PENERAPAN ALGORITMA JARO-WINKLER UNTUK AUTOCORRECT DAN SPELLING SUGGESTION PADA APLIKASI SPEECH RECOGNITION CMS BERBASIS WEBSITE

TUGAS AKHIR

Program Studi

S1 Sistem Informasi

Oleh :

I GEDE ADI WIJAYA

17410100035

FAKULTAS TEKNOLOGI DAN INFORMATIKA

UNIVERSITAS DINAMIKA

2021

PENERAPAN ALGORITMA JARO-WINKLER UNTUK AUTOCORRECT DAN SPELLING SUGGESTION PADA APLIKASI SPEECH RECOGNITION CMS BERBASIS WEBSITE

TUGAS AKHIR

Diajukan sebagai syarat untuk menyelesaikan Program Sarjana

Oleh :

Nama : I Gede Adi Wijaya

NIM : 17.41010.0035

Program : S1 Sistem Informasi

FAKULTAS TEKNOLOGI DAN INFORMATIKA

UNIVERSITAS DINAMIKA

2021

TUGAS AKHIR

PENERAPAN ALGORITMA JARO-WINKLER UNTUK AUTOCORRECT DAN SPELLING SUGGESTION PADA APLIKASI SPEECH RECOGNITION CMS BERBASIS WEBSITE

Dipersiapkan dan disusun oleh

I Gede Adi Wijaya

NIM: 17410100035

Telah diperiksa, dibahas dan disetujui oleh Dewan Pembahas

Pada: Jumat, 30 Juli 2021

Susunan Dewan Pembahas

Pembimbing

I. **Tri Sagirani, S.Kom., M.MT.**
NIDN. 0731017601

Digitally signed by Universitas Dinamika
Date: 2021.07.31 14:41:50 +07'00'

II. **Norma Ningsih , S.ST.,M.T**
NIDN. 0729099002

Digitally signed by Norma Ningsih
Date: 2021.07.31 22:12:05 +07'00'

Pembahas

I. **Dr. Jusak**
NIDN. 0708017101

Digitally signed by Universitas Dinamika
Date: 2021.08.02 08:53:14 +07'00'

Tugas Akhir ini telah diterima sebagai salah satu persyaratan

untuk memperoleh gelar Sarjana:

Digitally signed by Universitas Dinamika
Date: 2021.08.05 09:23:51 +07'00'

Tri Sagirani, S.Kom., M.MT.

NIDN. **0731017601**

Dekan Fakultas Teknologi dan Informatika

UNIVERSITAS DINAMIKA

The best way to predict

Your future is to create it.

- Abraham Lincoln

Ku persembahkan kepada

Ibu & Bapak,

Serta teman, sahabat yang

Selalu memberikan semangat disetiap perjuangan ku

SURAT PERNYATAAN

PERSETUJUAN PUBLIKASI DAN KEASLIAN KARYA ILMIAH

Sebagai mahasiswa Universitas Dinamika, saya :

Nama	:	I Gede Adi Wijaya
Nim	:	17410100035
Program Studi	:	S1 Sistem Informasi
Fakultas	:	Fakultas Teknologi dan Informatika
Jenis Karya	:	Tugas Akhir
Judul Karya	:	**PENERAPAN ALGORITMA *JARO-WINKLER* UNTUK *AUTOCORRECT* DAN *SPELLING SUGGESTION* PADA APLIKASI *SPEECH RECOGNITION CMS* BERBASIS WEBSITE**

Menyatakan dengan sesungguhnya bahwa:

1. Demi pengembangan Ilmu Pengetahuan, Teknologi dan Seni, saya menyetujui memberikan kepada Universitas Dinamika Hak Bebas Royalti Non-Eksklusif *(Non-Exclusive Royalti Free Right)* atas seluruh isi/ sebagian karya ilmiah saya tersebut di atas untuk disimpan, diahlimediakan dan dikelola dalam bentuk pangkalan data *(database)* untuk selanjutnya didistribusikan atau dipublikasikan demi kepentingan akademis dengan tetap mencantumkan nama saya sebagai penulis atau pencipta dan sebagai pemilik Hak Cipta
2. Karya tersebut di atas adalah karya asli saya, bukan plagiat baik sebagian maupun keseluruhan, Kutipan karya atau pendapat orang lain yang ada dalam karya ilmiah ini adalah semata hanya rujukan yang dicantumkan dalam Daftar Pustaka saya
3. Apabila dikemudian hari ditemukan dan terbukti terdapat tindakan plagiat pada karya ilmiah ini, maka saya bersedia untuk menerima pencabut terhadap gelar kerjasama yang telah diberikan kepada saya.

Demikian surat pernyataan ini saya buat dengan sebenarnya.

Surabaya, 11 Juli 2021

Yang menyatakan

I Gede Adi Wijaya

NIM : 17410100035

ABSTRAK

PT. Bukaloka Teknologi Indonesia merupakan situs jual beli online yang mendistribusikan produk dan jasa yang dilengkapi dengan *platform generate* toko online & web instant yang dimana layanan ini merupakan sebuah platform yang mengintegrasikan *marketplace* dengan layanan pembuatan *Company Profile Online* untuk usaha mikro, kecil, dan menengah (UMKM) yang ikut bergabung. Saat ini proses perubahan informasi *Company Profile* UMKM masih harus menggunakan *Content Management System* (CMS) standar yang mengharuskan perubahan dilakukan didalam *dashboard* CMS-nya sendiri dan pengisian data dilakukan dengan pengetikan di form data. Penelitian ini membahas mengenai pengimplementasian *Speech Recognition* yang didukung dengan Algoritma *Jaro-Winkler* kedalam CMS yang dapat mempermudah dan mempercepat, proses pemasukan kata yang akan menjadi konten dari *Company Profile* dibandingkan dari penggunaan *Content Management System* (CMS) tanpa perlu adanya penggunaan *keyboard*, yang meminimalkan potensi kesalahan kata serta proses pengetikan yang lama. Algoritma *Jaro-Winkler* sendiri dimaksudkan pembenahan kata dan proses pengakuratan kata pada potensi kesalahan kata yang dihasilkan oleh *Speech Recognition*. 30 calon pengguna disurvey untuk mengetahui kelayakan aplikasi di masyarakat dan mendapatkan hasil perhitungan survei dengan jumlah respon yang mengatakan "Ya" sebanyak 85.63%, sedangkan "Tidak" sebanyak 14.36%. Dengan total keseluruhan yang didapat yaitu 85.63% yang dapat disimpulkan mendapatkan hasil "Berhasil".

Kata Kunci : *Speech Recognition, Algoritma Jaro-Winkler*

KATA PENGANTAR

Puji syukur ke hadirat Tuhan Yang Maha Esa yang telah memberikan rahmat dan hidayah-Nya sehingga penulis mempu menyelesaikan Tugas Akhir dengan judul "**PENERAPAN ALGORITMA *JARO-WINKLER* UNTUK *AUTOCORRECT* DAN *SPELLING SUGGESTION* PADA APLIKASI *SPEECH RECOGNITION CMS* BERBASIS WEBSITE**".

Penyelesaian Tugas Akhir ini tidak terlepas dari bantuan berbagai pihak yang telah memberikan banyak masukan, nasihat, saran, kritik, dan dukungan moral maupun materil kepada penulis. Oleh karena itu penulis menyampaikan rasa terima kasih kepada:

1. Tuhan Yang Maha Esa yang telah memberikan petunjuk, kekuatan serta kesehatan kepada penulis dalam melaksanakan penelitian Tugas Akhir hingga penyusunan laporan ini.
2. Bapak dan Ibu tercinta serta keluarga yang selalu mendoakan, mendukung, dan memberikan semangat di setiap langkah dan aktivitas penulis.
3. Ibu Tri Sagirani, S.Kom., M.MT. selaku Dosen Pembimbing 1 yang selalu membimbing, mendukung, dan memberikan motivasi kepada penulis dalam menyelesaikan Tugas Akhir ini.
4. Ibu Norma Ningsih, S.ST., M.T. selaku Dosen Pembimbing 2 dan juga selalu membimbing, mendukung, memberikan motivasi dan arahan kepada penulis dalam menyelesaikan Tugas Akhir ini.
5. Bapak Dr. Jusak selaku Dosen Pembahas yang telah bersedia menjadi dosen pembahas dalam mengerjakan Tugas Akhir ini.
6. Teman-teman tercinta yang memberikan bantuan dan dukungan dalam menyelesaikan Tugas Akhir ini.
7. Pihak-pihak lain yang tidak dapat disebutkan satu-persatu yang telah memberikan bantuan dan dukungan kepada penulis.

Semoga Tuhan Yang Maha Esa memberikan balasan yang setimpal kepada semua pihak yang telah membantu dan memberikan bimbingan serta nasehat dalam proses menyelesaikan Tugas Akhir ini.

Penulis menyadari bahwa Tugas Akhir yang dikerjakan ini masih banyak terdapat kekurangan sehingga kritik yang bersifat membangun dan saran dari semua pihak sangatlah diharapkan agar aplikasi ini dapat diperbaiki menjadi lebih baik lagi. Semoga Tugas Akhir ini dapat diterima dan bermanfaat bagi penulis dan semua pihak.

Denpasar, 11 Juli 2021

I Gede Adi Wijaya

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Halaman

DAFTAR TABEL

Halaman

DAFTAR LAMPIRAN

Halaman

BAB I
PENDAHULUAN

1.1 Latar Belakang

PT. Bukaloka Teknologi Indonesia merupakan sebuah startup yang bertempat di pulau Bali dengan Putu Aditya Santana sebagai *Founder* & *CEO* PT. Bukaloka Teknologi Indonesia dan berdiri Oktober 2016 di Art Center, Denpasar, Bali. PT. Bukaloka Teknologi Indonesia merupakan situs jual beli online yang mendistribusikan produk dan jasa yang dilengkapi dengan *platform generate* toko online & web instant yang dimana layanan ini merupakan sebuah platform yang mengintegrasikan *marketplace* dengan layanan pembuatan *Company Profile Online* untuk usaha mikro, kecil, dan menengah (UMKM) yang ikut bergabung.

Proses Bisnis yang masih berjalan di PT. Bukaloka Teknologi Indonesia untuk saat ini dalam pembuatan situs toko online UMKM yang akan bergabung dengan PT. Bukaloka masih menggunakan *Content Management System* (CMS) yang biasa kita jumpai pada saat membuat blog pada www.blogger.com, Metode entry data yang akan ditampilkan ke *Company Profile* pada saat ini masih harus menuju ke halaman *CMS* dan harus menggunakan perangkat input seperti keyboard, yang kebanyakan orang yang masih awam dalam mengetik untuk menghasilkan beberapa kata-kata yang typo selain itu proses pengetikan untuk data yang banyak membutuhkan waktu yang lama, berdasarkan data observasi dari 30 Owner UMKM yang bergabung mempunyai kecepatan rata-rata 19-25 kata per menitnya, kecepatannya masih tergolong sangat rendah. *Speech Recognition* sangat cocok untuk menjawab permasalahan tersebut. dimana *Speech Recognition* sangat berguna untuk mempercepat proses pembuatan content, karena proses input data dengan berbicara jauh lebih mudah digunakan oleh orang yang awam dalam mengetik menggunakan keyboard, maka peneliti mengusulkan untuk pengimplementasian *Speech Recognition* kedalam *Content Management System* (CMS) menjadi pengganti perangkat input seperti keyboard.

Speech Recognition adalah salah satu bidang kecerdasan mesin yang sedang berkembang pesat, hal itu ditandai oleh hampir semua *device* teknologi dilengkapi oleh *voice command*. Hal ini telah menarik bagi para peneliti untuk menjadikan *Speech Recognition* sebagai disiplin ilmu yang penting untuk menciptakan dampak teknologi pada masyarakat dan diharapkan akan berkembang lebih jauh di bidang interaksi mesin dengan manusia. *Speech Recognition* sendiri adalah proses menangkap kata yang diucapkan melalui mikropon atau telepon dan mengubahnya ke dalam kata-kata yang tersimpan secara digital (Jelinek, 1997). Untuk membatasi kesalahan kata yang dihasilkan oleh *Speech Recognition* ini maka akan digunakan algoritma *Jaro-Winkler*. Algoritma *Jaro-Winkler* merupakan varian dari *Jaro Distance* metrik yaitu sebuah algoritma untuk mengukur kesamaan antara dua buah string, biasanya algoritma ini digunakan di dalam pendeteksian duplikat dan dapat dialih fungsikan menjadi fitur *Autocorrect* dan *Spelling Suggestion.*

Penggunaan *Speech Recognition* sendiri akan dapat mempermudah dan mempercepat, proses pemasukan kata yang akan menjadi content dari website dibandingkan dari penggunaan *Content Management System* (CMS) data tanpa perlu adanya penggunaan keyboard yang akan meminimalkan potensi kesalahan kata serta proses pengetikan yang lama. Algoritma *Jaro-Winkler* dimaksudkan pembenahan kata dan proses pengakuratan kata pada potensi kesalahan kata yang dihasilkan oleh *Speech Recognition.* Berdasarkan uraian di atas diperlukan penerapan suatu metode yang pengimplementasikan Teknologi *Speech Recognition* pada CMS berbasis website dengan menggunakan Algoritma *Jaro-Winkler* guna membantu proses *Autocorrect* dan *Spelling Suggestion*, yang memudahkan para UMKM pada proses input data dan meminimalisir potensi kesalahan kata dalam mengembangkan *Company Profile*.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan sebelumnya didapat rumusan masalah penelitian adalah bagaimana cara penerapan algoritma *jaro-winkler* untuk *autocorrect* dan *spelling suggestion* pada aplikasi *speech recognition* CMS berbasis website.

1.3 Batasan Masalah

Batasan masalah pada Penerapan Algoritma *Jaro-Winkler* Untuk *Autocorrect* dan *Spelling Suggestion* Pada Aplikasi *Speech Recognition CMS* Berbasis Website ini menjadi lebih terarah dan tidak menyimpang dari tujuan pembahasan adalah sebagai berikut :

1. Sistem tidak akan menyangkut pengembangan dari *marketplace.*
2. Sistem tidak melayani proses transaksi dari produk yang terdapat pada *Company Profile.*
3. Pengembangan sistem hanya menyangkut penerapan Teknologi *Speech Recognition* pada *Content Management System* (CMS) dan *Company Profile.*
4. Penggunaan Teknologi *Speech Recognition* hanya *work* pada *browser* Google Chrome.
5. Pembahasan lebih berfokus pada proses penerapan Algoritma *Jaro-Winkler* pada Teknologi *Speech Recognition.*

1.4 Tujuan

Tujuan dari penelitian ini adalah untuk menerapkan *Speech Recognition* pada *Content Management System* (*CMS*) dengan metode Algoritma *Jaro-Winkler* untuk pembenahan kata dan proses pengakuratan kata sehingga mengurangi potensi kesalahan kata dan mempercepat proses pembuatan dan perubahan konten.

1.5 Manfaat

Berikut manfaat dari Penerapan Algoritma *Jaro-Winkler* Untuk *Autocorrect* Dan *Spelling Suggestion* Pada Aplikasi *Speech Recognition CMS* Berbasis Website, yaitu :

1. Mengurangi mengurangi potensi kesalahan kata yang diakibatkan pada saat proses *input* data.
2. Menjadi alternatif jika tidak tersedianya perangkat *input* seperti *keyboard*.
3. Mempercepat proses pembuatan dan perubahan konten pada *company profile* tanpa harus kembali ke halaman dashboard untuk melakukan perubahan.

BAB II
LANDASAN TEORI

Landasan teori digunakan untuk dijadikan dasar dalam memberikan solusi yang ditawarkan untuk menyelesaikan permasalahan yang telah dijelaskan sebelumnya, yaitu penerapan algoritma *jaro-winkler* untuk *autocorrect* dan *spelling suggestion* pada aplikasi *speech recognition* CMS berbasis *website.*

2.1 Penelitian Terdahulu

Dalam penulisan proposal peneliti menggali informasi dari penelitian-penelitian sebelumnya sabagai bahan perbandingan, baik mengenai kekurangan atau kelebihan yang sudah ada. Penelitian terdahulu ini menjadi salah satu acuan penulis dalam melakukan penelitian sehingga penulis dapat memperkaya teori yang digunakan dalam mengkaji penelitian yang dilakukan. Dari penelitian terdahulu, penulis tidak menemukan penelitian dengan judul yang sama seperti judul penelitian penulis. Namun penulis mengangkat beberapa penelitian sebagai referensi dalam memperkaya bahan kajian pada penelitian penulis. Berikut merupakan penelitian terdahulu berupa beberapa jurnal terkait dengan penelitian yang dilakukan penulis.

Nama Peneliti	Judul Peneliti	Hasil Peneliti
Ratna, Sanjaya, Wirianata, dan Purnamasari (2020)	Word Level Auto-correction for Latent Semantic Analysis Based Essay Grading System	Penelitian ini menghasilkan sebuah teknik koreksi otomatis untuk memeriksa kata dari perpustakaan kata untuk pemerataan kata dengan arti yang sama atau tidak spesifik. Kemudian, algoritma jarak Jaro-Winkler digunakan untuk memeriksa kesalahan kata yang disebabkan oleh kecelakaan saat mengetik.

Perbedaan : penelitian ini menggunakan 4 tahapan dalam memilah arti dari esay yang akan diuji untuk digunakan sebagai patokan untuk diuji dengan Algoritma *Jaro-Winkler Distance* untuk proses autocorrect, sedangkan tahapan untuk

memilah arti dari kalimat tidak termasuk dalam penelitian yang diteliti penulis, karena proses *autocorrect* pada penelitian penulis hanya membandingkan kata dari hasil pengolahan *speech recognition* dengan sebuah database kata dasar yang sudah disiapkan.

Sumber: Department of Electrical Engineering, Faculty of Engineering Universitas Indonesia Depok, Indonesia

Nama Peneliti	Judul Peneliti	Hasil Peneliti
Prasetyo, Baihaqi, dan Iqbaluddin (2018)	Algoritma *Jaro-Winkler Distance*: Fitur Autocorrect Dan *Spelling Suggestion* Pada Penulisan Naskah Bahasa Indonesia Di Bms Tv	Penelitian ini menghasilkan sebuah sistem yang dapat membantu News Director MS TV dalam pemeriksaan kesalahan penulisan ejaan kata pada naskah bahasa Indonesia dan mempermudah News Director pusat dalam penghimpunan naskah dari berbagai kontributor BMS TV
Perbedaan : penelitian Agung Prasetyo, Wiga Maulana Baihaqi, Iqbaluddin Syam Had menggunakan *Tokenizing* sebagai sarana pemecah string sebelum di proses oleh Algoritma *Jaro-Winkler Distance*, sedangkan sarana pemecah string yang diteliti penulis adalah pemecahan string langsung menggunakan data hasil olahan *speech recognitionnya.*		

Sumber: Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK) Vol. 5, No. 4, September 2018, hlm. 435-444.

Nama Peneliti	Judul Peneliti	Hasil Peneliti
Friendly (2017)	Perbaikan Metode *Jaro–Winkler Distance* Untuk *Approximate* String Search Menggunakan Data Terindeks Aplikasi Multi User	Penelitian ini menghasilkan peningkatan kecepatan metode Jaro Winkler dengan penggunaan data terindeks dengan hasil

		pencarian mencapai 90% lebih cepat untuk jumlah data 16.926 dan 92% lebih cepat untuk jumlah data 69.145 dari pada penggunaan secara langsung metode Jaro Winkler.
Perbedaan : penelitian Friendly lebih berfokus dalam peningkatan kecepatan proses pengolahan string untuk pengakurasian data pada Algoritma *Jaro-Winkler Distance*, sedangkan penilitan yang ditulis oleh penulis adalah penerapan Algoritma *Jaro-Winkler Distance* yang tidak memerlukan peningkatan kecepatan.		

Sumber: Jurnal Teknovasi Volume 04, Nomor 02, 2017, 69 – 78

Nama Peneliti	**Judul Peneliti**	**Hasil Peneliti**
Leonardo dan Hansun (2017)	Text Documents Plagiarism Detection using Rabin-Karp and *Jaro-Winkler Distance* Algorithms	Penelitian ini menghasilkan sistem yang dapat mengetahui plagiarism kalimat yang dilakukan oleh penggunanya menggunakan Rabin-Karp dan *Jaro-Winkler Distance*
Perbedaan : penelitian Brinardi Leonardo, Seng Hansun menggunkan Algoritma *Jaro-Winkler Distance* sebagai acuan untuk menghitung potensi plagiarism kalimat sedangkan penggunaan Algoritma *Jaro-Winkler Distance* yang diteliti penulis akan menggunakan potensi plagiarism kalimat yang ditemukan menjadi acuan kata yang lebih akurat.		

Sumber: *Indonesian Journal of Electrical Engineering and Computer Science* Vol. 5, No. 2, February 2017, pp. 462 ~ 471

Nama Peneliti	Judul Peneliti	Hasil Peneliti
Fairly, Saptono, dan Sulistyo (2015)	*Jaro-Winkler Distance* Dan Stemming Untuk Deteksi Dini Hama Dan Penyakit Padi	Penelitian ini menghasilkan sistem yang dapat membantu mengubah input kata yang salah atau typo menjadi kata yang dimaksud dalam pendeteksian hama dan penyakit padi.
Perbedaan : penelitian Fairly Okta'mal , Ristu Saptono, Meiyanto Eko Sulistyo menggunakan Stemming untuk mencari kata-kata akar (*root word*) sebelum di proses oleh Algoritma *Jaro-Winkler Distance*, sedangkan penelitian yang diteliti penulis tidak menggunakan konsep Stemming untuk mencari kata-kata akar melainkan kata real yang diucapkan pengguna.		

Sumber: Seminar Nasional Sistem Informasi Indonesia, 2-3 November 2015

Nama Peneliti	Judul Peneliti	Hasil Peneliti
Tinaliah dan Elizabeth (2018)	Perbandingan Hasil Deteksi Plagiarisme Dokumen dengan Metode *JaroWinkler* Distance dan Metode *Latent Semantic Analysis*	Penelitian ini menghasilkan sistem yang dapat mengetahui plagiarism kalimat yang dilakukan oleh penggunanya dan membandingkan hasil plagiarism antara *Latent Semantic Analysis* dan *Jaro-Winkler Distance*
Perbedaan : penelitian Tinaliah, Triana Elizabeth menggunkan Algoritma *Jaro-Winkler Distance* sebagai acuan untuk menghitung potensi plagiarism kalimat yang akan dilakukan pembandingan dengan *Latent Semantic Analysis* sedangkan penggunaan Algoritma *Jaro-Winkler Distance* yang diteliti penulis akan menggunakan potensi plagiarism kalimat yang ditemukan menjadi acuan kata yang lebih akurat.		

Sumber: Jurnal Teknologi dan Sistem Komputer, 6(1), 2018, 7-12

2.2 Natural Language Processing

*Natural Language Processing (*NLP*)* memberi mesin kemampuan untuk membaca dan memahami bahasa yang digunakan manusia untuk berbicara. Banyak peneliti berharap bahwa sistem pemrosesan *natural language* yang cukup kuat akan dapat memperoleh pengetahuan dengan sendirinya, dengan membaca teks yang ada yang tersedia melalui internet. NLP adalah upaya untuk mengekstrak lebih jauh representasi dari suatu teks bebas. Hal ini dapat dimasukkan secara kasar seperti mencari siapa melakukan apa kepada siapa, kapan, di mana, bagaimana dan mengapa. NLP biasanya membuat penggunaan konsep-konsep linguistic seperti kata benda, kata kerja, kata sifat, dan lainnya dan struktur gramatikal (baik direpresentasikan sebagai ungkapan-ungkapan seperti frase nomina atau frase preposisional, atau hubungan ketergantungan seperti subjek dari- atau objek-dari). Beberapa aplikasi pemrosesan bahasa alami yang mudah dilakukan mencakup pencarian informasi (atau penambangan teks) dan terjemahan mesin (Pustejovsky dan Stubbs, 2012).

Pustejovsky dan Stubbs (2012) juga menjelaskan bahwa ada beberapa area utama penelitian pada field NLP, diantaranya:

1. *Question Answering Systems* (QAS). Kemampuan komputer untuk menjawab pertanyaan yang diberikan oleh user. Daripada memasukkan *keyword* ke dalam browser pencarian, dengan QAS, user bisa langsung bertanya dalam bahasa natural yang digunakannya, baik itu Inggris, Mandarin, ataupun Indonesia.
2. *Summarization*. Pembuatan ringkasan dari sekumpulan konten dokumen atau email. Dengan menggunakan aplikasi ini, user bisa dibantu untuk mengkonversikan dokumen teks yang besar ke dalam bentuk slide presentasi.
3. *Machine Translation*. Produk yang dihasilkan adalah aplikasi yang dapat memahami bahasa manusia dan menterjemahkannya ke dalam bahasa lain. Termasuk di dalamnya adalah Google Translate yang apabila dicermati semakin membaik dalam penterjemahan bahasa.

Contoh lain lagi adalah BabelFish yang menterjemahkan bahasa pada real time.

4. *Speech Recognition*. Field ini merupakan cabang ilmu NLP yang cukup sulit. Proses pembangunan model untuk digunakan telpon/komputer dalam mengenali bahasa yang diucapkan sudah banyak dikerjakan. Bahasa yang sering digunakan adalah berupa pertanyaan dan perintah.
5. *Document classification*. Sedangkan aplikasi ini adalah merupakan area penelitian NLP Yang paling sukses. Pekerjaan yang dilakukan aplikasi ini adalah menentukan dimana tempat terbaik dokumen yang baru diinputkan ke dalam sistem. Hal ini sangat berguna pada aplikasi *spam filtering, news article classification*, dan *movie review*.

2.3 Speech Recognition

Speech Recognition adalah proses menangkap kata yang diucapkan melalui mikropon atau telepon dan mengubahnya ke dalam kata-kata yang tersimpan secara digital dengan cara mengubah gelombang suara menjadi sekumpulan angka lalu disesuaikan dengan kode-kode tertentu dan dicocokkan dengan suatu pola yang tersimpan dalam suatu perangkat. Hasil dari identifikasi kata yang diucapkan dapat ditampilkan dalam bentuk tulisan atau dapat dibaca oleh perangkat teknologi (Jelinek, 1997).

Kualitas dari sistem *Speech Recognition* ditaksir dari dua faktor, yaitu akurasi (tingkat kesalahan dalam mengubah kata yang diucapkan ke dalam data digital) dan kecepatan (seberapa cepat *software* tersebut dapat mengikuti pembicaraan manusia). Teknologi *Speech Recognition* memiliki aplikasi yang tak ada habisnya. Umumnya, software tersebut digunakan untuk penerjemahan otomatis, pendiktean, komputasi bebas-tangan, transkripsi medis, robotika, layanan pelanggan otomatis, dan masih banyak lagi.

Ukuran kosakata (*vocabulary*) dari sistem pengenalan suara memengaruhi kompleksitas, parameter pelatihan dan akurasi sistem. Beberapa aplikasi pengenalan

suara hanya memerlukan beberapa kata, sedangkan yang lainnya memerlukan kamus yang sangat besar (misalnya mesin pendiktean). Terdapat 4 jenis ukuran kosakata, yaitu:

1. Kosakata ukuran kecil (*small vocabulary*) yang terdiri dari puluhan kata.
2. Kosakata ukuran sedang (*medium vocabulary*) yang terdiri dari ratusan kata.
3. Kosakata ukuran besar (*large vocabulary*) yang terdiri dari ribuan kata.
4. Kosakata ukuran sangat besar (*very large vocabulary*) yang terdiri dari puluhan ribu kata (Monika, 2014).

2.4 *Content Management System* (CMS)

Content management system (CMS) adalah sebuah aplikasi yang digunakan untuk membantu *user* dalam proses memanage konten. Dalam terminologi web secara lebih spesifik berarti sebuah *system* yang digunakan untuk memanage material web yang merupakan konten dari sebuah website. Dengan CMS, seorang *user* dapat mengkontrol, meng-audit, meng-upload, menyimpan, mengkategori, dan pada akhirnya mempublish data seperti text (artikel), gambar, sampai dengan multimedia sesuai timeline yang diinginkan (Nurrosat, 2017).

Aplikasi *Content Management System instant* yang banyak terdapat di internet saat ini kebanyakan dibuat menggunakan *scripting language PHP* dan database-nya adalah *MySQL*. Saat ini perkembangan *Content Management System* cukup pesat, banyak vendor yang membuat CMS *instant* yang didistribusikan secara gratis. Perkembangan CMS *instant* ini juga dipicu oleh perkembangan web 2.0 yang memungkinkan interaksi dalam arti yang cukup luas antara pengelola web dan pengunjung web.

Selain perkembangan teknologi web dan infrastruktur internet, perkembangan pesat *Content Management System* juga dipicu oleh kebutuhan masyarakat dan pelaku bisnis yang menginginkan web dapat mendukung kegiatan bisnis mereka secara mudah dalam hal pengelolaan content, cepat dalam pembuatan web, serta murah dalam pengadaannya.

2.5 Algoritma *Jaro-Winkler Distance*

Jaro-Winkler merupakan varian dari metrik *Jaro Distance* biasanya digunakan dibidang keterkaitan rekaman (duplikat) dirancang dan paling sesuai untuk string pendek. Pada *Jaro-Winkler* untuk dua string semakin tinggi jarak, semakin mirip data yang diperoleh dengan skor 0(nol) sama dengan tidak ada persamaan dan 1(satu) sama persis. Dasar dari algoritma ini memiliki kriteria antara lain menghitung panjang string, menemukan jumlah karakter yang sama didalam dua string dan menemukan jumlah transposisi (Yulianingsih, 2017).

Algoritma *Jaro-Winkler distance* memiliki kompleksitas waktu *quadratic runtime complexity* yang sangat efektif pada string pendek dan dapat bekerja lebih cepat dari algoritma edit distance. Dasar dari algoritma ini memiliki tiga bagian:

1. Menghitung panjang string.
2. Menemukan jumlah karakter yang sama di dalam dua string.
3. Menemukan jumlah transposisi.

Pada algoritma *Jaro-Winkler* digunakan rumus untuk menghitung jarak (dj) antara dua string yaitu s1 dan s2 dapat dilihat pada persamaan (1)

$$d_j = \frac{1}{3} x \left(\frac{m}{s1} + \frac{m}{s2} + \frac{m-t}{m}\right) \quad \text{.....(1)}$$

dimana m adalah jumlah karakter yang sama persis, |s1| adalah panjang string1, |s2| adalah panjang String2, T adalah jumlah transposisi, dan dj adalah nilai jarak antara dua buah string yang dibandingkan.

Bila mengacu kepada nilai yang akan dihasilkan oleh algoritma *Jaro-Winkler* maka nilai jarak maksimalnya adalah 1 yang menandakan kesamaan string yang dibandingkan mencapai seratus persen atau sama persis. Biasanya s1 digunakan

sebagai acuan untuk urutan di dalam mencari transposisi. Yang dimaksud transposisi di sini adalah karakter yang sama dari string yang dibandingkan akan tetapi tertukar urutannya.. Sebagai contoh, dalam membandingkan kata CRATE dengan TRACE, bila dilihat seksama maka dapat dikatakan semua karakter yang ada di s1 ada dan sama dengan karakter yang ada di s2 , tetapi dengan urutan yang berbeda. Dengan mengganti C dan T, dapat dilihat perubahan kata CRATE menjadi TRACE.Pertukaran dua elemen string inilah adalah contoh nyata dari transposisi yang dijelaskan. Dalam pencocokkan DwAyNE dan DuANE memiliki urutan yang sama D-A-N-E, jadi tidak ada transposisi.

Jaro-Winkler distance menggunakan *prefix scale* (p) yang memberikan tingkat penilaian yang lebih, dan *prefix length* (l) yang menyatakan panjang awalan yaitu panjang karakter yang sama dari string yang dibandingkan sampai ditemukannya ketidaksamaan. Bila string s1 dan s2 yang diperbandingkan, maka Jaro-Winkler distancenya (dw) seperti pada persamaan (2)

$$\mathbf{dw = dj + (lp(1 - dj))} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2)$$

dimana dw adalah nilai *Jaro-Winkler Distance*, dj adalah *Jaro distance* untuk *strings* s1 dan s2, l adalah panjang prefiks umum di awal string nilai maksimalnya 4 karakter(panjang karakter yang sama sebelum ditemukan ketidaksamaan max 4), dan p adalah konstanta *scaling factor*.

Nilai standar untuk konstanta ini menurut Winkler adalah p = 0,1. Berikut ini adalah contoh pada perhitungan *Jaro Winkler distance*. Jika string s1 MARTHA dan s2 MARHTA maka:

m = 6, s1 = 6, s2 = 6

karakter yang tertukar hanyalah T dan H maka t = 1.

Maka nilai Jaro distancenya adalah:

dj= 1/3 (6/6 + 6/6 + (6-1)/6) = 0.944

Kemudian bila diperhatikan susunan s1 dan s2 dapat diketahui nilai l = 3, dan dengan nilai konstan p = 0.1. Maka nilai Jaro-Winkler distance adalah :

$dw = 0.944 + (3 \times 0.1\ (1 - 0.944)) = 0.961$

2.6 Akurasi

Akurasi adalah penghitungan dari perbandingan antara jumlah data dokumen yang relevan dan jumlah keseluruhan dokumen dalam database (Hasugian, 2016). Tujuan dilakukannya pengujian akurasi adalah untuk melihat seberapa persis anatara dua data yang ingin dibandingkan. Persamaan akurasi persamaan (3).

$$Akurasi = \frac{Jumlah\ Dokumen\ Relevan}{Jumlah\ keseluruhan\ Dokumen} \quad \text{.....(3)}$$

		Nilai Sebenarnya	
		TRUE	FALSE
Nilai Prediksi	TRUE	**TP**	**FP**
	FALSE	**FN**	**TN**

TP = True Positive (Correct Result)

FP = False Positive (Unexpected Result)

FN = False Negative (Missing Result)

TN = True Negative (Correct absence of result

Penjelasan :

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \qquad Precision = \frac{TP}{FP+TP}$$

Dengan hasil yang didapatkan sebesar 99.5% maka dapat disimpulkan bahwa keakurasian data yang dicari sudah tepat, dikarenakan ketika presentase hasil akurasi melebihi 95%, data dapat dibilang sesuai (Hasugian, 2016).

2.7 Fitur *Autocorrect* dan *Spelling Sugestion*

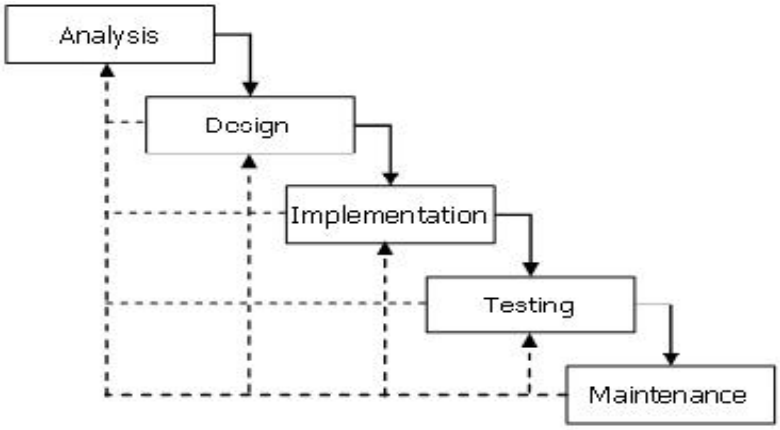
Autocorrect adalah suatu fitur yang dapat memeriksa dan memperbaiki kesalahan penulisan kata secara otomatis. Menurut Gueddah dkk (2015), pemeriksaan ejaan terdiri dari perbandingan antara kata yang salah dengan daftar kata pada basis data dan menyarankan kata-kata yang mirip dengan kata yang salah dengan menghitung kemiripan jarak antara kata-kata tersebut (Gueddah dkk, 2015).

Spelling Corrector atau *Spelling Checker* atau *Spelling Suggestion*. Fitur ini berfungsi sebagai pendeteksi kesalahan dan memberikan panduan bagi penggunanya dengan mendanai kata-kata yang tidak terdaftar dalam kamus suatu bahasa tertentu. Fitur ini juga disertaidengan sugesti kata yang berfungsi menyediakan rekomendasi kata-kata yang mendekati kata yang dimaksud (Mutammimah dkk, 2017). Pemeriksaan ejaan terdiri dari perbandingan antara kata yang salah dengan daftar kata pada basis data dan menyarankan kata-kata yang mirip dengan kata yang salah dengan menghitung kemiripan jarak antara kata-kata tersebut (Gueddah dkk, 2015).

2.8 Model Pengembangan *Waterfall*

Model *Waterfall* merupakan salah satu model pengembangan perangkat lunak yang ada di dalam model SDLC (*Sequencial Development Life Cycle*). Menurut Sukamto dan Shalahuddin (2016) mengemukakan bahwa "SDLC atau *Software Development Life Cycle* atau sering disebut juga *System Development Life Cycle* adalah proses mengembangkan atau mengubah suatu sistem perangkat lunak dengan menggunakan model-model dan metodologi yang digunakan orang untuk mengembangkan sistem-sistem perangkat lunak sebelumnya, berdasarkan *best practice* atau cara-cara yang sudah teruji baik."

Sedangkan menurut Bassil (2011) di jelaskan bahwa model *waterfall* sering juga disebut model sekuensi linear atau alur hidup klasik. Pengembangan sistem dikerjakan secara terurut mulai dari analisis, desain, pengkodean, pengujian dan tahap pendukung.



Gambar 2.1 Metode *Waterfall* (Bassil, 2011)

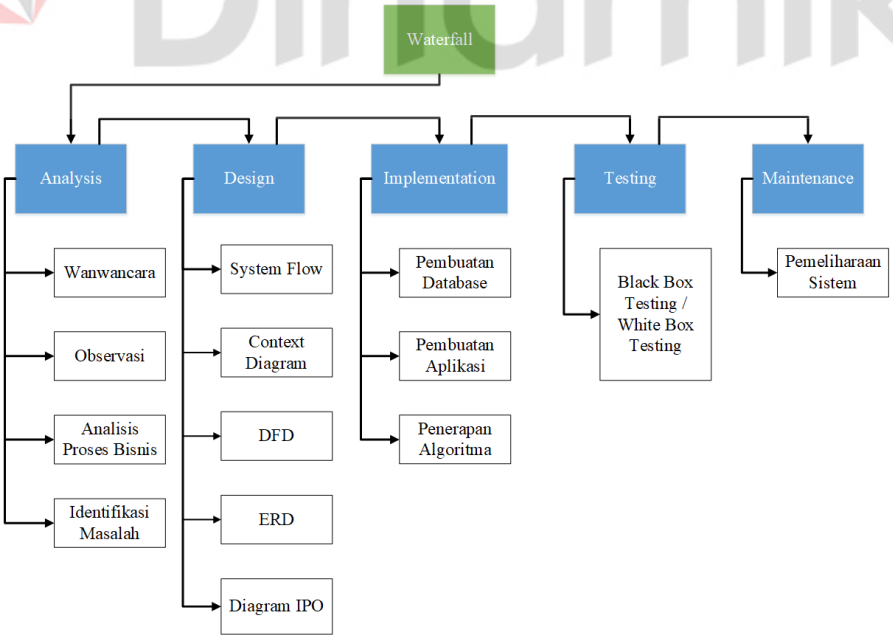
2.9 Black Box Testing

Blackbox testing adalah tahap yang digunakan untuk menguji kelancaran program. Pengujian ini menggunakan deskripsi eksternal perangkat lunak termasuk spesifikasi, persyaratan dan desain untuk menurunkan uji kasus (pengujian). Pengujian ini penting dilakukan agar tidak terjadi kesalahan alur program yang telah dibuat (Dhega, 2018).

BAB III
METODOLOGI PENELITIAN

Untuk pengembangan sistem penelitian ini menggunakan model SDLC (*Software Development Life Cycle*). *System Development Life Cycle* (SDLC) adalah proses pembuatan dan pengubahan sistem serta model dan metodologi yang digunakan untuk mengembangkan sebuah sistem. SDLC juga merupakan pola yang diambil untuk mengembangkan sistem perangkat lunak, yang terdiri dari tahap-tahap: rencana (*planning*), analisis (*analysis*), desain (*design*), implementasi (*implementation*), uji coba (*testing*) dan pengelolaan (*maintenance*).

Model SDLC yang dipakai dalam penelitian ini adalah model Waterfall. *Waterfall Model* atau *Classic Life Cicle* merupakan model yang paling banya dipakai dalam *Software Enginnering* (Bassil, 2017). Disebut *waterfall* karena tahap demi tahap yang harus dilalui menunggu selesainya tahap sebelumnya dan berjalan berurutan.



Gambar 3.1 Tahap Pengerjaan dengan Metode *Waterfal* (Bassil, 2017)

3.1 *Analysis*

Tahap *analysis* adalah tahap awal dimana pengembang sistem diperlukan suatu komunikasi yang bertujuan untuk memahami *software* yang dibutuhkan pengguna dan batasan *software*. Informasi ini biasanya dapat diperoleh melalui wawancara, *survey* (observasi) atau diskusi.

3.1.1 Wawancara

Proses wawancara dilakukan dengan ASM Pengembang dan Solusi TIK Back-End PT. Bukaloka Teknologi Indonesia pada 2 May 2020 pukul 11.40 WIB ~ 12.25 WIB via Whatapps Call, dikarenakan tidak mendukungnya kondisi dan permasalahan jarak kantor PT. Bukaloka Teknologi Indonesia. Selama proses berlangsung ASM Pengembang dan Solusi TIK Back-End PT. Bukaloka Teknologi Indonesia memeberikan penjelasan untuk mengetahui tentang bagaimana jalannya proses bisnis yang saat ini masih berjalan di dalam PT. Bukaloka Teknologi Indonesia, dan rencana pengembangan *Content Management System* (CMS) menggunakan *Speech Recognition.*

3.1.2 Observasi

Proses observasi perusahaan dilakukan pada 2 May 2020 pukul 11.40 WIB ~ 12.25 WIB via Whatapps Call. Observasi dilakukan ketika proses wawancara dengan ASM Pengembang dan Solusi TIK Back-End PT. Bukaloka Teknologi Indonesia dan Dilanjutkan dengan proses observasi beberapa kebutuhan data, untuk mencari berbagai kebutuhan data yang akan diperlukan pada saat pembuatan aplikasi dimulai seperti, pencarian nilai standar minimal dan maksimal pada proses *autocorrect & spelling suggestion*. Pengujian yang dilakukan adalah dengan meninjau dari 18 sample kata yang akan dicari nilai *Jaro-Winkler* yang dapat dilihat pada gambar 3.2.

Input	Target (tidur)	Nilai		Input	targer (sempat)	Nilai		Input	Target(dari)	nilai
tidyr	tidur	0.906		sempal	sempat	0.911		dary	dari	0.805
tidyr	tiru	0.826		sempal	sempol	0.893		dary	daba	0.688
tidyr	tiroid	0.824		sempal	semadi	0.669		dary	dada	0.727
tidyr	tidak	0.813		sempal	semen	0.556		dary	dadu	0.688
tidyr	tadir	0.805		sempal	semata	0.7		dary	daga	0.727
tidyr	timur	0.786		sempal	semula	0.782		dary	dana	0.688

Gambar 3.2 Data Sample Algoritma *Jaro-Winkler*

Berdasarkan hasil yang diberikan oleh algoritma *Jaro-Winkler* dari 18 sample kata yang dilakukan uji coba pada gambar 3, maka ditetapkan Nilai Standar Maksimal yang digunakan sebesar 0.950, mengingat hasil yang diberikan dari pengolahan kata "tidyr" menjadi "tidur" dan "sempal" menjadi "sempat" berada didalam range 0.9++. Nilai Standar Minimal yang digunakan sebesar 0.780, mengingat pengolahan dari kata "dary" menjadi "dari" mempunyai hasil 0.805 meskipun hanya berbeda 1 huruf,

3.1.3 Analisis Proses Bisnis

Proses bisnis pada PT. Bukaloka Teknologi Indonesia dimulai dengan ketika adanya UMKM yang ingin bergabung dengan *marketplace* PT. Bukaloka, dengan mendaftar melalui form yang dapat diakses online atau datang ke kantor secara langsung. yang dimana setelah pendaftaran selesai akan mendapatkan layanan pembuatan *Company Profile Online* untuk usaha UMKM tersebut. Proses pembuatan *Company Profile*-nya sendiri menggunakan *Content Management System* (CMS) yang sudah tersedia.

Staff IT PT. Bukaloka akan meng-edukasi UMKM yang bersangkutan dengan dasar dasar dari cara penggunaan, pembuatan konten, dan pengelolaan untuk pertama kalinya, sehingga UMKM yang bersangkutan dapat paham dan mudah menggunakan menggunakan *Content Management System* (CMS) standar yang sudah tersedia, sering kali juga ada keluhan yang sering dikatakan pelanggan yang dimana ketika proses input data adanya kata-kata yang typo dan proses pengetikan untuk data yang banyak membutuhkan waktu yang lama. Setelah proses edukasi selesai dan kerangka *Company Profile* sudah bisa dibilang

jadi maka, *Company Profile* akan seluruhnya diserahkan kepada pihak UMKM.

Proses pengelolaan data berserta konten *Company Profile* biasanya dilakukan UMKM bersangkutan, tetapi tidak jarang juga beberapa UMKM yang bergabung masih sangat awam atau tidak terlalu mengerti dengan penggunaan *Content Management System* (CMS) yang mengakibatkan timbulnya kebingungan penggunanya, biasanya pihak UMKM dapat meminta bantuan dari Staff IT PT. Bokaloka untuk membantu menjelaskannya lebih lanjut. Berikut merupakan document flow dari penjelasan diatas dapat dilihat pada Lampiran 1.

3.1.4 Identifikasi Masalah

Berdasarkan proses bisnis yang ada pada PT. Bukaloka Teknologi Indonesia tersebut, maka dapat dilakukan identifikasi permasalahan. Identifikasi permasalahan mencakup permasalahan yang ada, dampak dari permasalahan tersebut, dan solusi yang diusulkan. Hasil identifikasi tersebut dapat dilihat pada tabel 3.1.

Tabel 3.1 Identifikasi Masalah

NO	Permasalahan	Dampak	Solusi
1	Pada proses input data adanya kata-kata yang typo	Kata-kata typo dapat menyebabkan pembaca misunderstanding dan miscommunication.	Membuat sistem yang mempunyai fitur *autocorrect* dan *spelling suggestion* untuk mengurangi potensi kesalan kata
2	Proses pengetikan untuk data yang banyak membutuhkan waktu yang lama	Penyampaian Informasi dari pihak UMKM ke pelanggan menjadi lama	Membuat sistem yang dapat memasukan data tanpa menggunakan

			perangkat input seperti keyboard

3.1.5 Analisis Kebutuhan Pengguna

Analisis kebutuhan pengguna dilakukan dengan tujuan untuk mengetahui data dan informasi yang digunakan dan/atau dibutuhkan oleh pengguna sistem (perangkat lunak) yang akan dibuat. Selain itu juga untuk menganalisis *output* yang diproleh dari pengguna tersebut. Berdasarkan hasil analisis proses bisnis, pengguna dari sistem (perangkat lunak) yang akan dibuat adalah: Staff IT UMKM, Owner UMKM, Staff IT PT. Bukaloka.

1. Staff IT UMKM

Tabel 3.2 Analisis Kebutuhan Pengguna Staff IT UMKM

Kebutuhan Fungsi	Kebutuhan Data	Kebutuhan Informasi
Posting Konten & Produk menggunakan *Speech Recognition*	- Data Produk - Data About Us - Data Event - Data Testimoni - Data Perusahaan	- Daftar Produk - Infromasi harga - Informasi perusahaan - Informasi kegiatan - Daftar tesimoni (ulasan) pelanggan
Validasi Data	- Data Perubahan Kata	- Informasi perubahan kata
Menyimpan Data	- Data Valid	- Daftar kata valid

2. Owner UMKM

Tabel 3.3 Analisis Kebutuhan Pengguna *Owner* UMKM

Kebutuhan Fungsi	Kebutuhan Data	Kebutuhan Informasi
Posting Konten & Produk menggunakan *Speech Recognition*	- Data Produk - Data About Us - Data Event - Data Testimoni - Data Perusahaan	- Daftar Produk - Infromasi harga - Informasi perusahaan - Informasi kegiatan - Daftar tesimoni (ulasan)pelanggan
Validasi Data	- Data Perubahan Kata	- Informasi perubahan kata
Menyimpan Data	- Data Valid	- Daftar kata valid

3. Staff IT PT. Bukaloka

Tabel 3.4 Analisis Kebutuhan Pengguna Staff IT PT. Bukaloka

Kebutuhan Fungsi	Kebutuhan Data	Kebutuhan Informasi
Panduan posting	- Data Produk - Data About Us - Data Event - Data Perusahaan - Data Testimoni	- Daftar Produk - Informasi Perusahaan - Daftar Testimoni pelanggan - Daftar Kata Dasar

3.2 *Design*

Tahap *design* adalah proses yang digunakan untuk mengubah kebutuhan-kebutuhan diatas menjadi representasi ke dalam bentuk "*blueprint*" *software* sebelum coding dimulai. Desain dapat mengimplementasikan kebutuhan yang telah disebutkan pada tahap sebelumnya.

3.2.1 *System Flowchart*

Prosedur atau alur proses dalam Penerapan Algoritma *Jaro-Winkler* Untuk *Autocorrect* Dan *Spelling Suggestion* Pada Aplikasi *Speech Recognition CMS* Berbasis *Website* digambarkan dalam *system flow* yang terdiri dari dua proses diantaranya adalah sebagai berikut:

1. ***System Flow* Proses *Input* dengan *Speech Recognition* serta *Autocorrect & Spelling Suggestion***

   Adapun prosedur dari proses *input* dengan *speech recognition* serta *autocorrect & spelling suggestion* adalah sebagai berikut:

   a. User menginputkan data lisan (berbicara).

   b. Sistem dengan teknologi *speech recognition* yang menggunakan API google akan menggunakan data lisan yang diinputkan lalu diolah menjadi bentuk *data text*.

   c. Data text yang didapatkan akan memasuki proses pengakuratan data dengan membandingkan antara *data text* (standar) yang berasal dari *database* sistem dan *data text* (input) yang berasal dari pengolahan *speech recognition* menggunakan algoritma *jaro-winkler*.

   d. Data text yang dibandingkan akan menggunakan algoritma *jaro-winkler* akan menghasilkan nilai bobot.

   e. Nilai bobot akan diperiksa oleh sistem jika nilai bobot berada diantara nilai bobot yang sudah ditentukan pada sistem, maka *data text* akan ke tahap *autocorrect & spelling suggestion* yang dimana *data text* (standar) akan digunakan untuk menggantikan data text (*input*).

System flow proses *input* dengan *speech recognition* serta *autocorrect & spelling suggestion* ditunjukan pada Lampiran 2.

2. *System Flow Subprosess* Pencarian Nilai *Jaro-Winkler*

Adapun prosedur dari subprosess pencarian nilai *jaro-winkler* adalah sebagai berikut:

a. Proses pencarian dimulai dengan membandingkan antara data text (standar) yang berasal dari *database* sistem dan data text (*input*) yang berasal dari pengolahan *speech recognition* untuk mencari nilai *Jaro-Distance* (*dj*).

b. Sebelum Nilai *Jaro-Distance (dj)* memasuki proses selanjutnya akan dilakukan proses pengecekan keakuratan kata, proses ini akan menghasilkan nilai akurasi yang akan dibandingkan dengan nilai akurasi yang sudah ditetapkan, nilai akurasi yang melebihi atau sama dengan batas (95%) maka Nilai *Jaro-Distance (dj)* yang didapat pada proses sebelumnya akan memasuki proses berikutnya, jika tidak maka proses akan diulang dari tahap 1 kembali.

c. Nilai *Jaro-Distance* (*dj*) yang didapatkan selanjutkan akan digunakan untuk penghitungan mencari nilai *Jaro-Winkler Distance* (*dw*) yang diikuti juga dengan data text (standar) yang berasal dari *database* sistem dan data text (*input*) yang berasal dari pengolahan *speech recognition* sekali lagi.

d. Setelah perhitungan selesai maka akan mendapatkan nilai *Jaro-Winkler Distance* (*dw*) yang berfungsi sebagai nilai bobot untuk proses *autocorrect & spelling suggestion.*

System flow subprosess pencarian nilai *jaro-winkler* ditunjukan pada Lampiran 2.

3.2.2 *Context Diagram*

Context Diagram merupakan *level* tertinggi dari *Data Flow Diagram* yang menggambarkan seluruh *input* ke dalam sistem atau *output* dari sistem yang memberi gambaran tentang keseluruhan sistem. *Context diagram* dari Penerapan Algoritma *Jaro-Winkler* untuk *Autocorrect* dan *Spelling Suggestion* pada Aplikasi *Speech Recognition CMS* berbasis Website ini mempunyai tiga entitas pelaku diantaranya adalah Staff IT UMKM, Owner UMKM, dan Staff IT PT. Bukaloka yang dapat dilihat pada Lampiran 3.

3.2.3 *Hirarchy Input Proses Output* (HIPO)

Diagram HIPO memberikan gambaran proses dan sub-proses dari Penerapan Algoritma *Jaro-Winkler* untuk *Autocorrect* dan *Spelling Suggestion* pada Aplikasi *Speech Recognition CMS* berbasis Website. Pada aplikasi ini terdapat empat proses utama yaitu proses *Speech Recognition*, Algoritma *Jaro-Winkler*, *Autocorrect & Spelling Suggestion*, dan Validasi Data. Diagram HIPO dapat dilihat pada Lampiran 4.

3.2.4 *Data Flow Diagram*

*Data Flow Diagram (*DFD*)* adalah pengembangan dari *context diagram*. Pada DFD terdapat lima proses utama yaitu proses posting konten menggunakan *speech recognition*, perhitungan algoritma jaro-winkler (*dj*), perhitungan algoritma jaro-winkler (*dw*), *autocorrect & spelling suggestion*, dan validasi. Terdapat tiga entitas yang terlibat yaitu Staff IT UMKM, Owner, dan Staff IT PT.Bukaloka. Semua Entitas dapat melakukan posting konten menggunakan *speech recognition*, tetapi terbatas pada Staff IT PT.Bukaloka proses posting konten menggunakan *speech recognition* hanya dimaksudkan untuk panduan saja. *Data flow diagram* mulai dari level 0 sampai dengan level 1 dari Penerapan Algoritma *Jaro-Winkler* untuk *Autocorrect* dan

Spelling Suggestion pada Aplikasi *Speech Recognition CMS* berbasis Website dapat dilihat pada Lampiran 5.

3.2.5 *Entity Relationship Diagram* (ERD)

Entity Relationship Diagram (ERD) merupakan suatu model untuk menjelaskan hubungan antar data dalam basis data berdasarkan objek-objek dasar data yang mempunyai hubungan antar relasi. ERD untuk memodelkan struktur data dan hubungan antar data, untuk menggambarkannya digunakan beberapa notasi dan *symbol*. ERD disajikan dalam bentuk *Conceptual Data Model* (CDM) dan *Physical Data Model* (PDM).

1. *Conseptual Data Model* (CDM)

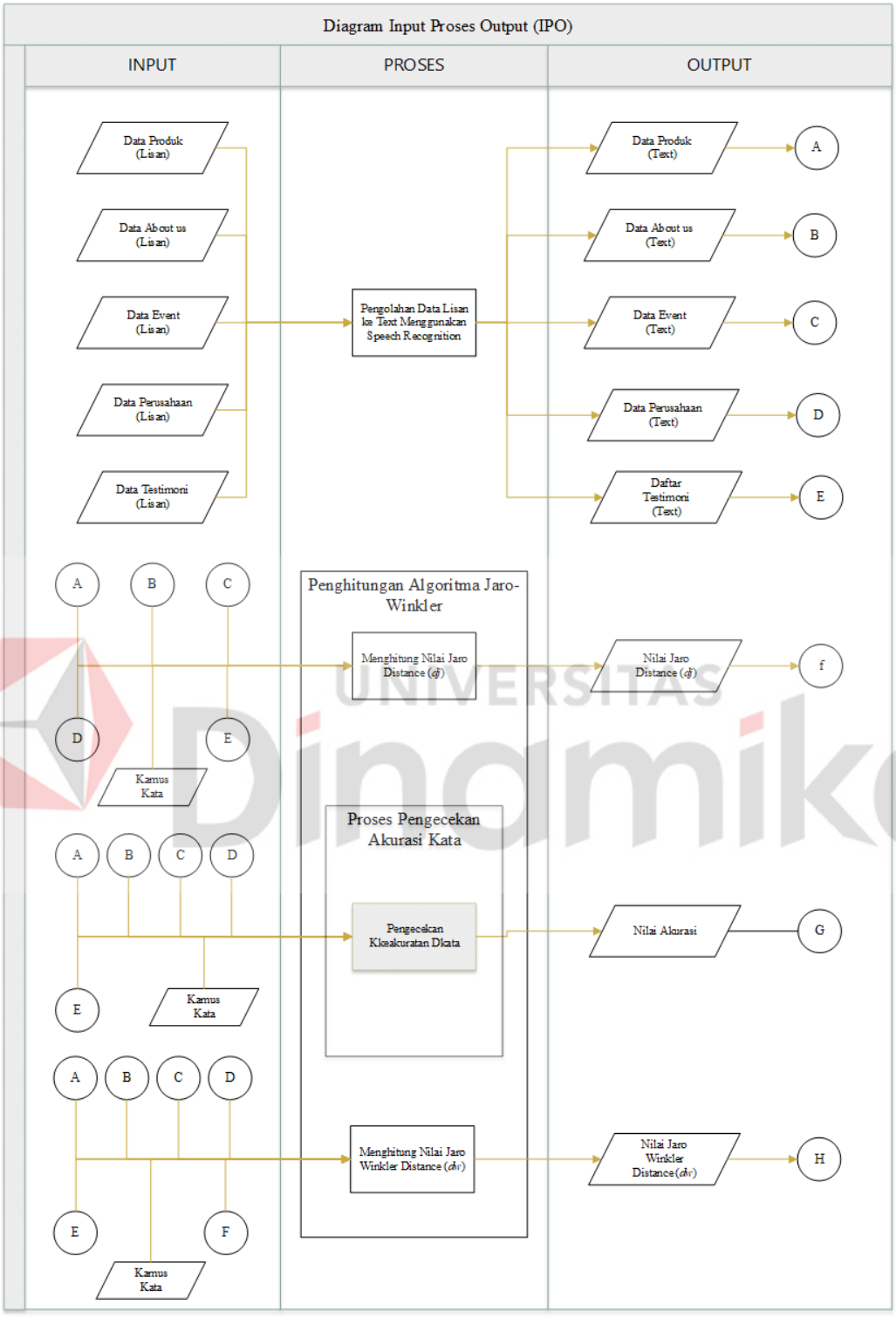
Desain *database* dari Penerapan Algoritma *Jaro-Winkler* untuk *Autocorrect* dan *Spelling Suggestion* pada Aplikasi *Speech Recognition CMS* berbasis Website ini disajikan dalam bentuk model logika yang digambarkan melalui *Conseptual Data Model* (CDM), yang berfungsi untuk melakukan identifikasi entitas, atribut dan relasi antar entitas. Untuk memberikan gambaran yang lebih jelas mengenai keseluruhan entitas yang dapat dilihaat pada Lampiran 6.

2. *Physical Data Model* (PDM)

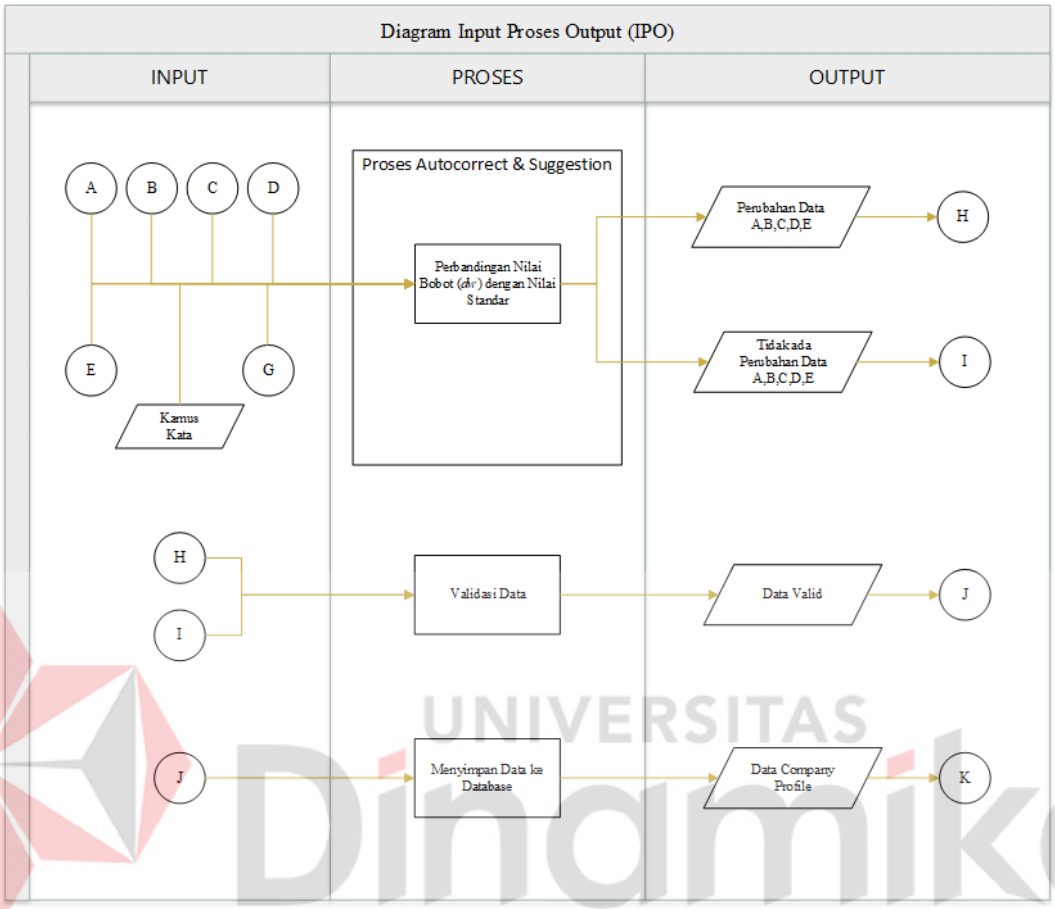
Physical Data Model (PDM) berguna untuk menggambarkan struktur antara table-tabel yang saling berhubungan yang akan diterapkan pada *Database Management System* (DBMS). PDM dapat dihasilkan dari generate CDM, gambar dari PDM dapat dilihat pada Lampiran 7.

3.2.6 Diagram Input Proses Output (IPO)

Pada proses *design* selanjutnya akan dilakukan perancangan sistem yang menggunakan pemodelan sistem IPO Diagram untuk menggambarkan kebutuhan input, proses, dan output dari setiap modul. IPO diagram dapat dilihat pada Gambar 3.7 dan 3.8.



Gambar 3.3 *Input Process Output part* 1



Gambar 3.4 *Input Process Output part* 2

Penjelasan IPO Diagram :

1. *Input*

Input merupakan tahap memasukan data untuk diolah menjadi data *output.* Berikut merupakan *inputan* yang digunakan, penjelasan dapat dilihat pada tabel 3.5.

Tabel 3.5 Penjelasan *Input* pada IPO Diagram

Data Produk (lisan)	Data produk (lisan) merupakan informasi yang dimaksudkan akan muncul pada daftar produk yang masih dalam bentuk suara yang akan dimasukkan oleh user menggunakan teknologi *speech recognition*
Data About us (lisan)	Data about us (lisan) merupakan informasi yang dimaksudkan akan muncul pada halaman about us yang masih dalam bentuk suara yang akan dimasukkan oleh user menggunakan teknologi *speech recognition*
Data Event (lisan)	Data event (lisan) merupakan informasi yang dimaksudkan akan muncul pada halaman event yang masih dalam bentuk suara yang akan dimasukkan oleh user menggunakan teknologi *speech recognition*
Data Perusahaan (lisan)	Data produk (lisan) merupakan informasi tentang perusahaan yang dimaksudkan akan muncul pada company profile perusahaan yang masih dalam bentuk suara yang akan dimasukkan oleh user menggunakan teknologi *speech recognition*
Data Testimoni (lisan)	Data testimoni (lisan) merupakan informasi yang dimaksudkan akan muncul pada daftar ulasan yang masih dalam bentuk suara yang akan dimasukkan oleh pengulas menggunakan teknologi *speech recognition*

Kamus Kata	Kamus kata merupakan kumpulan kata yang akan menjadi acuan dalam proses pembenahan kata yang dilakukan dalam algoritma *Jaro-Winkler*

2. *Proses*

Proses merupakan tahap mengolah data *input* menjadi data *output.* Berikut merupakan proses yang digunakan, penjelasan dapat dilihat pada tabel 3.6.

Tabel 3.6 Penjelasan Proses pada IPO Diagram

Mengelola data lisan ke text menggunakan *Speech Recognition*	Proses pengolahan/pengubahan dari data berbentuk lisan(suara) menjadi data text dengan *Speech Recognition*
Menghitung nilai Jaro Distance (*dj*)	Proses pencarian nilai *dj* dalam algoritma perhitungan *Jaro-Winkler*
Menghitung nilai Jaro-Winkler Distance (*dw*)	Proses pencarian nilai *dw* dalam algoritma prohitungan *Jaro-Winkler* yang akan menjadi nilai bobot
Perbandingan nilai bobot (*dw*) dengan nilai standar	Proses membandingan nilai bobot dan nilai standar yang sudah ditetapkan untuk memutuskan perubahan kata
Pengecekan Akurasi Kata	Proses yang dilakukan untuk melihat presentase kesamaan kata yang di olah oleh Algoritma *Jaro-Winkler Distance* sebelum menghasilkan nilai bobot/*dw*
Validasi Data	Mengkonfirmasi data yang akan digunakan sebelum proses penyimpanan
Menyimpan data ke *database*	Proses penyimapanan data yang telah selesai diproses.

3. *Output* :

Output merupakan tahap akhir dimana data sudah siap digunakan. Berikut merupakan *output* yang digunakan, penjelasan dapat dilihat pada tabel 3.7.

Tabel 3.7 Penjelasan *Output* pada IPO Diagram

Data company profile	Berisikan tentang informasi dari data lisan yang sudah dijadikan text dan telah melewati proses algoritma.

3.3 *Implementation*

Pada tahap ini, peneliti membangun sebuah aplikasi berdasarkan desain *system flowchart* dan IPO diagram yang telah dibuat. Proses akan terbagi menjadi 3 bagian dimana pertama-tama dilakukannya pembuatan *database* lalu pembuatan aplikasi secara plain lalu akan dilanjutkan dengan penerapan algoritma yang digunakan.

3.3.1 Pembuatan *Database*

Proses pembuatan *database* pada aplikasi ini menggunakan *software* PhpMyAdmin dengan range operasi berbasis MySQL dan MariaDB. Berdasarkan rancangan dari CDM dan PDM pembuatan database pada aplikasi ini bertotal 1 database "***db_admin***", dengan 8 tabel master yang digunakan untuk menyimpan data dari content-content yang akan ditampilkan pada company profile seperti "***about_us***", "***barang***", "***contact***", "***event***", "***event_content***", "***sale***", "***slider***",dan "***testimoni***". 2 tabel konfigurasi yang digunakan untuk mengatur halaman CMS seperti "***userconfig***" dan "***notifikasi***". Serta 1 tabel kamus yang berguna untuk menyimpan seluruh data kata dasar "***tb_katadasar***".

3.3.2 Pembuatan Aplikasi

Proses pembuatan aplikasi akan menyangkut seperti proses *coding back-end* dan *front-end* aplikasi tersebut. Pembuatan berbagai macam fungsi dan fitur-fitur yang akan digunakan oleh user di dalam aplikasi.

3.3.3 Penerapan Algoritma

Pada proses ini Algoritma *Jaro-Winkler* akan diimplementasikan kedalam sistem *speech recognition* yang digunakan untuk meningkatkan akurasi kata yang dihasilkan oleh *speech recognition*, dibantu dengan "tb_katadasar" sebagai kumpulan kata-kata dasar yang akan digunakan sebagai acuan dari kata yang ingin diperbaiki.

3.4 *Testing*

Setelah proses implementasi selesai, peneliti melakukan pengujian pada tahap ini. Aplikasi dapat diuji berdasarkan 2 metode yaitu *black box* atau *white box* tetapi peneliti hanya akan berfokus pada penggunaan metode *black box* untuk mengetahui tingkat keberhasilan dari bagian sistem.

Selain penggunaan *black box* testing yang digunakan untuk pengujian sistem, Survei kepada calon pengguna juga dilakukan untuk mengetahui kelayakan aplikasi menurut calon pengguna, survei akan dilakukan kepada 30 calon pengguna yang dimana akan diwawancarai dengan menggunakan *google form*. Hasil survei dapat dilihat pada Lampiran 7.

3.5 *Maintenance*

Rencana peneliti akan melakukan beberapa perbaikan tidak pada semua tahapan, namun hanya pada tahapan sebelum terjadi *error*. Sehingga peneliti tidak akan dipusingkan dengan melakukan tahapan dari awal hingga akhir kembali.

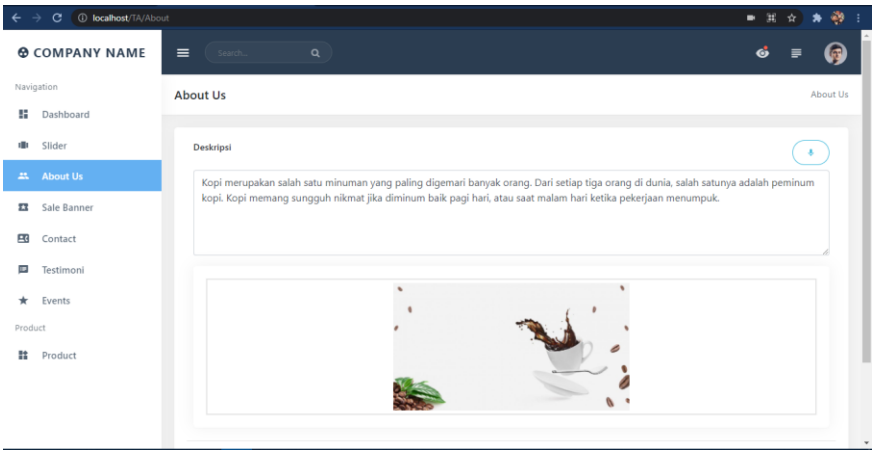
BAB IV
HASIL DAN PEMBAHASAN

4.1 Hasil Implementasi

Berikut merupakan hasil implementasi dari Penerapan Algoritma *Jaro-Winkler* Untuk *Autocorrect* dan *Spelling Suggestion* Pada Aplikasi *Speech Recognition CMS* Berbasis Website. Detail hasil implementasi dapat dilihat pada Lampiran 8.

4.1.1 Penerapan *Speech Recognition* di *Dashboard* CMS

Penerapan algoritma pada *Speech Recognition* di Dashboad CMS ini berlokasi pada seluruh halaman *dashboard* yang biasa digunakan oleh admin untuk mengubah informasi *company profile*. Penambahan fitur *speech recognition* yang disertai algortima bertujuan untuk menghindari proses pengetikan data yang banyak dengan hasil *output* text yang akurat. Pada tahap ini sama seperti proses *input/edit* data pada CMS tradisional pengguna diharuskan untuk login agar dapat memasuki halaman dashboard, lalu pengguna dapat memilih menu yang akan di input/edit datanya. Tampilan *Speech Recognition* pada *Dashboard* CMS dapat dilihat pada Gambar 4.1.

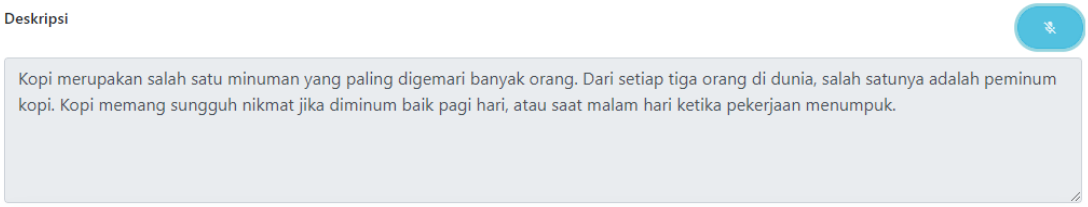


Gambar 4.1 Tampilan *Speech Recognition* pada *Dashboard* CMS

Penggunaan fitur *speech recogniton* ini tidak bersifat *default* melainkan optional, yang berarti fitur ini dapat diaktifkan jika ingin menggunakan fitur dan dapat dinonaktifkan jika tidak ingin menggunakan fitur ini. Proses pengaktifan *speech recognition* dilakukan dengan menekan tombol dengan *icon microphone* yang berada pada pojok kanan atas input teks. Tampilan dapat dilihat pada Gambar 4.2.

Gambar 4.2 *Speech Recognition* Tidak Aktif

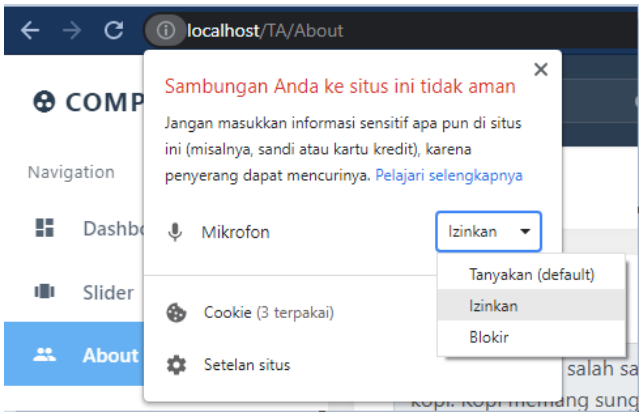
Icon microphone berubah menjadi tercentang dan input teks field akan berubah menjadi *disabled*, Tampilan dapat dilihat pada Gambar 4.3. Setelah *speech recognition* aktif maka penginputan data melalui suara dapat dilakukan.



Gambar 4.3 *Speech Recognition* Aktif

Perlu diperhatikan untuk memastikan bahwa *browser* yang digunakan harus sudah memberikan izin pada aplikasi untuk dapat mengakses *microphone* sebelum fitur digunakan, sehingga *speech recognition* dapat menangkap suara. Langkah paling mudah untuk mendapatkan izin pada *browser* Google Chrome adalah dengan menekan icon yang terdapat pada samping kiri URL aplikasi, lalu ubah izin mikrofon

dari pengaturan "Tanyakan(*default*) / *Ask(default)*" menjadi "Izinkan / *Allow*", tampilan dapat dilihat pada Gambar 4.4.

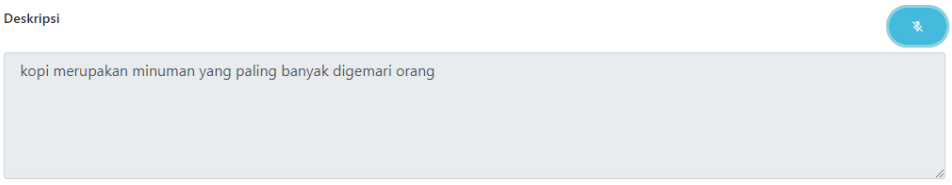


Gambar 4.4 Pemberian izin akses mikrofon

Dalam proses pengolahan data suara menjadi teks setelah data ditangkap oleh *Speech Recognition* diperlukan beberapa tahapan seperti *Natural Language Processing* dan Algoritma *Jaro-Winkler Distance*. Berikut merupakan contoh beserta tahapan prosesnya.

Contoh penggunaan *Speech Recognition* :

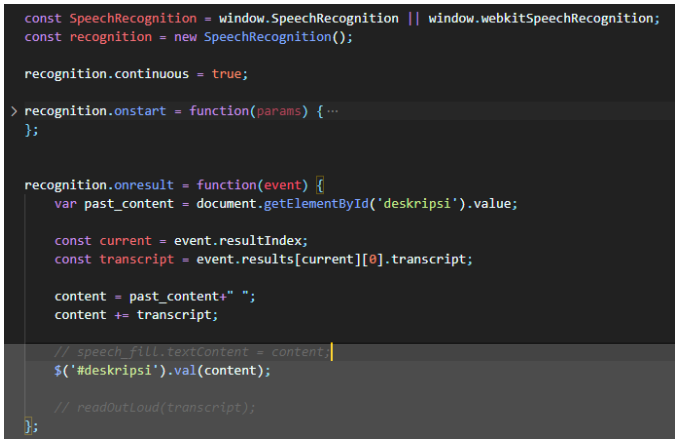
Sampel kata yang akan digunakan dalam uji coba adalah "**kopi merupakan minuman yang paling banyak digemari orang**" kata uji coba dapat dilihat pada Gambar 4.5.



Gambar 4.5 Uji coba sampel kata

1. Tahapan *Natural Language Processing*

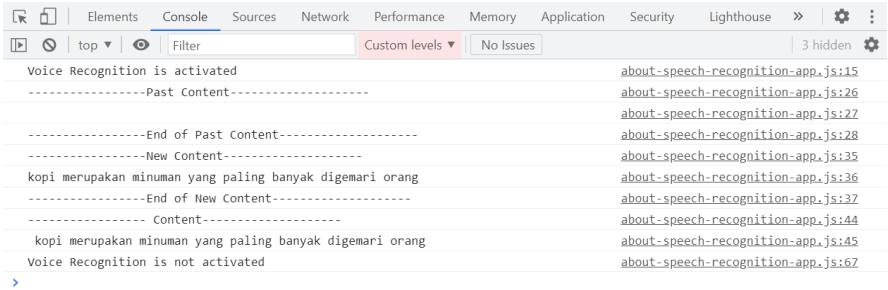
Sebelum dapat mengeluarkan output data berupa teks, sistem akan memproses terlebih dahulu data suara yang ditangkap oleh *speech recognition* lalu sistem mengekstrak gelombang suara lebih jauh dan merepresentasi data tersebut dari suatu teks bebas. Untuk dapat mengubah data suara menjadi data teks, dapat menggunakan kode seperti pada Gambar 4.6.



```
const SpeechRecognition = window.SpeechRecognition || window.webkitSpeechRecognition;
const recognition = new SpeechRecognition();

recognition.continuous = true;

recognition.onstart = function(params) {...
};


recognition.onresult = function(event) {
    var past_content = document.getElementById('deskripsi').value;

    const current = event.resultIndex;
    const transcript = event.results[current][0].transcript;

    content = past_content+" ";
    content += transcript;

    // speech_fill.textContent = content;
    $('#deskripsi').val(content);

    // readOutLoud(transcript);
};
```

Gambar 4.6 Kode mengubah suara menjadi teks

Setelah data diproses oleh sistem data teks akan terbentuk yang dimana merupakan hasil dari *Natural Language Processing* yang ditangkap oleh *Speech Recognition.* Hasil pemrosesan data dapat dilihat pada Gambar 4.7.



```
Elements  Console  Sources  Network  Performance  Memory  Application  Security  Lighthouse
top ▾  Filter  Custom levels ▾  No Issues  3 hidden
Voice Recognition is activated                                   about-speech-recognition-app.js:15
----------Past Content----------                                about-speech-recognition-app.js:26
                                                                 about-speech-recognition-app.js:27
----------End of Past Content----------                         about-speech-recognition-app.js:28
----------New Content----------                                 about-speech-recognition-app.js:35
kopi merupakan minuman yang paling banyak digemari orang         about-speech-recognition-app.js:36
----------End of New Content----------                          about-speech-recognition-app.js:37
---------- Content----------                                    about-speech-recognition-app.js:44
 kopi merupakan minuman yang paling banyak digemari orang        about-speech-recognition-app.js:45
Voice Recognition is not activated                               about-speech-recognition-app.js:67
>
```

Gambar 4.7 Hasil pemrosesan suara menjadi teks.

2. Tahapan Algoritma *Jaro-Winkler Distance*

Pada tahap ini data teks yang sudah didapatkan akan melalui pengecekan kata menggunakan Algoritma *Jaro-Winkler Distance*. Dalam tahap ini terdapat 2 proses yaitu pencarian nilai "*dj"* (*jaro-distance*). Pencarian nilai "*dj"* dapat menggunakan kode seperti pada Gambar 4.8.

```
function Jaro( $string1, $string2 ){

  $str1_len = strlen( $string1 );
  $str2_len = strlen( $string2 );

  // theoretical distance
  $distance = (int) floor(min( $str1_len, $str2_len ) / 2.0);

  // get common characters
  $commons1 = getCommonCharacters( $string1, $string2, $distance );
  $commons2 = getCommonCharacters( $string2, $string1, $distance );

  if( ($commons1_len = strlen( $commons1 )) == 0) return 0;
  if( ($commons2_len = strlen( $commons2 )) == 0) return 0;

  // calculate transpositions
  $transpositions = 0;
  $upperBound = min( $commons1_len, $commons2_len );
  for( $i = 0; $i < $upperBound; $i++){
    if( $commons1[$i] != $commons2[$i] ) $transpositions++;
  }
  $transpositions /= 2.0;

  // return the Jaro distance
  return ($commons1_len/($str1_len) + $commons2_len/($str2_len) + ($commons1_len - $transpositions)/($commons1_len)) / 3.0;
}
```

Gambar 4.8 Kode pencarian nilai *dj*

Lalu dilanjutkan dengan pencarian nilai "*dw"* yang dimana nilai "*dj"* yang didapatkan sebelumnya akan kembali dihitung kembali dengan nominasi kata dasar untuk menghasilkan nilai "*dw"*, semakin dekat nilai "*dw*" dengan nilai 1 maka dapat dikatakan kedua kata adalah serupa. Nilai "*dw*" ini juga berfungsi sebagai nilai bobot dari kata yang di cek. Pencarian nilai "*dw"* dapat menggunakan kode seperti pada Gambar 4.9

```
function JaroWinkler($string1, $string2, $PREFIXSCALE = 0.1 ){

  $JaroDistance = Jaro( $string1, $string2 );

  $prefixLength = getPrefixLength( $string1, $string2 );

  return $JaroDistance + $prefixLength * $PREFIXSCALE * (1.0 - $JaroDistance);
}

function getPrefixLength( $string1, $string2, $MINPREFIXLENGTH = 4 ){

  $n = min( array( $MINPREFIXLENGTH, strlen($string1), strlen($string2) ) );

  for($i = 0; $i < $n; $i++){
    if( $string1[$i] != $string2[$i] ){
      // return index of first occurrence of different characters
      return $i;
    }
  }

  // first n characters are the same
  return $n;
}
```

Gambar 4.9 Kode pencarian nilai *dw*

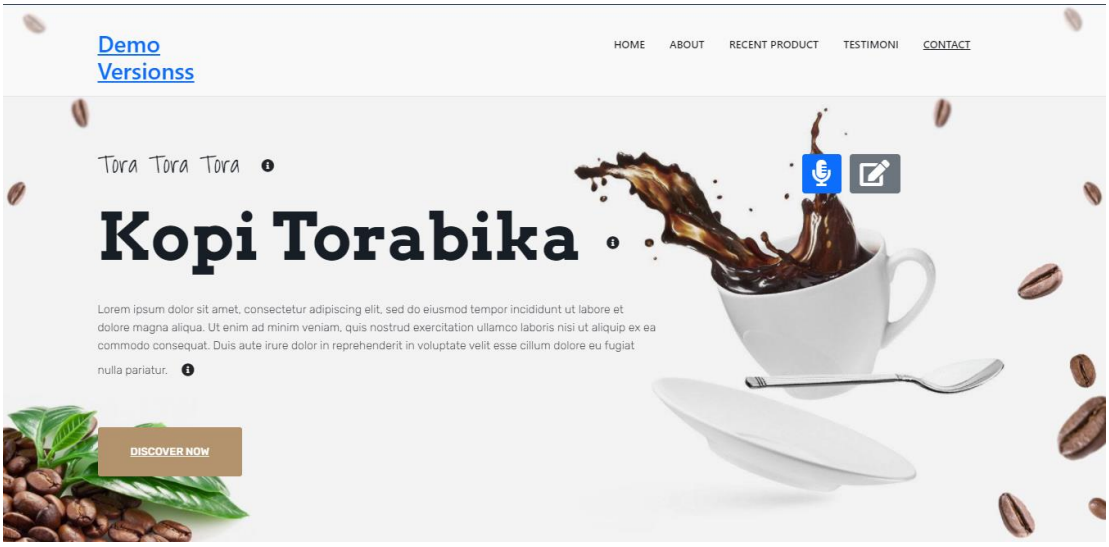
Dengan selesainya seluruh proses penghitungan maka didapatlah hasil dari penghitungan *Jaro-Winkler Distance.* Hasil dari perhitungan *Jaro-Winkler Distance* menghasilkan data seperti Tabel 4.1.

Tabel 4.1 Hasil perhitungan *Jaro-Winkler Distance*

No.	**Data**		***Jaro-Winkler Distance***	
	Input	**Kamus**	***Nilai dj***	***Nilai dw***
1	kopi	kopi	1	1
2	merupakan	merupakan	1	1
3	minuman	minuman	1	1
4	yang	yang	1	1
5	paling	paling	1	1
6	banyak	banyak	1	1
7	digemari > gemar	gemar	0.875 > 1	0.875 > 1
8	orang	orang	1	1

4.1.2 Penerapan *Speech Recognition* di *Company Profile*

Penerapan algoritma pada *Speech Recognition* di *Company Profile* ini berlokasi pada seluruh halaman *company profile* yang biasa dilihat pada saat pertama kali membuka website. Penambahan fitur *speech recognition* yang disertai algortima bertujuan untuk mempermudah pengeditan *company profile* tanpa harus pergi ke halaman dashboard CMS serta memberikan inovasi yang dimana pengguna akan bisa mengubah informasi didalamnya hanya dengan perintah suara. Penggunaan *speech recognition* di *company profile* ini hanya dapat diakses melalui "***live demo***" pada dashboard yang mempunyai icon mata. Tampilan *Speech Recognition* pada *Company profile* dapat dilihat pada Gambar 4.10.



Gambar 4.10 Tampilan *Speech Recognition* pada *Company Profile*

Proses penggunaan *speech recognition* pada *company profile* ini sedikit berbeda dengan penggunaan pada halaman *dashboard* CMS, penggunaan *speech recognition* pada halaman ini memiliki perintah kata yang harus digunakan untuk mengubah informasi sesuai dengan component yang ingin diubah. Untuk mengetahui perintah kata pengguna dapat berkata “bantuan” sebagai bantuan untuk mengetahui urutan perintah kata yang tepat dan juga sistem akan memberikan bantuan ketika pengguna menyebutkan perintah kata yang salah.

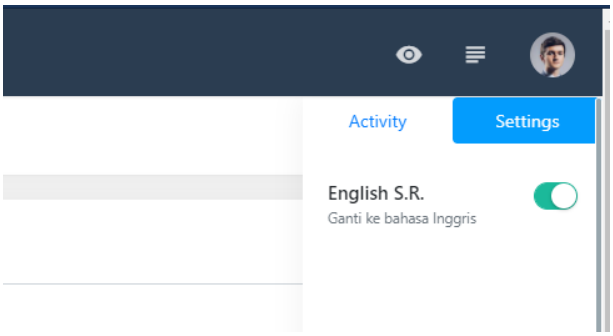
Proses pengolahan data yang terdapat di *speech recognition* pada *company profile* ini kurang lebih sama seperti pengolahan data yang terdapat di *speech recognition* pada *dashboard* CMS. Serta terdapatnya sebuah AI (*Artificial Intelegence*) *Top-down* sederhana yang bertugas sebagai asisten untuk memberikan bantuan untuk menginformasikan perintah kata yang harus digunakan untuk mengubah informasi pada *company profile.* Untuk dapat berinteraksi dengan pengguna AI *Top-down* sederhana menggunakan kode seperti pada Gambar 4.11.

```
function Slider_readOutLoud(message) {
    const speech = new SpeechSynthesisUtterance();
    speech.volume = 1;
    speech.rate = 1;
    speech.text = "Aku gak ngerti, bisa diulangi ?";

    if (message.includes('ubah') || message.includes('edit') || message.includes('Ubah') ) {
        if (message.includes('komponen') ) {
            if (message.includes('slogan') ) {
                const slogan = document.querySelector('.slider-speech-slogan-1');
                var newmessage = message.substring(21 , message.length);
                slogan.textContent = newmessage;
                speech.text = "Slogan siap";
            }else if (message.includes('nama produk')) {
                const produk = document.querySelector('.slider-speech-produk-1');
                var newmessage = message.substring(27 , message.length);
                produk.textContent = newmessage;
                speech.text = "nama produk siap";
            } else if (message.includes('deskripsi')) {
                const deskripsi = document.querySelector('.slider-speech-deskripsi-1');
                var newmessage = message.substring(24 , message.length);
                deskripsi.textContent = newmessage;
                speech.text = "deskripsi produk siap";
            } else {
                speech.text = "Perintah kurang lengkap // contoh : 'ubah'+'komponen'+'slogan. atau nama produk. atau deskripsi.'+'kata slogan baru'";
            }
        }else {
            speech.text = "Perintah kurang lengkap // contoh : 'ubah'+'komponen'+'slogan'+'kata slogan baru'";
        }
    }else if (message.includes('bantuan')) {
        speech.text = "Perintah kurang lengkap // contoh : 'ubah'+'komponen'+'slogan'+'kata slogan baru'";
    }
    // content.textContent = speech.text;
    console.log("--------------Jawaban---------------");
    console.log(speech.text);
    window.speechSynthesis.speak(speech);
}
```

Gambar 4.11 Kode AI *Top-down* sederhana

4.1.3 Fitur Ganti Bahasa Inggris

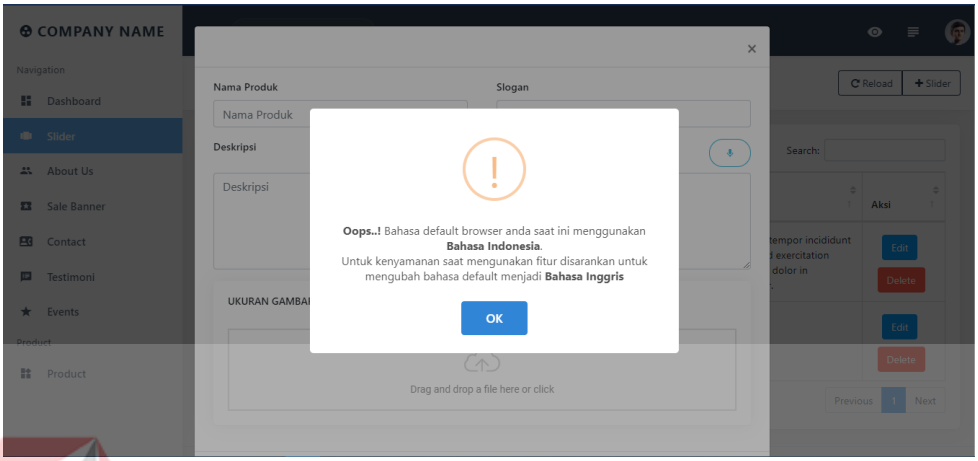
Penerapan fitur ganti bahasa pada aplikasi ini dapat diakses pada *dashboard header*. Fitur ini bertujuan untuk membuat agar *speech regocnition* dapat mengenali dan menganalisa suara dengan bahasa inggris. Tampilan fitur ganti bahasa pada *dashboard header* dapat dilihat pada Gambar 4.12.



Gambar 4.12 Fitur ganti Bahasa Inggris

Pengaktifan fitur ini berfungsi mengganti penggunaan kamus bahasa yang digunakan aplikasi, dimana data dari kamus tersebut akan digunakan dalam

perhitungan algoritma *jaro-winkler*. Untuk menggunakan fitur ini pengguna disarankan agar mengubah bahasa *default* browser pengguna mengikuti fitur ganti bahasa. Pada kasus bahasa browser dan fitur yang tidak sama, maka pengguna akan diberika peringatan untuk mengganti bahasa dari browser yang digunakan. Tampilan peringatan dapat dilihat pada Gambar 4.13.



Gambar 4.13 Fitur ganti Bahasa Inggris

4.2 Testing

Berikut merupakan hasil testing Penerapan Algoritma *Jaro-Winkler* Untuk *Autocorrect* dan *Spelling Suggestion* Pada Aplikasi *Speech Recognition CMS* Berbasis Website.

4.2.1 Testing Penerapan *Speech Recognition* di *Dashboard* CMS

Penerapan *algoritma* pada *speech recognition dashboard* CMS ini dapat membantu pengguna memasukan banyak data yang digunakan untuk informasi yang akan dimuncukan pada *company profile* hanya dengan menggunakan suara tanpa harus mengetik menggunakan *keyboard*. Berikut adalah desain testing untuk Penerapan Algoritma pada *Speech Recognition* di *Dashboard* CMS. Berikut adalah tahapan proses testing pada Tabel 4.2.

Tabel 4.2 Testing Penerapan *Speech Recognition* di *Dashboard* CMS

No.	Deskripsi	Tahapan	Input	Output	Hasil
1.	Masuk ke halaman dashboard CMS	Menuju ke halaman login	Masukkan email dan password	Halaman dashboard tampil	Sesuai
2.	Memilih bagin *Company Profile* yang akan diubah	Menuju ke halaman menu yang dituju	Klik menu tujuan.	Halaman menu tampil	Sesuai
3.	Menggunakan *speech recognition* untuk input data	Memilih komponen yang akan diubah informasinya	Klik button dengan icon mikrofon lalu bersuara	Sistem menghasilkan data text	Sesuai

4.2.2 Testing Penerapan *Speech Recognition* di *Company Profile*

Penerapan *algoritma* pada *speech recognition di company profile* ini dapat membantu pengguna memasukan data yang digunakan untuk informasi yang akan dimuncukan pada *company profile* hanya dengan menggunakan suara tanpa harus mengetik menggunakan keyboard dan tanpa perlu pergi ke halaman dashboard. Berikut adalah desain testing untuk Penerapan Algoritma pada *Speech Recognition* di *Company Profile*. Berikut adalah tahapan proses testing pada Tabel 4.3.

Tabel 4.3 Testing Penerapan *Speech Recognition* di *Company Profile*

No.	Deskripsi	Tahapan	Input	Output	Hasil
1.	Masuk ke halaman Live Demo	Menuju ke dashboard	Klik button dengan icon mata	Halaman Live Demo tampil	Sesuai
2.	Menggunakan *speech recognition* untuk input data	Memilih komponen yang akan diubah informasinya	Klik button dengan icon mikrofon lalu bersuara menggunakan urutan perintah suara.	Sistem menghasilkan data text	Sesuai

4.3 Hasil Uji Coba Lapangan

Hasil uji coba lapangan yang diperoleh peneliti terdapat 3 jenis, yang pertama adalah hasil ujicoba yang mencari informasi tentang beberapa kali *speech recognition* menghasilkan data yang *error* atau tidak sesuai dengan yang diinputkan saat bersuara. Uji coba ini dilakukakn berbarengan pada saat pengembangan sistem yang sejauh ini menghasilkan total 248 kali uji coba dengan aksen indonesian *native*, sistem memberikan kata yang salah hanya pada 14 kali uji coba dan hasil uji coba dengan menggunakan aksen *english native* yang diberikan arahan untuk menyebutkan kata Bahasa Indonesia total sebanyak 65 kali uji coba, yang dimana data yang salah hanya terdapat pada 9 kali uji coba. Dari kedua hasil uji coba untuk mencari berapa banyak kemungkinan *error* yang muncul didapat 5.64% kemungkinan kesalahan untuk penggunaan Indonesian *native* dan 13.84% kemungkingan kesalahan untuk penggunaan pada aksen *english native*. Berdasarkan kedua hasil tersebut dapat disimpulkan bahwa kemungkinan *error* yang muncul pada aplikasi ini dapat dikatakan berhasil karena belum melebihi 25%.

Hasil uji coba jenis kedua adalah uji coba yang hampir sama dengan uji coba pertama tetapi pada pengujian ini kata yang digunakan untuk uji coba sudah ditetapkan dan di test secara berulang-ulang lalu akan dihitung menggunakan rumus akurasi dan presisi untuk mengetahui seberapa akurat data yang dihasilkan. Total data yang digunakan adalah 600 data. Pengujian dapat dilihat pada perhitungan di bawah ini.

		Nilai Sebenarnya	
		TRUE	FALSE
Nilai Prediksi	TRUE	557	9
	FALSE	23	600

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

$$Accuracy = \frac{557 + 600}{557 + 600 + 9 + 23}$$

$$Accuracy = 0.9730 = 97.30\%.$$

$$Precision = \frac{TP}{FP+TP} \qquad Precision = \frac{557}{9+557} \qquad Precision = 0.9840 = 98.40\%.$$

Berdasarkan hasil pengujian diatas menghasilkan akurasi sebesar 97.30% dan hasil presisi sebesar 98.40%. Dengan hasil uji coba ini menunjukan bahwa nilai akurasi dan presisi yang melebihi 95%, maka dapat disimpulkan keakurasian data yang dicari sudah tepat.

Pengujian ketiga merupakan hasil uji coba dengan calon pengguna aplikasi "Penerapan Algoritma *Jaro-Winkler* Untuk *Autocorrect* dan *Spelling Suggestion* Pada Aplikasi *Speech Recognition CMS* Berbasis Website". Uji Coba yang akan ditanyakan kepada pengguna menggunakan *usability test script* (Steven Krug, 2020) dengan total 30 pengguna.

4.3.1 Pengujian dan Survei

Pembuatan tugas *usability testing* yang akan ditanyakan kepada pengguna yang telah dipilih bersarkan *learnability*, *efficiency*, *memorability*, *errors* dan *satisfaction*. Proses pengujian aplikasi dilakukan kepada 30 calon pengguna aplikasi untuk mencoba menggunakan aplikasi beserta fitur yang tersedia didalamnya. Survei dengan kuisioner merupakan tahap dimana dilakukannya penyebaran kuisioner kepada para pengguna untuk mendapatkan pendapat pengguna. Survei diberikan kepada total 30 calon pengguna yang akan memberikan pendapatnya.

4.3.2 Analisa Data Hasil *Usability Test* dan Survei

Proses analisa dilakukan dengan menghitung persentase pada *Task* di "Formulir Uji Coba Ketergunaan". Perhitungan ini dilakukan dengan menggunakan bentuk angka dari setiap jawaban dengan aturan angka 1 (satu) apabila jawaban "Ya" atau tanda centang (√) dan angka 0 (nol) diberikan kepada jawaban "Tidak". Dari setiap point Task dihitung persentase, berapa persen yang menjawab "Ya" dan berapa

persen yang menjawab “Tidak”. Berdasarkan analisa dilakukan mendapatkan hasil yang dapat dilihat pada Lampiran 7 dan rekap analisa pada Tabel 4.4.

Tabel 4.4 Rekap Hasil Perhitungan

No.	Kriteria	Respon	
		Ya (%)	Tidak (%)
1	Learnability	89.91	10.09
2	Memorability	77.7	22.3
3	Efficiency	77.42	22.57
4	Error	83.25	16.75
5	Satisfaction	99.9	0.1
Total Keseluruhan		**85.63**	**14.36**

Seperti pada tabel 4.4 rekap hasil perhitungan survei mendapatkan hasil dengan jumlah respon yang mengatakan “Ya” sebanyak 85.63%, sedangkan “Tidak” sebanyak 14.36%. Agar dapat disimpulkan aplikasi layak maka diperlukan pengelompokan kuantitatif yang mengidentifikasi apakah aplikasi sudah layak. Tabel Kuantitatif dapat dilihat pada Tabel 4.5.

Tabel 4.5 Tabel Kuantitatif

Skor	Kualifikasi	Hasil
85 – 100 %	Sangat Baik (SB)	Berhasil
65 – 84 %	Baik (B)	Berhasil
55 – 64 %	Cukup (C)	Tidak Berhasil
0 – 54 %	Kurang (K)	Tidak Berhasil

Dari perhitungan persentase jawaban respoden pada usability testing untuk mengukur penggunaan pada Penerapan Algoritma *Jaro-Winkler* Untuk *Autocorrect* dan *Spelling Suggestion* Pada Aplikasi *Speech Recognition CMS* Berbasis Website ini dapat disimpulkan bahwa sebanyak 85.63% pertanyaan mampu dilakukan oleh pengguna (menjawab “Ya”). 14.36% pertanyaan tidak dapat dilakukan oleh pengguna

(jawaban “Tidak”). Dengan total keseluruhan yang didapat yaitu 85.63%, berdasarkan tabel kuantitatif pada Tabel 4.5 maka dapat ditarik keseimpulan bahwa aplikasi “Penerapan Algoritma *Jaro-Winkler* Untuk *Autocorrect* dan *Spelling Suggestion* Pada Aplikasi *Speech Recognition CMS* Berbasis Website” mendapatkan hasil “Berhasil”.

BAB V
PENUTUP

5.1 Kesimpulan

Berdasarkan dari hasil implementasi dan testing Penerapan Algoritma *Jaro-Winkler* Untuk *Autocorrect* dan *Spelling Suggestion* Pada Aplikasi *Speech Recognition CMS* Berbasis Website, maka dapat dibuat kesimpulan sebagai berikut :

1. Penerapan Algoritma *Jaro-Winkler* pada *Speech Recognition* ini dapat memungkinkan pemasukan data dengan jumlah yang banyak tanpa menggunakan alat input seperti keyboard.
2. Penerapan *Speech Recognition* pada *company profile* memungkinkan proses perubahan konten tanpa perlu kembali ke halaman *dashboard* untuk melakukan perubahan.
3. Berdasarkan hasil uji coba lapangan yang dilakukan dengan penggunaan aksen mendapatkan 5.64% kemungkinan kesalahan untuk penggunaan Indonesian *native* dan 13.84% kemungkingan kesalahan untuk penggunaan pada aksen *English native.* Aplikasi ini dapat dikatakan berhasil karena belum melebihi 25%.
4. Berdasarkan hasil uji coba lapangan dimana dihitung dengan rumus akurasi dan presisi, mendapatkan akurasi sebesar 97.30% dan presisi sebesar 98.40%. Dengan hasil yang menunjukan nilai akurasi dan presisi yang melebihi 95%, maka dapat disimpulkan keakurasian data yang dicari sudah tepat.
5. Berdasarkan hasil testing dan survei yang dilakukan, respon yang mengatakan “Ya” sebanyak 85.63%, sedangkan “Tidak” sebanyak 14.36%, dengan total keseluruhan yang didapat yaitu 85.63% maka aplikasi dapat dinyatakan berhasil atau layak dipakai.

5.2 Saran

Dalam Penerapan Algoritma *Jaro-Winkler* Untuk *Autocorrect* dan *Spelling Suggestion* Pada Aplikasi *Speech Recognition CMS* Berbasis Website ini tentunya masih jauh dari kata sempurna dan memiliki beberapa kekurangan. Oleh sebab itu, untuk pengembangan aplikasi ini agar dapat menjadi lebih baik kedepannya, maka diberikan saran sebagai berikut :

1. Mengembangkan penggunaan *speech recognition* untuk dapat menjadi fitur dengan kegunaan yang lebih bervariatif.
2. Mengembangkan fitur *autocorrect & spelling suggestion* karena pada beberapa kasus masih bisa mengalami masalah seperti kata yang tidak terkoreksi.

DAFTAR PUSTAKA

Anak Agung Putri Ratna, R. S. (2020). Word Level Auto-correction for Latent Semantic Analysis Based Essay Grading System.

Bassil, Y. (2011). A Simulation Model for the Waterfall Software Development Life Cycle.

Dhega Febiharsa, I. M. (2018). Uji Fungsionalitas (Blackbox Testing) Sistem Informasi Lembaga Sertifikasi Profesi (SILSP) Batik Dengan Appperfect WEB TEST Dan Uji Pengguna. *Joined Jurnal*, Volume 1, Nomor 2.

Enterprise, J. (2014). *Lancar Java dan Javascript.* PT Elex Media Komputindo.

Gueddah, H. Y. (2015). The Filtered Combination of The Weighted Edit. Institute of Electrical and Electronics Engineers.

Hasugian, J. (2016). Penelusuran Informasi Ilmiah Secara Online: Perlakuan Terhadap Seorang Pencari Informasi Sebagai Real User . *Jurnal Studi Perpustakaan dan Informasi* .

James Pustejovsky, A. S. (2012). Natural Language Annotation for Machine Learning: A guide to corpus-building for applications. O'Reilly Media, Inc.

Jelinek, F. (1997). *Statistical Methods for Speech Recognition.* United States: Massachusetts.

Krug, S. (2020). Don’t Make Me Think! A Common Sense Approach to Web Usability. California.

Raharjo, B. (2018). *Belajar Otodidak Framework Codeigniter Edisi Revisi.* Bandung : Informatika.

Yulianingsih. (2017). Implementasi Algoritma Jaro-Winkler dan Levenstein . Jurnal String.