* membutuhkan *rule grammars* sebagai pembentuk potongan frasa dalam kalimat yang diolah. *Shallow parsing* sendiri juga disebut sebagai *chunking* yang merupakan metode pemenggalan kalimat menjadi frasa teks dengan aturan-aturan tertentu berdasarkan kelas kata atau *Part-of-Speech* (PoS). PoS *tag* adalah suatu proses yang memberikan label kelas kata secara otomatis pada setiap kata yang ada pada suatu teks atau dokumen. *Shallow parsing* melakukan ekstrasi informasi dengan memecah setiap kalimat menjadi beberapa bagian yang biasa disebut *phrasal chunks*. *Phrasal chunks* tersebut lalu dapat disimpan dan diambil bagian yang diinginkan (Dhara & Hendrawan, 2017). Pada metode *shallow parsing*, kata yang saling terkait masih berada dalam satu *phrasal chunk* yang sama sehingga dapat memberikan informasi yang lebih jelas. Hasil keluaran dari metode *shallow parsing* ini yakni berupa frasa-frasa yang kemudian diolah untuk mendapatkan jeda antar frasa yang sesuai. Penentuan frasa dalam sebuah kalimat sangatlah penting karena dapat memperjelas informasi dari makna pada suatu teks kalimat.

Berdasarkan pembahasan diatas, maka diperlukan pemenggalan kalimat menggunakan metode *shallow parsing*, agar dapat dihasilkan ucapan frasa-frasa yang sesuai. Pemenggalan kalimat menggunakan metode *shallow parsing* memerlukan sebuah aturan/ *rule grammars* yang dikembangkan secara spesifik untuk keperluan pembangkitan ucapan menggunakan *responsive voice*. Frasa-frasa hasil dari proses pemenggalan kalimat yang kemudian dibangkitkan menjadi ucapan menggunakan *responsive voice* ini dimaksudkan agar makna dari teks kalimat yang diucapkan dapat tersampaikan.

1.2 Rumusan Masalah

Berdasarkan latar belakang diatas, rumusan masalah yang dapat diambil adalah bagaimana mengembangkan *rule grammars* dengan metode *shallow parsing* agar dapat dihasilkan frasa-frasa ucapan yang sesuai sehingga dapat digunakan untuk pembangkitan ucapan menggunakan *responsive voice*.

1.3 Tujuan Penelitian

Adapun tujuan dari penelitian yang dilakukan adalah membangkitkan ucapan dengan frasa-frasa yang sesuai dari suatu teks pada website menggunakan pembangkit ucapan *responsive voice*.

1.4 Pembatasan Masalah

Beberapa hal yang menjadi batasan dalam penelitian ini adalah sebagai berikut.

1. Studi kasus website yang digunakan yaitu website berita.
2. Menggunakan PoS kelas kata dari penelitian Alfan Farizki Wicaksono (2010).
3. *Rule grammars* dikembangkan berdasarkan kalimat berita dalam Bahasa Indonesia.

1.5 Sistematika Penulisan Skripsi

Sistematika penulisan penelitian ini disusun untuk memberikan gambaran umum tentang penelitian yang dijalankan. Sistematika laporan tugas akhir ini disusun dalam 5 (lima) bab yang terdiri dari Bab I Pendahuluan, Bab II Tinjauan Pustaka, Bab III Metodologi Penelitian dan Perancangan Sistem, Bab IV Hasil Perancangan dan Analisis Sistem, serta Bab V Penutup.

Bab I Pendahuluan adalah bab yang berisi latar belakang penelitian, perumusan masalah, tujuan penelitian, pembatasan masalah, dan sistematika penulisan.

Bab II Tinjauan Pustaka adalah bab yang berisi landasan teori berkaitan dengan penelitian yang akan dilakukan. Beberapa teori-teori yang terkait adalah pengertian *natural language processing* (NLP), pengertian dan metode dalam membantu sistem *text-to-speech* (TTS), pengertian dan penggunaan *responsive voice*, pengertian *part-of-speech* (PoS) *tagger* dan *shallow parsing*, pengembangan *rule grammars* dengan bantuan *regular expression*, pengertian frasa pada Bahasa Indonesia, PHP, Python, Java dan pengujian sistem yaitu pengujian akurasi, pengujian *precision, recall* dan *f-measure*, pengujian subjektif dan pengujian *black box*.

Bab III Metodologi Penelitian dan Perancangan Sistem adalah bab yang berisi tentang bahan penelitian, alat yang dipergunakan, metode penelitian yang berisi diagram alir penelitian, perancangan sistem, perancangan antarmuka serta rencana pengujian dengan menggunakan pengujian akurasi, pengujian *precision*, *recall* dan *f-measure*, pengujian subjektif serta pengujian *black box*.

Bab IV Hasil dan Analisis adalah bab yang berisi penjelasan mengenai implementasi pada sistem, *screenshoot* tampilan antarmuka sistem yang sudah jadi, serta analisis hasil uji coba. Setiap bagian sistem yang ditampilkan akan dilakukan analisis terlebih dahulu untuk mengarah kepada suatu kesimpulan.

Bab V Penutup adalah bab yang berisi kesimpulan dari penelitian yang telah dilakukan dan saran/ rekomendasi untuk perbaikan, pengembangan atau kesempurnaan/ kelengkapan penelitian yang telah dilakukan.

BAB II
TINJAUAN PUSTAKA

2.1 Studi Literatur

M. Iqbal Kamiludin (2017), dalam penelitiannya yang berjudul Prediksi Jeda Pada Ucapan Bahasa Melayu Pontianak dengan Menggunakan Metode *Shallow Parsing* menjelaskan bahwa tujuan dari penelitian adalah untuk menghasilkan *rule grammar* untuk menentukan jeda dengan menggunakan metode *shallow parsing* sebagai fungsi penentu jeda pada ucapan Bahasa Melayu Pontianak dan sebagai informasi yang dapat digunakan oleh sistem TTS (*Text To Speech*). Penelitian ini menjelaskan bagaimana menemukan frasa jeda ucapan kalimat Bahasa Melayu Pontianak menggunakan metode *shallow parsing* dengan bantuan salah satu fungsi dari NLTK (*Natural Language Toolkit*) dan dalam penerapannya membutuhkan *rule* grammars dan PoS *tagger*. Hasil dari penelitian ini adalah kalimat yang telah terpotong-potong membentuk frasa-frasa jeda. Pengujian dilakukan dengan menggunakan *precision, recall* serta *F-measure* terhadap kalimat tunggal dan kalimat majemuk. Kalimat tunggal terdiri dari 47 kalimat dan kalimat majemuk terdiri dari 121 kalimat. Hasil dari pengujian diperoleh nilai *precision* dan nilai *recall* untuk kalimat tunggal adalah 0.74 dan 0.78 dengan nilai *f-measure* 0.75. Adapun pada kalimat majemuk memperoleh nilai *precision* dan *recall* sebesar 0.57 dan 0.67 dengan nilai *F-measure* 0.61.

Arry Akhmad Arman, dkk. (2013), melakukan penelitian yang berjudul *Syntactic Phrase Chunking for Indonesian Language.* Penelitian ini menjelaskan cara menemukan frasa dalam Bahasa Indonesia yang terkait dengan kategori sintaksis menggunakan metode *chunking*. Metode *chunking* membutuhkan aturan tata bahasa/ *rule grammars* dan PoS *tagger*. *Rule grammars* dibangun menggunakan *regular expression* untuk mengklasifikasikan kata-kata yang membentuk frase berdasarkan informasi dari PoS *tagger*. *Hidden Markov Model* (HMM) berdasarkan PoS *tagger* untuk bahasa Indonesia digunakan untuk *tagging* PoS setiap kata dalam kalimat. Pengujian dilakukan pada kalimat tunggal dan kalimat majemuk. Kalimat tunggal terdiri dari 31 kalimat dengan jumlah rata-rata kata per kalimat adalah 4,74 kata, kalimat majemuk terdiri dari 80 kalimat dengan

jumlah rata-rata kata per kalimat 15,74 kata. Hasil pengujian yang diperoleh nilai *precision* dan nilai *recall* untuk kalimat tunggal adalah 88,75 dan 78,89 dengan Nilai F adalah 83,53. Sedangkan untuk kalimat majemuk diperoleh nilai *precision* dan nilai *recall* adalah 80,57 dan 84,72 dengan nilai F 82,60.

Alfan Farizki Wicaksono dan Ayu Purwarianti (2010), melakukan penelitian yang berjudul *HMM Based Part-of-Speech Tagger for Bahasa Indonesia*. Penelitian ini menjelaskan cara mendapatkan set PoS dalam Bahasa Indonesia dengan lebih akurat. Beberapa metode digabungkan untuk meningkatkan keakuratan penandaan PoS berbasis HMM untuk Bahasa Indonesia. Metode pertama adalah menggunakan pohon imbuhan yang mencakup akhiran kata dan awalan. Yang kedua adalah menggunakan tag PoS berikutnya sebagai salah satu fitur untuk HMM. Metode terakhir adalah menggunakan leksikon tambahan (dari KBBI-Kateglo) untuk membatasi tag kandidat yang dihasilkan oleh pohon imbuhan. Model HMM dibangun di atas korpus data 15000-token. Dalam percobaan, pada korpus tes OOV 15%, akurasi terbaik adalah 96,50% dengan 99,4% untuk kata-kata dalam kosakata dan 80,4% untuk kata-kata OOV (di luar kosakata). Percobaan menunjukkan bahwa pohon imbuhan dan leksikon tambahan efektif dalam meningkatkan akurasi tagger PoS, sedangkan penggunaan tag PoS berikutnya tidak memberikan banyak perbaikan pada penanganan OOV.

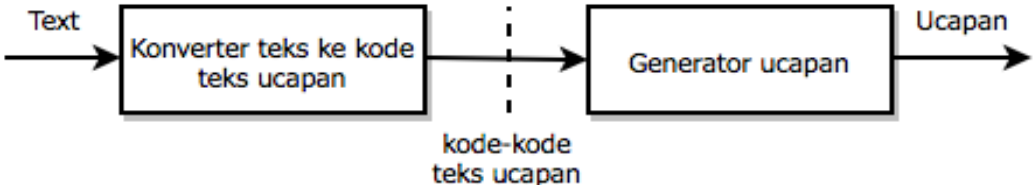
Tabel 2.1 Kajian Penelitian Terkait

No.	Penulis	Judul	Keterangan
1.	M. Iqbal Kamiludin (2017), Universitas Tanjungpura	Prediksi Jeda Pada Ucapan Bahasa Melayu Pontianak Dengan Menggunakan Metode *Shallow Parsing*	• Bahasa yang digunakan yaitu Bahasa Melayu Pontianak. • Menggunakan metode *shallow parsing* dengan *rule grammars* untuk membentuk frasa jeda pada kalimat

			Bahasa Melayu Pontianak. • Menggunakan PoS *tagger* dalam pemberian kelas kata.
2.	Arry Akhmad Arman, Arif Bijaksana Putra N, Ayu Purwarianti, Kuspriyanto (2013)	*Syntactic Phrase Chunking for Indonesian Language*	• Bahasa yang digunakan yaitu Bahasa Indonesia. • Menggunakan metode *chunking*, dengan 5 tipe frasa chunking TP, BP, KP, NP, dan VP. • Menggunakan PoS *tagger* Bahasa Indonesia.
3.	Alfan Farizki Wicaksono dan Ayu Purwarianti (2010)	*HMM Based Part-of-Speech Tagger for Bahasa Indonesia*	• Bahasa yang digunakan yaitu Bahasa Indonesia. • Menggunakan metode pohon imbuhan, tag PoS HMM, dan leksikon untuk keakuratan pada PoS.

2.2 *Text To Speech*

Text to speech merupakan sebuah sistem yang mengkonversi teks dalam format suatu bahasa menjadi ucapan sesuai dengan pembacaan teks dalam bahasa yang digunakan. TTS adalah sebuah sistem yang dapat mengubah suatu teks menjadi ucapan secara otomatis dengan cara fonetisasi (penyusunan fonem-fonem untuk membentuk ucapan). Diagram blok sistem konversi teks ke ucapan dapat dilihat pada Gambar 2.1 sebagai berikut.



Gambar 2.1 Diagram Blok Sistem Konversi Teks ke Ucapan (dimodifikasi dari Andayu, 2013)

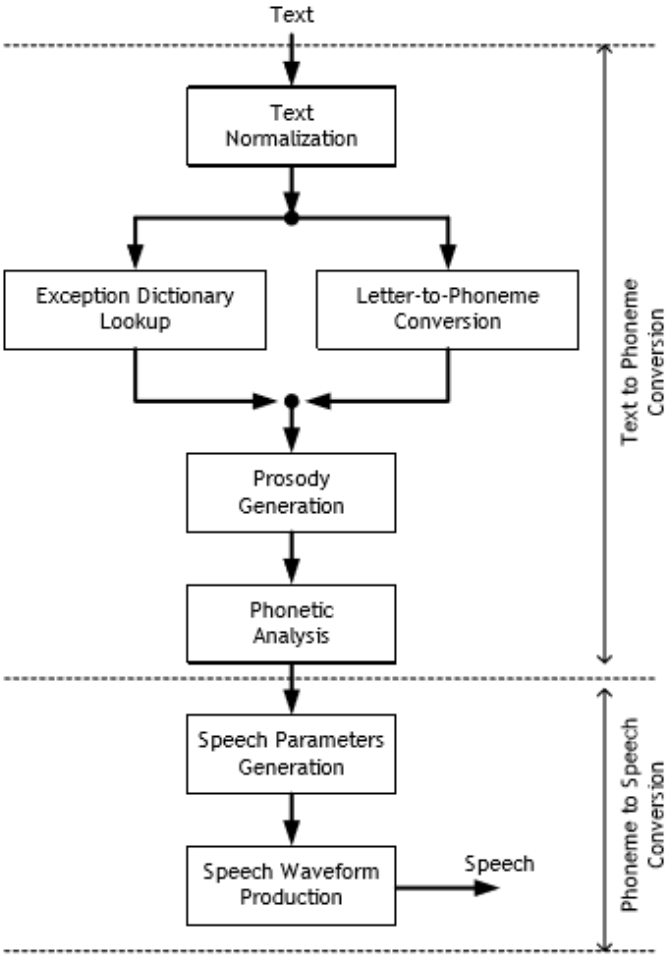
Seperti pada Gambar 2.1, sistem pensintesa ucapan atau *text to speech* pada prinsipnya terdiri dari dua subsistem, yaitu :

1. Bagian konversi teks ke kode teks ucapan

   Konverter teks ke kode teks ucapan berfungsi untuk mengolah kalimat masukan dalam suatu bahasa tertentu yang berbentuk teks menjadi urutan kode-kode ucapan yang sesuai pada suatu sistem pembangkit ucapan.
2. Bagian konverter kode teks ucapan ke ucapan

   Konverter kode teks ucapan ke ucapan akan menerima dan mengubah masukan kode-kode tersebut menjadi bunyi atau sinyal ucapan yang sesuai dengan kalimat yang ingin diucapkan.

Tahapan-tahapan utama konversi dari teks menjadi ucapan dapat dinyatakan dengan diagram seperti yang terlihat pada Gambar 2.2 berikut.



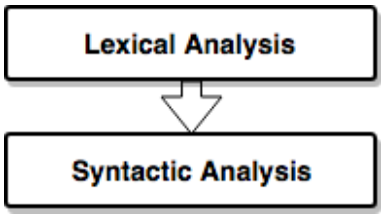
Gambar 2.2 Urutan Proses Konversi dari Teks ke Ucapan (Arman, Konversi dari Teks ke Ucapan, 2003)

Tugas sistem TTS secara umum dapat dibagi dalam 2 bagian besar, analisa teks dan sintesa ucapan. Analisa teks mentransformasi teks masukan menjadi representasi linguistik, selanjutnya bagian sintesis ucapan mentransformasi representasi linguistik tersebut menjadi gelombang sinyal ucapan. Dengan teknologi TTS, dimungkinkan sebuah komputer mampu berkomunikasi dan berinteraksi dengan manusia tidak hanya melalui tulisan, namun juga dalam bentuk lisan menggunakan bahasa yang digunakan sehari-hari.

2.3 *Natural Language Processing* (NLP)

Natural Language Processing (NLP) atau dalam bahasa Indonesia disebut dengan Pemrosesan Bahasa Alami (PBA) merupakan salah satu bidang ilmu komputer, kecerdasan buatan dan bahasa (*linguistic*) yang berkaitan dengan

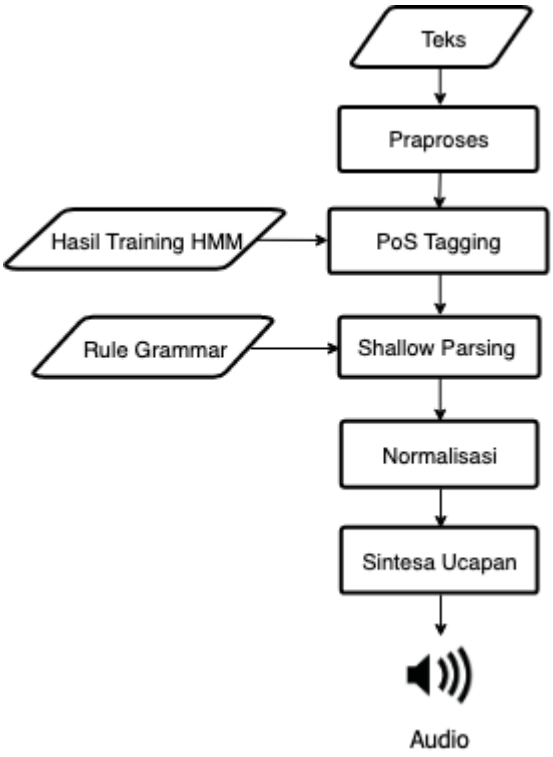
interaksi antara komputer dengan bahasa alami manusia. Dengan teknologi yang ada pada NLP, komputer dapat memahami bahasa manusia dan dapat membuat bahasa yang dimengerti manusia. Ada dua bentuk input dari NLP, yaitu input lisan (artinya AI juga harus dapat mengenali pelafalan kata manusia) dan input berupa teks (AI harus dapat mengenali huruf). *Natural Language Prossesing* (NLP) yang berupa modul konversi teks ke ucapan merupakan bagian dari sistem TTS. Ada lima tapahan pada NLP dalam pemrosesan bahasa manusia oleh komputer. Penelitian yang dilakukan peneliti berada pada tahapan yang kedua seperti pada Gambar 2.3 berikut.



Gambar 2.3 Tahapan dalam NLP (dimodifikasi dari Djohan, 2016)

1. *Lexical Analysis*. Pada tahapan ini, dilakukan identifikasi dan analisis terhadap struktur kata. Kata-kata secara individu dilakukan analisis berdasarkan komponennya, dan token yang tidak termasuk, seperti tanda baca dipisahkan dari kata-kata yang ada. Analisis ini sangat penting untuk menentukan aturan kata yang ada dalam pada kalimat, termasuk tata bahasa.
2. *Syntactic Analysis*. Pada tahapan ini, dilakukan analisis terhadap kumpulan kata dalam kalimat untuk mencari tahu bagaimana kata yang satu dengan yang lain terhubung. Dimana analisis ini mempelajari aturan untuk menggabungkan kata menjadi frase dan kalimat, serta menggunakan aturan tersebut untuk menguraikan (parse) dan membentuk kalimat.

Adapun tahapan penelitian yang dilakukan pada proses *syntactic analysis* seperti pada Gambar 2.4 berikut.



Gambar 2.4 Tahapan Penelitian Pada Proses *Syntactic Analysis*

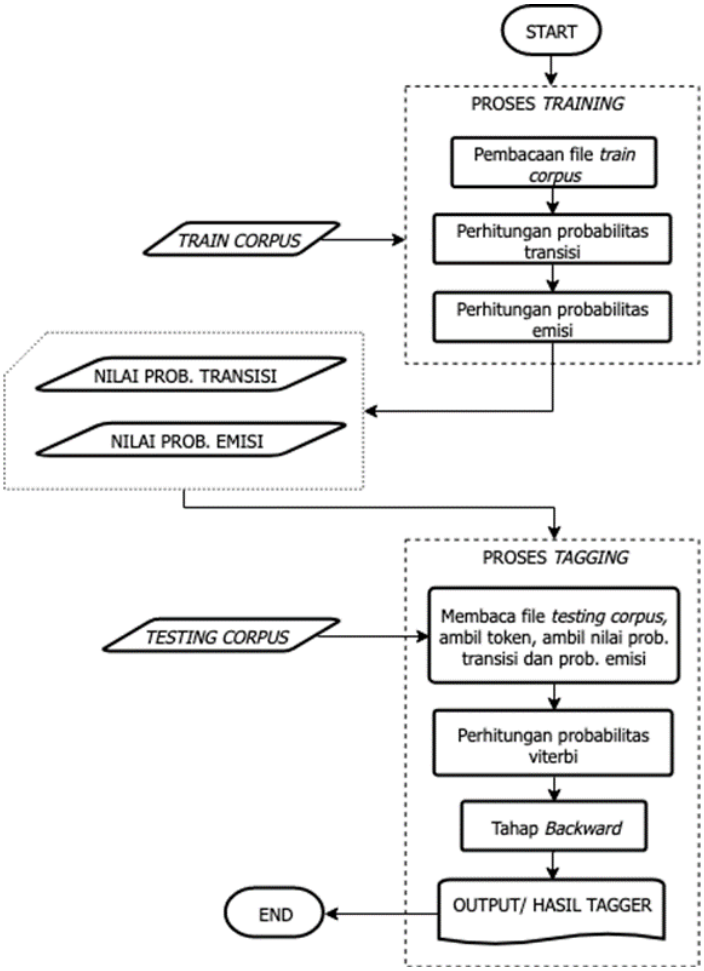
2.4 *Part of Speech* (PoS) *Tagging*

Part of speech (PoS) *tagging* merupakan bagian penting dari cakupan bidang ilmu *Natural Languange Processing* (NLP) yang diterapkan dalam pengenalan suara (*speech recognition*), pencarian informasi (*information retrieval*), pengucapan teks (*text to speech*), pencarian kata ambigu (*word sense disambiguation*), pengolahan semantik (*semantic processing*) dan mesin penerjemah (*machine translation*) (Jurafsky, dkk. 2000). Secara umum, terdapat 3 metode atau pendekatan yang bisa digunakan untuk melakukan PoS *tagging* yaitu diantaranya sebagai berikut.

a) Pendekatan berdasarkan aturan (*rule*-based). Memberikan tag POS ke sebuah kata berdasarkan beberapa aturan linguistik yang dibuat secara manual.
b) Pendekatan probabilistik. Menentukan tag yang paling mungkin dari token yang diberikan konteks sekitarnya, berdasarkan nilai-nilai probabilitas yang diperoleh dari corpus yang ditandai secara manual.

c) Pendekatan berbasis transformasi (transformation-based). Menggabungkan pendekatan berbasis aturan dan probabilistik untuk secara otomatis menurunkan aturan simbolik dari sebuah corpus.

Hidden Markov Model (HMM) adalah metode probabilistik yang ditetapkan untuk penandaan PoS otomatis. PoS tagger dengan metode HMM terbukti memiliki waktu proses yang lebih baik daripada metode probabilistik lainnya. Data training dari HMM digunakan untuk proses pelabelan kata. Model HMM ini terdiri dari tiga proses, yaitu inisialisasi, transisi, dan emisi. Inisialisasi merupakan proses mendapatkan jumlah label (*tags*) dari masing-masing kata yang terdapat pada data training. Dalam melakukan inisialisasi, perlu dimasukkan kata berlabel dan label kata. Kata berlabel merupakan kumpulan kalimat yang sudah diberi label. Dan label kata adalah jenis-jenis label yang terdapat dalam data set. Transisi adalah suatu proses mencari label (*tags*) kata setelah label kata tersebut. Transisi didapatkan dengan memasukkan kata berlabel dan label kata pada sistem. Emisi merupakan proses mencari jumlah kata dari masing-masing label (*tags*) yang terdapat pada data training. Kemudian nilai-nilai dari ketiga proses tersebut disimpan dan akan menjadi panduan yang digunakan pada saat proses tagging terhadap *testing corpus*. Berikut mekanisme kerja PoS *tagger* menggunakan HMM pada Gambar 2.5.



Gambar 2.5 Mekanisme Kerja PoS *Tagger* Menggunakan HMM (dimodifikasi dari Ramadhanti, 2019)

PoS *tagging* adalah suatu proses yang memberikan label kelas kata secara otomatis pada setiap kata yang ada pada suatu teks atau dokumen. Misalnya terdapat kalimat "saya makan nasi". Sistem akan menerima input berupa kalimat tersebut dan mengeluarkan output sebagai berikut.

saya/KG makan/VV nasi/NN

Pada *output* tersebut terdapat label yaitu KG=kata ganti, VV= kata kerja dan NN= kata benda. Label atau *tag* yang diberikan ke suatu kata dalam kalimat menunjukkan kelas kata (*part-of-speech*) dari kata yang bersangkutan. Secara umum kelas kata dapat dibedakan menjadi kelas kata benda (*noun*), kata kerja (*verb*), kata sifat (*adjective*), kata keterangan (*adverb*), kata tugas (*function words*) dan lainnya (*other*) termasuk didalamnya berupa tanda baca. Berikut penelitian Alfan Farizki Wicaksono (2010) mengenai PoS *tagger* berbahasa Indonesia dengan 35 set PoS berdasarkan kelas kata seperti pada Tabel 2.2.

Tabel 2.2 Set PoS Alfan Farizki Wicaksono (2010)

No	PoS	Deskripsi	Contoh Kata
1	OP	*Parenthesis*	({[
2	CP	*Close Parenthesis*	)}]
3	GM	*Slash*	/
4	;	*Semicolon*	;
5	:	*Colon*	:
6	“	*Quotation*	“ ’
7	.	*Sentence Terminator*	. ! ?
8	,	*Comma*	,
9	-	*Dash*	-
10	...	*Ellipsis*	...
11	JJ	*Adjective*	Kaya, Manis
12	RB	*Adverb*	Nanti, Sementara
13	NN	*Common Noun*	Mobil, Sepatu, Roda
14	NNP	*Proper Noun*	Bekasi, Indonesia
15	NNG	*Genitive Noun*	Bukunya
16	VBI	*Intransitive Verb*	Pergi
17	VBT	*Transitive Verb*	Membeli
18	IN	*Preposition*	Di, Ke, Dari
19	MD	*Modal*	Bisa
20	CC	*Coor-Conjunction*	Dan, Atau, Tetapi
21	SC	*Subor-Conjunction*	Jika, Ketika
22	DT	*Determiner*	Para, Ini, Itu
23	UH	*Interjection*	Wah, Aduh, Oi
24	CDO	*Ordinal Numerals*	Pertama, Kedua
25	CDC	*Collective Numerals*	Bertiga
26	CDP	*Primary Numerals*	Satu, Dua
27	CDI	*Irregular Numerals*	Beberapa
28	PRP	*Personal Pronouns*	Saya, Kamu
29	WP	*WH-Pronouns*	Apa, Siapa, Dimana
30	PRN	*Number Pronouns*	Kedua-duanya, Satu-satunya
31	PRL	*Locative Pronouns*	Sini , Situ, Sana
32	NEG	*Negation*	Bukan, Tidak
33	SYM	*Symbols*	@#$%^&
34	RP	*Particles*	Pun, Kah
35	FW	*Foreign*	Words

2.5 *Shallow Parsing*

Shallow parsing adalah proses untuk mengelompokkan secara berurutan kata-kata ke dalam bentuk frasa dengan sebuah *chunker*, atau bisa disebut *chunks* (Ye, 2009). Metode *shallow parsing* disebut juga dengan *chunking* yang merupakan metode pemenggalan/ pemotongan kalimat menjadi frasa teks dengan aturan-aturan tertentu berdasarkan kelas kata PoS *tagger*. Aturan-aturan yang dibutuhkan dalam penentuan frasa pada *shallow parsing* disebut juga *rule grammars* yang berfungsi sebagai pembentuk potongan frasa dalam kalimat yang diolah. Jenis-jenis frasa *shallow parsing* telah dikembangkan untuk membentuk frasa yang berhubungan dengan sintaksis kategori dan dapat digunakan untuk menginformasikan pembentukan frasa jeda ucapan kalimat. Jenis frasa yang dikembangkan pada penelitian Arman, dkk. (2013) adalah TP (*questioning phrases*), BP (*numeric phrase*), KP (*connection phrases*), NP (*noun phrases*), dan VP (*verb phrases*) seperti yang ditampilkan pada Tabel 2.3 berikut.

Tabel 2.3 Set PoS Frasa *Shallow Parsing* Arman, dkk. (2013)

Types of Phrases	**POS Tag**
Questioning Phrases (TP)	WP
Numeric Phrases (BP)	CDO, CDP, CDI, CDC
Connection Phrases (KP)	SC, CC, NEG, IN, MD, RB
Noun Phrases (NP)	NN, PRN, PRP, PRL, NNG, NNP, FW, RP, UH, JJ
Verb Phrases (VP)	VBI, VBT

Sebagai contoh kalimat berikut.

Wakil Presiden Budiono telah meresmikan jembatan tol Barito tadi sore.

Kalimat tersebut akan diinput, lalu ditag setiap kata melalui PoS *tagger*. Kemudian fungsi *shallow parsing* yang akan memberikan tag untuk membentuk frasa. *Output* dari proses *shallow parsing* sebagai berikut.

0 (NP Wakil/NN Presiden/NN (NP Budiono/NNP))

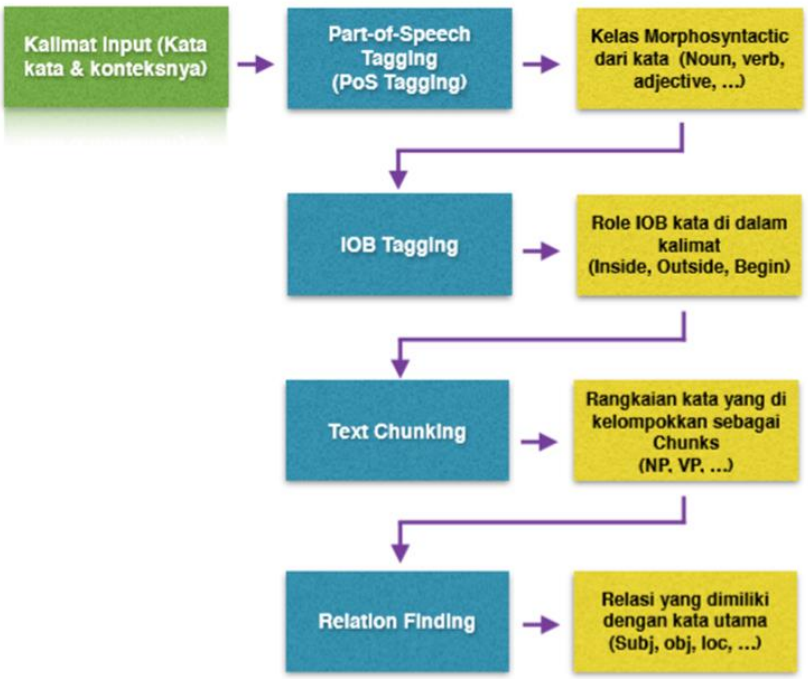
1 (VP (KP telah/MD) (VP meresmikan/VBT))

2 (NP jembatan/NN tol/NN Barito/NNP)

3 (NP (KP tadi/RB) (NP sore/NN))

4 .

Secara umum metode *shallow parsing* terdiri dari empat tahapan pemrosesan utama yaitu *part of speech tagging*, IOB *tagging*, *text chunking* dan *relation finding*. Masing-masing dari tahapan tersebut memiliki peran penting yang dilakukan secara sekuentif karena *output* dari suatu tahapan berpengaruh ke tahapan selanjutnya. Berikut Gambar 2.6 adalah gambaran dari arsitektur *shallow parsing* yang sering digunakan secara umum.



Gambar 2.6 Arsitektur *Shallow Parsing* (Antares, 2014)

2.6 *Regular Expression*

Regular Expression merupakan salah satu implementasi dari operasi pencocokan pola (*pattern recognation*) untuk sebuah *text* atau *string*. *Regular expression* atau yang lebih sering disebut regex ini memiliki 2 fungsi utama yakni mencari dan mengganti, mencari suatu pola tertentu dalam teks lalu menggantinya menjadi pola yang lain. *Regular expression* digunakan oleh banyak teks editor, *utilities*, dan bahasa pemrograman untuk pencarian dan memanipulasi teks berdasarkan pola. Regex merupakan salah satu alat yang paling berguna dalam ilmu komputer. Dalam NLP, regex digunakan dalam fonologi, morfologi, *text analysis*, *information extraction*, dan *speech recognition*. Selain itu, regex juga sangat manjur

(*powerful*) terutama untuk proses penguraian kata (*text parsing*). Berikut karakter *regular expressions* yang digunakan dalam *shallow parsing* pada Tabel 2.4.

Tabel 2.4 Makna Karakter *Regular Expressions* pada *Shallow Parsing*

Karakter	Makna karakter *Regular Expressions*
< >	Penentuan tag part-of-speech
?	Nol atau salah satu item sebelumnya
*****	Nol atau lebih dari item sebelumnya.
+	Satu atau lebih dari item sebelumnya
|	Mencocokkan satu item dengan yang lainnya,

2.7 *Rule Grammars*

Rule grammar merupakan aturan-aturan pembentukan suatu kalimat dalam sebuah bahasa, atau yang biasa disebut tata bahasa. Dengan adanya *grammar*, parsing dapat dilakukan secara cepat dengan hanya melakukan proses *searching*. Parser akan mencari aturan-aturan yang tepat untuk membentuk struktur suatu kalimat. *Rule grammar* ini nantinya akan digunakan pada proses *shallow parsing* untuk membentuk frasa yang sesuai. Adapun contoh implementasi *rule grammars* dalam bentuk penulisan *regular expressions* untuk proses *shallow parsing* pada NLTK seperti berikut.

```
NP: { <NN>+ <DT|RB>? }
```

Rule ini digunakan untuk mencari frasa kata benda (NP), jika kata benda (NN) ketemu determiner (DT) atau kata keterangan (RB) akan menjadi satu frasa. Didalam *regular expressions* pembacaan *rule* tersebut dapat dibaca seperti: frasa NP akan terbentuk jika label PoS <NN>+ dengan kemunculan minimal 1 atau N karakter dan diikuti label PoS <DT atau RB>? dengan kemunculan 0 atau 1 karakter.

Penelitian yang telah ada yaitu penelitian dari Arif Bijaksana Putra Negara dengan *rule grammars* berjumlah 9 *rule* yang sebelumnya digunakan untuk proses penentuan frasa *chunking* Bahasa Indonesia, kemudian dimodifikasi oleh peneliti disesuaikan dengan kebutuhan penelitian untuk mendapatkan potongan frasa yang sesuai. *Rule grammars* awal berjumlah 9 *rule* dengan penjabaran seperti pada Gambar 2.7 berikut.

```
TP : { <WP>+ }
NP : { <NNP>+ | <FW>+ | <RP>+ }
KP : { <SC|CC|NEG>+ <IN>* <MD|RB>* | <MD|RB>+ | <IN>+
<SC|CC|NEG>* <MD|RB>* }
CP : { <CDO|CDP|CDI|CDC>+ | <SYM> }
VP : { <VBI>+ | <VBT>+ }
NP : { <DT>? <NP|CP|NN|PRN|PRP|PRL|NNG|UH>+ <JJ>*
<NP|CP|NN|PRN|PRP|PRL|NNG|FW|UH>* <DT>? | <DT>? <CP>?
<JJ>+ <NP|CP|NN|PRN|PRP|PRL|NNG|FW|UH>* <DT>? | <DT> }
NP : { <KP> <NP> }
VP : { <VP> <VP> }
VP : { <KP> <VP> }
VP : { <VP> <NP> }
```

Gambar 2.7 Contoh Aturan *Rule Grammars Shallow Parsing*

Berikut contoh teks kalimat inputan:

Ibu sedang masak di dapur.

Kalimat inputan tersebut di tag setiap kata, kemudian fungsi regexpparser pada NLTK *shallow parsing* akan memproses kalimat dengan tag PoS ke bentuk potongan frasa dan pohon parsing sebagai berikut.

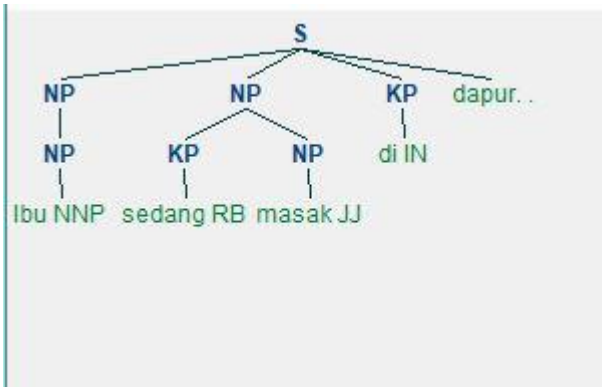
0 (NP (NP Ibu/NNP))

1 (NP (KP sedang/RB) (NP masak/JJ))

2 (NP (KP di/IN) (NP dapur/NN))

3 .

Pohon parsing yang terbentuk seperti pada Gambar 2.8 sebagai berikut.



Gambar 2.8 Pohon Parsing pada Kalimat Inputan

2.8 Bahasa Indonesia

Bahasa Indonesia adalah bahasa resmi yang dipakai sehari-hari oleh orang Indonesia dan bahasa persatuan bangsa Indonesia yang digunakan untuk menyatukan dan mempermudah komunikasi antardaerah yang ada di Indonesia. Pengertian bahasa menurut kamus besar Bahasa Indonesia (Alwi, 2002) berarti sistem lambang bunyi yang arbiter, yang digunakan oleh semua orang atau anggota masyarakat untuk bekerjasama, berinteraksi, dan mengidentifikasi diri dalam bentuk percakapan yang baik, tingkah laku yang baik, sopan santun yang baik. Bahasa Indonesia digunakan kurang lebih 230 juta masyarakat Indonesia yang tersebar di seluruh Indonesia dengan 23 juta native speaker. Bahasa Indonesia juga digunakan sangat luas di perguruan-perguruan, media massa, sastra, perangkat lunak, surat-menyurat resmi, dan berbagai forum publik lainnya. Sehingga dapat dikatakan bahwa Bahasa Indonesia digunakan oleh semua warga Indonesia.

2.9 Frasa

Pada struktur kalimat dasar, diantara kalimat dan kata biasanya ada satuan-antara yang berupa kelompok kata. Kelompok kata yang menjadi salah satu unsur pembentuk kalimat ini dikenal dengan sebutan frasa. Frasa merupakan rangkaian atau kesatuan dari beberapa kata. Contohnya adalah bapakku, sedang belajar, baju biru, selamat malam. Pada kata ‘selamat malam’, kedua kata penyusun ini memiliki arti yang berbeda ketika berdiri terpisah. Sedangkan ketika menjadi sebuah frasa, gabungan kata tersebut memiliki arti sebagai sebuah sapaan. Sebuah frasa memiliki beberapa ciri yaitu sebagai berikut.

- Terbentuk atas dua kata atau lebih dalam pembentukannya.
- Menduduki fungsi gramatikal dalam kalimat.
- Mengandung satu kesatuan makna gramatikal.
- Bersifat non-predikatif.

Terdapat beberapa cara untuk melakukan identifikasi jenis frasa dalam Bahasa Indonesia, salah satunya yaitu berdasarkan kelas kata. Frasa berdasarkan kelas kata dibedakan menjadi 9 sebagai berikut.

a. Frasa nomina, adalah kelompok kata benda yang dibentuk dengan memperluas sebuah kata benda.

b. Frasa verbal, adalah kelompok kata yang terbentuk dari kata-kata kerja.
c. Frasa adjektifa, adalah kelompok kata yang dibentuk oleh kata sifat atau keadaan sebagai inti (diterangkan) dengan menambahkan kata lain yang berfungsi menerangkan.
d. Frasa adverbial, adalah kelompok kata yang dibentuk dengan kata sifat.
e. Frasa pronominal, adalah frasa yang dibentuk dengan kata ganti.
f. Frasa numeralia, adalah kelompok kata yang dibentuk dengan kata bilangan.
g. Frasa interogatif koordinatif, adalah frasa yang berintikan pada kata tanya.
h. Frasa demonstratif koordinatif, adalah frasa yang dibentuk oleh dua kata yang tidak saling menerangkan.
i. Frasa preposisional koordinatif, adalah frasa yang dibentuk oleh kata depan yang tidak saling menerangkan.

2.10 Python

Python adalah bahasa pemrograman dinamis yang mendukung pemrograman berbasis objek. Python merupakan bahasa pemrograman yang dikembangkan secara khusus untuk membuat *source code* mudah dibaca. Python didistribusikan dengan beberapa lisensi yang berbeda dari beberapa versi. Namun pada prinsipnya python dapat diperoleh dan dipergunakan secara bebas, bahkan untuk kepentingan komersial.

Bahasa python mendukung hampir di semua sistem operasi, bahkan untuk sistem operasi linux, hampir semua distronya sudah menyertakan python didalamnya. Dengan kode yang simpel dan mudah diimplementasikan, seorang programmer dapat lebih mengutamakan pengembangan aplikasi yang dibuat. Saat ini kode python dapat dijalankan pada sistem berbasis Linux/ Unix, Windows, Mac OS X, OS/ 2, Amiga, Palm dan Symbian. Selain itu python merupakan salah satu produk yang open source juga multiplatform. Beberapa fitur yang dimiliki python adalah sebagai berikut.

- Memiliki kepustakaan yang luas. Dalam distribusi python telah disediakan modul-modul siap pakai untuk berbagai keperluan.
- Memiliki tata bahasa yang jernih dan mudah dipelajari.

- Memiliki aturan layout kode sumber yang memudahkan pengecekan, pembacaan kembali dan penulisan ulang kode sumber.
- Berorientasi obyek.
- Memiliki sistem pengelolaan memori otomatis (garbage collection, seperti java).
- Modular, mudah dikembangkan dengan menciptakan modul-modul baru. Modul-modul tersebut dapat dibangun dengan bahasa python maupun C/C++.
- Memiliki fasilitas pengumpulan sampah otomatis, seperti halnya pada bahasa pemrograman java, python memiliki fasilitas pengaturan penggunaan ingatan komputer sehingga para pemrogram tidak perlu melakukan pengaturan ingatan komputer secara langsung.

2.11 Java

Bahasa pemograman java adalah bahasa pemrograman berorientasi objek (PBO) atau *object oriented programming* (OOP). Java merupakan bahasa pemrograman objek murni karena semua kode programnya dibungkus dalam kelas. Kelas terdiri atas metode-metode yang melakukan pekerjaan dan mengembalikan informasi setelah melakukan tugasnya. Kelas-kelas ini diorganisasikan menjadi sekelompok yang disebut paket (*package*). Pada java, program *javac* digunakan untuk mengkompilasi file kode sumber java menjadi kelas-kelas *bytecode*. File kode sumber mempunyai ekstensi *.java. Kompilator *javac* menghasilkan file *bytecode* kelas dengan ekstensi *.class. Interpreter merupakan modul utama sistem java yang digunakan aplikasi java dan menjalankan program *bytecode* java.

Beberapa operator dalam bahasa pemrograman java sebagai berikut.

a. Operator Aritmatika

Menurut Siallagan (2009), operator aritmatika adalah operator-operator yang digunakan untuk mengoperasikan perhitungan (aritmatika). Bahasa pemograman java menyediakan operator-operator aritmatika untuk memanipulasi variabel data seperti pada Tabel 2.5 berikut.

Tabel 2.5 Operator Aritmatika

Operator	Keterangan
+	Penjumlahan
-	Pembagian
*	Perkalian
/	Pembagian
%	Modulus (sisa bagi)

b. Operator Relasional

Menurut Siallagan (2009), operator relasional adalah operator hubungan (relasi) yang membandingkan kedua nilai operand dan hasilnya berupa nilai boolean, yaitu benar (true) atau salah (false). Berikut Tabel 2.6 berisi daftar operator relasional.

Tabel 2.6 Operator Relasional

Operator	Keterangan
==	Sama dengan (membandingkan bukan penugasan)
!=	Tidak sama dengan
>	Lebih besar
<	Lebih kecil
>=	Lebih besar sama dengan
<=	Lebih kecil sama dengan

c. Operator Logiaka/Boolean

Menurut Siallagan (2009), operator logika adalah operator yang digunakan terhadap operand bertipe Boolean yang hasilnya benar (true) atau salah (false). Berikut Tabel 2.7 berisi daftar operator logika.

Tabel 2.7 Operator Logika

Operator	Keterangan
&	Logika AND
|	Logika OR
^	Logika XOR
!	Logika NOT

2.12 *Natural Language Toolkit* (NLTK)

NLTK mendefinisikan infrastruktur yang dapat digunakan untuk membangun program NLP (Natural Language Processing) menggunakan python. Meski python memiliki kemampuan untuk melakukan tugas-tugas NLP dasar, namun tidak cukup powerful untuk melakukan tugas-tugas standar NLP, maka dari itu muncul modul NLTK (*Natural Language Toolkit*). Modul ini menyediakan berbagai fungsi dan *wrapper*, serta corpora standar baik itu mentah ataupun *pre-processed* yang digunakan dalam materi pengajaran NLP. Salah satu modul dari NLTK yang digunakan pada penelitian yaitu *nltk.parse* yang digunakan untuk melakukan pengolahan bahasa atau parsing. NLTK juga menyediakan kelas dasar untuk mewakili data yang relevan dengan NLP, *interface* standar untuk melakukan tugas-tugas seperti penandaan *part-of-speech*, parsing sintaksis, dan klasifikasi teks serta implementasi standar untuk setiap tugas yang dapat dikombinasikan untuk memecahkan masalah yang kompleks.

2.13 Website

Website merupakan halaman situs sistem informasi yang dapat diakses secara cepat. Website ini didasari dari adanya perkembangan teknologi informasi dan komunikasi. Web adalah suatu metode untuk menampilan informasi di internet, baik berupa teks, gambar, suara maupun video yang interaktif dan mempunyai kelebihan untuk menghubungkan (link) satu dokumen dengan dokumen lainnya (*hypertext*) yang dapat diakses melalui sebuah browser. Berdasarkan sifatnya, website dibagi menjadi dua yaitu sebagai berikut.

- Website statis, adalah web yang halamannya tidak berubah, biasanya untuk melakukan perubahan dilakukan secara manual dengan mengubah kode. Website statis informasinya merupakan informasi satu arah, yakni hanya berasal dari pemilik softwarenya saja, hanya bisa diupdate oleh pemiliknya saja. Contoh website statis ini, yaitu profil perusahaan.
- Website dinamis, merupakan web yang halaman selalu *update*, biasanya terdapat halaman *backend* (halaman administrator) yang digunakan untuk menambah atau mengubah konten. Web dinamis membutuhkan database untuk menyimpan.

Secara umum situs web memiliki beberapa fungsi yaitu fungsi komunikasi, informasi, entertainment dan transaksi. Untuk membangun sebuah web atau situs diperlukan unsur yang harus ada agar situs dapat berjalan dengan baik. Unsur-unsur tersebut adalah *domain name*, *hosting*, *script* (bahasa program), desain web dan publikasi.Adapun dalam proses penelitian, website dibangun menggunakan bahasa HTML dan PHP. HTML (*Hyper Text Markup Language*) adalah sebuah bahasa markup yang digunakan untuk membuat sebuah halaman web dan menampilkan berbagai informasi di dalam sebuah browser internet. Contoh struktur penulisan HTML sebagai berikut.

```
<html>
<head>
<title> </title>
</head>
<body>
</body>
</html>
```

Sedangkan PHP merupakan *script* yang menyatu dengan HTML dan berada pada server (*server side HTML embedded scripting*) yangmana HTML digunakan sebagai pembangun atau pondasi dari kerangka layout web, sedangkan PHP difungsikan sebagai prosesnya sehingga dengan adanya PHP tersebut web akan sangat mudah dimaintenance. PHP dibangun sebagai modul pada *web server apache* dan sebagai binary yang dapat berjalan sebagai CGI. Dalam pembuatan file PHP sama dengan cara pembuatan file HTML dengan menuliskan *script* PHP di text editor seperti notepad dan lain-lain. Untuk penulisan *script*/ baris kode PHP, perlu diapit dengan tag "<?php" sebagai awal dan tag ?> sebagai akhir dari *script* PHP. Jika kita mengetikan kode di luar tag tersebut tidak akan dianggap sebagai sebagai *script* PHP oleh PHP *engine*, melainkan akan dianggap sebagai kode HTML. Biasanya di dalam sebuah file PHP mengandung tag HTML dan baris kode PHP. Berikut struktur penulisan *script* PHP dalam HTML.

```
<html>
<head>
<title> </title>
</head>
<body>
<?php
```

```
?>
</body>
</html>
```

2.14 *Responsive Voice*

Responsive voice ini merupakan sebuah TTS *tools* berbasis javascript yang dapat digunakan untuk membangkitkan teks menjadi ucapan dalam berbagai ucapan bahasa. Website *responsive voice* dapat diakses melalui https://responsivevoice.org/. Misi utamanya adalah menyediakan layanan teks berbicara yang mencakup semua negara. Tersedia banyak jenis bahasa yang bisa dipilih, termasuk bahasa Indonesia. Kualitas suara dinilai mencapai 10/10, yang artinya sudah selayaknya manusia berbicara dengan nada dan intonasi yang tepat. *Service* dari *responsive voice* dibagikan secara gratis dan bebas untuk digunakan secara nonkomersial sebagai solusi untuk sintesa ucapan sesuai dengan bahasa yang dibutuhkan. Satu hal yang menjadi kekurangan dari *responsive voice* ini adalah fitur berhenti sejenak atau terdapat jeda otomatis pada kalimat yang panjang, karena keseringan justru membuat panjang pendeknya menjadi tidak pas sehingga makna dari kalimat yang dibunyikan menjadi tidak sesuai. Kekurangan dari *responsive voice* ini mucul apabila jumlah karakter pada kalimat melebihi 100 karakter termasuk dengan tanda baca (“,” dan “.”). Namun ada beberapa kondisi lain dari *responsive voice* yaitu ketika suatu kalimat panjang dengan jumlah karakter diatas 100 karakter memiliki tanda baca buka kurung, tutup kurung, titik dua atau lain sebagainya didalam kalimat, maka *responsive voice* akan meletakkan jeda otomatis tepat setelah tanda baca tersebut. *Responsive voice* dapat diterapkan pada sebuah website yang bertugas membacakan isi berita yang ada hanya dengan sekali klik oleh pengunjung website. Keuntungannya adalah pengunjung website masih dapat *browsing* pada website lain sementara masih tetap bisa mendengarkan isi beritanya.

2.15 Pengujian *Black Box*

Pengujian *black box* berfokus pada fungsional perangkat lunak yang dibuat. Dengan demikian, pengujian *black box* memungkinkan perekayasa perangkat lunak mendapatkan serangkaian kondisi input yang sepenuhnya menggunakan semua persyaratan fungsional untuk suatu program. Pengujian *black box* merupakan

sebuah metode yang digunakan untuk menemukan kesalahan dan mendemonstrasikan fungsional aplikasi saat dioperasikan, apakah *input* diterima dengan benar dan *output* yang dihasilkan telah sesuai dengan yang diharapkan. Pengujian *black box* cenderung untuk menemukan hal-hal berikut.

- Fungsi yang tidak benar atau tidak ada.
- Kesalahan antarmuka (*interface errors*).
- Kesalahan performansi (*performance errors*).

2.16 Pengujian Akurasi

Pengujian akurasi merupakan sebuah pengujian yang dilakukan untuk menguji tingkat akurasi *rule grammars* yang dikembangkan. Pengujian akurasi dilakukan dengan mencocokkan kalimat antara *rule grammars* yang dikembangkan peneliti dengan data jeda keras yang didapat dari narasumber. Pengujian bertujuan untuk mendapatkan *rule grammars* terbaik yang digunakan untuk proses penjedaan.

Narasumber akan diberikan teks, kemudian setiap narasumber akan memberikan jeda berupa tanda koma (,) pada setiap kalimat dengan syarat jumlah karakter dari satu jeda ke jeda yang selanjutnya tidak lebih dari 99 karakter. Namun pemberian jeda oleh narasumber tetap harus memperhatikan frasa yang terbentuk. Data dari narasumber yaitu berupa teks berita yang telah terpenggal yang kemudian disatukan untuk mendapatkan jeda keras pada teks berita. Kalimat yang telah diberi jeda keras kemudian dihitung jumlah karakter setiap kalimat tunggalnya. Kalimat yang memiliki jumlah karakter ≤99 karakter akan masuk ke perhitungan akurasi.

Data kalimat dengan jeda keras yang didapat akan dicocokkan dengan kalimat terpenggal hasil dari proses *chunking* pada aplikasi. Beberapa hal yang perlu diperhatikan pada saat proses mencocokkan sebagai berikut.

- Letak/ posisi jeda (,) teks chunking harus sama dengan letak/ posisi jeda keras pada teks narasumber.
- Jumlah koma (,) teks chunking harus sama dengan jumlah koma pada teks narasumber.

Setelah proses pencocokkan selesai, maka dilakukan perhitungan untuk menghitung persentase tingkat akurasi *rule grammars*. Perhitungan persentase tingkat akurasi *rule grammars* dapat dilihat pada Persamaan (2.1) berikut.

Tingkat akurasi (%) = $\frac{Jumlah\ kalimat\ yang\ cocok}{Jumlah\ kalimat\ keseluruhan}$ x 100 (2.1)

2.17 *Precision* & *Recall*

Rozi (2012) menjelaskan bahwa, *precision* adalah rasio jumlah dokumen relevan yang ditemukan dengan total jumlah dokumen yang ditemukan oleh sistem. *Recall* adalah rasio jumlah dokumen relevan yang ditemukan kembali dengan total jumlah dokumen dalam kumpulan dokumen yang dianggap relevan. R*ecall* (perolehan) berhubungan dengan kemampuan sistem untuk memanggil dokumen yang relevan. Sedangkan *precision* (ketepatan) berkaitan dengan kemampuan sistem untuk tidak memanggil dokumen yang tidak relevan. Berikut tabel variabel yang digunakan pada perhitungan *precision* dan *recall*, dapat dilihat pada Tabel 2.8.

Tabel 2.8 Variabel untuk Perhitungan *Precision* dan *Recall*

	Relevan	Tak Relevan
Ditemukan	True positives (tp)	False positives (fp)
Tidak ditemukan	False negatives (fn)	True negatives (tn)

Persamaan (2.2) dan (2.3) untuk *precision* dan *recall* sebagai berikut.

$$Precision = \frac{\text{item yang relevan yang ditemukan}}{\text{item ditemukan}} = P\ (relevan|ditemukan) \quad (2.2)$$

$$Recall = \frac{\text{item yang relevan yang ditemukan}}{\text{semua item yang relevan}} = P\ (ditemukan|relevan) \quad (2.3)$$

Kemudian dapat disederhanakan ke bentuk Persamaan (2.4) dan (2.5).

$$Precision = \frac{\text{tp}}{\text{tp} + \text{fp}} \quad (2.4)$$

$$Recall = \frac{\text{tp}}{\text{tp} + \text{fn}} \quad (2.5)$$

Terdapat dua hal penting yang biasanya digunakan sebagai acuan dalam mengukur keefektifan suatu sistem temu kembali informasi yaitu perolehan (*recall*) dan ketepatan (*precision*). Keefektifan suatu sistem temu kembali informasi dinilai berdasarkan teori yang dicetuskan oleh Lancaster (1991) dalam Pendit (2008) yaitu relevan dan tidak relevan, jadi efektifitas temu kembali informasi dibedakan menjadi efektif jika nilai diatas 50% dan tidak efektif jika nilai dibawah nilai 50%. Selain itu, suatu sistem temu kembali dinyatakan efektif apabila hasil penelusuran

mampu menunjukkan ketepatan (*precision*) yang tinggi sekalipun perolehannya (*recall*) rendah (Rowley 1992) dalam Hasugian (2006).

2.18 *F-Measure*

F-measure (nilai F atau *F-Score* atau *F-measure*) merupakan sebuah nilai dari keakuratan sebuah tes. Nilai F menggunakan *precision* dan *recall* dari tes untuk menghitung nilainya, dengan *precision* yang menyatakan jumlah hasil benar dibandingkan dengan jumlah hasil yang ditemubalikkan dan *recall* yang menyatakan jumlah hasil benar dibandingkan dengan jumlah hasil yang harus ditemubalikkan. Nilai F dapat diartikan sebagai sebuah hasil rata-rata dari *precision* dan *recall*, dimana sebuah nilai F mendapat hasil 1 sebagai hasil terbaik dan 0 sebagai nilai terburuk. Persamaan (2.6) untuk menghitung nilai F sebagai berikut.

$$F = 2 . \frac{precision . recall}{precision+recall} \quad (2.6)$$

2.19 Pengujian Subjektif

Pengujian subjektif merupakan sebuah pengujian yang dilakukan secara langsung kepada pendengar untuk mendengar dan menilai kualitas ucapan dari suatu kalimat. Uji dengar pada pengujian subjektif bertujuan untuk menyatakan frasa ucapan suatu kalimat dapat diterima atau tidak dapat diterima oleh pendengar. Hal ini dimaksudkan untuk mendapatkan *rule grammars* terbaik yang digunakan untuk proses penjedaan kalimat. Pengujian subjektif dilakukan dengan memberikan lembaran kertas berupa *form* kepada pendengar yang isinya berupa tabel yang kemudian akan diisi oleh pendengar. Pendengar mengisi *form* dengan mengikuti instruksi yang diberikan. Pendengar akan mendengarkan suara dari *responsive voice* dengan kalimat inputan berupa kalimat asli dan kalimat hasil pemenggalan menggunakan *rule grammars*. Data yang didapat dari pendengar kemudian dihitung untuk mendapatkan persentase ucapan diterima dari kalimat yang diucapkan. Adapun yang perlu diperhatikan sebelum proses perhitungan yaitu konsistensi dari pendengar. Hal ini bertujuan untuk mendapatkan konsistensi isian dari pendengar dalam mengisi *form*.

BAB III
METODOLOGI PENELITIAN

3.1 Spesifikasi Perangkat Keras dan Perangkat Lunak

Alat yang digunakan dalam penelitian ini berupa perangkat lunak dan perangkat keras. Adapaun alat yang digunakan adalah sebagai berikut:

3.1.1 Perangkat keras

Pada dasarnya, perangkat keras adalah komponen komputer yang dapat dilihat secara langsung atau berbentuk nyata dan berfungsi untuk mendukung proses komputerisasi yang ada. Dalam penelitian ini, perangkat keras yang digunakan yaitu:

1. Laptop Asus A455LJ
2. Prosessor Intel(R) Core i5
3. RAM 4 GB
4. *Harddisk Drive* 500 GB
5. *Speaker*
6. LCD Monitor 14 inchi
7. *Mouse*

3.1.2 Perangkat lunak

Berbeda dengan perangkat keras, perangkat lunak adalah program komputer yang berfungsi sebagai sarana interaksi antara pengguna dan perangkat keras. Adapun perangkat lunak yang digunakan adalah sebagai berikut:

1. Sistem operasi Ubuntu Desktop versi 17.04 64-bit

Sistem operasi adalah perangkat lunak sistem yang mengelola perangkat keras komputer dan sumber daya perangkat lunak dan menyediakan layanan umum unutk program komputer. Tanpa sistem operasi, aplikasi tidak dapat digunakan. Sistem operasi Ubuntu Desktop akan digunakan sebagai media untuk membuat dan menjalankan aplikasi *text to speech* pada website.

2. Sistem operasi Ubuntu Server versi 16.04 64-bit

Sistem operasi Ubuntu Server merupakan salah satu distro sistem operasi

berbasis Linux. Sistem Operasi Ubuntu Server digunakan sebagai sarana untuk tahap implementasi sistem dari proses pemenggalan kalimat.

3. *Java Development Kit* (JDK) dan *Java Runtime Environment* (JRE) 8 (64-bit)

Java Development Kit (JDK) merupakan perangkat lunak untuk memanajemen dan membangun berbagai aplikasi java. JDK adalah paket instalan Java yang biasanya dibutuhkan oleh *programmer* dengan memanfaatkan *compiler* yang terdapat didalamnya untuk menjalankan *bytecode* dari program. Pada JDK juga terdapat JRE untuk menjalankan program java. JDK wajib terinstall pada komputer yang akan melakukan proses pembuatan aplikasi berbasis java, namun tidak wajib terinstall di komputer yang akan menjalankan aplikasi yang dibangun dengan java. Bahasa pemrograman java pada penelitian ini digunakan untuk mengembangkan dan menjalankan program java pada proses PoS *tagging*.

4. Sublime *Text*

Sublime *Text* adalah aplikasi editor untuk kode dan teks yang dapat berjalan diberbagai *platform* operating system. Perangkat lunak berfungsi untuk mempermudah tahap implementasi sistem yaitu mengubah perancangan sistem yang berupa diagram alir menjadi instruksi-instruksi dalam bahasa pemrograman. Sublime mendukung berbagai bahasa pemrograman, diantaranya yang digunakan pada penelitian yaitu antara lain HTML, PHP, python dan java.

5. Built-in Web Server

Web Server adalah suatu perangkat lunak (*software*) dalam server yang berfungsi untuk menerima permintaan (*request*) dari *client* atau *browser* berupa halaman website melalui protokol HTTP/ HTTPS, lalu merespon permintaan tersebut dalam bentuk halaman website berupa dokumen HTML atau PHP. Web server merupakan software yang berfungsi sebagai pemberi layanan kepada web *client* (*browser*) seperti chrome agar browser tersebut dapat menampilkan halaman website yang telah dibuat.

6. PHP 7.0.33

PHP merupakan bahasa pemrograman berbasis server-side yaitu kumpulan instruksi akan dieksekusi di server. PHP digunakan untuk mengolah inputan pengguna dari halaman web dan memberikan output kepada pengguna.

7. MariaDB 10.0.38

MariaDB adalah sistem manajemen database relasional yang dikembangkan dari MySQL. MariaDB adalah sebuah implementasi dari sistem manajemen basisdata relasional yang didistribusikan secara gratis. MariaDB dapat berjalan stabil pada berbagai sistem operasi seperti Linux. MariaDB pada penelitian ini digunakan sebagai database untuk menyimpan data pada website yang dibuat.

8. Python 3.7.2

Python adalah bahasa pemrograman tingkat tinggi yang ditujukan untuk tujuan umum (*general purpose*). Python pada penelitian ini digunakan untuk membuat dan menjalankan program untuk proses pemenggalan kalimat menjadi teks yang terpenggal.

9. NLTK (*Natural Language Toolkit*)

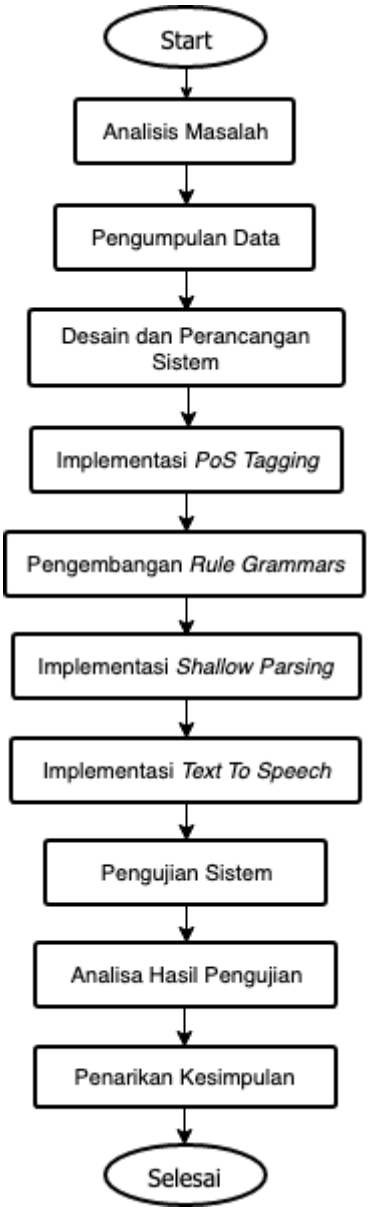
NLTK merupakan salah satu modul dalam python yang ada untuk mendukung teknologi NLP (*Natural Language Processing*). Modul NLTK perlu diinstall agar lebih *powerful* dalam melakukan tugas-tugas standar NLP.

10. *Google Chrome*

Google Chrome sebagai salah satu *browser* yang digunakan untuk menjalankan sistem yang sedang dibangun dalam website.

3.2 Metode Penelitian

Metode penelitian merupakan beberapa tahapan yang dirancang dan dijadikan sebagai panduan dalam melakukan sebuah penelitian. Metode penelitian yang dilakukan digambarkan pada diagram alir penelitian pada Gambar 3.1.



Gambar 3.1 Diagram Alir Penelitian

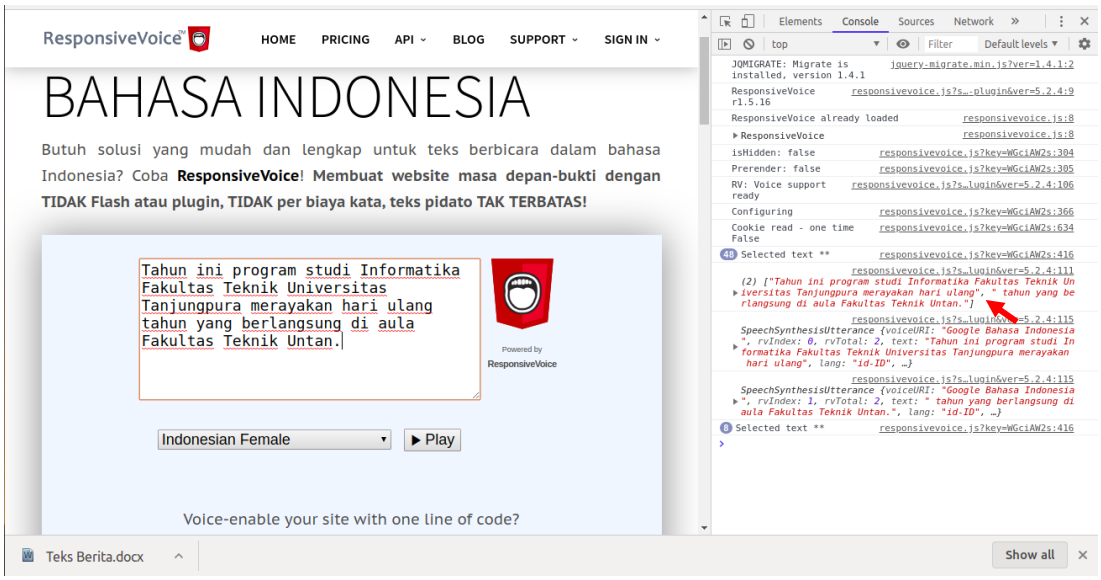
3.2.1 Analisis Masalah

Analisis masalah dilakukan untuk menemukan masalah yang ada pada proses implementasi *text to speech*. Dalam proses ini ditemukan kekurangan dari pembangkit ucapan yang digunakan dalam penelitian yaitu *responsive voice*.

Responsive voice merupakan sebuah *tools* yang dapat digunakan untuk membangkitkan teks menjadi ucapan dalam berbagai ucapan bahasa. Namun dalam penerapannya, terdapat beberapa kekurangan seperti ucapan pada frasa yang tidak sesuai khususnya untuk ucapan kalimat masukan yang panjang. Dalam hal ini, frasa merupakan satu-kesatuan yang tidak bisa dipisah karena dapat mempengaruhi makna dari kalimat yang diucapkan. Misalnya:

“Tahun ini program studi Informatika Fakultas Teknik Universitas Tanjungpura merayakan hari ulang tahun yang berlangsung di aula Fakultas Teknik Untan.”

Kalimat tersebut seharusnya diucapkan sesuai dengan frasa yang semestinya. Saat kalimat diatas dibunyikan, pada frasa “ulang tahun” yang seharusnya diucapkan dalam satu-kesatuan tetapi terjadi jeda dalam frasa tersebut. Hal ini dikarenakan *responsive voice* akan memberi jeda secara otomatis ketika jumlah karakter melebihi 100 karakter. Pemberian jeda otomatis oleh *responsive voice* dapat dilihat pada Gambar 3.2.



Gambar 3.2 Pemberian Jeda Otomatis pada *Responsive Voice*

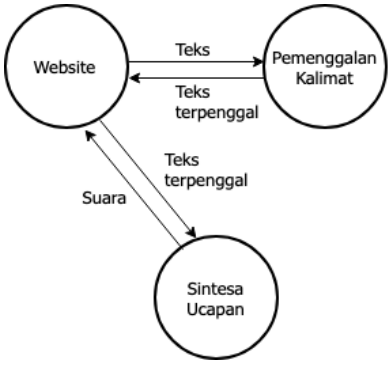
Pada kekurangan yang ada, peneliti menentukan metode agar pengucapan frasa yang dibunyikan sesuai dengan frasa yang ada. Sehingga dari permasalahan yang ditemukan, metode yang digunakan yaitu metode *shallow parsing*. Dalam mengimplementasikan metode *shallow parsing*, dibutuhkan *rule grammars* dan data kelas kata dari PoS *tagging*.

3.2.2 Pengumpulan Data

Pengumpulan data dilakukan dengan menggunakan metode wawancara dan observasi. Data yang digunakan berupa teks berita. Wawancara dan observasi bertujuan untuk mendapatkan data dari narasumber berupa data jeda keras dari teks berita. Teks berita yang digunakan pada penelitian adalah 5 berita yang ada dalam halaman berita website Teknik Informatika Untan pada http://informatika.untan.ac.id/?list&berita dengan jumlah kalimat sebanyak 74 kalimat yang sebagian besar berisi kalimat-kalimat panjang dengan jumlah kata sebanyak 1912 kata dan jumlah kata tiap kalimat rata-rata sebanyak 26 kata. Total narasumber sebanyak 3 orang. Setiap narasumber akan memenggal kalimat pada teks dengan memberikan tanda "," pada kalimat. Hasil dari observasi ini yaitu berupa data jeda keras. Pengumpulan data jeda keras digunakan untuk pengujian akurasi dan pengujian *precision*, *recall* dan *f-measure*.

3.2.3 Desain dan Perancangan Sistem

Sistem yang dibangun terdiri dari 3 buah subsistem diantaranya sistem website, sistem untuk pemenggalan kalimat dan sistem untuk sintesa ucapan. Sistem yang ada terdiri dari dua bagian yaitu *front-end* dan *back-end*. Bagian *front-end* merupakan bagian yang dapat dilihat dan dilakukan oleh pengguna yaitu meng-klik tombol suara pada halaman berita website untuk proses pengiriman teks dari website ke sistem pemenggalan kalimat dan mensintesa teks menjadi ucapan. Bagian *back-end* merupakan bagian yang mengolah teks menjadi teks terpenggal pada sistem pemenggalan kalimat dan mengirim kembali teks terpenggal ke website, kemudian teks terpenggal yang diterima dikirim ke *responsive voice* untuk dibunyikan. Berikut Gambar 3.3 sebagai gambaran sistem yang dibangun.



Gambar 3.3 Pemodelan Sistem yang Dibangun Oleh 3 Sub-Sistem

Adapun diagram alir proses pada *front-end* sampai *back-end* digambarkan dengan diagram alir pada Gambar 3.4.

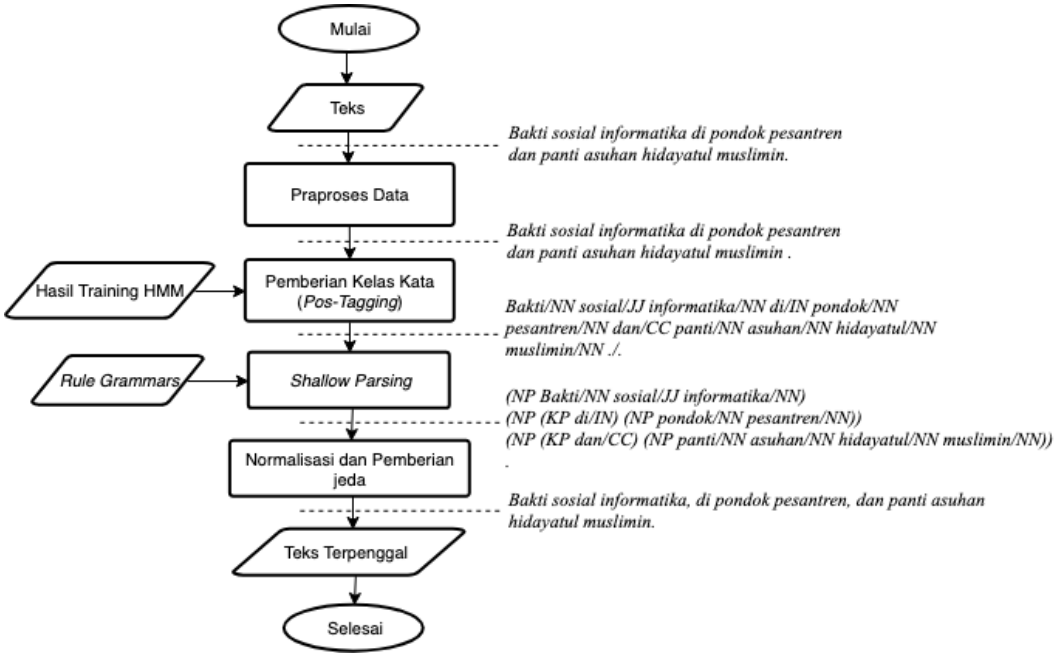
Gambar 3.4 Diagram Alir Sistem

3.2.3.1 Pengiriman Teks

Pengiriman teks adalah proses pengiriman teks berita pada website. Teks pada setiap paragraf diambil kemudian dikirimkan ke server untuk proses pemenggalan kalimat. Terdapat dua jenis teks berita yang diambil dari website yaitu teks dari judul berita yang terambil dan terkirim secara otomatis saat halaman detail berita dimuat dan teks dari isi berita yang diambil dan dikirim ketika *user* menekan tombol suara pada halaman detail berita.

3.2.3.2 Pemenggalan Kalimat

Pemenggalan kalimat adalah proses pemenggalan kalimat yang ada pada server. Setelah teks dikirim dari website, maka server kemudian akan menerima dan mengolah teks tersebut menjadi teks terpenggal. Berikut diagram alir proses pemenggalan kalimat dengan metode *shallow parsing* pada Gambar 3.5.



Gambar 3.5 Diagram Alir Proses Pemenggalan Kalimat dengan Metode *Shallow Parsing*

Dari diagram pada Gambar 3.5, dapat dijelaskan sebagai berikut.

1) Praproses data: data berupa teks yang dikirim dari website pertama kali akan masuk ke proses praproses data. Data teks yang masuk akan ditransformasi terlebih dahulu sebelum data tersebut masuk ke PoS *tagging*. Hal ini bertujuan untuk menghindari kesalahan saat proses PoS *tagging*. Pada praproses data, beberapa yang perlu diperbaiki diantaranya memberikan spasi pada simbol-simbol yang menyatu pada kata. Simbol-simbol yang dipisahkan yaitu tanda koma ",", buka kurung "(", tutup kurung ")", titik dua ":", tanda titik ".", tanda tanya "?", titik koma ";" dan *backslash* "\". Berikut contoh kalimat hasil dari praproses data.

Bakti sosial informatika di pondok pesantren dan panti asuhan hidayatul muslimin .

2) Pemberian kelas kata (PoS *tagging*): hasil transformasi pada praproses kemudian akan masuk ke PoS *tagging* untuk pemberian label kelas kata pada setiap kata ataupun tanda baca yang ada dalam kalimat. Pada proses PoS tagging dibutuhkan data hasil dari *training* HMM. Lalu kata yang sudah diberi label ini kemudian masuk ke proses *shallow parsing*. Berikut contoh dari hasil proses PoS *tagging*.

Bakti/NN sosial/JJ informatika/NN di/IN pondok/NN pesantren/NN dan/CC panti/NN asuhan/NN hidayatul/NN muslimin/NN ./.

3) *Shallow parsing*: hasil dari PoS *tagging* akan dikelompokkan menjadi frasa-frasa yang sesuai dengan aturan/ *rule grammars* Bahasa Indonesia yang telah dibangun. Berikut contoh dari hasil proses *shallow parsing*.

(NP Bakti/NN sosial/JJ informatika/NN)
(NP (KP di/IN) (NP pondok/NN pesantren/NN))
(NP (KP dan/CC) (NP panti/NN asuhan/NN hidayatul/NN muslimin/NN))
.

4) Normalisasi dan pemberian jeda: hasil dari proses *shallow parsing* yang berisi frasa-frasa kemudian dinormalisasi dan diberi jeda yaitu pemberian tanda koma (,) pada setiap frasa. Berikut contoh dari hasil proses normalisasi dan pemberian jeda.

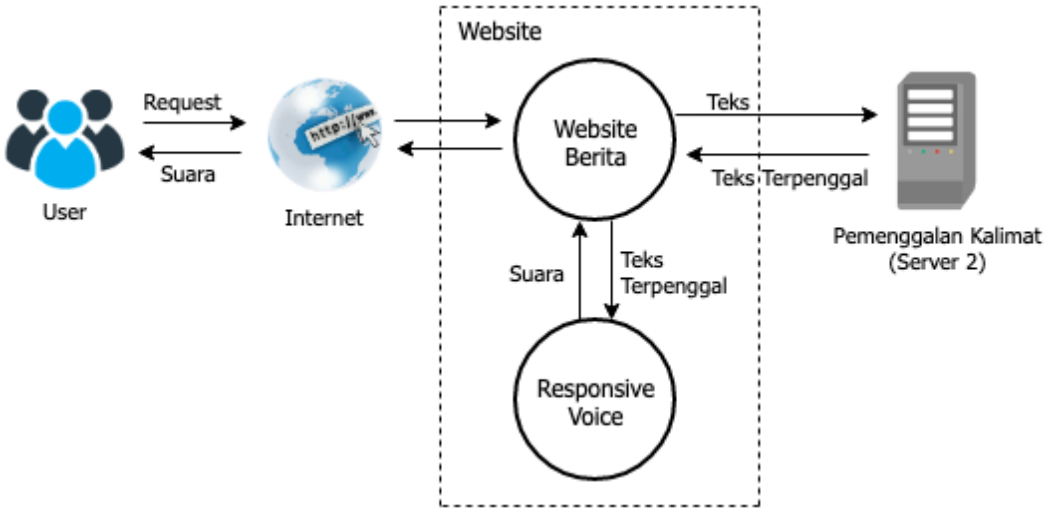
Bakti sosial informatika, di pondok pesantren, dan panti asuhan hidayatul muslimin.

3.2.3.3 Implementasi *Text To Speech*

Hasil dari proses normalisasi dan pemberian jeda pada sistem pemenggalan kalimat berupa teks yang sudah terpenggal akan dikirim kembali ke website dan dibangkitkan menjadi ucapan menggunakan *tools* dari *responsive voice*. Output *dari responsive voice* yaitu berupa suara dalam Bahasa Indonesia.

3.2.3.4 Arsitektur Sistem

Arsitektur sistem merupakan gambaran secara umum mengenai hubungan antar komponen yang terlibat dalam sistem yang dibangun. Sistem terdiri dari pengguna, website, server, dan *responsive voice*. Pengguna merupakan orang awam atau *end-user*. Setiap pengguna memiliki hak akses (*privilege*) yang sama. Halaman web dapat diakses melalui *browser* dengan bantuan jaringan internet. Berikut arsitekstur sistem penelitian pada Gambar 3.6.

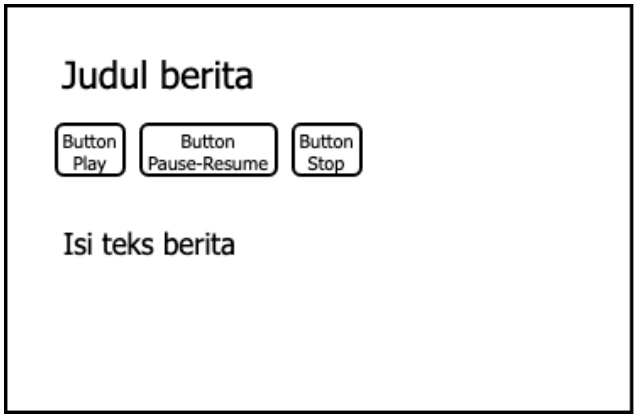


Gambar 3.6 Arsitektur Sistem Implementasi *Text To Speech* Pada Website Berita Menggunakan Metode *Shallow Parsing*

Pengguna mengakses website dengan bantuan jaringan internet. Saat pengguna meng-klik tombol bunyi pada halaman berita, maka teks pada website akan diambil dan kirim ke server pemenggalan kalimat untuk mendapatkan frasa menggunakan metode *shallow parsing*. Hasil dari proses pemenggalan yaitu berupa teks terpenggal. Teks terpenggal ini akan dikirim kembali ke website. Kemudian teks terpenggal akan dikirim ke *responsive voice* untuk dibunyikan. *Output*/ keluaran dari sistem ini adalah ucapan dalam Bahasa Indonesia.

3.2.3.5 Rancangan Antarmuka Pengguna Pada Website

Rancangan antarmuka pengguna pada website bertujuan merancang sebuah antarmuka (*user interface*) sistem untuk implementasi *text to speech* yang ditujukan kepada pengguna website. Rancangan antarmuka yang digunakan berupa GUI (*Graphic User Interface*) untuk memudahkan pengguna dalam menggunakan sistem, jadi pengguna tidak perlu mengetik teks perintah yang ingin dijalankan, akan tetapi pengguna hanya perlu mengklik gambar atau ikon yang sudah dibuat untuk menjalankan fungsinya. Antarmuka sistem *text to speech* yang dirancang berada pada halaman detail berita website. Beberapa tombol dirancang untuk menghubungkan antara pengguna dengan sistem. Rancangan antarmuka pada halaman detail berita dapat dilihat pada Gambar 3.7.



Gambar 3.7 Rancangan Antarmuka Pengguna pada Website

3.2.4 Implementasi PoS *Tagging*

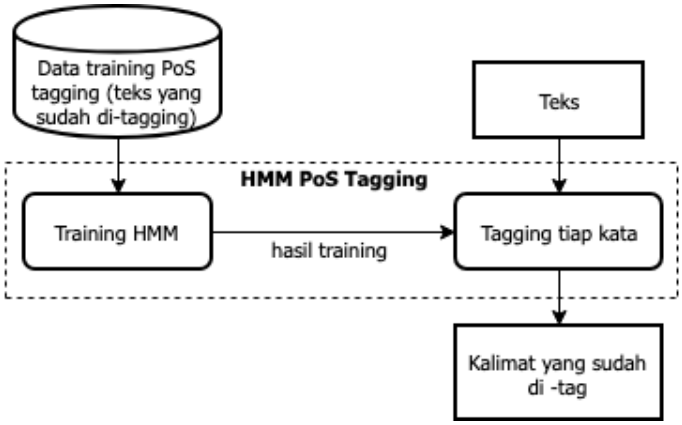
PoS *tagging* adalah proses memberikan label kelas kata secara otomatis pada setiap kata yang ada pada suatu teks atau dokumen. Implementasi PoS *tagging* bertujuan mengimplementasikan PoS *tagging* dari penelitian yang sudah ada sebelumnya untuk mendapatkan label kelas kata pada kalimat. Penelitian sebelumnya yaitu penelitian M. Iqbal Kamiludin (2017) pada bahasa melayu Pontianak menggunakan set PoS Bahasa Indonesia milik Alfan Farizki Wicaksono (2010). Dalam pengelompokkan kelas kata, set PoS Bahasa Indonesia yang dipakai yaitu set PoS dari penelitian Alfan Farizki Wicaksono (2010) yang berjumlah 35 tag yang terdiri dari kategori tata bahasa seperti kata kerja, kata benda, kata sifat, kata keterangan, kata tugas dan simbol. *Tagge*r ini dibangun oleh sekitar 15000 token data *corpus*. Set PoS Alfan Farizki Wicaksono (2010) dapat dilihat pada Tabel 3.1.

Tabel 3.1 Set PoS Alfan Farizki Wicaksono (2010)

No	PoS	Deskripsi	Contoh Kata
1	OP	*Parenthesis*	({[
2	CP	*Close Parenthesis*	)}]
3	GM	*Slash*	/
4	;	*Semicolon*	;
5	:	*Colon*	:
6	“	*Quotation*	“ ’
7	.	*Sentence Terminator*	. ! ?
8	,	*Comma*	,
9	-	*Dash*	-

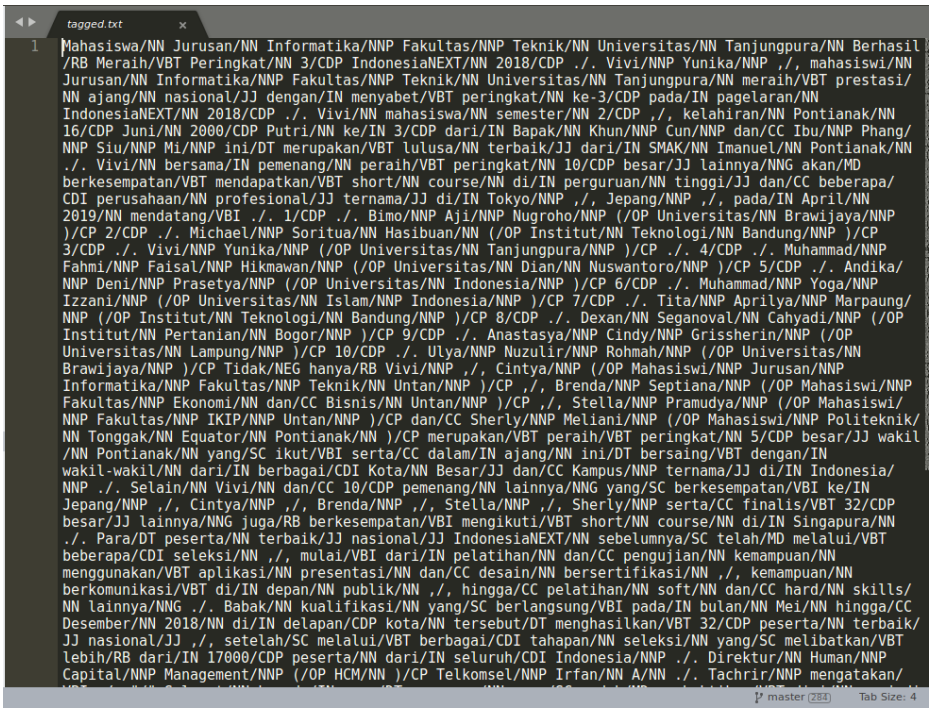
10	...	*Ellipsis*	...
11	JJ	*Adjective*	Kaya, Manis
12	RB	*Adverb*	Nanti, Sementara
13	NN	*Common Noun*	Mobil, Sepatu, Roda
14	NNP	*Proper Noun*	Bekasi, Indonesia
15	NNG	*Genitive Noun*	Bukunya
16	VBI	*Intransitive Verb*	Pergi
17	VBT	*Transitive Verb*	Membeli
18	IN	*Preposition*	Di, Ke, Dari
19	MD	*Modal*	Bisa
20	CC	*Coor-Conjunction*	Dan, Atau, Tetapi
21	SC	*Subor-Conjunction*	Jika, Ketika
22	DT	*Determiner*	Para, Ini, Itu
23	UH	*Interjection*	Wah, Aduh, Oi
24	CDO	*Ordinal Numerals*	Pertama, Kedua
25	CDC	*Collective Numerals*	Bertiga
26	CDP	*Primary Numerals*	Satu, Dua
27	CDI	*Irregular Numerals*	Beberapa
28	PRP	*Personal Pronouns*	Saya, Kamu
29	WP	*WH-Pronouns*	Apa, Siapa, Dimana
30	PRN	*Number Pronouns*	Kedua-duanya, Satu-satunya
31	PRL	*Locative Pronouns*	Sini , Situ, Sana
32	NEG	*Negation*	Bukan, Tidak
33	SYM	*Symbols*	@#$%^&
34	RP	*Particles*	Pun, Kah
35	FW	*Foreign*	Words

Dalam penelitian ini, pengembangan PoS *tagging* menggunakan metode HMM (*Hidden Markov Model*). HMM adalah metode probabilistik yang ditetapkan untuk penandaan PoS otomatis. Beberapa bahasa telah mengadaptasi metode HMM dalam membangun tagger PoS otomatis termasuk Bahasa Indonesia. Berikut tahapan proses tagging dengan model HMM pada Gambar 3.8.



Gambar 3.8 Tahapan PoS *Tagging* dengan Model HMM

Aplikasi PoS *tagging* dengan set PoS milik Wicaksono (2010) didapat dari peneliti yang sudah ada sebelumnya. Aplikasi tersebut menggunakan bahasa java, kemudian disesuaikan kembali menyesuaikan kebutuhan dalam penelitian. Berikut hasil proses PoS *tagging* dari kalimat yang sudah diinputkan pada Gambar 3.9.



Gambar 3.9 Hasil PoS *Tagging*

Output dari proses PoS *tagging* seperti Gambar 3.9 adalah kata yang sudah berlabel, digunakan untuk keperluan analisis pada perancangan *rule grammars*.

3.2.5 Pengembangan *Rule Grammars*

Pengembangan *rule grammars* bertujuan untuk mendapatkan *rule grammars* yang sesuai dalam menyelesaikan masalah pada penelitian. *Rule grammars* merupakan aturan-aturan tata bahasa yang dibutuhkan untuk membentuk frasa pada proses *shallow parsing*. Pada peneliti sebelumnya, telah dilakukan penelitian mengenai *shallow parsing* dalam Bahasa Indonesia yaitu penelitian milik Arman, dkk. (2013). Tipe-tipe frasa yang dikembangkan pada penelitian tersebut masih belum sesuai dengan yang diharapkan, sehingga perlu dikembangkan kembali tipe frasa yang sesuai. Hal ini dikarenakan penelitian tidak hanya berfokus pada pencarian frasa saja. Frasa yang didapat akan digunakan untuk mendapatkan penggalan kalimat yang sesuai yang tidak melebihi 99 karakter pada setiap penggalannya.

3.2.5.1 Pengembangan Tipe Frasa

Tipe-tipe frasa yang dikembangkan dalam penelitian ini dimaksudkan untuk membentuk frasa yang terkait dengan kategori sintaksis dan dapat digunakan untuk menginformasikan pembentukan frasa prosodik agar menghasilkan penggalan kalimat yang baik. Jenis tipe frasa yang digunakan dalam penelitian ini diambil dari penelitian yang dilakukan oleh Arman, dkk. (2013), dan ditambah dengan satu tipe frasa baru yakni G (frasa gabungan). Tujuannya untuk menghindari kesalahan dalam pembentukan frasa dan mendapatkan frasa yang sesuai. Berikut Tabel 3.2 berupa tipe frasa yang digunakan pada penelitian.

Tabel 3.2 Tipe Frasa 35 Set PoS

Tipe Frasa	35 Set PoS
Connection Phrases (KP)	SC, UH
Verb Phrases (VP)	VBI, VBT, MD
Noun Phrases (NP)	WP, PRP, PRN
Combined Phrases (G)	IN, CC, NEG, NN, RB, DT, VBT, CDP, CDI

- *Connection Phrases* (KP): merupakan frasa kata penghubung. Kelas kata yang digunakan untuk tipe frasa KP adalah kata tugas yakni kata hubung dan kata seru. Tipe frasa ini bertujuan untuk mendapatkan frasa dari kata hubung (label PoS SC) maupun kata seru (label PoS UH).
- *Verb Phrases* (VP): merupakan frasa kata kerja. Kelas kata yang digunakan untuk tipe frasa VP adalah kata kerja yakni kata kerja dan kata kerja ganti. Tipe frasa ini bertujuan untuk mendapatkan frasa dari kata kerja (label PoS VBI, VBT) dan kata kerja ganti (label PoS MD).
- *Noun Phrases* (NP): merupakan frasa kata benda. Kelas kata yang digunakan untuk tipe frasa NP adalah kata benda yakni kata ganti orang. Tipe frasa ini bertujuan untuk mendapatkan frasa dari kata ganti orang (label PoS WP, PRP, PRN).
- *Combined Phrases* (G): merupakan grasa gabungan. Kelas kata yang digunakan untuk tipe frasa G terdiri dari kata benda, kata keterangan dan kata tugas. Tipe frasa ini bertujuan untuk menghindari adanya kesalahan pada pemenggalan frasa yangmana terdapat frasa tertentu yang tidak boleh dipisah pada teks berita.

3.2.5.2 Perancangan *Rule Grammars*

Perancangan *rule grammars* pada penelitian bertujuan untuk mendapatkan frasa yang sesuai yang dapat membentuk penggalan kalimat dengan jumlah karakter tidak lebih dari 99 karakter. Perancangan *rule grammars* dibuat dengan menganalisis teks berita yang ada pada website teknik informatika. Dilihat dari teks yang ada, pemberian jeda dapat dilakukan dengan memperhatikan jenis kelas kata. Beberapa jenis kelas kata diambil untuk proses pemenggalan kalimat diantaranya kata hubung, kata kerja dan kata benda. Selain tiga kelas kata ini, terdapat *rule* gabungan dan *rule* lain yang digunakan untuk menghindari kesalahan saat pemenggalan kalimat.

Rule grammars yang dibuat pada penelitian yaitu sebanyak 3 opsi *rule grammars*. Ketiga opsi ini akan dipilih salah satu. Proses penentuan *rule grammars* dilakukan dengan melakukan pengujian. Hal ini bertujuan untuk mendapatkan satu *rule grammars* yang memiliki tingkat persentase paling tinggi. Dari proses

pengujian, didapat satu *rule grammars* yang akan diimplementasikan pada proses *shallow parsing*.

Dalam pengimplementasiannya, *rule grammars* menggunakan *regular expression*. *Regular expression* atau yang lebih sering disebut *regex* ini memiliki 2 fungsi utama yakni mencari dan mengganti, mencari suatu pola tertentu dalam *text* lalu menggantinya menjadi pola yang lain. Berikut karakter *regular expressions* yang digunakan dalam *shallow parsing* dapat dilihat pada Tabel 3.3.

Tabel 3.3 Makna karakter Regular Expressions pada Shallow Parsing

Karakter	Makna karakter *Regular Expressions*
< >	Penentuan tag part-of-speech
?	Nol atau salah satu item sebelumnya
*****	Nol atau lebih dari item sebelumnya.
+	Satu atau lebih dari item sebelumnya
|	Mencocokkan satu item dengan yang lainnya,

Rule grammars yang digunakan pada proses *shallow parsing* seperti pada Gambar 3.10 berikut.

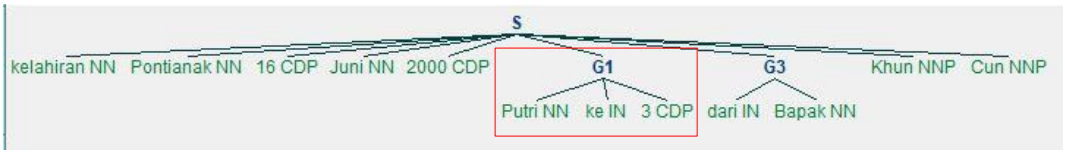
```
G1 : { <NN> <IN> <CDP> }
G2 : { <NN> <VBT> <IN> }

KP : { <SC|UH>+ }
VP : { <VBI|VBT|MD>+ <CC>* <NN|RB>* }
NP : { <WP|PRP|PRN>+ }

G3 : { <RB>* <IN>+ <NN>+ <VP>*}
G4 : { <NN>* <IN> <VP> <G3>* <NP>*}
G5 : { <CC> <JJ>* <G3|G4|VP|G1|NP|NN|CDI|DT|RB|NEG>+ }
G6 : { <RB>* <NP|VP|NEG>+ <RB>* <VP|KP|NP|DT>*
<G3|G4>* }
G7 : { <KP|G3>+ <G3|G4|G6|VP|NP|NN|RB>+ }
```

Gambar 3.10 *Rule Grammars* yang Digunakan pada Proses *Shallow Parsing*

***Rule* 1, G1 : { <NN> <IN> <CDP> }**

Frasa G1 adalah frasa gabungan. Frasa ini akan terbentuk jika ada 1 label PoS NN bertemu 1 label PoS IN dan kemudian bertemu 1 label PoS CDP. Jika 3 label PoS ini bertemu, maka frasa ini harus terbentuk. Contoh frasa G1 pada kalimat dan bentuk pohon parsingnya sebagai berikut.

Kalimat: kelahiran Pontianak 16 Juni 2000 Putri ke 3 dari Bapak Khun Cun



Gambar 3.11 Pohon Parsing *Rule* 1

***Rule* 2, G2 : { <NN> <VBT> <IN> }**

Frasa G2 adalah frasa gabungan. Frasa ini akan terbentuk jika ada 1 label PoS NN bertemu 1 label PoS VBT dan kemudian bertemu 1 label PoS IN. Jika 3 label PoS ini bertemu, maka frasa ini harus terbentuk. Contoh frasa G2 pada kalimat dan bentuk pohon parsingnya sebagai berikut.

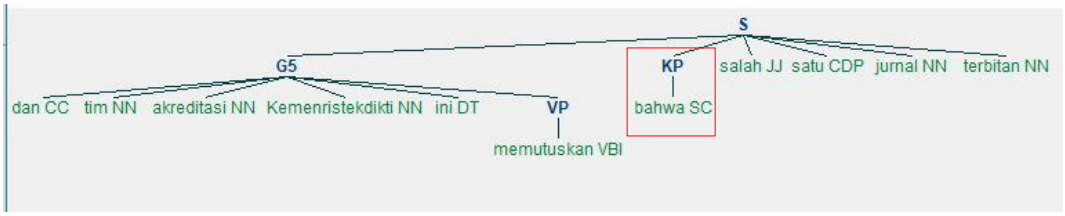
Kalimat: kemampuan berkomunikasi di depan publik

Gambar 3.12 Pohon Parsing *Rule* 2

***Rule* 3, KP : { <SC|UH>+ }**

Frasa KP adalah frasa yang terdiri dari kata penghubung. Frasa ini akan terbentuk jika terdapat label PoS SC|UH dengan keterangan *regular expression* "+" yang berarti harus muncul minimal 1 atau lebih dari satu item sebelumnya. Contoh frasa KP pada kalimat dan bentuk pohon parsingnya sebagai berikut.

Kalimat: dan tim akreditasi Kemenristekdikti ini memutuskan bahwa salah satu jurnal terbitan

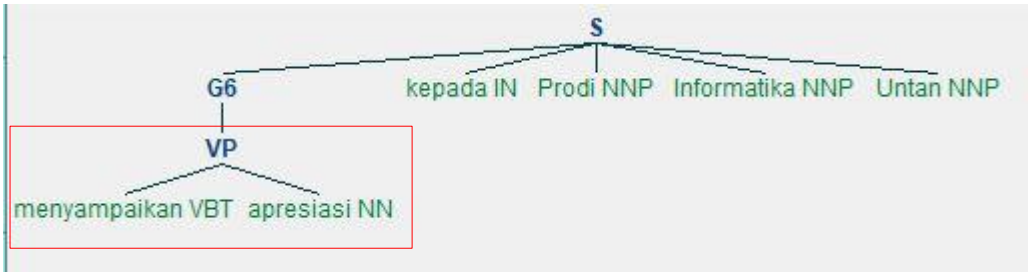


Gambar 3.13 Pohon Parsing *Rule* 3

***Rule* 4, VP : { <VBI|VBT|MD>+ <CC>* <NN|RB>* }**

Frasa VP adalah frasa yang terdiri dari kata kerja. Frasa ini akan terbentuk jika terdapat label PoS VBI|VBT|MD dengan keterangan *regular expression* "+" yang berarti harus muncul minimal 1 atau lebih dari satu item sebelumnya, kemudian bertemu label PoS <CC>* dengan kemunculan minimal 0 atau lebih item dan label PoS <NN|RB>* dengan kemunculan minimal 0 atau lebih dari item sebelumnya. Contoh frasa VP pada kalimat dan bentuk pohon parsingnya sebagai berikut.

Kalimat: menyampaikan apresiasi kepada Prodi Informatika Untan

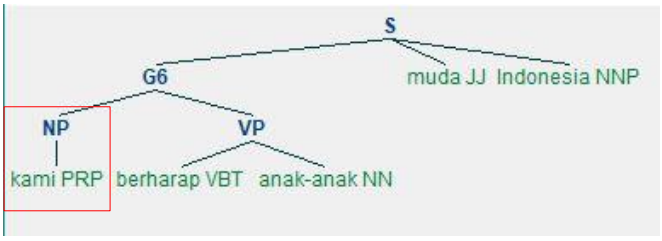


Gambar 3.14 Pohon Parsing *Rule* 4

***Rule* 5, NP : { <WP|PRP|PRN>+ }**

Frasa NP adalah frasa yang terdiri dari kata ganti orang. Frasa ini akan terbentuk jika terdapat label PoS WP|PRP|PRN dengan keterangan regular expression "+" yang berarti harus muncul minimal 1 atau lebih dari satu item sebelumnya. Contoh frasa NP pada kalimat dan bentuk pohon parsingnya sebagai berikut.

Kalimat: kami berharap anak-anak muda Indonesia



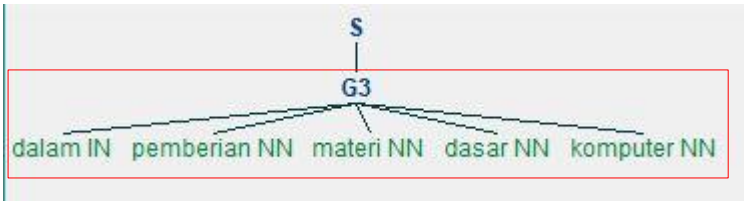
Gambar 3.15 Pohon Parsing *Rule* 5

***Rule* 6, G3 : { <RB>* <IN>+ <NN>+ <VP>*}**

Frasa G3 adalah frasa gabungan. Frasa ini akan terbentuk jika ada label PoS <RB>* dengan kemunculan minimal 0 atau lebih item, kemudian bertemu label PoS <IN>+ dengan kemunculan minimal 1 atau lebih dari satu item dan label PoS <NN>+ dengan kemunculan minimal 1 atau lebih dari satu item serta label PoS <VP>*

dengan kemunculan minimal 0 atau lebih item. Sehingga dapat dikatakan untuk membentuk frasa G3, <IN> harus bertemu dengan <NN>. Contoh frasa G3 pada kalimat dan bentuk pohon parsingnya sebagai berikut.

Kalimat: dalam pemberian materi dasar komputer



Gambar 3.16 Pohon Parsing *Rule* 6

***Rule* 7, G4 : { <NN>* <IN> <VP> <G3>* <NP>*}**

Frasa G4 adalah frasa gabungan. Frasa ini akan terbentuk jika ada label PoS <NN>* dengan kemunculan minimal 0 atau lebih item, kemudian bertemu 1 label PoS <IN> dan 1 frasa <VP> serta frasa <G3>* dengan kemunculan minimal 0 atau lebih item dan frasa <NP>* dengan kemunculan minimal 0 atau lebih item. Sehingga dapat dikatakan untuk membentuk frasa G4, <IN> harus bertemu dengan <NP>. Contoh frasa G4 pada kalimat dan bentuk pohon parsingnya sebagai berikut.

Kalimat: hal senada disampaikan oleh Kepala Sekolah SDN 7

Gambar 3.17 Pohon Parsing *Rule* 7

***Rule* 8, G5 : { <CC> <JJ>* <G3|G4|VP|G1|NP|NN|CDI|DT|RB|NEG>+ }**

Frasa G5 adalah frasa gabungan. Frasa ini akan terbentuk jika ada label PoS <CC>+ dengan kemunculan minimal 1 atau lebih dari satu item, kemudian bertemu label PoS <JJ>* dengan kemunculan minimal 0 atau lebih item dan <G3|G4|VP|G1|NP|NN|CDI|DT|RB|NEG>+ dengan kemunculan minimal 1 atau lebih dari satu item sebelumnya. Sehingga dapat dikatakan untuk membentuk frasa

G5, <CC> harus bertemu dengan <G3|G4|VP|G1|NP|NN|CDI|DT|RB|NEG>. Contoh frasa G5 pada kalimat dan bentuk pohon parsingnya sebagai berikut.

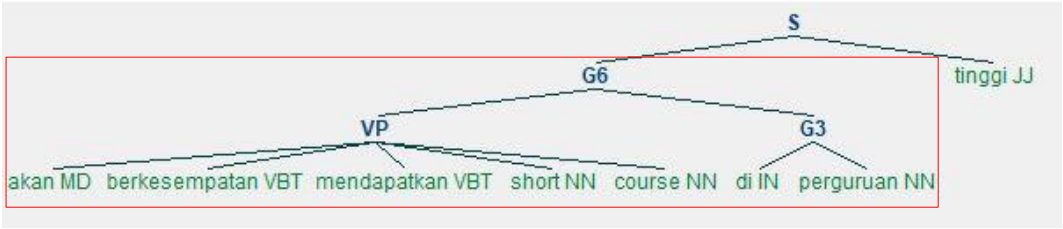
Kalimat: dan pengujian kemampuan menggunakan aplikasi presentasi

Gambar 3.18 Pohon Parsing *Rule* 8

***Rule* 9, G6 : { <RB>* <NP|VP|NEG>+ <RB>* <VP|KP|NP|DT>* <G3|G4>* }**

Frasa G6 adalah frasa gabungan. Frasa ini akan terbentuk jika ada label PoS <RB>* dengan kemunculan minimal 0 atau lebih item, kemudian bertemu <NP|VP|NEG>+ dengan kemunculan minimal 1 atau lebih dari satu item sebelumnya dan <RB>* dengan kemunculan minimal 0 atau lebih item serta <VP|KP|NP|DT>* dengan kemunculan minimal 0 atau lebih dari item sebelumnya dan <G3|G4>* dengan kemunculan minimal 0 atau lebih dari item sebelumnya. Contoh frasa G6 pada kalimat dan bentuk pohon parsingnya sebagai berikut.

Kalimat: akan berkesempatan mendapatkan short course di perguruan tinggi



Gambar 3.19 Pohon Parsing *Rule* 9

***Rule* 10, G7 : { <KP|G3>+ <G3|G4|G6|VP|NP|NN|RB>+ }**

Frasa G7 adalah frasa gabungan. Frasa ini akan terbentuk jika ada <KP|G3>+ dengan kemunculan minimal 1 atau lebih dari satu item sebelumnya dan <G3|G4|G6|VP|NP|NN|RB>+ dengan kemunculan minimal 1 atau lebih dari satu item sebelumnya. Contoh frasa G7 pada kalimat dan bentuk pohon parsingnya sebagai berikut.

Kalimat: sehingga enggan untuk melanjutkan transaksi

Gambar 3.20 Pohon Parsing *Rule* 10

3.2.6 Implementasi *Shallow Parsing*

Metode *shallow parsing* disebut juga dengan *chunking* yang merupakan metode pemenggalan/ pemotongan kalimat menjadi frasa teks dengan aturan-aturan tertentu berdasarkan kelas kata PoS *tagger*. Implementasi *shallow parsing* bertujuan mengimplementasikan *shallow parsing* dari penelitian yang sudah ada sebelumnya untuk mendapatkan frasa-frasa dalam Bahasa Indonesia dan mendapatkan hasil pemenggalan kalimat yang sesuai. Penelitian sebelumnya yaitu penelitian M. Iqbal Kamiludin (2017) yang menggunakan bahasa melayu Pontianak. Pada proses *shallow parsing*, data *input* yaitu berupa data PoS *tag* yang berisi teks yang sudah diberi label kelas kata dan *output* dari proses *shallow parsing* adalah frasa-frasa dalam Bahasa Indonesia sesuai dengan *rule grammars* yang dirancang.

Bahasa pemrograman yang digunakan untuk implementasi *shallow parsing* adalah bahasa python. *Source code shallow parsing* yang ada pada penelitian sebelumnya menggunakan python 2, kemudian diubah menjadi python 3 menyesuaikan kebutuhan pada penelitian. Pemenggalan kalimat pada proses *shallow parsing* dibantu dengan menggunakan *tools* NLTK dari python. Fungsi yang dipakai pada NLTK adalah *regexpparser*, dimana *rule grammars* diimplementasikan menggunakan *regular expression* untuk memenggal kalimat pada teks. Penampakan implementasi *rule grammars* pada *shallow parsing* dapat dilihat pada Kode program 3.1 berikut.

Kode program 3.1. Implementasi *Rule Grammars* pada *Shallow Parsing*

```
import nltk
import sys

grammar = r"""
        G1 : { <NN> <IN> <CDP> }
        G2 : { <NN> <VBT> <IN> }
```

```
        KP : { <SC|RP|UH>+ }
        VP : { <VBI|VBT|MD>+ <CC>* <NN|RB>* }
        NP : { <WP|PRP|PRN>+ }

        G3 : { <RB>* <IN>+ <NN>+ <VP>*}
        G4 : { <NN>* <IN> <VP> <G3>* <NP>*}
        G5 : { <CC> <JJ>* <G3|G4|VP|G1|NP|NN|CDI|DT|RB|NEG>+ }
        G6 : { <RB>* <NP|VP|NEG>+ <RB>* <VP|KP|NP|DT>* <G3|G4>* }
        G7 : { <KP|G3>+ <G3|G4|G6|VP|NP|NN|RB>+ }

        """

def chunk(text):
   for line in text.split("\n"):

# Tag setiap kata yang ada dari kalimat input yg dibersihkan dari
spasi di sisi kiri dan kanan
   sentence = [nltk.tag.str2tuple(t) for t in line.strip().split()]

# Melakukan parsing terhadap kalimat
   result = nltk.RegexpParser(grammar).parse(sentence)

   text_result = ""

# Memproses satu per satu hasil parsing; Jika hasil berupa tanda
baca, maka tanda baca tersebut akan ditambahkan ke output beserta
dengan tanda akhir baris ('\n'). Jika hasil bukan merupakan tanda
baca maka hasil langsung ditambahkan ke ke output beserta dangan
tanda akhir baris ('\n')
   for i, item in enumerate(result):
      if item.count(',') != 0:
         text_result += (',' + "\n")
      else:
          if item.count('?') != 0:
             text_result += ('?' + "\n")
          else:
              if item.count('!') != 0:
                 text_result += ('!' + "\n")
              else:
                  if item.count('.') != 0:
                     text_result += ('.' + "\n\n")
                  else:
                      text_result += (str(item) + "\n")
   return text_result

if __name__ == "__main__":
print(chunk(sys.argv[1]))
# print(chunk(sys.argv[1]) )
```

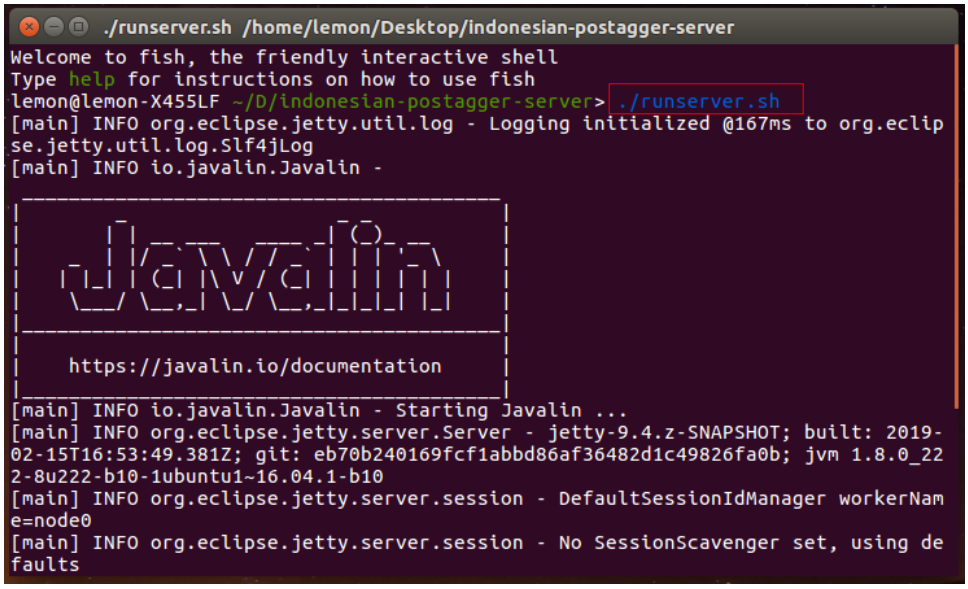
Hasil dari implementasi *rule grammars* pada *shallow parsing* diatas dapat dilihat dengan menjalankan aplikasi. Hal ini bertujuan agar *rule grammars* yang dibangun dapat sesuai dengan yang dibutuhkan. Berikut Kode program 3.2 untuk melihat hasil implementasi metode *shallow parsing*.

Kode program 3.2. *Source Code* File Process.py

```
from preprocessor import preprocess
from tagger import tag
from chunker import chunk
from normalizer import normalize

input_file = open("input.txt")
preprocessed_file = open("preprocessed.txt", "w")
tagged_file = open("tagged.txt", "w")
chunked_file = open("chunked.txt", "w")
normalized_file = open("normalized.txt", "w")

input_text = "\n".join([line for line in input_file])

# Praproses
temp = preprocess(input_text)
preprocessed_file.write(temp)

# Tagging
temp = tag(temp, "http://localhost:7000")
tagged_file.write(temp)

# Chunking
temp = chunk(temp)
chunked_file.write(temp)

# Normalisasi
temp = normalize(temp)
normalized_file.write(temp)
```

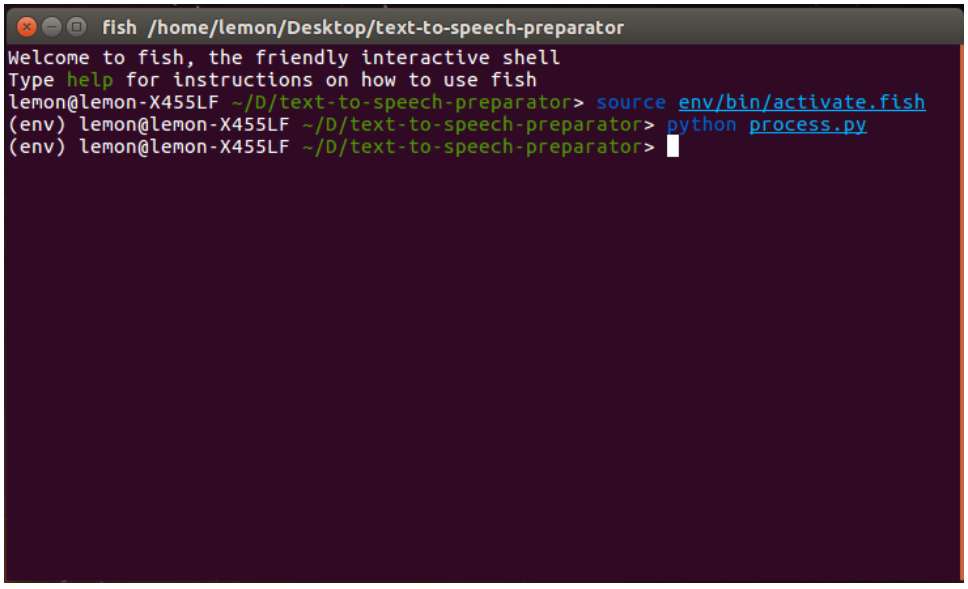
Output dari praproses tersimpan dalam file preprocessor.txt, *output* dari proses PoS *tagging* tersimpan dalam file tagged.txt, *output* dari proses *shallow parsing* tersimpan dalam file chunked.txt dan *output* dari proses normalisasi tersimpan dalam file normalized.txt.

Sebelum menjalankan file process.py untuk proses pemenggalan kalimat, terlebih dulu aktifkan java dengan cara klik kanan pada folder indonesian-postagger-server dan pilih buka di terminal, lalu ketikkan kode seperti pada Gambar 3.21 berikut.



Gambar 3.21 Aktivasi Java Pada Linux Ubuntu

Kemudian masukkan teks uji ke dalam input.txt. Jalankan file process.py pada folder text-to-speech-preparator dengan cara klik kanan folder dan pilih buka di terminal, lalu ketikkan kode seperti pada Gambar 3.22 berikut.



Gambar 3.22 *Running* File Process.py pada Folder Pemenggalan Kalimat

3.2.7 Implementasi *Text To Speech*

Implementasi *text to speech* merupakan proses mengimplementasikan sistem *text to speech* kedalam website. Implementasi dilakukan berdasarkan kepada desain dan perancangan sistem yang telah dilakukan pada tahap sebelumnya. Teks pada website diambil dan dikirim ke server pemenggalan kalimat, kemudian teks

terpenggal dari server akan dikirim kembali ke website dan dibunyikan dengan menggunakan *tools* dari *responsive voice* untuk membangkitkan teks menjadi ucapan. Pada tahap ini, sistem *text to speech* pada website dibangun dengan menggunakan bahasa pemrograman JavaScript, PHP dan HTML. Website yang digunakan untuk melakukan uji coba implementasi *text to speech* dibangun menggunakan PHP native dan framework CSS.

3.2.8 Pengujian Sistem

Pengujian sistem adalah sebuah tahap mengevaluasi sistem yang dibangun. Terdapat dua jenis objek penilaian yaitu kinerja sistem dan kualitas *rule grammars*. Pengujian terhadap kinerja sistem akan menggunakan pengujian *black box* sedangkan pengujian terhadap kualitas *rule grammars* akan menggunakan pengujian akurasi, pengujian *precision, recall* dan *f-measure* dan pengujian subjektif.

3.2.8.1 Pengujian *Black Box*

Pengujian *black box* merupakan pengujian fungsionalitas sistem. Pengujian *black box* bertujuan untuk memeriksa sistem dapat berjalan dan menjalankan fungsi-fungsi sistem dengan benar yakni input dapat diterima dengan benar dan output yang dihasilkan telah sesuai dengan yang diinginkan. Sistem menyediakan tiga tombol pada halaman detail berita website untuk mengatur jalannya sistem. Tiga tombol diantaranya adalah *play*, *pause-resume* dan *stop*. Pengujian fungsionalitas sistem dimulai dari pengambilan dan pengiriman teks berita ke server, pengolahan teks menjadi teks terpenggal dalam server, mensintesa teks terpenggal menjadi ucapan, dapat menghentikan sementara dan dapat melanjutkan kembali ucapan/ suara serta dapat menghentikan sepenuhnya ucapan/ suara yang sedang diputar.

Bentuk tabel pengujian *black box* dapat dilihat pada Tabel 3.4. Tabel pengujian *black box* terdiri dari tiga kolom yaitu skenario pengujian, hasil yang diharapkan dan kesimpulan. Skenario pengujian berisi deskripsi dari bentuk masukan (*input*). Hasil yang diharapkan berupa deskripsi mengenai hasil keluaran (*output*) yang diharapkan dari sistem. Bagian kesimpulan terdiri dari dua keluaran

(*output*) yaitu valid atau tidak valid. Hasil valid didapatkan ketika hasil yang diharapkan sesuai dengan hasil keluaran (*output*), sedangkan jika tidak sesuai maka hasil yang didapatkan adalah tidak valid.

Tabel 3.4 Pengujian *Black Box*

No.	Skenario Pengujian	Hasil yang diharapkan	Kesimpulan
1	Meng-klik salah satu jenis berita dan masuk ke halaman detail berita	Deskripsi hasil yang diharapkan	Validasi
2	Tombol *play* di-klik	Deskripsi hasil yang diharapkan	Validasi
3	Tombol *pause* di-klik	Deskripsi hasil yang diharapkan	Validasi
4	Tombol *resume* di-klik	Deskripsi hasil yang diharapkan	Validasi
5	Tombol *stop* di-klik	Deskripsi hasil yang diharapkan	Validasi

3.2.8.2 Pengujian Akurasi

Pengujian akurasi merupakan sebuah pengujian yang dilakukan untuk menguji tingkat akurasi *rule grammars* yang dikembangkan. Pengujian akurasi dilakukan dengan mencocokkan kalimat antara *rule* yang dikembangkan peneliti dengan data jeda keras yang didapat dari narasumber. Pengujian bertujuan untuk mendapatkan *rule* terbaik yang digunakan untuk proses penjedaan. Pengujian berdasarkan pada kalimat dengan jeda keras dan kalimat hasil proses pemenggalan menggunakan *rule*. Narasumber berjumlah 3 orang dan *rule* yang dirancang sebanyak 3 opsi *rule*. Data dari narasumber yaitu berupa teks berita yang telah terpenggal yang kemudian disatukan untuk mendapatkan jeda keras pada teks berita. Jeda keras didapat dari proses pencocokkan letak jeda antara narasumber pertama, kedua dan ketiga. Jeda keras pada pengujian adalah jeda keras dari kesepakatan tiga narasumber. Jika terdapat jeda keras pada kalimat teks berita yaitu jeda keras dari tiga narasumber, maka kalimat tersebut adalah data kalimat dengan jeda keras dari

tiga orang yang digunakan untuk proses pencocokkan *rule grammars* pada pengujian akurasi. Berikut Tabel 3.5 untuk mendapatkan jumlah kalimat yang memiliki jeda keras dari tiga orang.

Tabel 3.5 Skenario dalam Penentuan Jumlah Kalimat dengan Jeda Keras dari 3 Narasumber

Kalimat ke-	**Orang ke-**			**Kesimpulan**
	1	**2**	**3**	
1	✓	✓	✓	Masuk
2	✓	✓	X	Tidak
3	X	X	X	Tidak
Total Kalimat				1

Total pada Tabel 3.5 menunjukkan jumlah kalimat dengan jeda keras dari 3 orang. Maksud dari total kalimat = 1 dari kolom kesimpulan adalah terdapat sebanyak 1 kalimat yang memiliki jeda keras dari kesepakatan tiga orang narasumber. Setelah mendapatkan kalimat-kalimat dengan jeda keras tiga orang, data kalimat tersebut akan diseleksi terlebih dahulu. Proses seleksi yang dilakukan yakni menyaring kalimat jeda keras, mana kalimat yang memiliki jumlah karakter ≤99 karakter dan jumlah karakter >99 karakter pada kalimat tunggalnya. Kalimat dengan jeda keras dari 3 orang yang lulus dengan jumlah karakter kalimat tunggal ≤99 karakter akan masuk perhitungan akurasi.

Data kalimat dengan jeda keras 3 orang masing-masing akan dicocokkan dengan teks terpenggal hasil dari proses *chunking rule* pertama, kedua dan ketiga untuk mendapatkan satu *rule* yang memiliki persentase tingkat kecocokkan paling baik. Setelah proses pencocokkan selesai, maka dilakukan perhitungan untuk menghitung persentase tingkat akurasi *rule*. Perhitungan persentase tingkat akurasi *rule grammars* dapat dilihat pada Persamaan (3.1) berikut.

$$\text{Tingkat akurasi } \textit{rule} \text{ ke-n (\%)} = \frac{Jumlah\ kalimat\ yang\ cocok}{Jumlah\ kalimat\ keseluruhan} \times 100 \qquad (3.1)$$

Skenario pada pengujian akurasi untuk jeda keras 3 orang dapat dilihat pada Tabel 3.6 sebagai berikut.

Tabel 3.6 Skenario Perhitungan pada Pengujian Akurasi

No.	Kalimat ke-	*Rule Grammars ke-*		
		1	**2**	**3**
1.	1	✓	✓	✓
2.	2	✓	✓	X
3.	3	X	✓	X
Total		2	3	1
Persentase (%)		67	100	33

3.2.8.3 Pengujian *Precision, Recall* dan *F-Measure*

Pengujian *precision* dan *recall* merupakan pengujian untuk mendapatkan informasi hasil pencarian dokumen yang relevan dengan dokumen asli yang ingin dibandingkan. *Precision* dapat dianggap sebagai ukuran ketepatan atau ketelitian, sedangkan *recall* adalah pengulangan kejadian yang sama dengan dokumen asli. Nilai F dapat diartikan sebagai sebuah hasil rata-rata dari *precision* & *recall*, dimana sebuah nilai F mendapat hasil 1 sebagai hasil terbaik dan 0 sebagai nilai terburuk. Dokumen asli yang digunakan adalah kalimat yang telah diberikan jeda keras oleh narasumber. Dokumen yang diuji merupakan teks berita sebanyak 42 kalimat berdasarkan jeda keras dari kesepakatan 3 orang. Rancangan pengujian *precision* dan *recall* dapat dilihat pada Tabel 3.7 berikut.

Tabel 3.7 Skenario Pengujian *Precision, Recall* dan *F-Measure* Terhadap Data Kalimat

No.	Kalimat Uji	Relevan		Tidak Relevan
		Terambil (a)	**Tidak Terambil (b)**	**Terambil (c)**
1.	1	…	…	…
2.	2	…	…	…
dst.	…	…	…	…
Total		a	b	c
Precision		$\frac{a}{a+c}$		

Recall	$\frac{a}{a+b}$
F-Measure	$2 \cdot \frac{precision \cdot recall}{precision + recall}$

Keterangan:

a = frasa jeda yang relevan ditemukan

b = frasa jeda yang relevan tidak ditemukan

c = frasa jeda yang tidak relevan ditemukan

Berdasarkan Tabel 3.7 diatas, untuk menghitung rasio *precision* yakni jumlah dokumen relevan yang terambil (a) dibagi dengan jumlah dokumen yang terambil dalam mesin (a+c). Sedangkan untuk menghitung rasio *recall* yakni jumlah dokumen relevan yang terambil (a) dibagi dengan jumlah dokumen relevan dalam dokumen asli (a+b). Rumus *precision*, *recall* dan *f-measure* dapat dinyatakan seperti pada Persamaan 3.2, 3.3 dan 3.4 berikut.

$$Precision = \frac{a}{a+c} \quad (3.2)$$

$$Recall = \frac{a}{a+b} \quad (3.3)$$

$$F\text{-}Measure = 2 \cdot \frac{precision \cdot recall}{precision + recall} \quad (3.4)$$

Berikut dua hal yang digunakan sebagai acuan dalam mengukur keefektifan dari sistem yang dibangun yaitu ketepatan (*precision*) dan perolehan (*recall*). Keefektifan suatu sistem temu kembali informasi dinilai berdasarkan teori yang dicetuskan oleh Lancaster (1991) dalam Pendit (2008) yaitu relevan dan tidak relevan, jadi efektifitas temu kembali informasi dibedakan menjadi efektif jika nilai diatas 50% dan tidak efektif jika nilai dibawah nilai 50%. Selain itu, suatu sistem temu kembali dinyatakan efektif apabila hasil penelusuran mampu menunjukkan ketepatan (*precision*) yang tinggi sekalipun perolehannya (*recall*) rendah (Rowley 1992) dalam Hasugian (2006).

3.2.8.4 Pengujian Subjektif

Pengujian subjektif merupakan sebuah pengujian yang dilakukan secara langsung kepada pendengar untuk mendengar dan menilai kualitas ucapan dari suatu kalimat yang dibunyikan menggunakan aplikasi *responsive voice*. Uji dengar

pada pengujian subjektif bertujuan untuk menyatakan frasa ucapan suatu kalimat dapat diterima atau tidak dapat diterima oleh pendengar. Hal ini dimaksudkan untuk mendapatkan *rule grammars* terbaik yang digunakan untuk proses penjedaan kalimat. Pengujian subjektif dilakukan dengan memberikan lembaran kertas berupa *form* kepada pendengar yang isinya berupa tabel yang kemudian akan diisi oleh pendengar. Pendengar mengisi *form* dengan mengikuti instruksi yang diberikan. Pendengar akan mendengarkan suara dari *responsive voice* dengan kalimat inputan berupa kalimat asli (kalimat yang bukan hasil pemenggalan) dan kalimat hasil pemenggalan menggunakan *rule grammars*. Jumlah pendengar pada pengujian sebanyak 39 orang. Kalimat yang akan diuji sebanyak 25 kalimat berita. *Rule chunking* yang digunakan untuk dilakukan perbandingan adalah *rule grammars* 1 dan *rule grammars* 2. Ucapan yang akan didengar oleh pendengar adalah sebanyak 75 ucapan berupa suara diantaranya 25 ucapan dari 25 kalimat berita asli, 25 ucapan dari 25 kalimat *chunking* menggunakan *rule grammars* 1 dan 25 ucapan dari 25 kalimat *chunking* menggunakan *rule grammars* 2. Pengujian dibagi menjadi 2 sesi, sesi pertama adalah membandingkan suara dari kalimat asli dan kalimat *chunking* dari *rule grammars* 1, sedangkan sesi kedua adalah membandingkan suara dari kalimat asli dan kalimat *chunking* dari *rule grammars* 2. Skenario pada *form* yang diberikan kepada pendengar dapat dilihat pada Tabel 3.8 berikut.

Tabel 3.8 Skenario pada Lembaran Isian Pendengar Sesi ke-n

No.	Suara 1	Suara 2
1.	✓ / X	✓ / X
2.	✓ / X	✓ / X
dst.	✓ / X	✓ / X

Terdapat tanda centang dan tanda silang pada kolom suara 1 dan suara 2. Hal ini dimaksudkan jika penggalan frasa pada kalimat yang diucapkan dapat diterima maka pendengar dapat memberi tanda centang pada kolom, sedangkan jika penggalan frasa pada kalimat yang diucapkan tidak dapat diterima maka pendengar dapat memberi tanda silang pada kolom. Kolom suara 1 dan suara 2 diisi dengan suara kalimat asli dan suara kalimat *chunking* dengan posisi suara diacak kiri kanannya namun tetap menyesuaikan urutan per kalimat, sehingga tidak dapat

dipastikan isi dari kolom 1 adalah suara dari kalimat asli atau suara dari kalimat *chunking*. Hal ini bertujuan untuk mendapatkan konsistensi isian dari pendengar dalam mengisi *form*, apabila isian pada ucapan kalimat asli di sesi 1 dan sesi 2 yakni sama, maka kalimat tersebut akan masuk ke proses perhitungan. Kemudian data yang didapat dari pendengar akan dihitung untuk mendapatkan persentase ucapan diterima (per kalimat) dengan rumus seperti pada Persamaan 3.5 sebagai berikut.

$$\text{Persentase ucapan diterima} = \frac{\sum ucapan\ diterima}{\sum ucapan\ diterima + \sum ucapan\ tidak\ diterima} \times 100 \quad (3.5)$$

Setelah mendapatkan persentase ucapan diterima dari tiap kalimat, kemudian dihitung rata-rata persentase dari seluruh kalimat dengan rumus seperti pada Persamaan 3.6 sebagai berikut.

$$\text{Rata-rata ucapan diterima} = \frac{\sum nilai\ persentase\ ucapan\ diterima\ dari\ seluruh\ kalimat}{\sum kalimat\ keseluruhan} \quad (3.6)$$

Berikut Tabel 3.9 untuk skenario perhitungan persentase pada pengujian subjektif.

Tabel 3.9 Skenario Perhitungan Persentase Ucapan Diterima *Rule Grammars* ke-n

Kalimat ke-	**Jumlah Konsistensi (Orang)**	**Jumlah Ucapan**		**Persentase Diterima**
		Diterima	**Tidak Diterima**	
1	a+b	a	b	$c = \frac{a}{a+b}$
2	a+b	a	b	$c = \frac{a}{a+b}$
3	a+b	a	b	$c = \frac{a}{a+b}$
Rata-Rata				$\frac{c}{total\ kalimat}$

3.2.9 Analisis Hasil Pengujian

Pada tahap ini, hasil pengujian sistem akan dianalisis secara keseluruhan untuk mempermudah penarikan kesimpulan.

3.2.10 Penarikan Kesimpulan

Kesimpulan dirumuskan berdasarkan tahapan-tahapan yang telah dilakukan sebelumnya apakah sistem yang dirancang dan dibangun dapat berjalan baik sesuai dengan yang diharapkan.

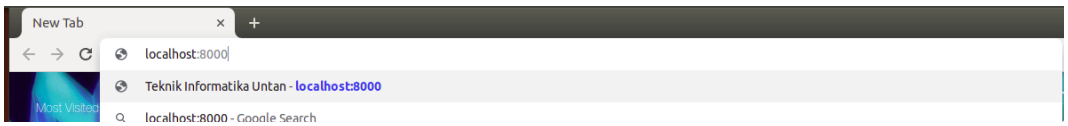
BAB IV
IMPLEMENTASI DAN HASIL PENGUJIAN

4.1 Implementasi

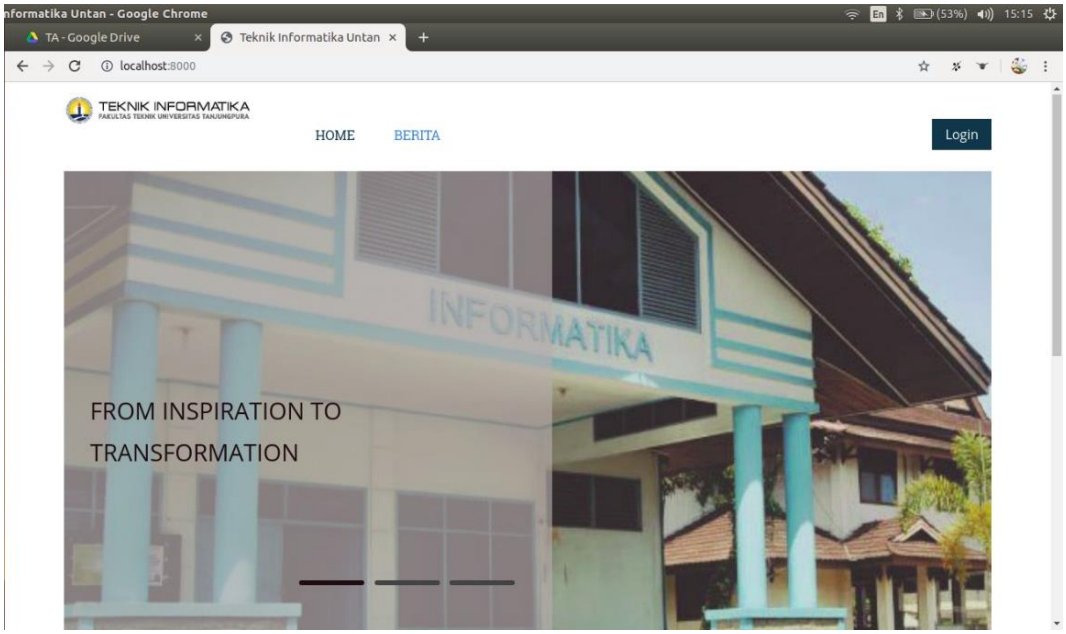
Sistem yang dirancang merupakan "Sistem implementasi *text to speech* pada website dengan metode *shallow parsing*" yang bertujuan untuk mengimplementasi text to speech pada halaman berita website dari masukan berupa teks berita dan keluaran berupa suara dalam Bahasa Indonesia. Sistem yang dirancang terdiri dari tiga subsistem diantaranya sistem pada website, pemenggalan kalimat dan sintesa ucapan. Sistem pemenggalan kalimat disimpan pada server melalui alamat http://203.24.50.138:8001/ . Berikut antarmuka pengguna pada website, tampilan file pemenggalan kalimat yang diterapkan ke server dan tampilan hasil dari sintesa ucapan pada website. Adapun hasil implementasinya sebagai berikut:

4.1.1 Antarmuka Pengguna Pada Website

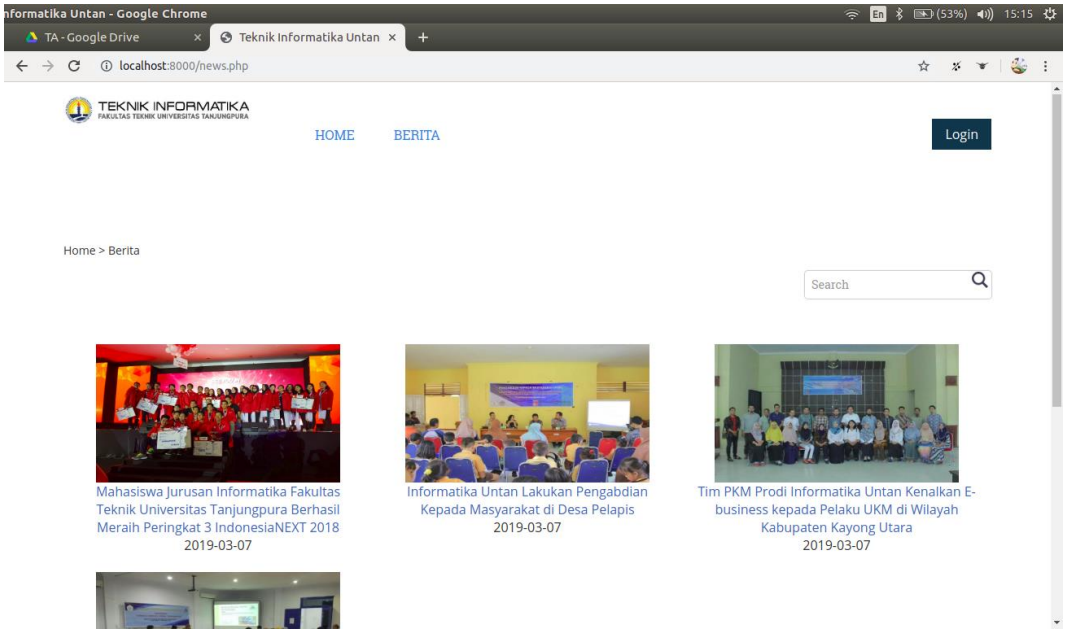
Antarmuka pengguna pada website merupakan sebuah sarana bagi pengguna untuk menggunakan beberapa fitur atau fungsi yang ditawarkan oleh sistem. Antarmuka yang dibuat berupa ikon yang diterapkan di halaman detail berita pada website. Ikon yang dibuat untuk antarmuka pengguna adalah sebanyak 3 ikon yaitu *play*, *pause-resume* dan *stop*. Ikon *play* berfungsi untuk mengirim teks ke server pemenggalan kalimat dan mensintesa teks terpenggal menjadi suara dengan bantuan dari *responsive voice*. Ikon *pause-resume* berfungsi untuk menghentikan sementara dan memainkan kembali suara yang keluar. Ikon *stop* berfungsi untuk menghentikan suara sepenuhnya. Tampilan antarmuka pengguna dapat dilihat pada Gambar 4.2, 4.3, 4.4, 4.5 dan 4.6. Masuk ke dalam website dapat diakses dengan mengetikkan linknya seperti pada Gambar 4.1 sebagai berikut.



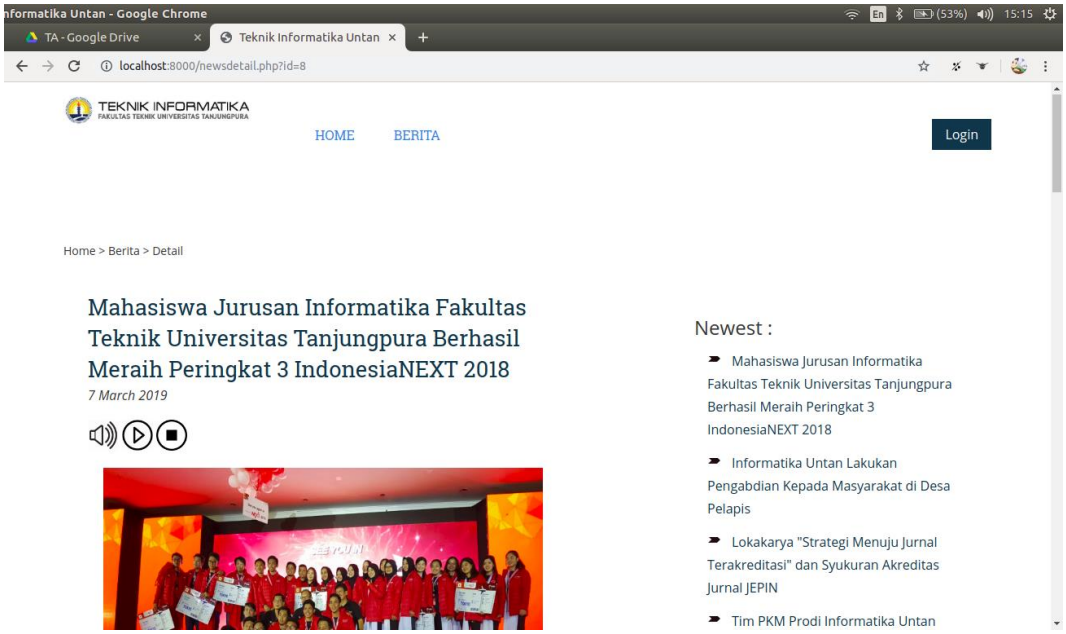
Gambar 4.1 Link Masuk ke Halaman Website



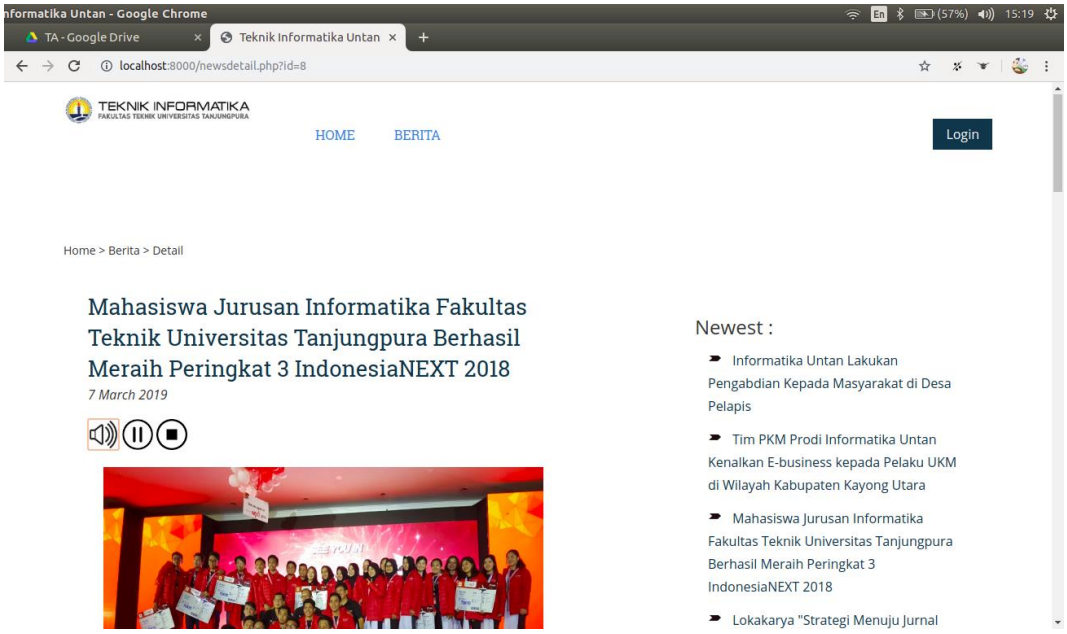
Gambar 4.2 Halaman Beranda Website



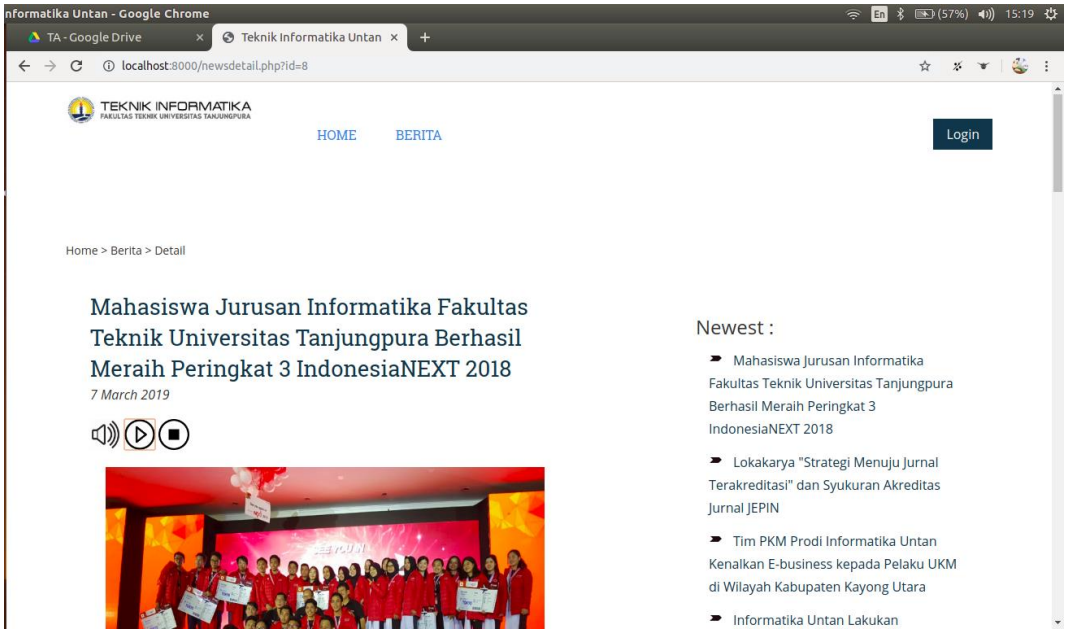
Gambar 4.3 Halaman Berita Website



Gambar 4.4 Antarmuka Pengguna pada Halaman Detail Berita Website

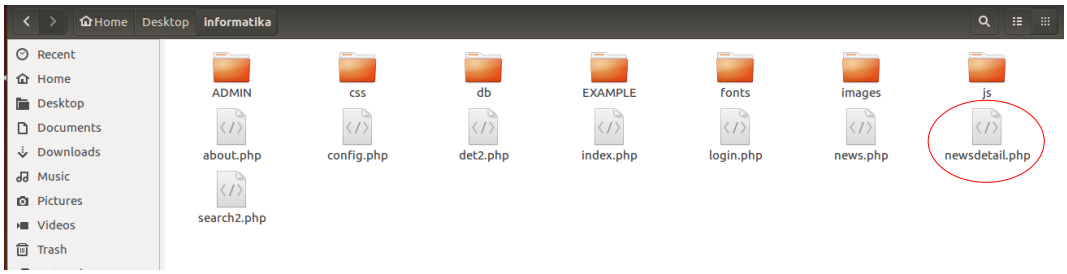


Gambar 4.5 Tampilan Antarmuka ketika Tombol *Play* Di-Klik



Gambar 4.6 Tampilan Antarmuka ketika Tombol *Pause* Di-Klik

Proses implementasi *text to speech* dilakukan dalam halaman detail berita web pada file newsdetail.php seperti pada Gambar 4.7 sebagai berikut.



Gambar 4.7 Implementasi *Text To Speech* pada File newsdetail.php

Beberapa *source code* yang perlu disesuaikan dan ditambahakan untuk menampilkan judul dan isi teks berita dalam *database*, menampilkan ikon tombol dan beberapa *script* yang perlu ditambahkan untuk implementasi *text to speech* ke dalam halaman detail berita pada website seperti pada Kode program 4.1, 4.2, 4.3, 4.4 dan 4.5 berikut.

Kode Program 4.1. Memperhatikan *Source Code* Judul Teks Berita dalam *Database*

```
<h8 id="title">
    <?php echo $data['judul']; ?>
</h8>
```

Kode Program 4.2. Memperhatikan *Source Code* Isi Teks Berita dalam *Database*

```
<p>
    <article class="isi">
        <?php echo $data['isi']; ?>
    </article>
</p>
```

Kode Program 4.3. Koneksi ke Server *Responsive Voice*

```
<head>

<script src='https://code.responsivevoice.org/responsivevoice.js'>
</script>

</head>
```

Kode Program 4.4. Menampilkan Ikon Tombol *Play*, *Pause-Resume* dan *Stop*

```
< body>

<input    id="play_button"    type="image"    src="images/play.png"
alt="Submit" width="38" height="38">
<input            id="pause_resume_button"            type="image"
src="images/pause.png" alt="Submit" width="38" height="38">
<input    id="stop_button"    type="image"    src="images/stop.png"
alt="Submit" width="38" height="38">

</body>
```

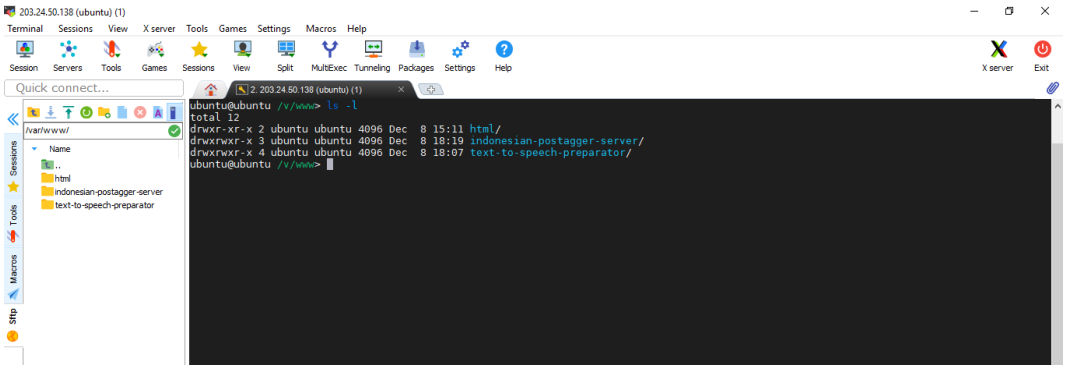
Kode Program 4.5. Mengirimkan Teks ke Server dan Membunyikan Teks

```
<body>
    <script>
        var state = "stopped"
        var text_list = []
        var pause_resume_button = $("#pause_resume_button");
        var play_button = $("#play_button");
        var stop_button = $("#stop_button");
        var pause_image = "images/pause.png";
        var resume_image = "images/resume.png";
        /* URL service / web pemroses teks */
        var service_url = "http://203.24.50.138:8001/text-to-
        speech-preparator";

        /* Mengirimkan teks artikel ke server dan membunyikan teks
        artikel */
        function textToSpeech(text_list, index) {
            $.post(service_url, {
                /* Data teks yang hendak dikirim */
                text: text_list[index]
            }, function({
                /* Data teks yang dikembalikan oleh server pemroses
                */
                data
            }) {
                responsiveVoice.speak(
```

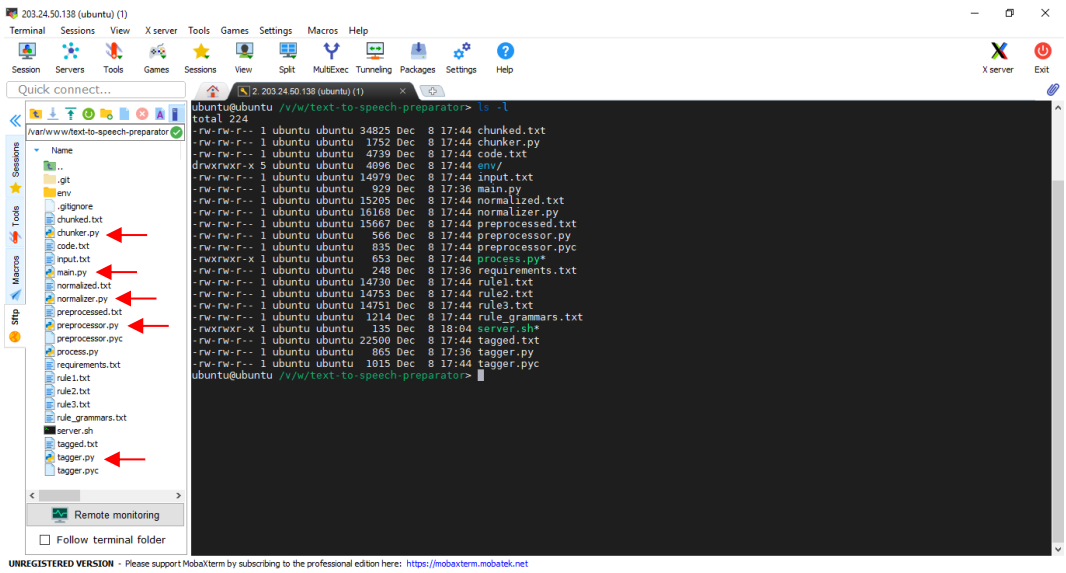
```
                data.text,
                "Indonesian Female", {
                    onend: function() {
                        if (index + 1 < text_list.length) {
                            textToSpeech(text_list, index + 1)
                        }
                    }
                }
            )
        })
    }

    $(document).ready(function() {
        $(".isi p").each(function(index, elem) {
            text_list.push($(elem).text())
        })

        pause_resume_button.attr("src", resume_image)

        /* Mengirimkan teks judul ke server dan membunyikan
        teks judul */
        $.post(service_url, {
            text: $("#title").text().trim()
        }, function({
            data
        }) {
            console.log("TEST")
            responsiveVoice.speak(
                data.text, "Indonesian Female"
            )
        })

        play_button.click(function() {
            textToSpeech(text_list, 0)
            state = "playing"
            pause_resume_button.attr("src", pause_image)
        });

        stop_button.click(function() {
            responsiveVoice.cancel()
            state = "stopped"
        })

        /* Memberhentikan / memulai suara sesuai dengan situasi
        */
        pause_resume_button.click(function() {
            if (state == "playing") {
                state = "paused"
                pause_resume_button.attr("src", resume_image)
                responsiveVoice.pause()
            } else if (state == "paused") {
                state = "playing"
                pause_resume_button.attr("src", pause_image)
                responsiveVoice.resume()
            }
        })
    })
</script>
</body>
```

4.1.2 Tampilan File Pemenggal Kalimat Pada Server

Pemenggalan kalimat pada server merupakan sebuah sistem pemenggalan kalimat yang program aplikasinya diupload dan disimpan di dalam server. Aplikasi pada server ini akan dihubungkan dan digunakan untuk proses *text to speech* pada website. Proses pemenggalan kalimat pada server dimulai dari menerima teks dari website, kemudian masuk praproses, PoS *tagger*, *shallow parsing* dan normalisasi. Hasil normalisasi berupa teks terpenggal yang akan dikirimkan ke website kembali untuk dibunyikan menggunakan *responsive voice*. Tampilan file pemenggal kalimat pada server dapat dilihat seperti Gambar 4.8 dan 4.9 sebagai berikut.



Gambar 4.8 Tampilan File Program Pemenggal Kalimat dalam Server



Gambar 4.9 File Program Proses *Shallow Parsing* pada Server

Panah merah pada Gambar 4.9 menunjukkan file-file yang dibutuhkan dalam pemenggalan kalimat. File preprocessor.py merupakan file program untuk melakukan proses praproses pada teks website. File tagger.py merupakan file

program yang terhubung ke server PoS *tagger* untuk menjalankan proses PoS *tagging*. File chunker.py merupakan file program untuk proses *shallow parsing*. File normalizer.py merupakan file program untuk proses normalisasi. Dan file main.py merupakan file program yang menghubungkan dan menjalankan semua proses yang diperlukan untuk proses pemenggalan kalimat dalam server. Berikut Kode program 4.6 yang ada pada file main.py.

Kode Program 4.6. *Source Code* pada File Program main.py

```
from flask import Flask, jsonify, request
from preprocessor import preprocess
from tagger import tag
from chunker import chunk
from normalizer import normalize
from flask_cors import CORS
import waitress

app = Flask(__name__)
CORS(app)

@app.route("/", methods=["POST"])
def process():
    # Mengakses data form dari request HTTP
    text = request.form.get("text", "")

    # Melakukan preprocessing
    text = preprocess(text)

    # Melakukan tagging
    text = tag(text, "http://localhost:7000")

    # Melakukan chunking
    text = chunk(text)

    # Melakukan proses normalisasi
    text = normalize(text)


    # Membuat response HTTP dengan format JSON yang berisi teks
    yang telah diproses
    return jsonify({
        "status": "success",
        "message": "Request successful",
        "data": {
            "text": text
        }
    })
# Menjalankan server
waitress.serve(app, host="0.0.0.0", port=8070)
```

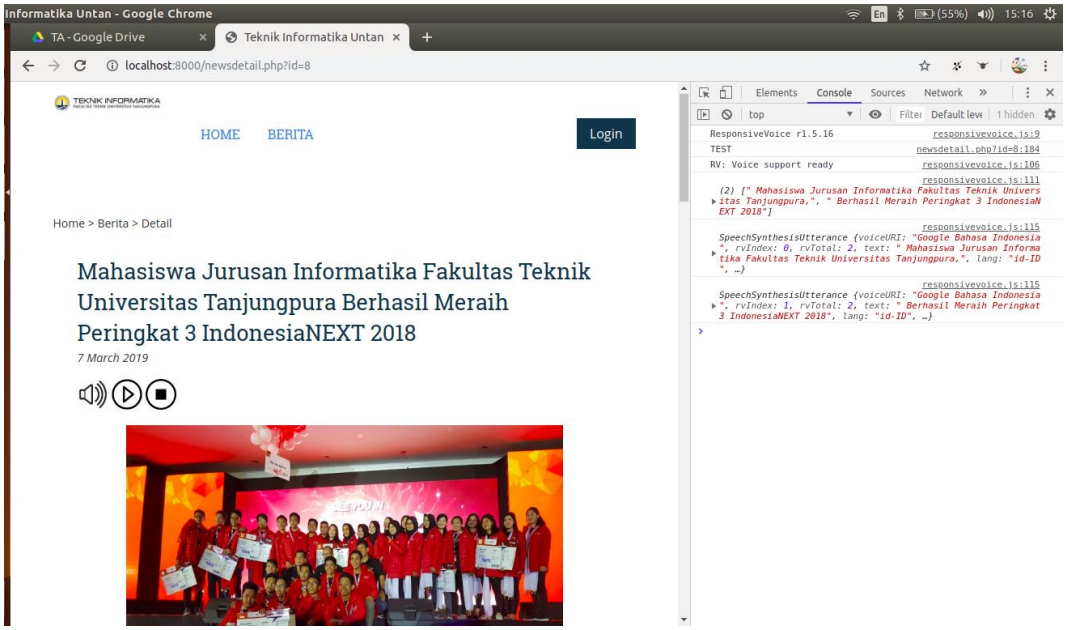
Pada folder program pemenggalan kalimat dalam server, terdapat file *rule*_grammars.txt yang didalamnya disediakan 2 model *rule grammars*. Berikut Kode program 4.7 berisi dua model *rule grammars* dalam file *rule*_grammars.txt.

Kode Program 4.7. Dua Model *Rule Grammars* pada File *rule*_grammars.txt

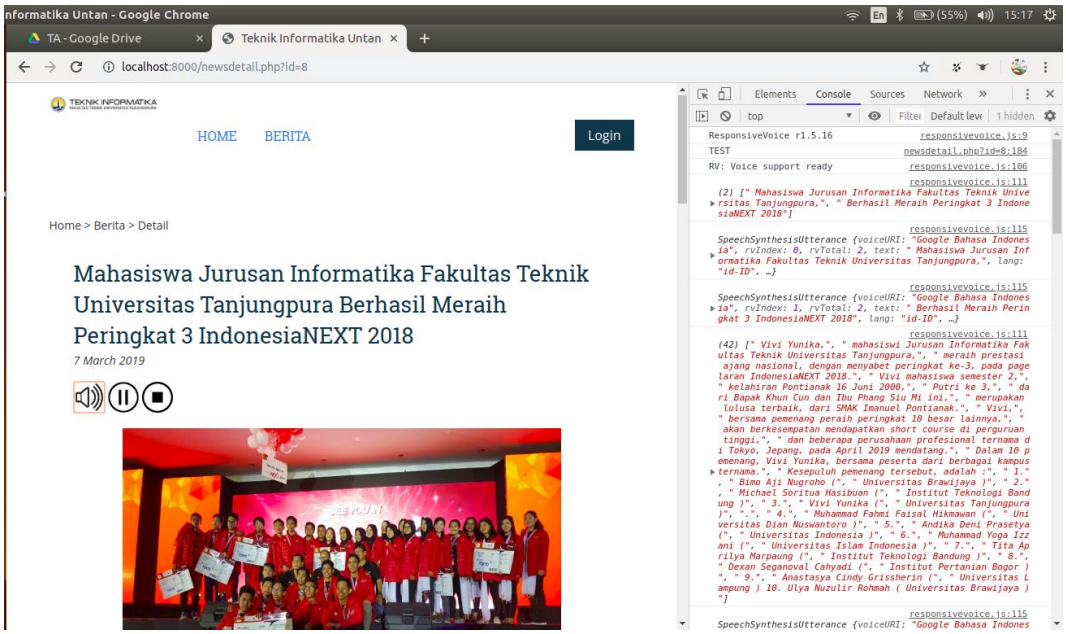
```
Rule Grammars 1:
G1 : { <NN> <IN> <CDP> }
G2 : { <NN> <VBT> <IN> }

KP : { <SC|RP|UH>+ }
VP : { <VBI|VBT|MD>+ <NN|RB>*}
NP : { <WP|PRP|PRN>+ }

G3 : { <IN> <NN>* <VP>+ }
G4 : { <KP>+ <NP|VP|KP|G3|NN|PRL>+ }
G5 : { <NP|CC|RB|NEG|MD>+ <VP|G1|NP|G3|CDI|DT|PRL>+ }
G6 : { <VP> <G3> }

Rule Grammars 2:
G1 : { <NN> <IN> <CDP> }
G2 : { <NN> <VBT> <IN> }

KP : { <SC|RP|UH>+ }
VP : { <VBI|VBT|MD>+ <CC>* <NN|RB>* }
NP : { <WP|PRP|PRN>+ }

G3 : { <RB>* <IN>+ <NN>+ <VP>*}
G4 : { <NN>* <IN> <VP> <G3>* <NP>*}
G5 : { <CC> <JJ>* <G3|G4|VP|G1|NP|NN|CDI|DT|RB|NEG>+ }
G6 : { <RB>* <NP|VP|NEG>+ <RB>* <VP|KP|NP|DT>* <G3|G4>* }
G7 : { <KP|G3>+ <G3|G4|G6|VP|NP|NN|RB>+ }
```

4.1.3 Tampilan Hasil Sintesa Pada Website

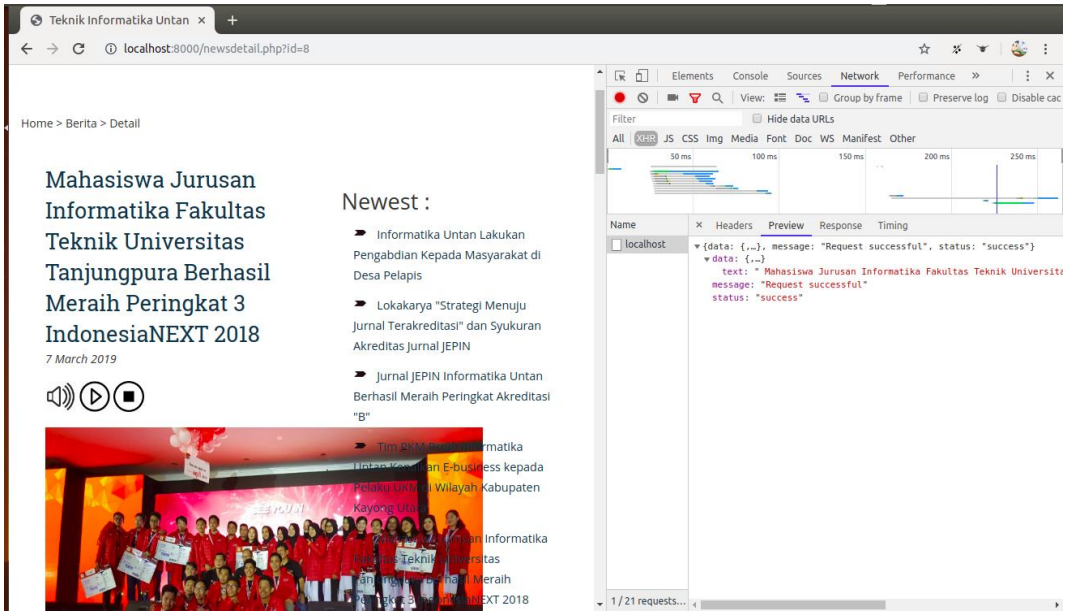
Sintesa pada website merupakan proses sintesa dari teks berita menjadi ucapan pada website tepatnya pada halaman detail berita. Teks terpenggal dari server yang diterima akan dikirim ke *responsive voice* dan mendapatkan *output* berupa ucapan dalam Bahasa Indonesia sesuai yang dibutuhkan. Terdapat dua jenis teks berita yang dibunyikan yaitu teks dari judul berita yang diputar otomatis ketika *user* membuka halaman detail berita pada website dan teks dari isi berita yang dibunyikan ketika *user* mengklik ikon *play* pada halaman detail berita. Berikut tampilan ketika website sudah terhubung dengan *responsive voice* dan proses *text to speech* sedang berjalan, yang dapat dilihat dengan membuka *inspect element* pada bagian *console* seperti pada Gambar 4.10, 4.11, 4.12 dan 4.13 sebagai berikut.



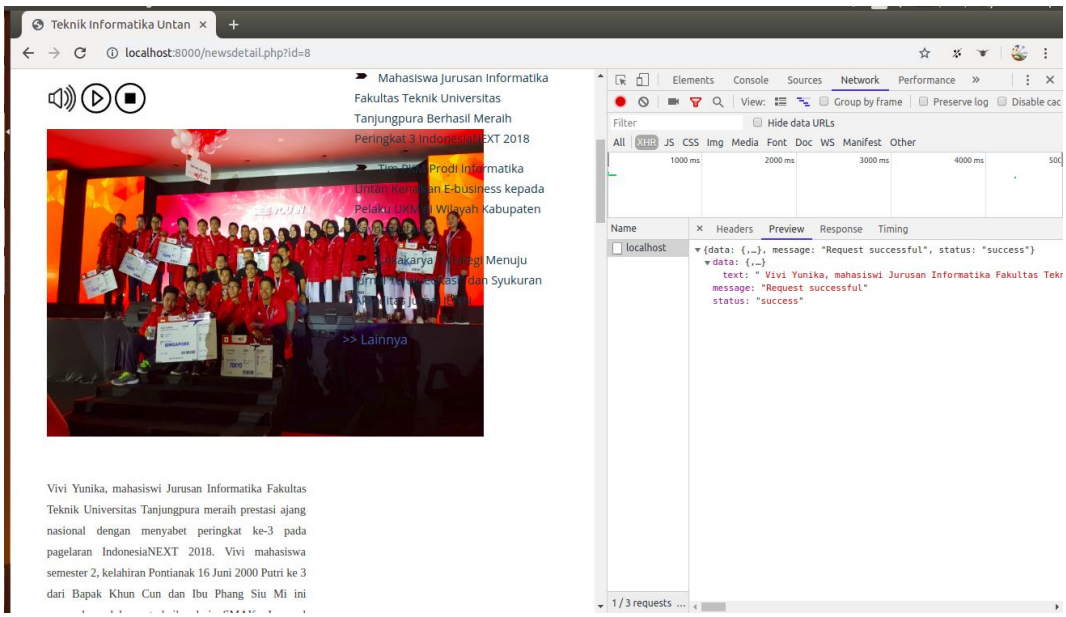
Gambar 4.10 Tampilan Sintesa Ucapan Teks Judul Berita dengan *Responsive Voice* pada *Console Inspect Element*



Gambar 4.11 Tampilan Sintesa Ucapan Teks Isi Berita dengan *Responsive Voice* pada *Console Inspect Element*



Gambar 4.12 Hasil Pemenggalan Kalimat pada Judul Berita yang Dikirim dari Server ke Website



Gambar 4.13 Hasil Penggalan Kalimat pada Isi Berita yang Dikirim dari Server ke Website

4.2 Tahap Pengujian

Pengujian yang dilakukan terdiri dari tiga jenis yaitu pengujian *black box*, pengujian akurasi dan pengujian *precision*, *recall* dan *f-measure*. Narasumber berasal dari mahasiswa Teknik UNTAN dengan jurusan Informatika yang telah

menyelesaikan masa studi. Total narasumber untuk pengujian berjumlah 3 orang mahasiswa yakni 2 orang mahasiswa yang telah melakukan penelitian mengenai PBA (Pemrosesan Bahasa Alami) khususnya pada prediksi jeda dan 1 orang mahasiswa biasa. Daftar narasumber dapat dilihat pada lampiran N. Total kalimat yang diuji sebanyak 5 topik berita dengan jumlah kalimat sebanyak 74 kalimat dan total kata sebanyak 1912 kata dan jumlah kata tiap kalimat rata-rata sebanyak 26 kata.

Sistem yang dibangun terdiri dari dua bagian secara umum yaitu *front-end* dan *back-end*. Bagian *front-end* terdiri dari proses pengiriman teks dari website ke sistem pemenggalan kalimat dan mensintesa teks menjadi ucapan. Bagian *back-end* terdiri dari proses pengolahan teks menjadi teks terpenggal pada sistem pemenggalan kalimat dan mengirim kembali teks terpenggal ke website, kemudian teks terpenggal yang diterima dikirim ke *responsive voice* untuk dibunyikan.

4.3 Hasil Pengujian

4.3.1 Pengujian *Black Box*

Pengujian *black box* bertujuan untuk menguji fungsionalitas dari sistem. Pengujian *black box* berdasarkan aksi dari pengguna. Sistem memiliki beberapa jenis aksi berupa tombol pada halaman detail berita website sehingga pengujian *black box* dilakukan dengan melihat hasil proses dari aksi yang dilakukan oleh pengguna. Setiap tombol memiliki fungsi dan proses yang berbeda. Hasil pengujian *black box* dapat dilihat pada Tabel 4.1 berikut.

Tabel 4.1 Hasil Pengujian *Black Box*

No.	Skenario Pengujian	Hasil yang diharapkan	Kesimpulan
1.	Meng-klik salah satu jenis berita dan masuk ke halaman detail berita	Sistem secara otomatis mengirim teks judul berita ke server pemenggalan kalimat dan mensintesa teks terpenggal menjadi ucapan/ suara.	Valid
2.	Tombol *play* di-klik	Sistem dapat mengirim teks isi berita ke server pemenggalan kalimat dan mensintesa teks	Valid

		terpenggal ke bentuk ucapan/ suara.	
3.	Tombol *pause* di-klik	Sistem dapat menghentikan sementara ucapan/ suara yang sedang berjalan.	Valid
4.	Tombol *resume* di-klik	Sistem dapat melanjutkan kembali ucapan/ suara yang sedang diberhentikan sementara.	Valid
5.	Tombol *stop* di-klik	Sistem dapat menghentikan ucapan/ suara yang sedang berjalan.	Valid

4.3.2 Pengujian Akurasi

Pengujian akurasi bertujuan untuk menguji tingkat akurasi *rule grammars* yang dikembangkan dan mendapatkan *rule* terbaik yang digunakan untuk proses penjedaan, dilakukan dengan mencocokkan kalimat antara *rule* yang dikembangkan peneliti dengan data jeda keras yang didapat dari narasumber. *Rule grammars* yang dibuat yaitu sebanyak 3 opsi seperti pada Kode program 4.8 berikut.

Kode Program 4.8. Tiga Opsi *Rule Grammars*

```
Rule Grammars 1:
G1 : { <NN> <IN> <CDP> }
G2 : { <NN> <VBT> <IN> }

KP : { <SC|RP|UH>+ }
VP : { <VBI|VBT|MD>+ <NN|RB>*}
NP : { <WP|PRP|PRN>+ }

G3 : { <IN> <NN>* <VP>+ }
G4 : { <KP>+ <NP|VP|KP|G3|NN|PRL>+ }
G5 : { <NP|CC|RB|NEG|MD>+ <VP|G1|NP|G3|CDI|DT|PRL>+ }
G6 : { <VP> <G3> }

Rule Grammars 2:
G1 : { <NN> <IN> <CDP> }
G2 : { <NN> <VBT> <IN> }

KP : { <SC|RP|UH>+ }
VP : { <VBI|VBT|MD>+ <CC>* <NN|RB>* }
NP : { <WP|PRP|PRN>+ }

G3 : { <RB>* <IN>+ <NN>+ <VP>*}
```

```
G4 : { <NN>* <IN> <VP> <G3>* <NP>*}
G5 : { <CC> <JJ>* <G3|G4|VP|G1|NP|NN|CDI|DT|RB|NEG>+ }
G6 : { <RB>* <NP|VP|NEG>+ <RB>* <VP|KP|NP|DT>* <G3|G4>* }
G7 : { <KP|G3>+ <G3|G4|G6|VP|NP|NN|RB>+ }

Rule Grammars 3:
G1 : { <NN> <IN> <CDP> }
G2 : { <NN> <VBT> <IN> }

KP : { <SC|RP|UH>+ }
VP : { <RB>* <VBI|VBT|MD>+ <CC>* <RB|NN>* }
NP : { <WP|PRP|PRN>+ }

G3 : { <RB>* <IN>+ <NN>+ <KP>* <VP>* }
G4 : { <NN>* <IN> <VP> }
G5 : { <CC> <JJ>* <G3|G4|VP|G1|CDI|NEG|NN|DT>+ <NP>* }
G6 : { <G4>* <VP|NP|NEG>+ <VP|NP|KP>* <G3|G4|DT>* }
G7 : { <KP|G3>+ <G6>? <G3|G4|VP|NN|DT>* }
```

Data jeda keras yang didapat dari narasumber adalah data jeda keras dari kesepakatan 3 orang, dapat dilihat pada lampiran F. Hasil dari pengujian akurasi dapat dilihat pada lampiran H. Data kalimat dengan jeda keras dari kesepakatan tiga orang narasumber yang didapat dari teks berita (74 kalimat) adalah sebanyak 50 kalimat, yang artinya semua kalimat teks berita memiliki jeda keras dari kesepakatan tiga orang. Kemudian data kalimat sebanyak 50 kalimat ini akan disaring untuk mendapatkan kalimat yang memiliki jumlah karakter ≤99 karakter dan jumlah karakter >99 karakter. Data kalimat jeda keras ≤99 karakter masuk ke perhitungan akurasi. Dari perhitungan jumlah karakter ini, didapat data seperti pada Tabel 4.2 berikut. Tabel lengkap terdapat pada lampiran G.

Tabel 4.2 Hasil Perhitungan Karakter

Jenis Jeda Keras	**Jumlah kalimat**	**Jumlah Kalimat pada Perhitungan Karakter**	
		≤99 Karakter	**>99 Karakter**
Dari kesepakatan 3 Orang	50	42	8

Pada Tabel 4.2, jumlah kalimat yang digunakan untuk perhitungan akurasi (jumlah karakter kalimat tunggal ≤ 99 karakter) adalah sebanyak 42 kalimat (jeda keras dari 3 orang).

Pada pengujian akurasi, dilakukan perhitungan persentase kecocokkan antara kalimat hasil *chunking* (*rule grammars* 1, 2 dan 3) dan kalimat dengan jeda

keras dari narasumber. Rumus yang digunakan untuk setiap *rule grammars* dengan kalimat jeda keras dari 3 narasumber dapat dilihat pada Persamaan (4.1) berikut.

$$\text{Tingkat akurasi (\%)} = \frac{Jumlah\ kalimat\ yang\ cocok}{Jumlah\ kalimat\ keseluruhan} \times 100 \quad (4.1)$$

Hasil perhitungan persentase tingkat akurasi dapat dilihat pada Tabel 4.3 berikut.

Tabel 4.3 Hasil Persentase Tingkat Akurasi

Jenis Jeda Keras	**Jumlah Kalimat**	**Jumlah Kalimat Cocok**			**Persentase Akurasi (%)**		
		***Rule* 1**	***Rule* 2**	***Rule* 3**	***Rule* 1**	***Rule* 2**	***Rule* 3**
Dari 3 Orang	42	13	9	8	30,952	21,429	19,048

Pada Tabel 4.3, persentase tingkat akurasi paling tinggi terletak pada *rule grammars* 1 dengan persentase sebesar 30,952% dari kalimat uji dengan jeda keras dari kesepakatan 3 orang.

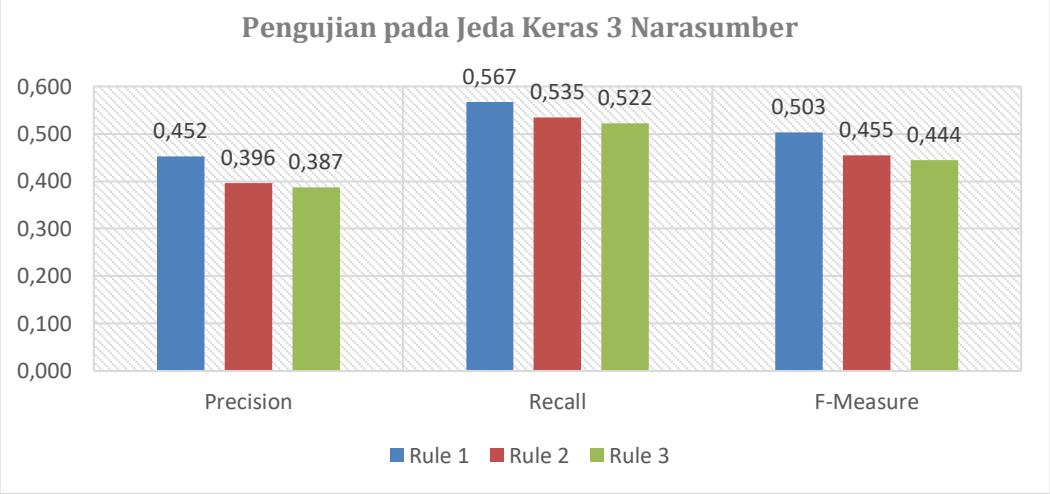
4.3.3 Pengujian *Precision*, *Recall* dan *F-Measure*

Pengujian *precision*, *recall* dan *f-measure* bertujuan untuk mendapatkan informasi hasil pencarian dokumen yang relevan dengan dokumen asli yang ingin dibandingkan. Pengujian *precision*, *recall*, dan *f-measure* dilakukan dengan membandingkan frasa jeda dari narasumber dan frasa jeda hasil implementasi *rule* jeda. Pengujian *precison*, *recall* dan *f-measure* pada penggalan frasa dilakukan untuk melihat persamaan penggalan frasa kalimat pada kalimat dari narasumber yang sudah ditandai kejadian jeda keras sesuai dari kesepakatan narasumber dan kalimat hasil prediksi dari *shallow parsing*. Jenis jeda keras yang digunakan adalah jeda keras dari kesepakatan 3 narasumber. Jumlah kalimat dengan jeda keras dari kesepakatan 3 narasumber yang diuji yaitu sebanyak 42 kalimat. Kalimat hasil prediksi dari *shallow parsing* dibentuk oleh 3 *rule grammars* yaitu *rule grammars* 1, 2 dan 3, yang dapat dilihat pada Kode program 4.8. Tabel hasil pengujian penggalan frasa terhadap jeda keras dari 3 narasumber dapat dilihat pada Tabel 4.4. Untuk hasil lengkap pengujian dapat dilihat di lampiran I, J dan K.

Tabel 4.4 Hasil Pengujian Penggalan Frasa Terhadap Jeda Keras dari 3 Narasumber

Rule	**Kalimat Uji**	***Precision*** $\left(\frac{a}{a+c}\right)$	***Recall*** $\left(\frac{a}{a+b}\right)$	***F-Measure*** $\left(2.\frac{p.r}{p+r}\right)$
1	1-42	0,452	0,567	0,503
2	1-42	0,396	0,535	0,455
3	1-42	0,387	0,522	0,444

Berikut grafik dari hasil pengujian penggalan frasa terhadap jeda keras dari kesepakatan 3 narasumber seperti pada Gambar 4.14.



Gambar 4.14 Grafik Hasil Pengujian Penggalan Frasa Terhadap Jeda Keras dari 3 Narasumber

Pada Gambar 4.14, nilai *precision* pada *rule* 1,2 dan 3 adalah sebesar 0,452 dan 0,396 dan 0,387, sedangkan nilai *recall* pada *rule* 1, 2 dan 3 adalah sebesar 0,567 dan 0,535 dan 0,522, sehingga *rule grammars* 1, 2 dan 3 dapat dikatakan masih belum efektif dilihat dari nilai *precision* yang masih dibawah 0,5. Hal ini disebabkan karena jumlah frasa yang tidak relevan lebih besar dibandingkan dengan jumlah yang relevan.

4.3.4 Pengujian Subjektif

Pengujian subjektif bertujuan untuk mendapatkan *rule grammars* terbaik yang digunakan untuk proses penjedaan kalimat. Pengujian dilakukan secara langsung kepada pendengar untuk mendengar dan menilai kualitas ucapan dari

suatu kalimat yang dibunyikan menggunakan aplikasi *responsive voice* untuk menyatakan frasa ucapan suatu kalimat dapat diterima atau tidak dapat diterima oleh pendengar. Kalimat yang diuji diambil dari kalimat berita. Kalimat yang dibunyikan diantaranya yakni 25 kalimat asli, 25 kalimat hasil pemenggalan menggunakan *rule grammars* 1 dan 25 kalimat hasil pemenggalan menggunakan *rule grammars* 2. Jumlah pendengar pada pengujian sebanyak 39 orang. Daftar pendengar dapat dilihat pada lampiran N. Pengujian dibagi menjadi 2 sesi, sesi 1 adalah mendengarkan ucapan dari kalimat asli dan kalimat *chunking rule grammars* 1, sedangkan pada sesi 2 adalah mendengarkan ucapan dari kalimat asli dan kalimat *chunking rule grammars* 2. Perhitungan persentase dilakukan dengan memperhatikan konsistensi isian dari setiap pendengar. Tabel perhitungan pengujian subjektif dapat dilihat pada lampiran M. Hasil perhitungan persentase ucapan diterima dapat dilihat pada Tabel 4.5 sebagai berikut.

Tabel 4.5 Hasil Perhitungan Persentase Ucapan Diterima pada Pengujian Subjektif

Rule Grammars	Jumlah Kalimat	Rata-Rata Persentase Ucapan Diterima
1	25	73,90
2	25	73,64

Pada Tabel 4.5, dapat dikatakan bahwa *rule grammars* 1 memiliki nilai persentase ucapan diterima lebih tinggi daripada *rule grammars* 2. Dari 25 kalimat yang diuji, terdapat 15 kalimat yang nilai pada *rule grammars* 1 lebih tinggi daripada *rule grammars* 2, 8 kalimat yang nilai pada *rule grammars* 1 lebih rendah daripada *rule grammars* 2 dan terdapat 2 kalimat yang nilai pada *rule grammars* 1 sama dengan *rule grammars* 2.

4.4 Analisis Hasil Pengujian

Dari beberapa pengujian yang telah dilakukan pada sistem *text to speech* Bahasa Indonesia pada website, berikut adalah beberapa analisa sistem sintesa ucapan.

1. Hasil pengujian *black-box* dapat dilihat pada Tabel 4.1. Hasil pengujian berupa sistem dapat menjalankan setiap proses yang ada pada website sesuai dengan

jenis aksi yang telah definisikan sesuai Tabel 3.4. Pengujian pada setiap skenario memiliki nilai valid karena telah sesuai dengan hasil/ keluaran yang diharapkan. Sehingga fungsional sistem pada website dapat dikatakan sistem telah berjalan dan menjalankan fungsi-fungsi sistem dengan benar.

2. Pengujian akurasi dilakukan dengan menghitung persentase tingkat akurasi kecocokkan antara kalimat dengan jeda keras yang diperoleh dari kesepakatan 3 narasumber dan kalimat hasil dari proses *shallow parsing*. Hasil yang diperoleh pada *rule grammars* 1, 2 dan 3 dengan jeda keras dari kesepakatan 3 narasumber adalah sebesar 30,952% ; 21,429% dan 19,048%. Hasil pengujian akurasi dapat dilihat pada Tabel 4.2 dan 4.3. Nilai persentase pada pengujian akurasi yang berada dikisaran 19%-31% disebabkan penggalan kalimat hasil proses *shallow parsing* masih banyak yang letak jedanya tidak sama seperti beda posisi jeda, kelebihan dan kekurangan dalam pemberian jeda. Adapun kesalahan lain dalam pemenggalan kalimat dikarenakan beberapa faktor diantaranya jeda keras memiliki pola jeda yang bervariasi dari narasumber yang menyebabkan kemunculan jeda menjadi tidak sama dan *rule grammars* membentuk frasa sesuai tipe PoS yang muncul pada kalimat. Pemberian kelas kata yang belum sempurna pada kalimat mengakibatkan *rule grammars* yang dibuat belum bisa memotong frasa secara akurat seperti pada Tabel 4.6 berikut.

Tabel 4.6 Perbandingan Hasil PoS *Tagger*

No	Kejadian PoS Tagger 1	Kejadian PoS Tagger 2
1	kelahiran/NN Pontianak/NN 16/CDP Juni/NN **2000/CDP**	pada/IN bulan/NN Mei/NN hingga/CC Desember/NN **2018/NN**
2	Pemanfaatan/NN Situs/NN Online/NN **Store/NNP** dan/CC **Market/NNP Place/NNP**	dikenal/VBT dengan/IN online/NN **store/NN** dan/CC digital/JJ **market/VBT place/NN**

3. Pengujian *precision*, *recall* dan *f-measure* dilakukan dengan menghitung tingkat akurasi frasa pada kalimat hasil penggalan *rule grammars* dengan

kalimat yang letak jeda kerasnya diperoleh dari kesepakatan 3 narasumber. Hasil pengujian dengan menghitung nilai *precision*, *recall* dan *f-measure* dari penggalan frasa jeda dapat dilihat pada Tabel 4.4 dan 4.5 yakni nilai prediksi jeda *shallow parsing* pada *rule grammars* 1, 2 dan 3 terhadap jeda keras dari 3 narasumber memiliki hasil yang berbeda. Berdasarkan Tabel 4.4 dan 4.5, nilai *precision* tertinggi terletak pada *rule grammars* 1 sebesar 0,452, nilai *recall* tertinggi terletak pada *rule grammars* 1 sebesar 0,567 dan nilai *f-measure* tertinggi terletak pada *rule grammars* 1 sebesar 0,503. Nilai *precision* yang berada dibawah 0,5 dapat dinyatakan masih belum efektif, dikarenakan jumlah frasa yang tidak relevan lebih besar dibandingkan dengan jumlah yang relevan. Nilai *precision* dan *recall* berada pada kisaran 0,49-0,51 disebabkan oleh beberapa faktor diantaranya *rule grammars* untuk *shallow parsing* yang dirancang masih belum sempurna untuk menghasilkan potongan frasa jeda yang tepat, pemberian kelas kata yang belum sempurna pada kalimat sehingga *rule grammars* yang dibuat belum bisa memotong frasa secara akurat, dan jeda keras memiliki pola jeda yang bervariasi dari narasumber yang menyebabkan kemunculan jeda menjadi tidak sama.

4. Pengujian subjektif dilakukan secara langsung kepada pendengar untuk mendengar dan menilai kualitas ucapan dari suatu kalimat yakni kalimat berita asli dan kalimat berita hasil penggalan *shallow parsing* menggunakan *rule grammars*, yang dibunyikan menggunakan aplikasi *responsive voice* untuk menyatakan frasa ucapan suatu kalimat dapat diterima atau tidak dapat diterima oleh pendengar. Hasil dari perhitungan uji dengar pada pengujian subjektif adalah sebanyak 73,90% pada *rule grammars* 1 dan 73,64% pada *rule grammars* 2. Nilai persentase pada uji dengar berada diatas 50%, sehingga dapat dikatakan bahwa *rule grammars* 1 dan *rule grammars* 2 dapat diterima dengan cukup baik oleh pendengar dengan nilai persentase pada *rule grammars* 1 lebih besar daripada *rule grammars* 2.
5. Proses implementasi *text to speech* pada website berhasil dan dapat berjalan sesuai dengan yang diharapkan serta proses pemenggalan kalimat menjadi kalimat terpenggal pada server dapat berjalan sesuai dengan yang diharapkan.

6. Proses sintesa ucapan berupa suara menggunakan bantuan dari *responsive voice* dalam suara wanita dengan Bahasa Indonesia. Hasil sintesa ucapan dapat berjalan sesuai dengan yang diharapkan, namun masih terkendala oleh beberapa hal yang merupakan kekurangan dari *responsive voice* sendiri. Beberapa diantaranya yaitu sebagai berikut.
    a. Proses *pause* atau memberhentikan sementara ucapan yang keluar pada website hanya dapat bertahan selama kurang lebih atau sama dengan 15 detik.
    b. Tidak dapat membaca angka yang didalamnya memakai tanda titik (“.”), contohnya yaitu tidak dapat membaca dengan baik pada angka “17.000” tetapi dapat membaca dengan baik jika angka “17000”.
    c. Pembacaan pada kata yang bercetak tebal dan kata yang tidak bercetak tebal akan dibaca sama.
    d. Sudah dapat membaca kata dalam bahasa inggris dengan baik, namun terdapat beberapa kata yang masih belum dapat dibaca degan baik.
    e. Sudah dapat membaca singkatan dengan baik, namun terdapat beberapa singkatan yang masih belum dapat dibaca dengan baik.
7. Hasil perhitungan pada pengujian akurasi dan *precision*, *recall* dan *f-measure*, menunjukkan rentang angka persentase dari kecil-sedang. Hal ini terjadi dikarenakan beberapa faktor yang telah disebutkan pada kesimpulan poin ke 2 dan ke 3. Meskipun nilai persentase yang didapatkan belum menunjukkan angka yang efektif, namun kualitas dari hasil sistem pemenggalan kalimat berupa teks terpenggal masih dapat dikategorikan cukup. Hal ini terjadi karena pada perhitungan uji dengar dalam poin 4, ketika teks terpenggal dibunyikan, penggalan kalimat yang diucapkan dapat diterima dengan cukup baik oleh pendengar.

BAB V
PENUTUP

5.1 Kesimpulan

Berdasarkan hasil analisis dan pengujian terhadap sistem implementasi *text to speech* pada website menggunakan metode *shallow parsing*, maka dapat ditarik kesimpulan sebagai berikut:

1. Berdasarkan pengujian *black box*, sistem dapat menjalankan setiap proses yang ada pada website dengan baik sesuai dengan jenis aksi yang telah definisikan. Nilai valid pada setiap skenario menunjukkan bahwa fungsional pada sistem telah berjalan sesuai dengan yang diharapkan.
2. Berdasarkan pengujian akurasi, persentase tingkat akurasi tertinggi antara *rule grammars* 1, 2 dan 3, terletak pada *rule grammars* 1 yaitu sebesar 30,952%. Angka persentase ini didapat dari proses pencocokkan antara kalimat dengan jeda keras dan kalimat hasil dari proses *shallow parsing*.
3. Berdasarkan pengujian *precision*, *recall* dan *f-measure*, nilai tertinggi terletak pada *rule grammars* 1 dengan nilai *precision* sebesar 0,452, nilai *recall* sebesar 0,567 dan nilai *f-measure* sebesar 0,503. Nilai *precision* yang berada dibawah 0,5 atau 50% dapat dinyatakan masih belum efektif, dikarenakan jumlah frasa yang tidak relevan lebih besar dibandingkan dengan jumlah yang relevan.
4. Berdasarkan pada pengujian subjektif, hasil dari perhitungan persentase ucapan telah diterima oleh pendengar pada *rule grammars* 1 dan 2 adalah sebesar 73,90% dan 73,64%. Nilai persentase pada uji dengar memiliki nilai yang cukup besar, sehingga dapat dikatakan bahwa *rule grammars* 1 dan 2 dapat diterima dengan cukup baik oleh pendengar dengan nilai persentase pada *rule grammars* 1 lebih besar daripada *rule grammars* 2.
5. Implementasi *text to speech* pada website berhasil dan dapat berjalan sesuai dengan yang diharapkan serta proses pemenggalan teks menjadi teks terpenggal pada server dapat berjalan sesuai dengan yang diharapkan. Sintesa ucapan yakni berupa suara menggunakan bantuan dari *responsive voice* dalam suara wanita dengan Bahasa Indonesia yang telah berjalan sesuai dengan yang diharapkan.

6. Berdasarkan hasil perhitungan pada pengujian *precision*, *recall*, dan *f-measure* serta pengujian akurasi, menunjukkan rentang angka persentase dari kecil-sedang. Meskipun nilai persentase yang didapatkan belum menunjukkan angka yang efektif, namun kualitas dari hasil sistem pemenggalan kalimat berupa teks terpenggal dapat dikategorikan cukup. Hal ini terjadi karena pada perhitungan pengujian subjektif, saat teks terpenggal dibunyikan, penggalan kalimat yang diucapkan dapat diterima dengan cukup baik oleh pengguna.

5.2 Saran

Adapun beberapa hal yang perlu ditambahkan dalam pengembangan sistem implementasi *text to speech* pada website menggunakan metode *shallow parsing* adalah sebagai berikut:

1. Perlu adanya pengembangan PoS *tagger* yang lebih spesifik untuk pelabelan kelas kata dalam kalimat supaya pengembangan *rule grammars* dapat lebih akurat.
2. Perlu adanya pengembangan *rule grammars* berdasarkan teks berita dalam Bahasa Indonesia yang lebih spesifik dengan jumlah karakter tiap penggalannya tidak lebih dari 99 karakter, bertujuan agar hasil penggalan pada kalimat menjadi lebih baik.
3. Perlu adanya tambahan narasumber yang ahli dalam bidang bahasa khususnya Bahasa Indonesia untuk memperkuat data jeda keras pada kalimat berita dan penambahan referensi dalam pengembangan *rule grammars*.
4. Perlu adanya analisis yang lebih mendalam pada *responsive voice* agar perancangan aplikasi dan pengembangan *rule grammars* menjadi lebih efektif dan efisien.