Pariwisata merupakan sektor yang berkaitan erat dengan kehidupan masyarakat modern. Semakin tinggi tingkat pendidikan dan ekonomi masyarakat, maka kebutuhan pada pariwisata akan semakin besar pula. Namun pengembangan informasi pariwisata yang ada saat ini dirasa belum efektif dan efisien. Dalam penyampaian informasinya masih menggunakan sistem manual, seperti brosur, pamflet, dan poster. Oleh karena itu dibutuhkan sebuah teknologi yang dapat menyampaikan informasi kepada wisatawan secara efektif dan efisien. Salah satu perkembangan teknologi terkini adalah kecerdasan buatan (artificial intelligence) yang diaplikasikan pada sistem percakapan pintar yang lebih dikenal dengan chatter bot (chatbot).

Chatbot merujuk kepada Bayan Abu Shawar dan Eric Atwell dalam tulisannya berjudul chatbots: Are they Really Useful? dimana merupakan program komputer yang berinteraksi dengan pengguna melalui bahasa natural. Dalam arti lain chatbot merupakan robot yang dirancang untuk bekerja dengan tema tertentu. Chatbot merupakan robot yang dirancang untuk berinteraksi atau bercakap-cakap dengan manusia.

Pada buku ini dikembangkan chatbot berbasis Telegram dan WhatsApp sebagai upaya mendukung pariwisata di Sumatera Utara. Dimana pembaca akan diberikan langkah demi langkah membangun chatbot dengan menggunakan bahasa pemrograman Python dan Visual Studio Code beserta library nya. Dengan buku ini diharapkan pembaca dapat membangun chatbot untuk wisata di daerahnya, sehingga wisatawan bisa mendapatkan informasi objek wisata dengan melakukan tanya jawab kepada sistem layaknya sebuah model diskusi.pengetahuan secara digital di lingkungan keluarga dan masyarakat.

Pustaka Aksara

ISBN 978-623-5471-14-3

MEMBANGUN CHATTER ROBOT (CHATBOT)

Pustaka Aksara

Pustaka Aksara

MEMBANGUN CHATTER ROBOT (CHATBOT)

WHATSAPP DAN TELEGRAM UNTUK INFORMASI PARIWISATA

Dr. Hesti Fibriasari, M.Hum. | Bakti Dwi Waluyo, M.T.
Prof. Dr. Baharuddin, S.T., M.Pd. | Tansa Trisna Astono Putri, S.Kom., M.T.I.
Merdy Roy Sunarya Togatorop, M.Sn.

MEMBANGUN *CHATTER ROBOT* (CHATBOT) WHATSAPP DAN TELEGRAM UNTUK INFORMASI PARIWISATA

Dr. Hesti Fibriasari, M.Hum.
Bakti Dwi Waluyo, M.T.
Prof. Dr. Baharuddin, S.T., M.Pd.
Tansa Trisna Astono Putri, S.Kom., M.T.I.
Merdy Roy Sunarya Togatorop, M.Sn.

Pustaka Aksara

MEMBANGUN CHATTER ROBOT (CHATBOT) WHATSAPP DAN TELEGRAM UNTUK INFORMASI PARIWISATA

Penulis : Dr. Hesti Fibriasari, M.Hum.
Bakti Dwi Waluyo, M.T.
Prof. Dr. Baharuddin, S.T., M.Pd.
Tansa Trisna Astono Putri, S.Kom., M.T.I.
Merdy Roy Sunarya Togatorop, M.Sn.
Desain Sampul : 21 Production
Tata Letak : Adam Akbar

ISBN : 978-623-5471-14-3

Diterbitkan oleh : **PUSTAKA AKSARA, 2022**
Redaksi:
Jl. Karangrejo Sawah IX nomor 17, Surabaya
Telp. 0858-0746-8047
Laman : www.pustakaaksara.co.id
Surel : info@pustakaaksara.co.id

Anggota IKAPI

Cetakan Pertama : 2022

KATA PENGANTAR

Puji syukur penulis ucapkan kepada Tuhan Yang Maha Esa, karena berkat dan rahmat-Nya, penulis dapat menyelesaikan buku ini. Penulis menyadari tanpa bantuan dan bimbingan dari berbagai pihak sangatlah sulit bagi penulis untuk menyelesaikan karya ini. Oleh karena itu, penulis mengucapkan banyak terimakasih pada semua pihak yang telah membantu penyusunan buku ini. Sehingga buku ini dapat hadir di hadapan pembaca.

Penggunaan komputer sebagai media informasi digital merupakan salah satu cara untuk pengaksesan informasi di masyarakat. Permasalahan dimana pencarian informasi yang berada di satu website dengan mengandalkan *search box* manual dirasa belum optimal. Pencarian hanya mengandalkan *keyword* dan tidak bisa berdasarkan kueri. Salah satu solusi adalah dengan menerapkan sistem pencarian berbasis *chatter robot* (chatbot). Chatbot merupakan aplikasi/layanan yang berinteraksi dengan pengguna melalui percakapan teks. Chatbot bekerja untuk menggantikan peran manusia dalam melayani pembicaraaan melalui aplikasi pesan. Buku ini bertujuan untuk mengembangkan chatbot dengan tujuan untuk menghasilkan alternatif pencarian informasi yang interaktif terkait tentang pariwisata di Sumatera Utara.

Pada perancangan dan implementasi perangkat lunak ini menghasilkan chatbot yang dibangun dengan menggunakan *artificial intelligence markup language* (AIML). AIML ini menyebabkan chatbot dapat mengintegrasikan input yang diterima berupa input text. Sehingga akan menghasilkan percakapan antara pengguna dan program.

Dalam buku ini pembaca diberikan pengalaman secara praktis, bagaimana membangun chatbot dengan menggunakan bahasa pemrograman Python beserta *library* pendukung yang cukup populer bagi sebagian *programmer* dan memanfaatkan Telegram Bot dan WhatsApp Bot sebagai media komunikasi antara pengguna dengan bot. Selain itu pembaca juga akan diberikan petunjuk praktis instalasi dan konfigurasi *visual studio code* sebagai

bot. harapannya, pembaca berhasil membangun chatbot baik di jaringan lokal hingga ke jaringan internet.

DAFTAR ISI

DAFTAR GAMBAR

MEMBANGUN *CHATTER ROBOT* (CHATBOT) WHATSAPP DAN TELEGRAM UNTUK INFORMASI PARIWISATA

Dr. Hesti Fibriasari, M.Hum.
Bakti Dwi Waluyo, M.T.
Prof. Dr. Baharuddin, S.T., M.Pd.
Tansa Trisna Astono Putri, S.Kom., M.T.I.
Merdy Roy Sunarya Togatorop, M.Sn

BAB I
PENDAHULUAN

A. Pendahuluan

Chatbot mempunyai banyak nama, diantaranya adalah agen percakapan (*conversational agents*), sistem dialog komputer dan manusia (*human computer dialogue systems*), agen interaktif (*interactive agents*), agen virtual (*virtual agents*), manusia virtual (*virtual humans*), dan asisten virtual (*virtual ssistants*) (Palanica et al, 2019). Meskipun terdapat perbedaan kecil diantara istilah-istilah tersebut, namun dapat diterima secara umum karena dapat digunakan secara bergantian. Chatbot mengaburkan batas antara manusia dan bukan manusia, karena chatbot berusaha meniru pola bicara manusia (Abu Shawar & Atwell, 2005). Pemahaman paling mendasar tentang chatbot adalah kerangka kerja yang merespons masukan bahasa alami dengan luaran seperti bahasa alami (Reshmi and Balakrishnan, 2018).

Chatbot (*chatter bot*) adalah layanan yang didukung oleh peraturan dan kecerdasan buatan, yang berinteraksi dengan kita melalui antarmuka obrolan. Layanan ini bisa berupa sejumlah hal, mulai dari yang fungsional hingga menyenangkan dan bisa hidup di aplikasi pesan instan yang memberikan rumah bagi chatbot ini. Chatbot merujuk kepada apa yang telah disampaikan oleh Bayan Shawar dan Eric Atwell dalam tulisannya berjudul "*Chatbots*: *Are They Really Useful?*", merupakan program komputer yang berinteraksi dengan pengguna yang memanfaatkan bahasa natural. Sementara itu, Jennifer Hill dalam "*Real Conversations with Artificial Intelligence: A Comparison Between Human-Human Online Conversation and Human-Chatbot Conversation*" mengatakan bahwa chatbot merupakan mesin sistem percakapan.

Bot dalam kata chatbot merupakan kata yang diambil dari "robot". Philip Auslander dalam jurnalnya berjudul "*Live from Cyberspace: Or, I Was Sitiing at My Computer this Guy Appeared He Touhgt I Was a Bot*" mengatakan bahwa terdapat banyak ragam "bot" di dalam ranah komputer, termasuk diantaranya warbots, channelbots, spambots, cancelbots, clonebots, collidebots,

floodbots, gamebots, barbots, eggdrop bots, dan modbots. Berbagai bot ini merupakan robot yang dirancang untuk berinteraksi atau bercakap-cakap dengan manusia.

Chatbot adalah program kecerdasan buatan dan model interaksi manusia-komputer (*human-computer interaction*) (Bansal & Khan, 2018). Kecerdasan buatan (*artificial intelligence*) telah mempengaruhi cara kita terlibat dalam aktivitas sehari-hari dengan merancang dan mengevaluasi aplikasi dan perangkat canggih, yang disebut agen cerdas, yang dapat melakukan berbagai fungsi. Secara umum chatbot adalah program komputer yang dirancang untuk mensimulasikan percakapan dengan manusia, terutama melalui internet. Chatbot menggunakan *natural language processing* (NLP) dan analisis sentimen untuk berkomunikasi dalam bahasa manusia melalui teks atau ucapan lisan dengan manusia atau chatbot lainnya (Khanna et al., 2015).

Selain bagian dari teknologi yang dapat meniru interaksi manusia, chatbot juga dapat diterapkan diberbagai bidang lain, diantaranya bidang pendidikan, bisnis dan digital marketing, kesehatan, dan hiburan (Shawar & Atwell, 2010). Selain itu, dalam dunia bisnis chatbot menjadi sangat umum karena mengurangi biaya layanan dan dapat menangani banyak pelanggan secara bersamaan. Chatbot lebih ramah dan menarik bagi pengguna dari pada pencarian konten statis di dalam daftar pernyatanyaan yang sering diajukan (*frequently asked questions*). Chatbot menawarkan bantuan yang nyaman dan efisien kepada pengguna saat berkomunikasi karena chatbot dapat memberikan jawaban yang menarik serta dapat menanggapi masalah pengguna secara langsung (R. Ranoliya, Raghuwanshi, & Singh, 2017).

Kebanyakan dari pengguna merasa bahwa chatbot adalah teman yang ramah dan bukan hanya sebagai asisten belaka (Costa, 2018). Tingkat kepercayaan yang diperoleh suatu sistem chatbot berasal dari faktor-faktor yang terkait dengan perilaku, penampilan, developer, privasi penggua, dan perlindungan terhadap pengguna (Wallace, 2009). Tingkat kepercayaan

terhadap chatbot juga bergantung terhadap karakteristik visual dan efektif dalam menangani bahasa manusia (Go & Sundar, 2019).

Perkembangan AI meningkatkan kemampuan chatbot untuk meniru agen manusia dalam percakapan. Namun, komunikasi manusia-chatbot memiliki perbedaan mencolok dalam konten dan kualitas dibandingkan dengan diskusi manusia-manusia. Durasi percakapan manusia-chatbot panjang. Orang sering menggunakan bahasa yang singkat dengan kosa kata yang buruk atau bahkan bahasa yang buruk. Perlu dicatat, bahwa perbedaan penting antara chatbots dan manusia adalah persepsi empati, karena chatbots kurang mampu memahami percakapan dibandingkan manusia. Namun, kemajuan sedang dibuat, dan chatbots secara bertahap menjadi lebih menyadari perasaan lawan bicara mereka (Fernandes, 2018).

B. Sejarah Chatbot

Pada tahun 1950, Alan Turing berfikir apakah program komputer dapat berbicara dengan sekelompok orang tanpa menyadari bahwa lawan bicaranya adalah palsu atau bukan manusia. Awalnya, chatbot merupakan percobaan program komputer yang bertujuan untuk memperdaya orang yang *chatting* seolah-olah dengan manusia, padahal sesungguhnya mereka berbicara dengan mesin. Ide tersebut bernama tes turing yang sebagian orang menyakini bahwa hal tersebut merupakan ide dasar dari chatbot. Chatbot dikembangkan sebagai simulasi percakapan manusia yang sesungguhnya. Hal ini untuk menjawab keinginan manusia untuk bisa berbicara dengan komputer menggunakan bahasa yang digunakan oleh manusia.

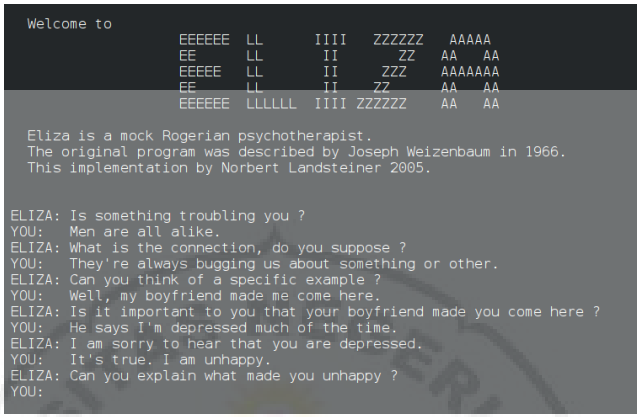
Chatbot pertama yang dikembangkan oleh Profesor MIT Joseph Weizenbaum pada tahun 1966 diberi nama ELIZA. Pada tahun 2009, sebuah perusahaan *WeChat* di China menciptakan chatbot yang lebih canggih. Sejak diluncurkan, *WeChat* telah menaklukkan hati banyak pengguna yang menunjukkan kesetiaan yang tak tergoyahkan, sehingga mendukung *WeChat* menjadi platform media sosial yang sangat populer dan

berkembang. Meskipun banyak paltform sosial media lainnya, seperti facebook messenger, slack, dan telegram, hal tersebut tidak membuat *WeChat* kalah dalam pengembangan chatbot. Chatbot milik *WeChat* saat ini dibantu oleh perusahaan Chumen Wenmen yang berdiri sejak 2012.

1. **ELIZA**

   Pada tahun 1966 Joseph Weizenbaum dari Massachusetts Institute of Technology (MIT) telah merilis sebuah chatbot bernama ELIZA. ELIZA dirancang sebagai chatbot yang memiliki tabiat sebagai seorang psikoterapis dalam berinteraksi atau ber-*chatting* dengan lawan bicara manusia. Chatbot ELIZA bekerja dengan meneruskan kata-kata yang dimasukkan pengguna ke komputer dan kemudian memasangkannya ke daftar kemungkinan tanggapan tertulis.

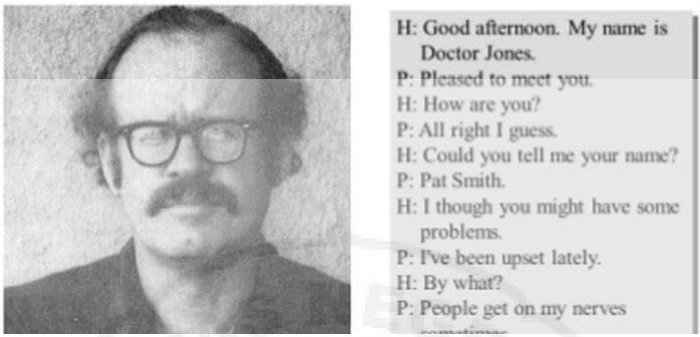
   Pada perkembangannya, pembuat WLIZA Weizenbaum merasa terganggu oleh reaksi pengguna. Pengguna banyak memanfaatkan ELIZA sebagai tempat mengungkapkan dan mencurahkan pemikiran mereka secara mendalam. Padahal ELIZA adalah sebuah chatbot yang berbasis mesin, sehingga tidak mempunyai hati dan perasaan. Weizenbaum menekankan bahwa pemahaman bahasa komputer sepenuhnya tergantung pada konteks dimana chatbot digunakan. ELIZA mempunyai kelemahan dimana mempunyai pengetahuan yang terbatas. Oleh karena itu, ELIZA hanya mampu membahas topik tertentu dan tidak dapat digunakan untuk percakapan yang panjang.



Gambar 1. Chatbot ELIZA

2. PARRY

PARRY dibangun oleh psikiater Amerika Kenneth Colby pada tahun 1972, yang disimulasikan sebagai pasien dengan *skizofrenia*. PARRY bekerja melalui sistem asumsi, atribusi, dan respon emosional yang diaktifkan. Lima puluh tahun yang lalu Kenneth Mark Colby adalah satu-satunya psikiater yang berfikir tentang bagaimana komputer dapat berkontribusi pada permasalahan penyakit mental. Namun setelah PARRY diuji oleh lima ahli psikiater pada tahun 1979, mendapatkan hasil bahwa PARRY tidak dapat digunakan dengan baik karena orang dengan *skizofrenia* memiliki tingkat inkoherensi dalam berbicara. Secara umum, PARRY dianggap sebagai chatbot dengan kemampuan rendah dalam hal pemahaman bahasa dan kemampuan untuk mengekspresikan emosi. PARRY juga memiliki kecepatan merespon yang rendah dan tidak dapat mempelajari percakapan secara baik.



Gambar 2. Chatbot PARRY

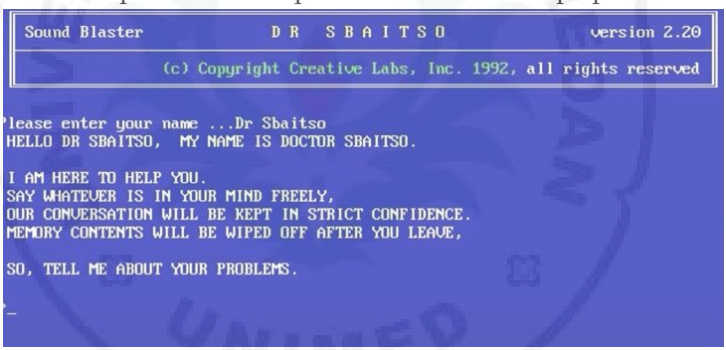
3. Jabberwacky

Gambar 3. Chatbot Jabberwacky

Kecerdasan Buatan pertama kali digunakan dalam domain chatbots dengan pembangunan Jabberwacky pada tahun 1988 (Jabberwacky, 2019). Jabberwacky ditulis dalam CleverScript, bahasa berbasis spreadsheet yang memfasilitasi pengembangan chatbot, dan menggunakan pencocokan pola kontekstual untuk merespons berdasarkan diskusi sebelumnya. Tetap saja, Jabberwacky tidak dapat membalas dengan kecepatan tinggi apabila digunakan dengan banyak pengguna.

4. Dr. Sbaitso

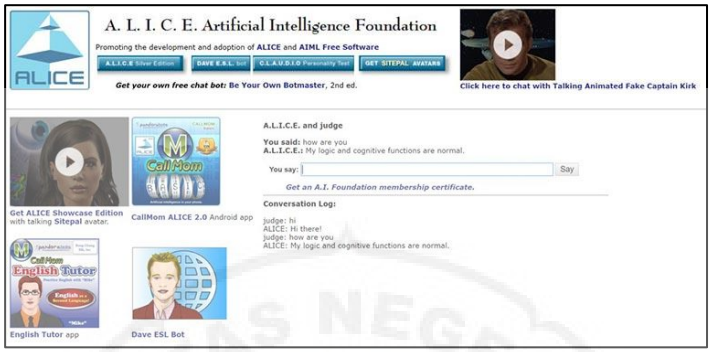
Istilah Chatterbot pertama kali disebutkan pada tahun 1991. Itu adalah pemain buatan TINYMUD (*multiplayer real-time virtual world*), yang fungsi utamanya adalah untuk mengobrol. Dr. Sbaitso (*Sound Blaster Artificial Intelligent Text to Speech Operator*) adalah chatbot yang dibuat oleh Creative Labs untuk MS-Dos pada tahun 1992 (Dr. Sbaitso, 2019). Ini adalah salah satu upaya paling awal untuk memasukkan AI ke dalam chatbot dan diakui untuk program obrolan yang dioperasikan dengan suara penuh. Program akan berbicara dengan pengguna seolah-olah itu adalah seorang psikolog. Sebagian besar tanggapannya adalah seperti "Mengapa Anda merasa seperti itu?" daripada interaksi rumit apa pun.



Gambar 4. Chatbot Dr. Sbaitso

5. ALICE

Basis pengetahuan ALICE terdiri dari 41.000 template dan pola, jumlah ini lebih besar dibanding dengan ELIZA yang hanya memiliki 200 kata kunci dan aturan. Pada tahun 1998, program ALICE diedit di Java, dan pada tahun 2001 Wallace mencetak spesifikasi AIML. Dari sana, ALICE menjadi program dengan sumber terbuka sehingga pengembangan lainnya menjadikan ALICE dapat dikembangkan ke beberapa bahasa asing. Program ini mensimulasikan mengobrol dengan orang sungguhan melalui internet dan menjawab dialog penggunanya.



Gambar 5. Chatbot ALICE

6. SmarterChild

Gambar 6. Chatbot SmarterChild

Pada tahun 2001, ada evolusi nyata dalam teknologi chatbot dengan pengembangan SmarterChild, yang tersedia di Messenger seperti America Online (AOL) dan Microsoft (MSN). Hal ini merupakan pertama kalinya chatbot dapat membantu orang dengan praktis, karena dapat mengambil informasi dari database tentang waktu pemutaran film, skor olahraga, harga saham, berita terkini, dan cuaca. Kemampuan ini menandai perkembangan yang signifikan baik dalam kecerdasan mesin dan lintasan interaksi manusia-

komputer karena sistem informasi dapat diakses melalui diskusi dengan chatbot.

7. **Siri**

Gambar 7. Chatbot Siri

Siri dikembangkan oleh Apple untuk iOS pada tahun 2010. Siri merupakan asisten pribadi cerdas dan navigator pembelajaran yang menggunakan UI bahasa alami. Aplikasi paten oleh Kantor Paten dan Merek Dagang Amerika Serikat merinci layanan Apple baru dimana pengguna dapat mengajukan pertanyaan dan percakapan dengan Siri melalui pesan. Paten baru ini menambahkan integrasi dengan file audio, vidio, dan gambar. Mirip dengan SMS dan Facebook Messenger, paten Apple menggambarkan Siri yang dapat melakukan tugas saat ini tanpa pengguna harus mengobrol dengan suara keras. Siri dapat membalas teks, audio, gambar, dan vidio saat ditransfer oleh pengguna. Pihak Apple menjelaskan bahwa hal ini akan mampu menghasilkan pengalaman interaktif yang lebih bermanfaat diantara konsumen dan asisten digital. Meski Siri canggih, namun Siri

masih memiliki beberapa kelemahan yaitu membutuhkan koneksi internet, ada beberapa bahasa yang tidak didukung, serta memiliki kesulitan dalam menterjemahkan bahasa jika dalam kondisi bising dan aksen yang berbeda.

8. **Watson**

Gambar 8. Chatbot Watson

Pada tahun 2011, sebuah chatbot bernama Watson diciptakan oleh IBM. Watson dapat memahami bahasa alami manusia dengan cukup baik untuk memenangkan dua juara sebelumnya pada kompetisi kuis "Jeopardy", di mana peserta menerima beberapa informasi dalam bentuk jawaban dan harus menebak pertanyaan yang sesuai. Bertahun-tahun kemudian, Watson memungkinkan bisnis untuk membuat asisten virtual yang lebih baik. Selain itu, Watson Health dirancang untuk membantu dokter dalam mendiagnosis penyakit di layanan kesehatan. Namun, kelemahan Watson adalah hanya mendukung bahasa Inggris.

9. Google Assistant

Google Now diluncurkan oleh Google Inc. pada tahun 2012, pada awalnya digunakan untuk memberikan informasi kepada pengguna dengan mempertimbangkan waktu, lokasi, dan preferensi. Google Assistant yang dikembangkan pada tahun 2016 merupakan generasi berikutnya dari Google Now. Google Assistant adalah asisten virtual yang didukung oleh kecerdasan buatan dan dikembangkan oleh Google yang terutama tersedia di perangkat selular dan perangkat rumah pintar. Tidak seperti Google Now, Google Assistant dapat terlibat dalam percakapan dua arah. Google Assistant awalnya memulai debutnya pada bulan mei 2016 sebagai bagian dari aplikasi perpesan Google Allo, dan pembicara yang diaktifkan suara Google Home.

Setelah periode eksklusif pada ponsel pintar Pixel dan Pixel XL, lalu mulai diluncurkan di perangkat Android lainnya pada bulan Februari 2017, termasuk ponsel pintar pihak ketiga dan Android Wear, dan dirilis sebagai aplikasi yang berdiri sendiri pada sistem operasi iOS pada bulan Mei. Di samping pengumuman pengembangan perangkat lunak pada bulan April 2017, Google Assistant telah, dan sedang, diperluas untuk mendukung berbagai macam perangkat, termasuk mobil dan peralatan rumah. Meski hanya tersedia eksklusif di Pixel, kehadiran asisten virtual ini cukup mencuri perhatian karena memiliki kemampuan yang lebih baik ketimbang pendahulunya, Google Now. Namun Google ternyata menjawab rasa penasaran sebagian besar pengguna ponsel pintar Android. Akhir Februari 2017, raksasa internet itu pun memastikan kehadiran Google Assistant untuk pengguna Android selain Pixel. Setelah lama dinanti, Google Assistant akhirnya tersedia dalam bahasa Indonesia.

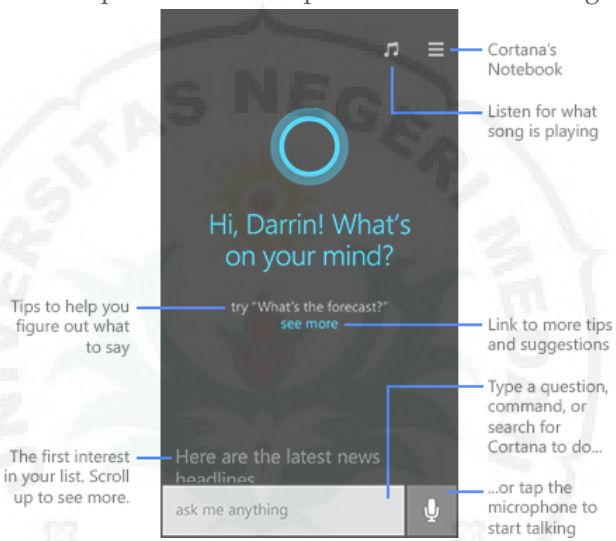
Gambar 9. Google Assistant

10. Cortana

Microsoft merancang asisten pribadi Cortana yang dikembangkan pada tahun 2014, dan langsung terintegrasi ke dalam perangkat telepon Windows dan PC Windows 10. Cortana dapat melakukan berbagai pekerjaan pada perangkat melalui perintah suara. Cortana dirancang dengan teknologi AI yang berusaha meniru kecerdasan dan perilaku manusia. Dilansir dari situs web Microsoft, Cortana punya ragam fungsi diantaranya mengatur pengingat jadwal pada kalender, mengadakan rapat *online*, mengatur alarm dan pengingat kegiatan, menemukan fakta, definisi, dan info. Tak hanya itu, Cortana dapat memutar musik dan radio, membacakan email masuk, membuat catatan-catatan, hingga menebak judul film atau lagu.

Akan tetapi, Cortana hanya tersedia di negara atau wilayah tertentu lantaran terkendala soal bahasa dan beberapa fitur Cortana mungkin tidak tersedia sepenuhnya. Cortana dapat ditemukan di Australia, Kanada, India, Inggris

Raya, dan Amerika Serika dengan perintah suara dalam bahasa Inggris. Cortana juga dijumpai di Brasil (bahasa Portugis), Cina, Perancis, Jerman, Italia, Jepang, serta Meksiko dan Spanyol (bahasa Spanyol). Selain Cortana, asisten virtual berbasis AI yang dapat mengerjakan "segala hal" melalui perintah suara seperti Assistant dari Google.

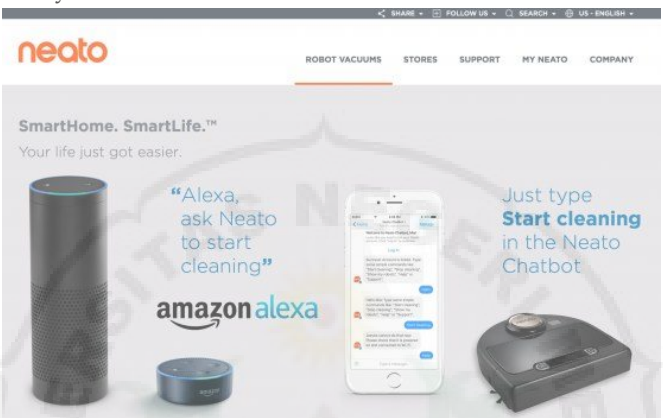


Gambar 10. Chatbot Cortana

11. Alexa

Alexa Internet, Inc. adalah sebuah anak perusahaan dari Amazon.com yang berbasis di California dan menyediakan data komersial terkait *traffic web*. Didirikan sebagai sebuah perusahaan independen pada tahun 1966, Alexa kemudian diakusisi oleh Amazon pada tahun 1999. Toolbar yang diciptakan oleh Alexa mengumpulkan data berupa kebiasaan berselancar penggunanya di internet dan mengirimkannya ke pusat Alexa, dimana data-data tersebut disimpan dan dianalisis, sehingga menjadi dasar dari laporan *web traffic* yang diberikan oleh perusahaan kepada pelanggannya. Sebagaimana tercatat pada tahun 2014, Alexa menyediakan data *traffic*, ranking situs web secara global maupun dalam satu negara tertentu, dan berbagai informasi

lainnya pada lebih dari 30 juta situs web yang terdaftar. Situs web Alexa dikunjungi lebih dari 8,8 juta orang setiap bulannya.



Gambar 11. Chatbot Alexa

Alexa Internet didirikan pada tahun 1996 oleh Brewster Kahle dan Bruce Gilliat. Nama dari perusahaan ini dipilih terinspirasi dari sebuah perpustakaan bernama "*Alexandria*" yang ada di California, dengan tujuan untuk menggambarkan hubungan antara pusat penyimpanan data sejak zaman kuno dengan potensi dari internet untuk menjadi pusat data menggantikan posisi perpustakaan. Operasi Alexa termasuk juga mengarsipkan halaman web yang mereka telusuri. Database ini disimpan dalam sebuah server yang kemudian disebut "*Internet Archive*" dan bisa diakses melalui sebuah mesin pencari yang dinamakan "*Wayback Machine*". Pada tahun 1998, perusahaan ini menyumbangkan sebuah salinan yang hanya berukuran 2 *terabytes* dari arsip tersebut kepada perpustakaan kongres amerika. Alexa kemudian memasok arsip internet tersebut secara rutin dengan menggunakan mesin penelusuran situs.

C. Cara Kerja Chatbot

Secara sederhana, cara kerja Chatbot adalah dengan mengandalkan keyword alias kata kunci yang sudah tertanam pada sistem. Maka, setiap kali chatbot memperoleh pertanyaan dari pengguna, secara otomatis ia akan menyesuaikan jawaban mana yang sesuai dengan keyword pertanyaan yang diajukan. Selain dirancang dengan kemampuan analisis dan identifikasi yang begitu responsif, ada tiga macam metode sistem operasioal yang dianut oleh Chatbot yaitu:

1. ***Pattern Matching*** **(Penyesuaian Pola)**

   Pada metode ini, bot menggunakan strategi penyesuaian pola (*pattern matching*) saat mengelompokkan teks. Dengan begitu, bot bisa menghasilkan jawaban yang tepat dengan permintaan pengguna. Pola tersebut dikenal dengan istilah *Artificial Intelligence Markup Language* (AIML). Kurang lebih, seperti ini contoh pola AIML pada Chatbot.

```
<aiml version = "1.0.1" encoding =
"UTF-8"?>
  <category>
   <pattern> WHO IS ABRAHAM LINCOLN
</pattern>
   <template> Abraham Lincoln was the
US President during
   American Civil War. </template>
 </category>

 </category>
  <pattern> DO YOU KNOW WHO * IS
</pattern>
  <template>
   <srai> WHO IS <star/></srai>
  </template>
 </category>
</aiml>
```

   Maka, setiap kali Chatbot memperoleh pertanyaan, ia akan memberikan respons apapun yang sekiranya tepat dengan pola terkait. Namun, jika ada bentuk permintaan

yang di luar dari bentuk pola, maka ia tidak akan mampu memberikan jawaban yang sesuai.

Gambar 12. Chatbot Decision Tree-Based

2. ***Decision Tree-Based***

Dapat dikatakan bahwa cara kerja Chatbot satu ini merupakan cara yang kurang ramah bagi pengguna. Karena pengguna harus mengikuti urutan jawaban yang sudah terprogram oleh mesin bot. Metode ini terkadang bisa begitu sederhana atau pun rumit, tergantung bagaimana konsep tersebut dirancang. Namun, banyak pula pemilik bisnis yang menggunakan metode ini karena kerumitannya yang rendah, lebih cepat, dan tetap berguna dalam melayani pertanyaan para penggua. Saat pemakaiannya, pengguna akan dihadapkan dengan beberapa *widget* berupa tombol yang berisikan teks jawaban. Kurang lebih contoh dari Chatbot dengan sistem Decision Tree-Based dapat dilihat pada Gambar 12.

3. Contextual

Metode kontekstual chatbot mengandalkan sistem kecerdasan buatan dengan *machine learning* (ML) untuk bisa menciptakan percakapan secara natural. Metode ini memang

tergolong sebagai sistem terbaik dibandingkan kedua metode sebelumnya. Namun, untuk merancangnya, pihak developer membutuhkan perencanaan yang sangat strategis dan terarah. Salah satu teknologi yang sering diandalkan untuk menciptakan metode kerja kontekstual pada chatbot yaitu *Natural Language Processing* (NLP). NLP meruoakan teknologi yang memudahkan AI untuk bisa memahami setiap konteks dan maksud penggua dalam bentuk bahasa yang sesuai.

Ketika pengguna mengetik kata "halo", otomatis NLP akan membantu bot untuk memahami bahwa penggua tersebut telah mengirim ucapan, dan AI akan menentukan jawaban yang tepat. Supaya metode kontekstual dapat berjalan secara maksimal, pihak developer perlu merancang database yang cukup luas untuk mencakup segala bentuk permintaan pengguna. Anda bisa mengumpulkan data pelanggan, transkrip obrolan pada fitur *live chat*, dan berbagai data pendukung lainnya. Semakin banyak data yang dikumpulkan pada database, maka semakin berkembang pula kapasitas bot untuk memberikan respon.

4. **Markov Chain**

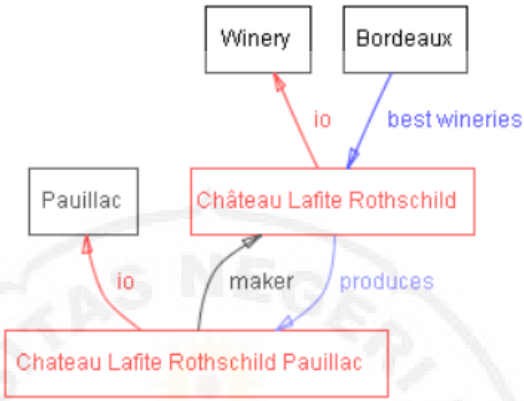
Model Markov Chain adalah pendekatan yang mengacu pada penelitian Anreyevich Markov tahun 1906. Gagasannya, ia mencatat dan mencermati tiap perubahan yang terjadi. Kemudian catatan tersebut dijadikan sebuah model untuk memprediksi perubahan yang terjadi di masa mendatang. Perubahan-perubahan itu diwakili dalam variabel dinamis dalam kurun waktu tertentu. Pada konteks chatbot, pendekatan ini kerap digunakan dengan tujuan membangun konteks percakapan antara chatbot dan pengguna menjadi layak. Meskipun terkadang chatbot menghasilkan kalimat yang cenderung omong kosong, namun ia terlihat alami.

5. **Ontologi**

Ontology atau ontologi berasal dari kata "*ontos*" dan "*logos*" (bahasa Yunani). Dalam bidang teknologi, ontologi ialah sebuah cara yang secara eksplisit dapat menjelaskan suatu domain pengetahuan. Ontologi juga dapat diartikan dalam suatu konsep yang dapat dibuat dengan cara memberikan makna, properti, serta relasi pada konsep sehingga sebuah ontologi dapat berkumpul pada suatu domain pengetahuan dan membentuk sebuah basis pengetahuan atau yang biasa disebut sebagai *knowledge base*. Sebuah ontologi bersama dengan satu set dari individu atau *instance* dari *class* merupakan sebuah *knowledge base*, hanya ada garis tipis dimana ontologi berakhir dan sebuah basis pengetahuan (*knowledge base*) dimulai (Noy & McGuinness, 2001). Ontologi juga dapat digunakan sebagai susunan pemersatu dalam pemecahan suatu masalah.

Secara umum ontologi memiliki kegunaan sebagai *controller vocabulary*, *sematic*, *interoperability*, *knowledge sharing*, dan *reuse*. Gambar 13 adalah dimana contoh ontologi dengan beberapa *class*, *instance*, dan *relation* berdasarkan *wine* domain. Warna hitam menggambarkan *classi* dan merah menggambarkan *instance*. Garis lurus mewakili sebuah slot dan garis internal seperti *instance-of* dan *subclass-of*. Pada intinya terdapat beberapa cara praktis dalam membangun sebuah ontologi, yaitu:

a. Mendefinisikan *class* dalam ontologi
b. Mengatur *class* ke dalam hierarki taksonomi (*subclass-superclass*)
c. Mendefinisikan slot dan menjelaskan jenis *value* yang diizinkan
d. Mengisi *value* untuk slot pada *instance*.



Gambar 13. Contoh Ontologi (Noy & McGuinness, 2001)

BAB II
ARSITEKTUR DAN PENDEKATAN TEKNIS

A. Pendekatan Algoritma

Terdapat beberapa pendekatan yang umum dilakukan dalam proses pengembangan chatbot. Pertama, pencocokan pola (*pattern matching*) yang merupakan pendekatan paling banyak digunakan dan juga diimplementasikan oleh Eliza. Pendekatan ini dilakukan dengan mencocokkan data yang masuk dengan data yang ada basis data. Jika terdapat pola yang cocok, maka pola tersebut akan dikembalikan ke pengguna (Bradesko, 2012).

Kedua, penguraian tekstual (*textual parsing*) yang merupakan pendekatan di mana teks asli yang dikirim ke chatbot ditransformasikan (*lexical parsing*) menjadi seperangkat kata. Tujuan dari transformasi ini sebagian besar adalah untuk menentukan struktur tata bahasa. Contohnya adalah ketika pengguna mengirim teks "tolong ambilkan minum", maka kalimat tersebut akan ditransformasikan menjadi "ambil minum".

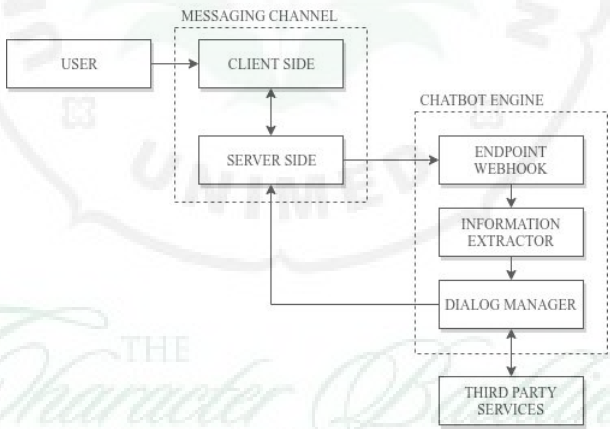
Ketiga, model Markov Chain adalah pendekatan yang mengacu pada penelitian Andreyevich Markov tahun 1906. Gagasannya adalah Markov mencatat dan mencermati tiap perubahan yang terjadi. Kemudian catatan tersebut dijadikan sebuah model untuk memprediksi perubahan yang terjadi di masa mendatang. Perubahan-perubahan ini diwakili dalam variabel dinamis dalam kurun waktu tertentu. Pada konteks percakapan antara chatbot dan pengguna menjadi lebih layak. Meskipun terkadang chatbot menghasilkan kalimat yang cenderung omong kosong, namun ia terlihat alami.

Keempat, ontologi yang merupakan pendekatan dengan mengambil keuntungan dari kemiripan pada suatu konsep (pada chatbot kita bisa menyebutnya kata atau kalimat) yang berhubungan dan disimpan secara hierarki. Dengan menggunakan pendekatan ini, chatbot diuntungkan karena setiap konsep memiliki interkoneksi di dalam suatu *graph*.

B. Arsitektur

Secara umum, chatbot dibangun berdasarkan dua komponen utama, yaitu komponen ekstraksi informasi dan komponen manajemen dialog. Komponen ekstraksi informasi atau kerap disebut identifikasi intent merupakan proses untuk menentukan maksud pesan yang dikirim oleh pengguna. Sedangkan komponen manajemen dialog merupakan komponen yang berfungsi mengelola kebutuhan-kebutuhan dialog antara pengguna dan chatbot (Chan, 2017).

Seperti pada gambar 14, proses interaksi pengguna dan chatbot akan terjadi melalui medium chat, dimana chatbot akan mengirimkan *webhok* kepada suatu *endpoint* yang ada di sistem chatbot. Selanjutnya, akan ada proses identifikasi *intent* di mana ia bertugas mengekstrak informasi dari masukan pengguna. Proses ekstraksi ini kerap dilakukan dengan metode *statistical*, *non-statistical*, atau penggabungan keduannya.



Gambar 14. Arsitektur Umum yang Ada pada Chatbot

Setelah proses identifikasi *intent* selesai, chatbot harus melakukan proses manajemen dialog yang di dalamnya terdapat berbagai strategi komunikasi dengan pengguna. Strategi komunikasi yang dimaksud antara lain adalah untuk menangani beberapa hal seperti pergantian topik pembahasan, memberikan pertanyaan atau konfirmasi kepada pengguna, atau memberi

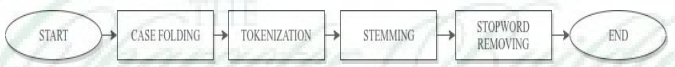
tahu pengguna bahwa chatbot tidak mengerti maksud pengguna jika *intent* tidak ditemukan.

C. Information Retrieval

Information Retrieval (IR) merupakan topik yang mempelajari bagaimana komputer menyimpan dan mencari data secara efisien. Selain itu, IR juga mempelajari bagaimana komputer memberikan jawaban yang sesuai dengan kebutuhan pengguna (Asian, 2007). Terdapat tiga tujuan utama yang melatarbelakangi kemunculan IR.

Pertama, untuk melakukan pencarian informasi dengan cepat seiring dengan pertumbuhan data di Internet. Komputer harus bisa melakukan pencarian informasi dengan cepat bahkan saat ia harus mencari informasi dari teks yang jumlahnya miliaran atau triliunan. Kedua, untuk meningkatkan fleksibilitas operasi pencocokan sebuah data. Ketiga, untuk memungkinkan proses pencarian informasi dengan hasil bertingkat. Di dalam IR terdapat beberapa tugas yang umum dilakukan untuk mengurai sebuah informasi dalam suatu teks yaitu *case folding*, *tokenization*, *stemming*, dan *stopword removing*.

Mengacu pada Gambar 15, langkah pertama yang perlu dilakukan dalam proses IR adalah *case folding*. Ia merupakan proses yang bertujuan menyamakan teks menjadi huruf kecil atau huruf besar. Proses ini berguna untuk menyamakan nilai *binary* dari masing-masing karakter.



Gambar 15. Alur Pemrosesan *Information Retriveral*

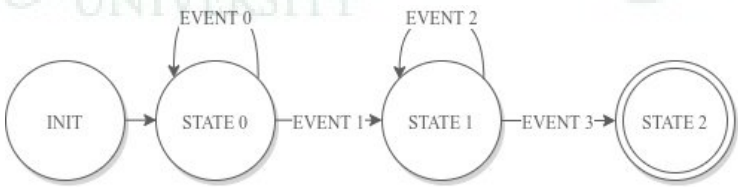
Setelah proses *case folding* selesai, langkah berikutnya adalah *stemming*. Ia adalah proses yang dilakukan untuk mengubah suatu kata menjadi kata dasar, misalnya kata "pengetahuan" akan di-*stem* menjadi "tahu". *Stemming* bertujuan untuk meningkatkan sensitivitas proses pencarian informasi yang relevan, meski kerap menurunkan selektivitas karena dalam beberapa kasus *stemming* dapat menyebabkan

hilangnya makna yang berguna. Selanjutnya, proses yang perlu dijalankan adalah *stopword removing*. Ia adalah proses penghapusan kata-kata yang tidak diperlukan dari sebuah kumpulan kata. Proses ini bertujuan menghemat ukuran indeks, waktu pemrosesan, dan mengurangi tingkat *noise* dalam suatu teks.

D. Finite State Machine

Kemunculan *finite state machine* (FSM) atau kerap disebut *finite state automata* bermula dari deskripsi *finite automata* pada makalah berjudul "*A Logical Calculus Immanent in Nervous Activity*" yang ditulis oleh dua orang ahli neurofisiologi Warren McCulloch dan Walter Pitts pada tahun 1943. Makalah yang berkontribusi secara signifikan terhadap penelitian-penelitian tentang teori jaringan syaraf tiruan, automata, dan komputasi ini kemudian digeneralisasi oleh Mealy dan Moore ke dalam bidang ilmu komputer pada tahun 1955-1956.

Secara formal, FSM merupakan salah satu pemodelan dalam komputer yang berfungsi untuk menentukan suatu tindakan yang harus dilakukan oleh mesin pada suatu keadaan. FSM dilandasi oleh kesadaran bahwa mesin memiliki keterbatasan dalam melakukan pekerjaan. Keterbatasan itu juga bersinggungan dengan lambatnya pertumbuhan kecepatan komputasi dari suatu mesin. Artinya dengan menggunakan FSM, mesin dapat diatur sedemikian rupa agar melakukan tindakan-tindakan tertentu pada suatu keadaan. Kemunculan FSM membuat proses komputasi menjadi lebih sederhana karena semesta kemungkinan yang ada dapat dibatasi.



Gambar 16. Alur Pemrosesan *Finite State Machine*

Pada FSM terdapat dua kata kunci utama, pertama adalah *state* atau keadaan yang merupakan sebuah kondisi dimana ia akan mengarahkan mesin. Kedua adalah *action* atau tindakan yang harus dilakukan komputer saat berada dalam suatu *state*. Dua kata kunci itu kemudian diturunkan menjadi empat variabel yang diperlukan untuk menjalankan pemodelan FSM.

Variabel pertama seperti Gambar 16 adalah *set of all states* (Q) yang merupakan kumpulan keadaan yang didaftarkan ke mesin. Variabel kedua adalah iinputs (Σ) atau daftar masukan yang datang ke suatu mesin. Variabel ketiga adalah *initial state* ($q_0 \epsilon Q$) dimana sebuah keadaan yang merupakan proses dimulainya FSM. Keempat yaitu *set of final states* ($F \epsilon Q$) yang merupakan keadaan dimana FSM selesai berproses. Lalu kelima yaitu *transition function* ($\delta: Q \times \Sigma \rightarrow Q$) yang merupakan suatu *action* yang perlu dilakukan oleh mesin dari satu *state* ke *state* lainnya (Wright, 2005).

Di dalam FSM terdapat dua buah kategori pemodelan yang populer yaitu Deterministic FSM (DFA) dan Non-Deterministic FSM (NFA). Perbedaan keduanya terletak pada pertama, jumlah *action* yang diperbolehkan pasa suatu *state*, dimana DFA hanya diperbolehkan memiliki satu *action* sedangkan NFA bisa lebih dari satu. Kedua, diizinkan atau tidaknya keberadaan *string* kosong pada suatu *action*. DFA memungkinkan *action*-nya diisi oleh *string* kosong, sedangkan NFA tidak.

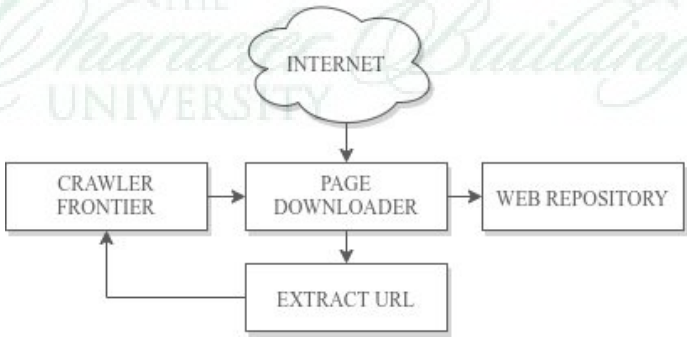
Ketiga, jumlah mesin yang dapat digunakan untuk memproses *action* pada suatu *state* dimana DFA hanya bisa satu mesin, sedangkan NFA bisa lebih dari satu mesin. Keempat, DFA harus memiliki satu buah *final state* sedangkan NFA dapat memiliki lebih dari satu. Kelima, DFA tidak diperbolehkan adanya alur mundur. Misalnya, jika *state* A sudah terlewati dan sekarang mesin berada pada *state* B, maka di DFA *action* yang dilakukan oleh B tidak boleh kembali ke *state* A. sedangkan pada NFA, hal itu diperbolehkan.

Pada *chatbot* sendiri, FSM bukanlah model yang umum digunakan. Perkembangan model jaringan syaraf tiruan yang cukup cepat membuat kebanyakan pengembang *chatbot* lebih memilih tidak menggunakan FSM. Namun, terdapat salah satu penelitian menarik dimana ternyata FSM kerap kali lebih efektif dibandingkan model jaringan syaraf tiruan. Dalam hal ini Sanghyun Yi dan Kyomin Jun dalam penelitiannya pada 2017 coba mengombinasikan penggunaan FSM, *information retrieval* (IR), dan *bot-initiave strategy*.

Bot-Initiative Strategy mengacu pada sistem yang dapat mengambil inisiatif dalam memulai percakapan untuk mendapatkan informasi yang diperlukan. Namun dalam keadaan ini pengguna juga dapat mengambil inisiatif. Pada penelitian tersebut, sebagian besar model percakapan yang ditanamkan di *chatbot* menggunakan FSM. Artinya, hampir seluruh masukan dari pengguna akan masuk ke dalam suatu *state*. Pengguna IR hanya dilakukan ketika masukan dari pengguna berada di luar jangkauan model FSM.

E. Web Crawling

Web Crawler merupakan program yang bertugas untuk mencari dan mengunduh data yang terdapat disuatu halaman *website* secara otomatis. Teknik *web crawling* utamanya digunakan dalam membangun mesin pencari di mana halaman-halaman *website* yang telah ditentukan kemudian disimpan dalam repositori lokal yang terstruktur.



Gambar 17. Arsitektur *Web Crawler*

Terdapat tiga komponen utama untuk membangun *web crawler* seperti yang tersaji pada Gambar 17. Pertama, *crawler frontier* yang bertugas untuk menyimpan daftar URL yang perlu dikunjungi. Kedua, *page downloader* yang bertugas untuk mengunduh data pada setiap URL yang telah didaftarkan sebelumnya. Ketiga, *web repository* yang bertugas untuk menerima hasil *crawling* yang dilakukan *page downloader* lalu menyimpannya di dalam repositori lokal.

Inisiasi proses *web crawling* dilakukan menggunakan mekanisme penjadwalan. Misalnya, setiap 60 menit sekali sistem penjadwalan akan meminta *page downloader* untuk memulai proses *crawling*. *Page downloader* kemudian akan mengecek daftar URL yang perlu dikunjungi dan unduh pada *crawler frontier*. Setelah berhasil mengunduh data pada halaman yang ada, *page downloader* akan memilih data-data pada saja yang dibutuhkan lalu menyimpannya di repositori web lokal. Selanjutnya, *page downloader* akan mengekstrak URL-nya dan memberi tahu komponen *crawler frontier* bahwa ia telah mengunduh dan menyimpan halaman tersebut di repositori lokal.

F. Chatbot Builder

Chatbot adalah asisten virtual dapat membantu dalam mengotomatiskan berbagai elemen utama bisnis seperti penjualan, pemasaran, dan pelayanan. Ada enam tahap penting yang perlu dilakukan sebelum menggunakan teknologi bot ini:

1. Tentukan tujuan penggunaan chatbot. Uraikan terlebih dahulu apa saja fungsi bisnis yang perlu didukung dengan bantuan bot. Selain itu, fikirkan pula konsep bot yang tepat agar bisa melayani pengguna dengan hasil maksimal.
2. Pilih saluran yang tepat untuk berinteraksi dengan pelanggan. Perlu diputuskan platform apa nanti yang akan terintegrasi dengan bot. Apakah melalui website, aplikasi pada *smartphone*, facebook messenger, telegram atau platform media sosial lainnya.

3. Latih bot untuk bisa merespon dengan tepat. Sebelum bot siap digunakan, lakukan A/B testing untuk menilai bagaimana performa bot Anda. Bantu dengan penggunaan format *frequently asked questions* (FAQ) sehingga jawaban yang dihasilkan bisa sesuai.
4. Buat pendekatan yang seimbang. Akan ada beberapa titik yang mungkin pertanyaan pelanggan tidak bisa dicerna oleh bot. Maka itu, tentukan tahapan mana saja yang membutuhkan bantuan bot dan bantuan manusia secara langsung.
5. Tes, rilis, dan evaluasi. Setelah Anda berhasil merisil bot, pastikan Anda rutin melakukan pengecekan dan evaluasi untuk menilai seberapa jauh performa yang dihasilkan.

Dengan melewati keenam tahapan tersebut, diharapkan Anda bisa menciptakan bot yang sesuai dengan yang diharapkan dengan minimnya kesalahan. Setelah itu, barulah Anda bisa menentukan platform chatbot mana yang ingin digunakan. Berikut ini adalah beberapa rekomendasi mengenai platform **chatbot builder**.

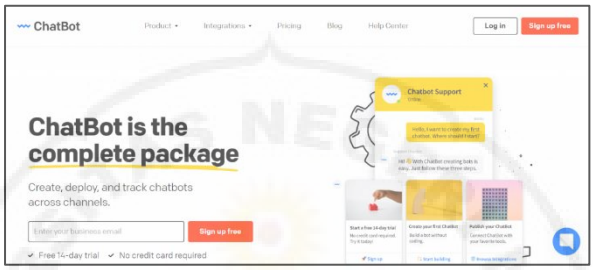
1. Chatbot

Chatbot merupakan salah satu platform terpopuler yang paling banyak digunakan oleh berbagai bisnis digital di dunia. Dari namanya sudah menunjukkan kalau platform ini secara eksplisit memberikan solusi tepat untuk menciptakan chatbot yang multi fungsi. Anda bisa mengintegrasikan chatbot pada berbagai platform pesan singkat seperti Slack, Facebook Messenger, Livechat, Skype, Twitter, hingga YouTube.

Bila Anda belum merasa yakin untuk melakukan pembelian, chatbot menawarkan layanan *free trial* selama 14 hari dengan fitur unggulan seperti:

a. Tersedia rangkaian *template* siap pakai maupun kustom.
b. Pembuatan yang mudah dengan adanya fitur *drag and drop*.

c. Memiliki visual chatbot builder untuk menambahkan gambar, menu, dan wieget.
d. Terintegrasi dengan segala macam bahasa.
e. Proses tes yang cepat dan mudah.

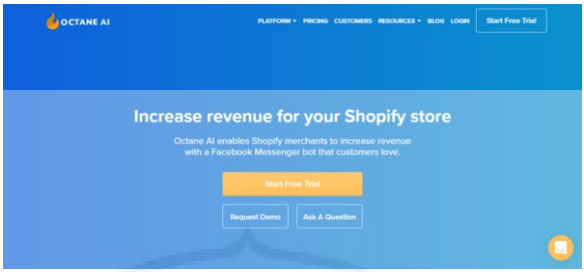


Gambar 18. Chatbot Builder Chatbot

2. Octane AI

Octane AI merupakan platform chatbot yang umumnya digunakan oleh pengguna shopify dan facebook messenger. Sama dengan platform sebelumnya, Octane AI tidak membutuhkan keterampilan coding sama sekali untuk menggunakannya. Berikut adalah fitur unggulan yang bisa Anda coba dengan Octane AI:

a. Memiliki tool analytics yang dapat menyediakan data *revenue* dan *customer behavior* secara rinci.
b. Dapat mengirimkan notifikasi seputar *shipping* dan *receipt order* kepada pelanggan.
c. Bisa menargetkan konsumen berdasarkan demografi dan perilaku.
d. Menjalankan *campaign* di halaman facebook.
e. Menciptakan sistem pelayanan secara kustom dengan *automation flows*.

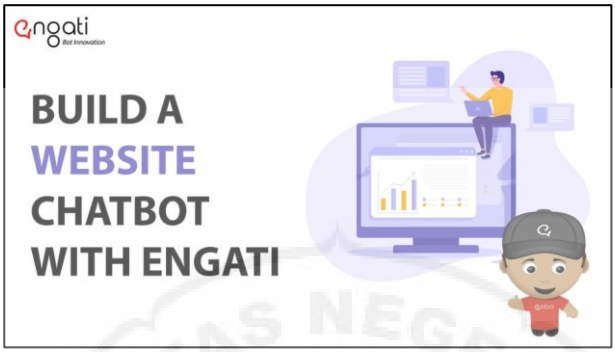


Gambar 19. Chatbot Builder Octane AI

3. Engati

Berbeda dengan platform chatbot sebelumnya, Engati merupakan platform yang tidak hanya terintegrasi dengan aneka platform chatiing seperti Facebook Messenger, Telegram, Line, Viber, Skype, dan Slack, tetapi juga dengan website dan blog. Engati juga menawarkan lisensi gratis, tapi hanya untuk berinteraksi dengan 1000 pelanggan saja. Kalau sudah lebih itu, maka Anda harus membeli lisensi yang Preminum. Banyak sekali fitur mumpuni yang bisa Anda manfaatkan dengan Engati, yaitu:

a. Tersedia lebih dari 150 *templates* untuk segala jenis sektor bisnis.
b. Bisa dioperasikan dengan banyak macam bahasa.
c. Ada fitur bot dalam bentuk teks dan suara.
d. Tersedia dalam versi mobile.
e. Terintegrasi dengan lebih dari 14 platform media sosial.
f. Promosi bisnis lebih fleksibel dengan fitur automasi broadcast dan campaign.
g. Bekerja dengan berbagai *deployment model* seperti *public cloud, on premise*/private cloud dan hybrid.
h. Tersedia tools analytics.

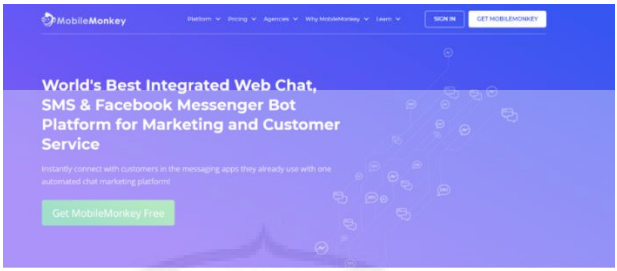


Gambar 20. Chatbot Builder Engati

4. **MobilMonkey**

MobileMonkey adalah teknologi bot yang bisa digunakan pada facebook messenger, website, dan sms. Kesempatan Anda untuk memperoleh konversi tinggi akan semakin besar dengan platform ini. Ditambah lagi jika Anda menghubungkannya dengan *ad campaign, drip marketing, list* kontak pelanggan dan tools pesan *broadcast*. Sayangnya pada versi gratis, Anda hanya bisa menggunakan list kontak pelanggan dan fitur terbatas. Kecuali bila Anda mengupgrade ke Premium Plan dengan harga $149 per bulan untuk menikmati fiturnya yang lengkap, termasuk kapasitas 5000 list kontak pelanggan. Selain itu, mobilemonkey chatbot juga menawarkan jejeran keunggulan lain, seperti:

a. Bisa mengatur setiap percakapan melalui aplikasi mobile di iOS dan Android.
b. Menciptakan percakapan yang interaktif dan mudah dipahami.
c. Memberikan respon secara *real-time*.
d. Menawarkan platform yang *mobile-friendy* di berbagai perangkat pengguna.
e. Membuat bot untuk *follow up* setiap pesan untuk *nurture leads* sesuai jadwal dan segmen setiap *campaign*.
f. Menawarkan fitur *drag and drop, widgets,* dan *tools visual campaign*.

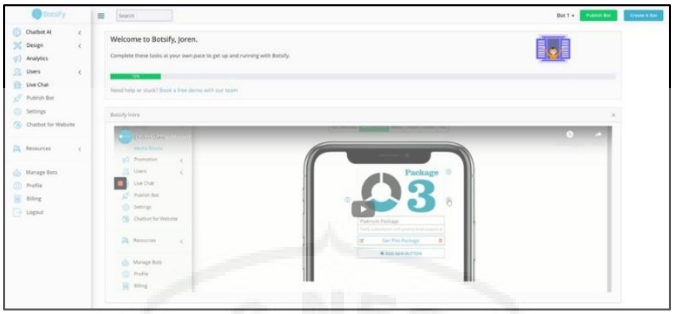


Gambar 21. Chatbot Builder MobileMonkey

5. **Botsify**

Botsify adalah teknologi bot yang instan dan berkualitas. Platform ini bisa Anda integrasikan di Facebook Messenger, Slack, dan Website. Penggunaannya yang sederhana tidak mengurangi kualitas dalam menanggapi pesan pelanggan yang kompleks. Sekalipun ada pesan sulit ditangani, bot ini akan mengarahkan otomatis ke pihak *customer service* yang Anda punya. Platform Botsify bisa dicoba dengan layanan *free trial* untuk 14hari. Sisanya, Anda bisa mengupgrade atau meningkatkan layanan menjadi layanan pro. Berikut adalah fitur-fitur lainnya yang bisa Anda nikmati dari botsify:

a. Terintegrasi dengan lebih dari 1000 platforms.
b. Penggunaan yang mudah dan sederhana dengan fasilitas yang dapat disesuaikan dengan selera pembuatnya.
c. Menyediakan CMS untuk menyimpan data secara lengkap.
d. Tools analytics untuk tracking data *real time*.
e. Fitur *hybrid* untuk beralih dari chatbot ke agen yang dioperasikan manusia dengan cepat.

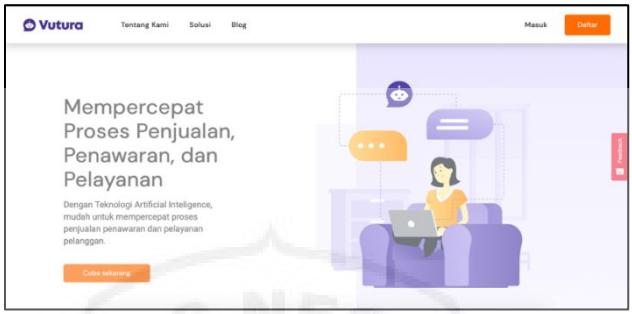


Gambar 22. Chatbot Builder Botsify

6. Vutura Chatbot Indonesia

Chatbot Vutura adalah chatbot asli buatan Indonesia. Vutura memiliki banyak keunggulan yang juga tidak kalah dengan platform lainnya, yaitu memberikan kemudahan bagi para pebisnis untuk menciptakan teknologi bot secara instant. Anda dapat menggunakan platform ini untuk menciptakan bot pada Line, Facebook Messenger, Telegram, hingga website. Aplikasi yang dirintis di Jakarta pada tahun 2018 ini mengusung rangkaian fitur terbaik yang bisa menjunjang kualitas pelayanan bisnis. Berikut ini adalah beberapa fitur yang dapat digunakan ketika menggunakan Vutura Chatbot:

a. Menyediakan template siap pakai (*ready to use*).
b. Penggunaannya yang fleksibel dengan membuat pesan dalam bentuk teks dan gambar.
c. Analisa lebih effisien dengan fitur analytics.
d. Terdapat tools helpdesk untuk menggabungkan kapasitas bot dengan manusia secara bersamaan. Seperti tracking agen, fitur antrian agen, ambil alih percakapan dan analisis interaksi pelanggan.
e. Cocok untuk berbagai macam tipe bisnis.
f. Sudah dipercayai oleh berbagai perusahaan ternama di Indonesia.

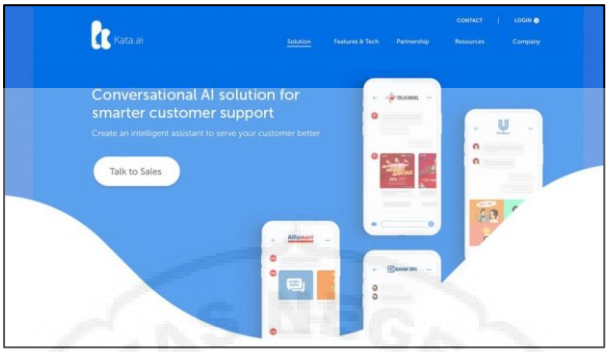


Gambar 23. Chatbot Builder Vutura

7. Kata.ai

Selain vutura, ada juga platform chatbot kedua terbaik asli kreasi lokal, yaitu Kata.ai. berdiri sejak tahun 2015, platform yang awalnya bernama Yesboss ini menawarkan teknologi asisten virtual alis bot dengan jejeran kelebihan. Anda bisa memanfaatkan platform ini untuk mendukung dari setiap elemen bisnis mulai dari pemasaran, pelayanan, hingga sebagai alat untuk mengumpulkan data kebiasaan-kebiasan dari pelanggan. Berikut ini adalah beberapa fitur-fitur terbaiknya:

a. Bekerja dengan sistem NLP (*natural language processing*) sehingga bot bisa berbicara dengan senatural mungkin layaknya manusia.
b. Pemakaian yang simpel dengan fitur *drag and drop*.
c. Ada dukungan kode JavaScript untuk menciptakan CMS secara kustom.
d. Memiliki tools analytics.
e. Bisa mengintegrasikan projek pengguna dengan *prebuilt module* kata.ai untuk menghasilkan kualitas bot yang lebih canggih.
f. Tersedia dua pilihan bahasa yaitu Bahasa Indonesia dan Bahasa Inggris.



Gambar 24. Chatbot Builder Kata.ai

BAB III
METODE DAN ALGORITMA CHATBOT

A. *Artificial Intelligence* (AI)

Kecerdasan buatan atau *artificial intelligence* adalah suatu bidang studi yang mempelajari cara membuat sebuah mesin yang dapat berinteraksi layaknya manusia, serta memberi respon pada situasi tertentu (Naven dkk, 2016). Menurut Bayu (2016), metode-metode yang digunakan dalam pengimplementasian AI adalah sebagai berikut:

1. *Fuzzy Logic*

Teknik ini digunakan oleh mesin untuk mengadaptasi bagaimana makhluk hidup menyesuaikan kondisi dengan memberikan keputusan yang tidak kaku 0 atau 1. Sehingga dimunculkan sistem logika *fuzzy* yang tidak kaku. Penerapan logika *fuzzy* ini salah satunya adalah untuk sistem pengereman kereta di jepang.

2. *Evolutionary Computing*

Metode ini menggunakan skema evolusi yang menggunakan jumlah individu yang banyak dan memberikan sebuah ujian untuk menyeleksi individu terbaik untuk membangkitkan generasi selanjutnya. Seleksi tersebut digunakan untuk mencari solusi dari suatu permasalahan. Contoh dari pendekatan ini adalah Algoritma Genetika yang menggunakan ide mutasi dan kawin silang, *Particle Swarm Optimization* (PSO) yang meniru kumpulan binatang seperti burung dan ikan dalam mencari mangsa, *Simulated Anneling* yang menirukan bagaimana logam ditempa.

3. *Machine Learning*

Metode ini merupakan metode paling populer karena banyak digunakan untuk menggantikan atau menirukan perilaku manusia untuk menyelesaikan masalah. Ciri khas dari *Machine Learning* adalah adanya proses pelatihan, pembelajaran, atau *training*, sehingga *machine learning*

membutuhkan data untuk dipelajari disebut sebagai data *training*.

B. *Artificial Intelligence Markup Language* (AIML)

Artificial Intelligence Markup Language (AIML) adalah bahasa *scripting interpreter* yang merupakan turunan dari *Extensible Markup Language* (XML) dengan fungsi lebih spesifik. Salah satu fungsinya adalah membuat sistem *stimulus-response* berbasis pengetahuan. Dokumen AIML terdiri dari objek-objek yang dipisahkan oleh *tag-tag* tertentu layaknya dokumen XML atau HTML (Fernando, 2008).

Objek AIML tersusun atas unit-unit yang disebut *topics* dari *categories*, berisi data yang sudah ter-*parsing*. Data yang ter-*parsing* berisi karakter-karakter, beberapa diantaranya berupa data karakter, yang lainnya dapat berupa elemen AIML. Elemen AIML mengkapsulasi pengetahuan dalam bentuk *stimulus-response* di dokumen.

AIML berisi kumpulan pola dan respon yang dapat digunakan oleh chabot untuk penelusuran jawaban setiap kalimat yang diberikan. *Interpreter* AIML diperlukan untuk menerima input dan melakukan penelusuran jawaban pada dokumen AIML. Saat ini tersedia banyak *interpreter* AIML dalam berbagai bahasa pemrograman sehingga proses pembuatan *chatbot* dapat terfokus pada penyusunan dokumen AIML.

Tag-tag penting dari AIML adalah sebagai berikut:

1. AIML

Setiap file AIML dimulai dengan *tag* AIML dan diakhiri dengan *tag* AIML. Gambar 25 menunjukkan penggunaan *tag* AIML. Pada baris 1 menunjukkan versi dari AIML dan atribut *encoding* yang digunakan. Atribut *encoding* tersebut menunjukkan tipe karakter *encoding* yang digunakan, pada Gambar 1 tipe encoding yang digunakan adalah "UTF-8".

```
1 <aiml version="1.0.1" encoding="UTF-8"?>
2 <category>
3    <pattern> HELLO BOT </pattern>
4    <template>
5       Hello my new friend!
6    </template>
7 </category>
8 </aiml>
```

Gambar 25. Penggunaan *Tag* AIML

2. *Category*

Tag category adalah unit dasar dari pengetahuan AIML. Satu *tag category* terdiri dari:

a. *Input* pengguna yang dapat berupa kalimat pernyataan, pertanyaan maupun kalimat lainnya.
b. Respon dari *chatbot* terhadap *input* dari pengguna.
c. Konteks-konteks tambahan.

Pada Gambar 25, baris ke 2 dan ke 7 menunjukkan pembuka dan penutup dari *tag category*. Kemudian *tag category* terdapat *tag pattern* yang merupakan *input* pengguna dan *tag template* yang merupakan respon dari *chatbot* ke pengguna jika *input* pengguna sesuai dengan isi dari *tag pattern*.

3. *Pattern*

Tag Pattern merupakan *tag* yang berisi kalimat *input* pengguna. Setiap *tag category* harus memiliki 1 *tag pattern*, dan harus merupakan *tag* pertama yang dideklarasikan setelah *tag category*. Pada Gambar 25, baris ke 3 merupakan *input* dari pengguna. Jika pengguna memasukkan "*Hallo Bot*" sebagai *input* maka chatbot akan mengerti dan menjawab sesuai dengan jawaban yang sudah disediakan.

4. *Template*

Tag Template merupakan *tag* yang berisi dengan jawaban yang akan diberikan oleh chatbot kepada pengguna. Pada Gambar 25, baris ke 4 sampai ke 6 merupakan jawaban chatbot jika pengguna memasukkan masukan berupa "*Hello Bot*".

5. *Star/Wildcard*

Tag Star menangkap kata-kata tertentu dari masukan pengguna. *Tag Star* bisa juga ditambahkan atribut *index*= ”n” sehingga menjadi *star index*= ”n”, yang berfungsi untuk menangkap kata-kata tertentu pada indeks ke-n, dimana n berupa angka 1 sampai dengan n.

Gambar 26 merupakan contoh implementasi *tag star* dan *star index*= “n”.

```
1  <category>
2     <pattern> I LIKE * </pattern>
3     <template>
4        I like <star/> too.
5     </template>
6  </category>
7
8  <category>
9     <pattern> A * IS A * </pattern>
10    <template>
11       When a <star index="1"/> is not a <star index="2"/>?
12    </template>
13 </category>
```

Gambar 26. Pengguna *Tag Star* dan *Star Index*

Pada Gambar 26, jika pengguna memasukkan *input* berupa “*I Like Programming*” maka chatbot akan memberikan respon berupa “*I Like Programming too*”. Dapat dilihat bahwa tanda * pada *tag pattern* merupakan kata yang akan ditangkap oleh *tag star*, sehingga jika dituliskan “*I Like Programming*” maka kata programming yang akan ditangkap oleh chatbot dan menggantikan *tag star* yang terdapat pada *tag template* sehingga respon dari chatbot menjadi “*I Like Programming too*”. Sedengkan pada baris ke 8 sampai ke 13, *star index*= “2” akan menangkap tanda * kedua pada *tag pattern* juga. Sehingga jawaban yang dikeluarkan chatbot jika pengguna memasukkan *input* berupa “*A Man Is A Male*” adalah “*When a man is not a male?*”.

6. SRAI

Tag SRAI digunakan untuk memanggil jawaban pada *tag template* lain, sehingga membuat chatbot AIML menjadi lebih efisien. Terdapat beberapa teknik yang digunakan dengan memanfaatkan *tag* SRAI, yaitu:

a. *Symbolic Reduction*
b. *Divide and Conquer*
c. *Synonyms Resolutions*
d. *Keyword Detection*

7. *Random* dan *Li*

Tag Random digunakan supaya chatbot bisa memberikan jawaban bervariasi, dan setiap variasi jawaban harus dibatasi dengan *tag li*. Gambar 27 menunjukkan penggunaan *tag random*. Jika pengguna memasukkan *input* berupa "*Hi*" maka chatbot akan memberikan salah satu jawaban yang dibatasi *tag li* secara acak.

```
<category>
  <pattern> HI </pattern>
  <template>
     <random>
        <li> Hi! Nice to meet you </li>
        <li> Hello, How are you? </li>
        <li> Hello! </li>
     </random>
  </template>
</category>
```

Gambar 27. Penggunaan *Tag Random*

8. *Set* dan *Get*

Tag Set dan *Tag Get* memungkinkan chatbot untuk menggunakan variabel yang sudah dideklarasikan sebelumnya, yang biasa digunakan untuk menyimpan informasi tentang chatbot seperti nama atau properti lainnya. *Tag set* digunakan untuk mendeklarasikan variabel. *Tag set* harus digunakan diantara *tag template*. Gambar 28 menunjukkan penggunaan *tag set*.

```
<category>
  <pattern> MY NAME IS * </pattern>
  <template>
     Hello <set name="nameUser"> <star/> </set>
  </template>
</category>
```

Gambar 28. Penggunaan *Tag Set*

Pada Gambar 28, *tag set* akan mendeklarasikan *nameUser* dengan * yang dimasukkan oleh pengguna. Jika

pengguna memasukkan *input* berupa "*My Name is Andrew*", maka variabel *nameUser* akan berisi "*Andrew*".

Tag get akan mengembalikan variabel yang telah dideklarasikan dengan *tag set*. *Tag get* juga harus digunakan diantara *tag template*. Gambar 29 menunjukkan penggunaan *tag get*.

```
<category>
   <pattern> GOOD NIGHT </pattern>
   <template>
      Good night <get name="nameUser"/>
   </template>
</category>
```

Gambar 29. Penggunaan *Tag Get*

Pada Gambar 29, *tag get* akan memanggil variabel *nameUser* yang sebelumnya telah dideklarasikan dengan *tag set*. Jika pengguna memasukkan *input* berupa "*Good Night*", dimana "*Andrew*" adalah variabel *nameUser* yang telah dideklarasikan sebelumnya menggunakan *tag set*.

9. *That*

Tag That digunakan supaya chatbot dapat memberikan jawaban tambahan sesuai dengan *input* dari pengguna. Gambar 30 menunjukkan contoh implementasi *tag that* pada AIML.

```
<category>
   <pattern> MAKE SOME QUESTION </pattern>
   <template>
      Do you like movies?
   </template>
</category>

<category>
   <pattern> YES </pattern>
   <that> Do you like movies? </that>
   <template>
      Nice, I like movies too.
   </template>
</category>

<category>
   <pattern> NO </pattern>
   <that> Do you like movies? </that>
   <template>
      OK. But I like movies.
   </template>
</category>
```

Gambar 30. Penggunaan *Tag That*

Pada Gambar 30, jika pengguna memasukkan *input* berupa "*Make Some Question*" maka chatbot akan bertanya "*Do you like movies*?" kepada pengguna, jika pengguna

menjawab "*Yes*" maka chatbot akan memberi respon tambahan berupa "*Nice, I Like Movie too*", jika pengguna menjawab "*No*" maka respon chatbot adalah "*Ok, But I Like Movies*".

10. *Topic*

Tag Topic digunakan untuk mengelompokkan subyek atau topik yang bisa chatbot bicarakan. Untuk memanfaatkan *tag that, tag category* dengan topik yang sama dikelompokkan menjadi satu. Gambar 31 adalah contoh penggunaan *tag topic* pada AIML. Pada Gambar 31, jika pengguna memasukkan *input* berupa "*Let talk about flowers*" maka chatbot akan memberikan respon serta mendeklarasikan "*flowers*" sebagai variabel "*topic*", sehingga mempercepat pencarian jawaban, dimana jawaban yang dicari hanya terdapat diantara *tag topic* dengan *name*= "*flowers*".

```
<category>
   <pattern> LET TALK ABOUT FLOWERS. </pattern>
   <template>
      Yes <set name="topic">flowers</set>
   </template>
</category>

<topic name="flowers">
   <category>
      <pattern> * </pattern>
      <template>
         Flowers have a nice smell.
      </template>
   </category>

   <category>
      <pattern> I LIKE IT SO MUCH! </pattern>
      <template>
         I like flowers too.
      </template>
   </category>
</topic>
```

Gambar 31. Penggunaan *Tag Topic*

11. *Think*

Tag Think memiliki fungsi untuk menyimpan variabel tanpa memberitahu pengguna. Gambar 32 menunjukkan contoh *tag think* pada AIML.

```
<category>
   <pattern> MY NAME IS * </pattern>
   <template>
      <think> <set name="nameUser"> * </set> </think>
</template>
</category>
```

Gambar 32. Penggunaan *Tag Think*

12. *Condition*

Tag Condition digunakan jika:

a. Terdapat banyak jawaban yang bisa diberikan ke pengguna.
b. Jawaban yang paling tepat tergantung terhadap analisis dari variabel tertentu yang diperbaharui dari pembicaraan antara user dengan chatbot.

13. *Bot*

Tag Bot digunakan untuk mendeklarasikan informasi tentang chatbot, yang bisa ditampilkan kepada pengguna jika pengguna memasukkan *input* yang sesuai dengan *pattern* yang tersimpan.

C. *Levenshtein Distance*

Levenshtein Distance adalah sebuah matriks string yang digunakan untuk mengukur perbedaan atau jarak (*distance*) antara dua *string*. Nilai *distance* antara dua *string* ini ditentukan oleh jumlah minimum dari operasi-operasi perubahan yang diperlukan untuk melakukan transformasi dari suatu *string* menjadi *string* lainnya. Operasi-operasi tersebut adalah penyisipan (*insertion*), penghapusan (*deletion*), atau penukaran (*substitution*). Penjelasan tiap operasi dari *levenshtein distance*:

1. *Insertion*

Insertion berarti menyisipkan karakter ke dalam suatu *string*. Contohnya *string* 'disrit' menjadi *string* 'diskrit', dilakukan penyisipan karakter 'k' dibagian tengah *string*. Penyisipan bisa dilakukan di bagian *string* manapun. Tabel 1 menunjukkan ilustrasi dari operasi *insertion*.

Tabel 1. Ilustrasi dari Operasi *Insertion*

STRING 1	D	I	S	K	R	I	T
STRING 2	D	I	S	-	R	I	T
Insertion				K			

2. *Deletion*

Deletion adalah operasi yang dilakukan untuk menghilangkan karakter dari suatu *string*. Contohnya *string* 'nilais', karakter terakhir dihilangkan sehingga menjadi *string* 'nilai' dengan menghilangkan karakter 's'. Tabel 2 menunjukkan ilustrasi dari operasi *deletion*.

Tabel 2. Ilustrasi dari Operasi *Deletion*

STRING 1	N	I	L	A	I	-
STRING 2	N	I	L		I	S
Insertion						S

3. *Substitution*

Substitution adalah operasi penukaran karakter dengan karakter lain. Contohnya *string* 'gabel' menjadi *string* 'kabel'. Dalam kasus ini karakter 'g' yang terdapat pada awal *string* di substitusikan dengan karakter 'k'. Tabel 3 menunjukkan ilustrasi dari operasi *substitusi*.

Tabel 3. Ilustrasi dari Operasi *Substitution*

STRING 1	K	A	B	E	L
STRING 2	G	A	B	E	L
Insertion	K				

Cara perhitungan *distance* dari *levenshtein distance* dimulai dari pojok kiri atau sebuah matriks yang telah diisi sejumlah karakter *string* awal dan *string* terget. Entri-entri matriks tersebut mempresentasikan nilai terkecil dari transformasi *string* awal menjadi *string* target. Entri yang terdapat pada ujung kanan bawah matriks adalah nilai *distance* yang menggambarkan jumlah perbedaan dua *string*. Berikut ini adalah langkah-langkah dalam mendapat nilai *distance*:

Langkah 1: inisialisasi

a. Hitung panjang *string* awal dan *string* target, misalkan panjang *string* awal adalah m dan panjang *string* target adalah n.
b. Buat matriks berukuran m x n.

Langkah 2: proses

a. Periksa S[i] untuk 1 < I < n
b. Periksa T[j] untuk 1 < j < n
c. Jika S[i] = T[j], maka entrinya adalah nilai yang terletak pada tepat diagonal sebelah kiri, yaitu d[i,j] = d [i-1, j-1]
d. Jika S[i] !=T[j], maka entrinya adalah d[I,j] minimum dari:
    1) Nilai yang terletak tepat diatasnya, ditambah satu, yaitu d[i, j-1]+1
    2) Nilai yang terletak tepat dikirinya, ditambah satu, yaitu d[i-1, j]+1
    3) Nilai yang terletak pada tepat diagonal atas sebelah kirinya, ditambah satu, yaitu d[i-1, j-1] +1

Langkah 3: hasil dari entri matriks pada baris ke-I dan komlom ke-j, yaitu d[i, j].

Langkah 2 diulang hingga entri d[m, n] ditemukan.

Levenshtein distance melakukan perhitungan bobot *similarity* setelah mendapatkan nilai *distance* dari kedua string yang dibandingkan. Kemudian menggunakan suatu persamaan dalam menentukan bobot *similarity* yang ditunjukkan pada Persamaan 1.

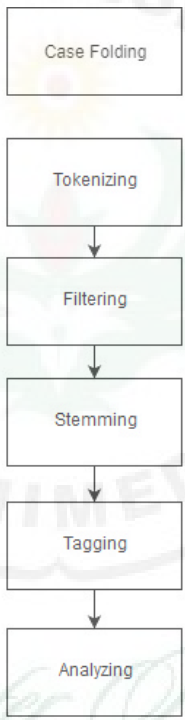
$$Bobot\ Similarity = \left(1 - \frac{d[m,n]}{Max\ (S,T)}\right) * 100\% \text{ (Pers. 1)}$$

Dengan d[m, n] adalah nilai *distance*, terletak pada baris ke m dan kolom ke n, S adalah panjang *string* awal, T adalah panjang *string* target, dan Max (S, T) adalah panjang terbesar *string* awal dan *string* target.

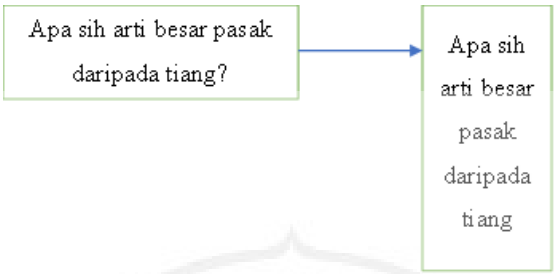
Bobot *similarity* diasumsikan pada rentang 0 (nol) hingga 100 (seratus) persen, yang artinya nilai 100 adalah nilai maksimum yang menunjukkan bahwa dua *string* tersebut adalah sama identik.

D. *Text Mining*

Text mining merupakan proses pencarian pola atau penggalian informasi baru. Tujuan dari *text mining* adalah menemukan informasi yang penting dari teks dengan mengubah teks menjadi data yang dapat digunakan untuk analisis yang lebih lanjut (Elisabet, 2013). Secara umum tahapan yang dilakukan pada *text mining*, yaitu *case folding*, *tokenizing*, *stemming*, *tagging*, dan *analyzing*.



Gambar 33. Tahapan *Text Mining*



Gambar 34. Tahapan *Case Folding*

1. ***Case Folding***

   Case Folding merupakan proses penghilangan karakter selain huruf 'a' sampai 'z'. karakter seperti tanda ',', '.', ':' dan lainnya akan dihilangkan. Proses *case folding* dapat dilihat pada Gambar 34.

2. ***Tokenizing***

   Tokenizing adalah proses mengurai kalimat menjadi kata-kata yang menyusunnya. Pada tahap ini juga dilakukan proses merubah semua huruf menjadi huruf kecil. Hanya huruf 'a' sampai 'z' yang diterima. Proses *tokenizing* seperti Gambar 35.

Gambar 35. Tahapan *Tokenizing*

3. *Filtering*

Filtering merupakan tahap pengambilan kata-kata penting dari hasil proses *tokenizing*. Pada tahapan *filtering* terdapat dua algoritma yang dapat digunakan yaitu algoritma *stoplist*/*stopword* (membuang kata yang kurang penting) atau *wordlist* (menyimpan kata penting). Contoh *stopwords* adalah "apa", "di", "pada", dan seterusnya. Jika terdapat kata *stoplist* dalam kalimat maka kata-kata tersebut akan di-*remove* dari kalimat sehingga kata-kata yang tersisa di dalam deskripsi dianggap sebagai kata-kata penting atau *keywords*.

4. *Stemming*

Stemming melibatkan pemotongan dan penghapusan imbuhan afiks guna untuk menghasilkan bentuk akar kata (kata dasar) dari sebuah istilah. Dalam mengambil informasi, *stemming* dapat digunakan untuk mengurangi bentuk dari istilah dan untuk menghindari terjadinya ketidak cocokan. Hal ini terjadi karena imbuhan bisa berisi informasi yang merupakan bagian dari perbincangan, plurarisme, dan lain-lain. Penting untuk diingat bahwasannya, pada proses *stemming* disini bukanlah tugas dari etimologi tata bahasa. Oleh karena itu, ketika algoritma *stemming* menghasilkan kata-kata yang tidak begitu berarti, maka akan dapat ditoleransi. Ada dua bagian pada algoritma *stemming* yaitu algoritma bebas konteks (*context-free*) dan sensitif konteks (*context-sensitive*). Algoritma *context free* bisa menghapus sufiks tanpa ada batasan, sedangkan pada *contect sensitive* menyertakan batasan yang banyak dengan konteks agar akhiran tidak dihapus, sehingga merusak hasil dari *stem*.

Dalam kasus *stemming* bahasa Indonesia ada beberapa pendekatan berbeda dipakai pada proses *stemming*-nya. Adapun pendekatan berbeda itu ialah ada yang menggunakan kamus pada proses *stemming*-nya dan ada yang tidak menggunakan kamus. Contohnya pada algoritma Vega yang tidak memerlukan kamus dalam prosesnya.

Sedangkan untuk algoritma lainnya seperti algoritma Nazief dan Andriani, algoritma *cinfix stripping*, algoritma *enhanced confix stripping*, dan algoritma idris akan memerlukan kamus dalam memproses kata. Algoritma *enchanced confix stripping* dikembangkan pada tahun 2008 oleh Putu Adhi Kerta Mahendra merupakan hasil perbaikan dari penelitian sebelumnya yakni memperbaiki beberapa kekurangan yang ada pada algoritma *confix stripping stemmer*. Adapaun hal yang dilakukan untuk memperbaikinya adalah dengan menambahkan juga mengubah aturan pemenggalannya. Kemudian menambahkan fitur pengembalian sufiks untuk antisipasi apabila terjadi kesalahan pemotongan yang tidak seharusnya dilakukan.

Proses *stemming* bahasa Indonesia yang dapat dilakukan dalam struktur morfologi Indonesia:

a. Penghapusan partikel
   Yang dapat dilakukan dalam proses ini yaitu dengan menghilangkan partikel seperti "-lah", "-kah", "-pun".
   Contoh:
   Kata awal= apakah, partikel = (-kah), kata setelah *stemming* = apa
b. Pembuangan posesif pronoun
   Yaitu dengan menghilangkan kata. Kata yang dihilangkan menbacu pada kepemilikan sesuatu. Seperti "-ku" dan "-mu".
   Contoh:
   Kata awal = rumahku, possessive pronoun = -ku, kata setelah *stemming* = rumahku, possessive pronoun = -ku, kata setelah *stemming* = rumah.
c. Pembuangan circumfix
   Yaitu kombinasi prefiks dan sufiks.
   Contoh:
   Kata awal = kecepatan, circumfix = ke -an, kata setelah *stemming* = cepat.

d. Pembuangan Prefiks

Nama lain dari awalan disebut prefiks. Akan dilakukan pemotongan prefiks pada proses stemming bahasa Indonesia.

Contoh:

Kata awal = memperkuat, prefix = memper, kata setelah *stemming* = kuat

e. Kombinasi Terlarang

Kombinasi antara prefiks dan sufiks disebut juga dengan kombinasi terlarang. Kombinasi ini mengakibatkan akar kata mengambil bentuk baru jika diberikan tambahan ini tetapi menjadi kata non-baku.

Contoh:

Awalan be-, akhir -i

Awalan me-, akhiran -an

E. Boyer Moore

Algoritma boyer moore merupakan algoritma pencocokan *string* yang paling efisien. Algoritma ini mencocokan karakter mulai dari karakter paling kanan hingga karakter paling kiri pada pola (*pattern*). Jika terjadi ketidak cocokan (*missmatch*), maka akan dicek nilai pergeserannya berdasarkan tabel *bad character shift* dan *good suffix shift*.

Algoritma boyer moore menggunakan dua buah tabel untuk mengolah informasi saat terjadi kegagalan pencocokan *pattern*. Tabel pertama disebut *badcharacter shift* juga sering disebut *occurrence heuristic* (OH). Tabel kedua disebut dengan istilah *good suffix shift* juga disebut *match heuristic* (MH).

Secara sistematis, langkah-langkah yang dilakukan algoritma boyer moore pada saat mencocokkan string:

1. Algoritma boyer moore mulai mencocokkan *pattern* pada awal teks.
2. Dari kanan ke kiri, algoritma ini akan mencocokan karakter per karakter *pattern* dengan karakter di teks yang bersesuaian, sampai salah satu kondisi berikut terpenuhi:

a. Karakter di *pattern* dan di teks yang dibandingkan tidak cocok (*missmatch*).
b. Semua karakter di *pattern* cocok kemudian algoritma akan memberitahukan penemuan di posisi ini.

3. Algoritma kemudian menggeser *pattern* dengan memaksimalkan nila penggeseran *good suffix* dan penggeseran *bad character*, lalu mengulangi langkah 2 sampai *pattern* berada di ujung teks (Susanto, 2014).

Perhitungan dengan algoritma boyer moore dapat dilihat pada contoh berikut ini:

Contoh,

Pattern (y) = "asa" → n = 3

Text (x) = "putus asa"

a. Perhitungan pergeseran *bad character*

i	0	1	2
c	a	s	a
bmBc [c]	2	1	0

Nilai pergeseran pada tabel didapat dengan perhitungan sebagai berikut:

bmBc[*y*[*i*]] = *n* -1 -*i*

bmBc[*y*[*0*]] = *3 -1 -0 = 2*

bmBc[y[*1*]] = *3 -1 -1 = 1*

bmBc[*y*[*2*]] = *3 -1 -2 = 0*

dan untuk karakter selain yang terdapat pada *pattern*, karakter tersebut bernilai sejumlah karakter *pattern* yaitu 3. Jika ada karakter yang berulang ambil nilai *BmBc* terkecil, dalam kasus ini adalah karakter "a" yang bernilai 2 dan 0. Maka jadikan karakter "a" bernilai 0.

i	0	1	2
c	a	s	a
bmBc [c]	0	1	0

b. Perhitungan nilai *suffix*

i	0	1	2
c	a	s	a
bmBc [c]	1	0	3

Langkah-langkah untuk menghitung nilai *suffix* adalah sebagai berikut:

1) Isi kolom *suffix* paling kanan dengan nilai *n* yaitu panjang karakter *pattern*

i	0	1	2
c	a	s	a
bmBc [c]			3

2) Membandingkan karakter dari kanan ke kiri
   Bandingkan karakter a dan s jika tidak cocok maka beri nilai 0
   Bandingkan karakter a dan a jika cocok maka jangan beri nilai

i	0	1	2
c	a	s	a
bmBc [c]		0	0

3) Pengisian kolom yang tidak bernilai dilakukan dengan memakai rumus **i -I -s. s** adalah nilai berapa kali pergeseran karakter yang sama. i – (i-s) = 1 – (1 -1) = 1

i	0	1	2
c	a	s	a
bmBc [c]	1	1	3

c. Perhitungan pergeseran *good suffix*

i	0	1	2
c	a	s	a
bmBc [c]	1	0	3

Langkah-langkah untuk menghitung nilai *bmGs* adalah sebagai berikut:

1) Isi *bmGs* sesuai dengan panjang karakter yaitu 3

i	0	1	2
c	a	s	a
Suff [i]	1	0	3
bmGs	3	3	3

2) Lakukan pengecekan dari kanan, lakukan pengecekan hingga i=0
   i=2, jika i + 1 = 3 dan ternyata *suffix* bernilai 3 maka tandai kolom
   i = 1, jika i + 1 = 2 dan ternyata *suffix* bernilai 0 maka jangan beri tanda pada kolom
   i = 0, jika i+1=1 dan ternyata *suffix* bernilai 1 maka tandai kolom

i	0	1	2
c	a	s	a
Suff [i]	1	0	3
bmGs	3	3	3

3) Masukkan rumus **n -1 -i** pada nilai yang telah diberi ceklist n -1-i = 3 -1 -2 = 0. Karena hasilnya 0 maka tidak melakukan perhitungan -1 -i = 3 -1 -1 = 1.
4) Ubahlah nilai indeks 0 dengan rumus **n -1 -i**, n -1 -i = 3 -1 -0 = 2

i	0	1	2
c	a	s	a
Suff [i]	1	0	3
bmGs	3	3	3

5) Lakukan pengecekan sampai **n -2** = 3 -2 = 1, artinya menghitung indeks dari 0 sampai 1
6) Msukkan rumus **n -1 -suffix** untuk mengisi kolom *bmGs*
   a) i = 0
      3 -1 -0 = 2, 2 adalah nilai i yang akan diisi pada kolom *bmGs*
      n -1 -i = 3 -1 -0 = 2

i	0	1	2
c	a	s	a
Suff [i]	1	0	3
bmGs	2	3	2

b) i = 1

3 -1 -1 = 1, 1 adalah nilai i yang akan diisi pada kolom *bmGs*

n -1 -i = 3 -1 -1 = 1

i	0	1	2
c	a	s	a
Suff [i]	1	0	3
bmGs	2	1	2

Contoh implementasi pencarian pada teks proses ke -1

Teks	p	u	t	u	s		a	s	a
Pattern	a	s	a						
Indeks	0	1	2						

bmBc = t; (teks) = 3

bmGs = indeks (2); *pattern* = 2

3 > 2 dengan begitu *pattern* digeser 3 karakter

Contoh implementasi pencarian pada teks proses ke -3

Teks	p	u	t	u	s		a	s	a
Pattern							a	s	A
Indeks							0	1	2

Semua karakter telah cocok, artinya *pattern* telah ditemukan di dalam teks.

F. Fuzzy String Matching

Fuzzy string matching adalah pencocokan *string* buram, artinya *string* yang disamakan mempunyai suatu kemiripan dari segi penyusunan karakter yang memiliki perbedaan (kemungkinan urutan atau jumlahnya), namun *string* tersebut tetap mempunyai kemiripan baik dari secara tekstual atau pun penulisan (*approximate string matching*) atau pun secara ucapan (*phonetic string matching*). Nilai bisa bernilai '*true*' dan '*false*' secara seiringan pada logika *fuzzy*. Tingkatan nilai '*true*' atau '*false*' pada logika *fuzzy* terkait dengan bobot kedudukan yang dipunya. Berbeda dengan logika digital yang hanya memiliki dua kedudukan 0 atau 1 saja pada satu waktu logika *fuzzy* memiliki tingkatan kedudukan *range* antara 0 hingga 1. Logika *fuzzy* sering dipakai dalam menyatakan suatu nilai dari terjemahan bahasa (*linguistic*), misalnya guna menyatakan suhu pada suatu ruangan, yang suhu ruangan tersebut dingin, hangat, atau kah panas.

Pada proses mencari kata yang dikerjakan oleh *bot*, bisa menggunakan metode *fuzzy string-matching* yang melakukan proses mengolah data inputan menghasilkan *output* yang *valid* dan *complete*. Bagaimana cara menentukan *string* yang ingin ditemukan mempunyai sesuatu yang sama dengan *string* yang ada di data, walaupun *string* tidak mirip persis dalam hal penyusunan karakter merupakan kunci dari konsep pencarian ini. Untuk menentukan kemiripan ini dipakai sebuah fungsi yang digunakan dengan istilah *similarity function*. Fungsi ini akan menentukan *string* hasil proses mencari jika didapati *string* hasil pendekatan (aproksimasi). *Fuzzy string matching* selalu beroperasi dengan mengkonsumsi jumlah sumber komputasi yang besar (Sebastian, 2016).

Ada beberapa variasi dari *fuzzy string matching*, diantaranya yakni *levensthein, damerau-levenshtein*, *hamming*, dan *sellers*. Algoritma-algoritma ini sebenarnya melakukan hal yang sama hanya menggunakan cara yang berbeda. Pada algoritma *fuzzy string matching* ukuran kuantitatif yang digunakan untuk mengukur kemiripan antara suatu *string* dengan *string* lainnya

dihitung dengan algoritma *levenshtein distance* atau sering disebut juga algoritma *edit distance*. Algoritma ini menghitung jumlah operasi penghapusan, penyisipan, dan penukaran yang harus dilakukan terhadap suatu *string* agar sama dengan *string* lain yang menjadi pembanding. Sebagai contoh, *string* "komputer" dengan "*computer*" memiliki distance 1 dikarenakan hanya membutuhkan dikerjakan satu pengoperasian saja agar terjadi proses perubahan satu *string* ke *string* yang lain. Pada yang terjadi dengan dua *string* di atas, *string* "*computer*" bisa menjadi "komputer" hanya dengan memproses satu penukaran karakter "c" ke karakter "k".

Perhitungan skor kemiripan antar dua *string* i dan j dilakukan dengan melakukan operasi perhitungan seperti dalam rumus berikut ini:

$$(1 - p/max(i,j))x\ 100\%$$

Dengan p adalah jumlah operasi yang dilakukan agar *string* i sama dengan j, dan max(i,j) adalah ukuran maksimum antara *string* i dan j, dimana *string* terpanjang menjadi patokan ukuran dalam perhitungan skor kemiripan. Skor kemiripan tertinggi adalah 100% dan skor kemiripan terendah adalah 0%.

sContoh:

Teks: HUKUM SHALAT DI RUMAH

Pattern: HUKUMNYA SOLAT DI RUMAH

HUKUM	SHALAT	DI	RUMAH
HUKUMNYA	SHOLAT	DI	RUMAH

Pada contoh di atas perbandingan karakter pertama sudah mengalami kecocokan, sebagaimana penjelasan di atas mengenai konsep *fuzzy string matching* ialah mencari kecocokan atau kemiripan antara *string* yang dicari dengan *string* yang ada di kamus data. Dalam kasus di atas kita bisa melihat bahwa ada kemiripan dalam beberapa kata yaitu:

HUKUM dengan HUKUMNYA

SHALAT dengan SHOLAT

BAB IV
CHATBOT TELEGRAM UNTUK PARIWISATA

A. Tahapan Persiapan

Hal pertama yang perlu diketahui sebelum membuat chatbot ini ialah spesifikasi minimum dari perangkat yang dibutuhkan. Tujuannya adalah agar pemakaian suatu program dapat berjalan dengan semestinya tanpa adanya interupsi dari alokasi hardware yang kurang. Berikut spesifikasi minimun perangkat keras dan perangkat lunak yang dibutuhkan untuk membangun chatbot telegram:

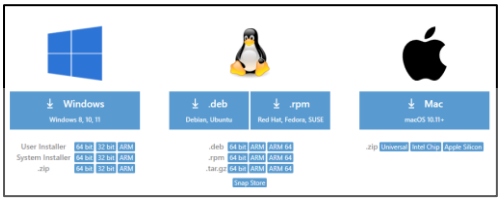
1. Prosesor 1,6 GHz
2. RAM 2 GB
3. HDD 250 GB
4. Sistem Operasi Windows 7 (32 bit / 64 bit) / MAC OS (10.11) / Ubuntu Desktop 16.04
5. Microsoft .NET Framework 4.5.2 diperlukan untuk VS Code. Jika Anda menggunakan Windows 7, pastikan .NET Framework 4.5.2 diinstal.

Dalam melakukan pembuatan chatbot ini, kita harus mempersiapkan beberapa kebutuhan yang akan digunakan dalam pembangunan chatbot telegram ini. Kebutuhan tersebut ialah sebagai berikut:

1. Visual Studio Code (VS Code)
2. Python
3. Python Library aiohttp dan pyTelegramBotAPI
4. Bot chat dari BotFather.

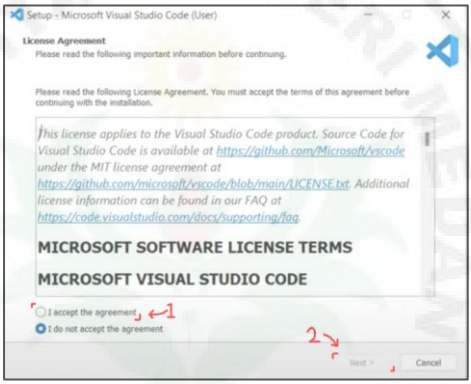
B. Instalasi Visual Studio Code

1. Pertama, buka link "https://code.visualstudio.com/" dan scroll down ke halaman bawah website untuk mendownload VS Code. Berikut ini untuk OS Windows.

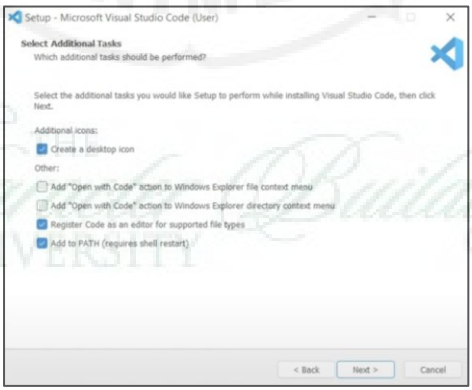


Gambar 36. Download Visual Studio Code

2. Jalankan file setup yang telah didownload dan pilih "*I accept the agreement*" lalu klik next.



Gambar 37. *SetUp* Visual Studio Code Bagian *License Agreement*



Gambar 38. SetUp Visual Studio Code Bagian *Additional Task*

3. Selanjutnya memilih tempat VS Code, secara default akan terletak pada "C:\Users\...\AppData\Local\Programs\Microsoft VS Code". Klik next, lalu klik next lagi. Setelah itu akan diberikan pemilihan additional task, pilih saja "Create a desktop icon" lalu klik next. Kemudian klik Install.

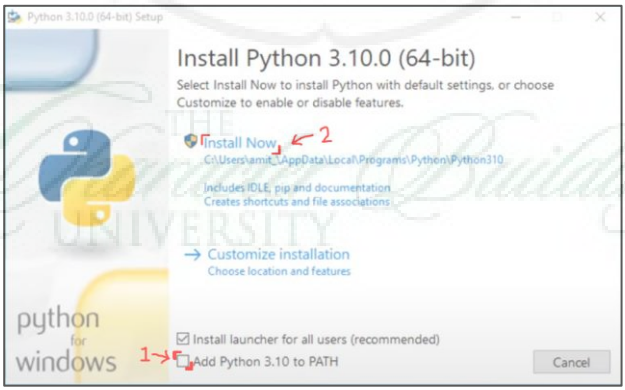
4. Setelah itu klik *Finish*. Dan penginstallan VS Code selesai.

C. Instalasi Python

1. Download Python pada link "https://www.python.org/downloads/", lalu pilih "Download Python".

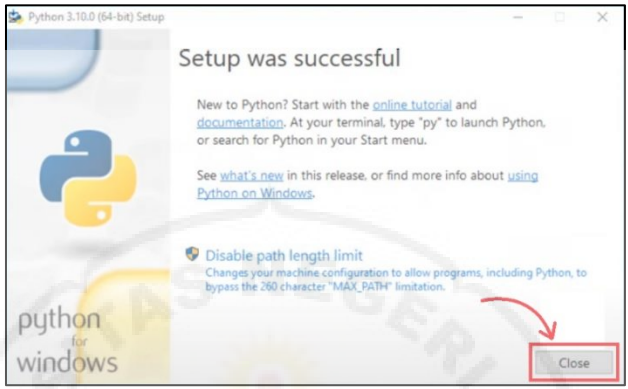
Gambar 39. Halaman Website untuk Mendownload Python

2. Buka file setup Python lalu pilih "Add Python to PATH" dan klik "*Install Now*".



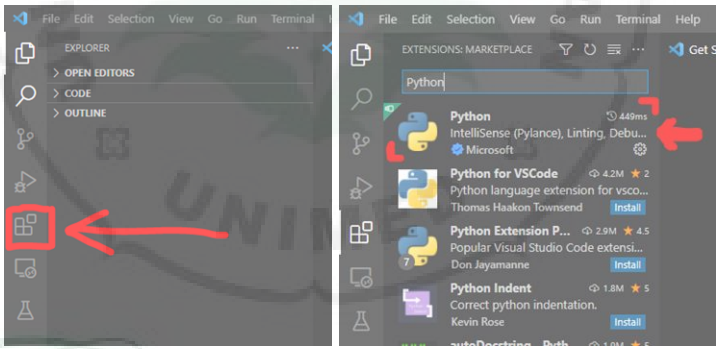
Gambar 40. *SetUp* Penginstallan Python

3. Terakhir klik "*Close*" dan penginstallan Python selesai.



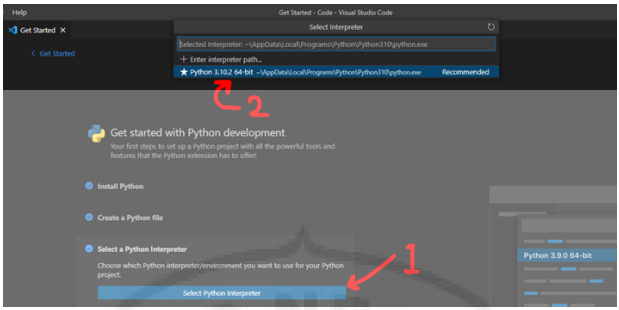
Gambar 41. Python Berhasil Diinstall

4. Setelah kalian mengistall software Visual Studio Code dan Python, buka VS Code lalu istall extension Python di bagian Extension.



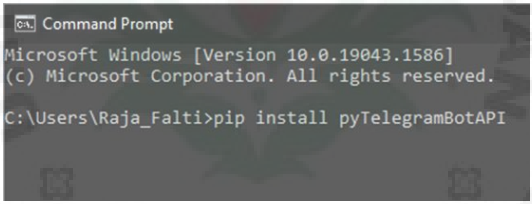
Gambar 42. Menginstall Extension Python pada VS Code

5. Setelah itu, masuk ke Get Started Python lalu klik di bagian "Select a Python Interpreter" dan pilih interpreter yang direkomendesikan.



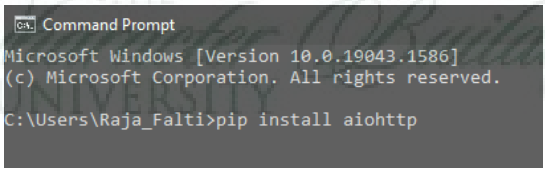
Gambar 43. Memilih Interpreter Python

6. Sekarang install library aiohttp dan pyTelegramBotAPI dengan cara membuka Command Prompt. Setelah Command Prompt terbuka, ketiklah didalamnya dengan:
   `pip install pyTelegramBotAPI`
   lalu tekan enter.



Gambar 44. Menginstall pyTelegramBotAPI pada Command Prompt

Setelah selesai, ketik lagi ini pada Command Prompt:
`pip install aiohttp`
lalu tekan enter.

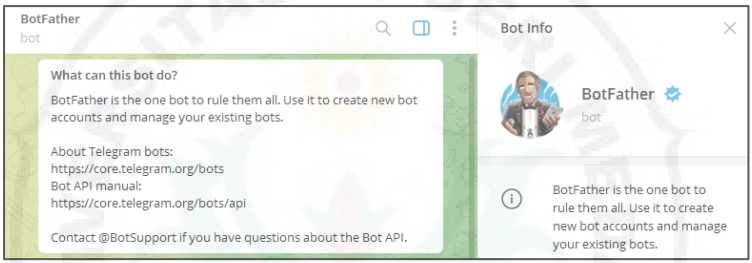


Gambar 45. Menginstall aiohttp pada Command Prompt

Untuk selanjutnya, kita akan masuk ke dalam tahapan pembuatan chatbot telegram.

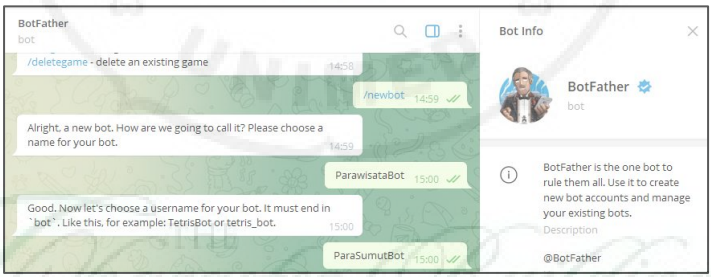
D. Pembuatan Kode Program

Hal pertama yang harus dilakukan sebelum membuat chatbot telegram ini, kita harus membuat bot-nya terlebih dahulu. Untuk membuat bot, kita cukup datang ke telegram "BotFather". BotFather merupakan telegram bot yang dapat membuat bot secara gratis. Untuk membuat bot, kita masuk ke telegram BotFather (https://t.me/botfather/). Jika masuk ke dalam chat BotFather pertama kali, akan ada opsi "START". Klik opsi tersebut lalu bot akan menjelaskan fungsinya.



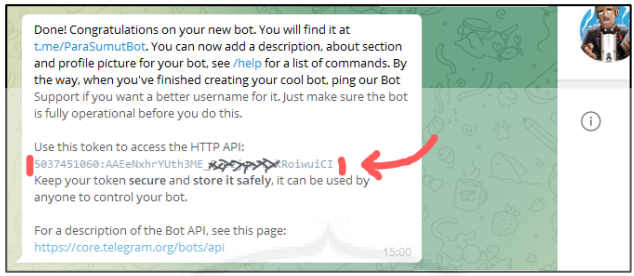
Gambar 46. Masuk Chat BotFather

Selanjutnya klik "/newbot" untuk membuat bot baru. Lalu masukan nama bot Anda dan juga username bot Anda.



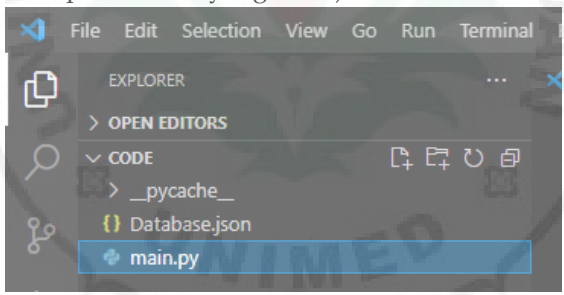
Gambar 47. Membuat Bot Baru dengan BotFather

Setelah membuat bot. Kita akan diberikan token untuk mengakses HTTP API. Token ini berguna untuk memprogram bot mu sesuai keinginanmu jadi simpan token tersebut (Token disensor).



Gambar 48. Bot telah berhasil dibuat

Klik "t.me/namabotmu" pada chat box seperti di atas untuk masuk ke chat bot mu. Setelah kita membuat bot baru. Selanjutnya kita akan melakukan koding kepada bot tersebut. Buka VS Code dan buat file Python baru (main.py) dan file JSON (Database.json) baru untuk databasenya (Ingat, semua file ini harus berada pada folder yang sama).



Gambar 49. Membuat File main.py dan Database.json

Setelah itu kita masukan code untuk file python (main.py (Full Code berada di Lampiran)). Pertama tulis kode untuk mengakses library python. Tulis code-nya seperti di bawah ini:

```
import json
import telebot
from telebot.async_telebot import AsyncTeleBot
from telebot import asyncio_filters
import asyncio
from telebot.types import InlineKeyboardMarkup,
InlineKeyboardButton
from telebot.asyncio_storage import StateMemoryStorage
from telebot.asyncio_handler_backends import State,
StatesGroup
```

Selanjutnya, kita buat variabel dengan memasukan token yang Anda dapat dari BotFather. Codenya ialah seperti yang ada di bawah ini (Saya sensor token milik saya).

```
tokenku = "5037451060:AAEeNxhrYUth3ME_8-
eejqx7LukRoiwuiCI"
parabot = AsyncTeleBot(tokenku,
state_storage=StateMemoryStorage())
```

Lalu buat class untuk membuat state bagi program untuk membaca file database yang dalam bentuk format json.

```
class Statusnya(StatesGroup):
 names = State()
 picts = State()
 descs = State()
 mapss = State()
```

Kemudian masukan code untuk membuka file database yang berformat json lalu masukan dalam bentuk variabel.

```
Sala = {}

def db(file):
 ope = open(file, )
 aps = json.load(ope)
 ope.close()
 return aps

loca = db('Database.json')
```

Selanjutnya kita buat bot untuk memulai chat jika kita menulis command "/start" pada chatbot. Tulis code seperti di bawah ini:

```
@parabot.message_handler(commands=['start'])
async def start(message):
 loca = db('Database.json')
```

Kemudian tulis code ini untuk menghapus chat yang diberikan chatbot sebelumnya.

```
for key in Sala.copy():
if not Sala[key]:
Sala.pop(key)
await parabot.delete_message(Sala[key], key)
del Sala[key]
```

Lalu masukan code untuk membuat chatbot memberikan kata pembuka.

```
sp = await parabot.send_message(message.chat.id, "Halo, nama saya Parasya. Saya suatu bot yang akan memberikan beberapa tempat wisata yang mungkin anda ingin kunjungi", parse_mode='HTML')
Sala[sp.message_id] = sp.chat.id
Sala[message.message_id] = message.chat.id
print(Sala)
```

Selanjutnya buat code untuk menampilkan opsi list-list parawisata yang telah dibuat dalam database. Codenya sebagai berikut:

```
mark = InlineKeyboardMarkup()
mark.row_width = 1
for i in loca:
mark.add(InlineKeyboardButton(f"{i}", callback_data=f"{i}"))
mark.add(InlineKeyboardButton("[Tidak ada pertanyaan lagi]", callback_data="Selesai"))
```

Code di bawah ini merupakan code dimana chatbot akan mengirimkan chat berikut sebelum list parawisata dikirimkan.

```
await parabot.send_message(message.chat.id, "Berikut merupakan beberapa tempat wisata yang bisa anda kunjungi.", parse_mode='HTML', reply_markup = mark)
```

Selanjutnya code berikut merupakan kode untuk membuat opsi kembali setelah tempat parawisata dipilih.

```
@parabot.callback_query_handler(func=lambda call: True)
async def callback_query(call):
 mark = InlineKeyboardMarkup()
 mark.row_width = 1
 mark.add(InlineKeyboardButton("[Kembali ke Menu]",
callback_data="kembali"))
```

Kemudian tulis code untuk menampilkan gambar, deskripsi, dan juga peta lokasi dari parawisata yang telah dipilih oleh user.

```
 await parabot.delete_message(call.from_user.id,
call.message.message_id)
 if call.data in loca:
 send_pict_load = await
 parabot.send_message(call.from_user.id,
 "<code>Processing...</code>", parse_mode='HTML')
 await
 parabot.send_photo(call.from_user.id,f'{loca[call.data]["picts"]}',
 f"<b>{call.data}</b> (<a href='{loca[call.data]['links']}'>Titik
 Lokasi</a>)\n\n{loca[call.data]['descs']}",
 parse_mode='HTML',reply_markup = mark)
 await parabot.delete_message(send_pict_load.chat.id,
 send_pict_load.message_id)
```

Lalu, buat code untuk menampilkan kembali opsi list-list parawisata ketika opsi kembali ditekan oleh user.

```
 elif call.data == "kembali":
 mark = InlineKeyboardMarkup()
 mark.row_width = 1
 for i in loca:
 mark.add(InlineKeyboardButton(f"{i}", callback_data=f"{i}"))
 mark.add(InlineKeyboardButton("[Tidak ada pertanyaan lagi]",
callback_data="Selesai"))
```

```
await parabot.send_message(call.from_user.id, "Silakan lihat
lokasi wisata lainnya", parse_mode='HTML', reply_markup =
mark)
```

Code berikut merupakan code dimana jika opsi [Tidak ada pertanyaan lagi] dipilih oleh user maka seluruh chat yang diberikan oleh chatbot akan dihapus.

```
elif call.data == "Selesai":
for key in Sala.copy():
if not Sala[key]:
Sala.pop(key)
await parabot.delete_message(Sala[key], key)
del Sala[key]
```

Lalu code di bawah ini akan mengirim chat kata sampai jumpa ketika opsi [Tidak ada pertanyaan lagi] dipilih oleh user.

```
cya = await parabot.send_message(call.from_user.id, "Semoga
perjalanan wisata anda menyenangkan!", parse_mode='HTML')
Sala[cya.message_id] = cya.chat.id
```

Code ini merupakan code dimana chatbot tidak mengerti input yang dimasukan oleh user.

```
else:
await parabot.answer_callback_query(call.id, "Pardon?")
```

Ini merupakan code untuk menjalankan chatbot dengan code yang telah ditulis dalam telegram.

```
parabot.add_custom_filter(asyncio_filters.StateFilter(parabot))
print("Bot Now Running")
asyncio.run(parabot.polling())
```

Selanjutnya merupakan kode yang akan ditulis dalam file JSON (Database.json). Code tersebut berupa nama; foto; deskripsi; dan lokasi wisata. Format kodenya sebagai berikut:

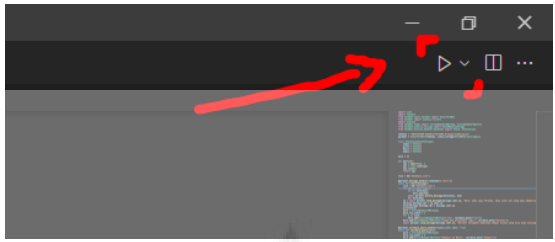
```
{
 "Nama Wisata": {
 "picts": "Link URL Gambar",
 "descs": "Deskripsi Wisata",
 "links": "Link URL Lokasi Wisata"
 }
}
```

Jika ingin menambahkan tempat wisata lagi, kita masukan koma setelah bracket wisata.

```
{
 "Nama Wisata": {
 "picts": "Link URL Gambar",
 "descs": "Deskripsi Wisata",
 "links": "Link URL Lokasi Wisata"
 },
 "Nama Wisata": {
 "picts": "Link URL Gambar",
 "descs": "Deskripsi Wisata",
 "links": "Link URL Lokasi Wisata"
 }
}
```

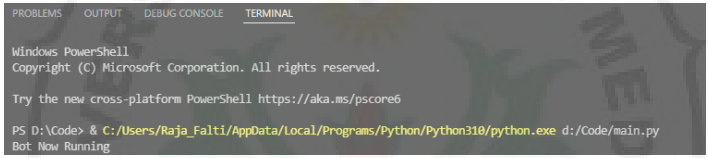
E. Tahap Akhir

Dalam tahapan ini, kita akan menjalankan program yang telah kita buat. Buka telegram, lalu buka chatbot yang telah dibuat. Lalu dalam VS Code, kita jalankan program utama yaitu program Python (main.py) dengan cara menekan "Run" di bagian kanan atas VS Code.



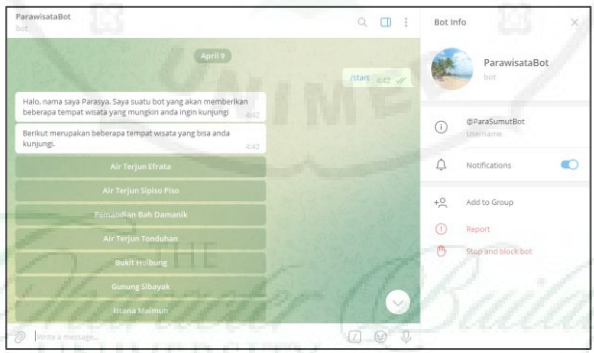
Gambar 50. Tombol Menjalankan (*Run*) Program main.py pada VS Code

Lalu liat bagian terminal yang ada di bawah sampai kata "Bot Now Running" muncul pada terminal.



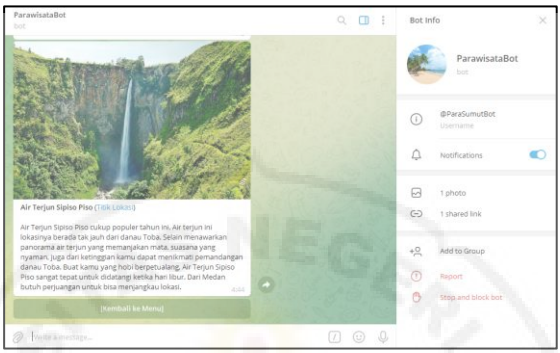
Gambar 51. Terminal VS Code

Selanjutnya kita buka chatbot telegram yang kita buat, lalu ketik "/start" untuk memulai chatbotnya.



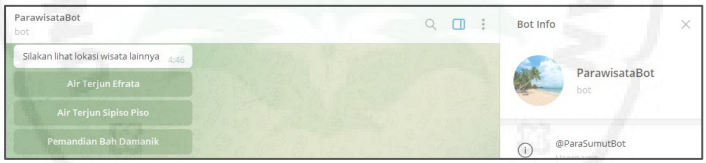
Gambar 52. Bot Telah Dibuat dan Dimulai

Kita dapat menekan opsi yang ada dalam chat. Ini jika saya tekan opsi "Air Terjun Sipiso Piso.



Gambar 53. Opsi "Air Terjun Sipiso Piso" ditekan dan ditampilkan

Tekan opsi [Kembali ke Menu] untuk kembali ke opsi pilihan wisata.

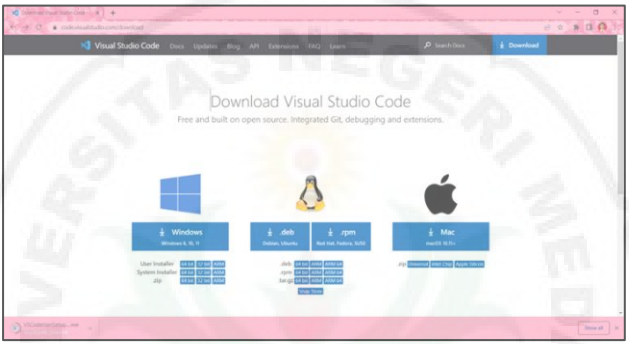


Gambar 54. Opsi "[Kembali ke Menu]" Ditekan dan Menampilkan Menu Utama

BAB V
CHATBOT WHATSAPP UNTUK PARIWISATA

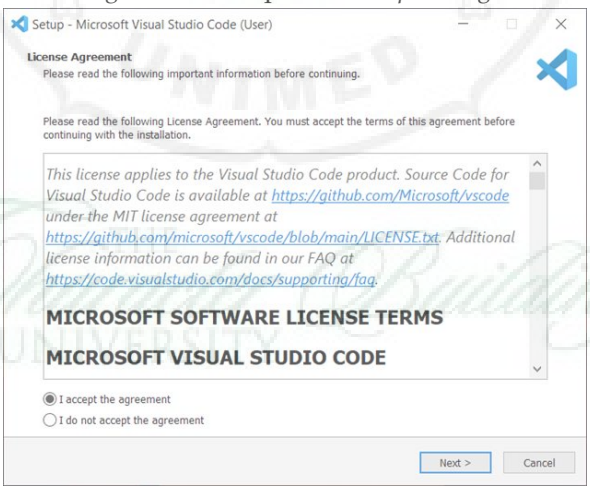
A. Konfigurasi Visual Studio Code

1. Mengunduh Visual Studio Code melalui: https://code.visualstudio.com/download sesuai dengan sistem operasi yang digunakan.



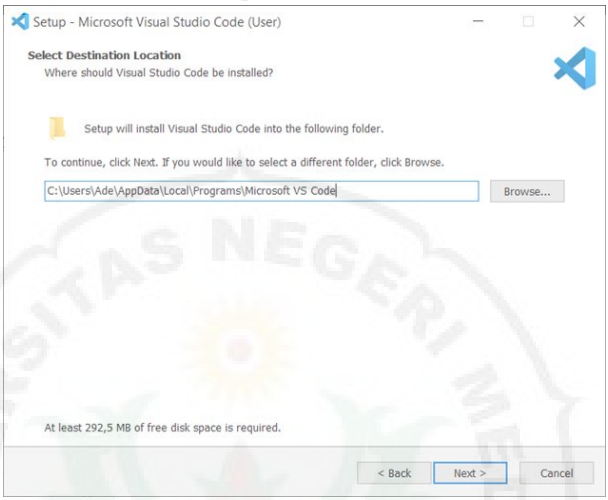
Gambar 55. Mengunduh VS Code

2. Melakukan instalasi visual studio code yang telah diunduh.
   a. Baca *license agreement* dan pilih "*I accept the agreement*".



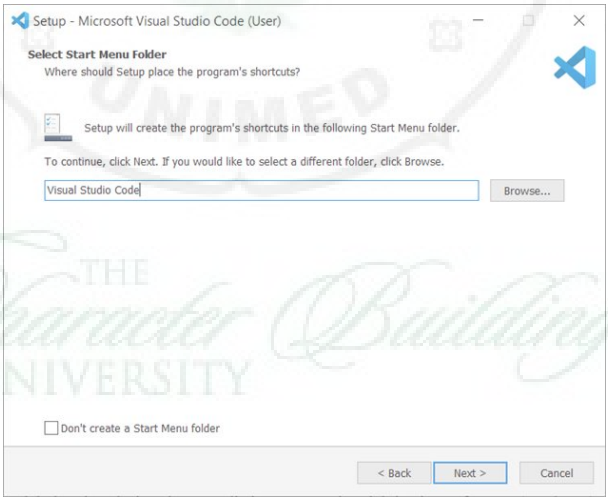
Gambar 56. *License Agreement*

b. Pilih lokasi penyimpanan untuk penginstalan visual studio code, kemudian pilih *Next*.



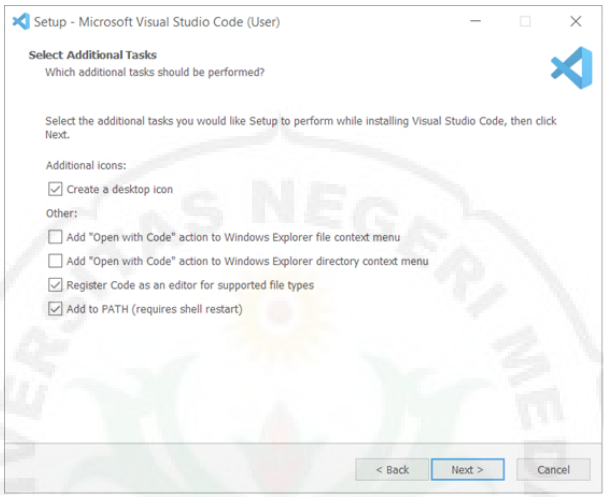
Gambar 57. *Select Destination Location*

c. Memilih lokasi penyimpanan *Start Menu Folder*, kemudian klik *Next*.



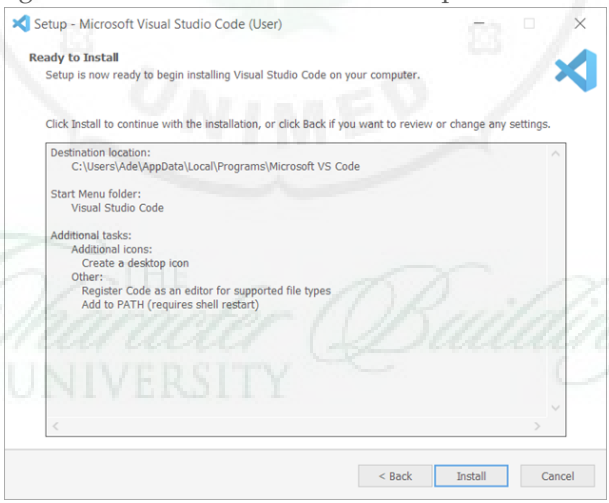
Gambar 58. *Select Start Menu Folder*

d. Mencentang pada bagian *Create a deskstop icon* untuk menampilkan *icon* visual studio code pada *desktop* setelah proses instalasi selesai.



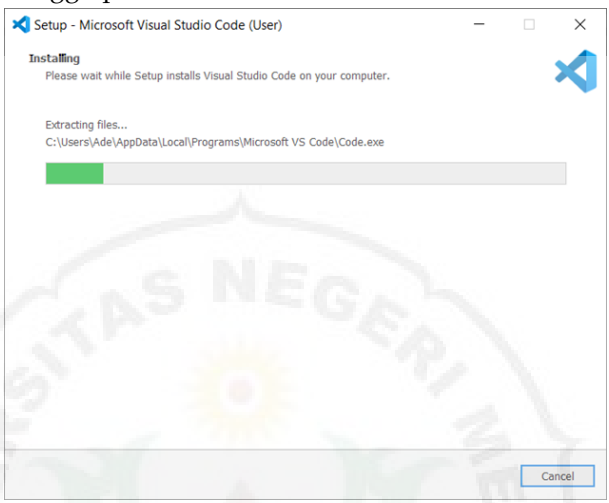
Gambar 59. *Select Additional Task*

e. Mengklik tombol *Install* untuk memulai proses instalasi.



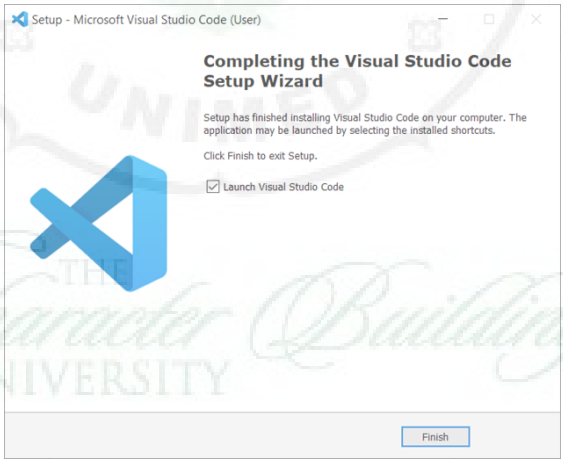
Gambar 60. *Ready to Install*

f. Menunggu proses instalasi selesai.



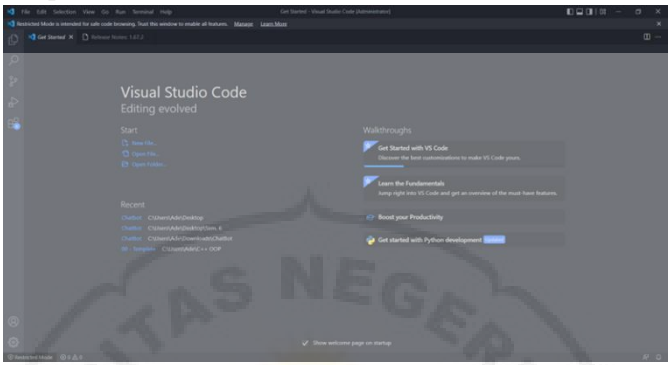
Gambar 61. *Installing*

g. Proses instalasi selesai, centang pada bagian *Launch Visual Studio Code*, untuk membuka aplikasi visual studio code. Kemudian klik "*Finish*".



Gambar 62. *Complete Setup*

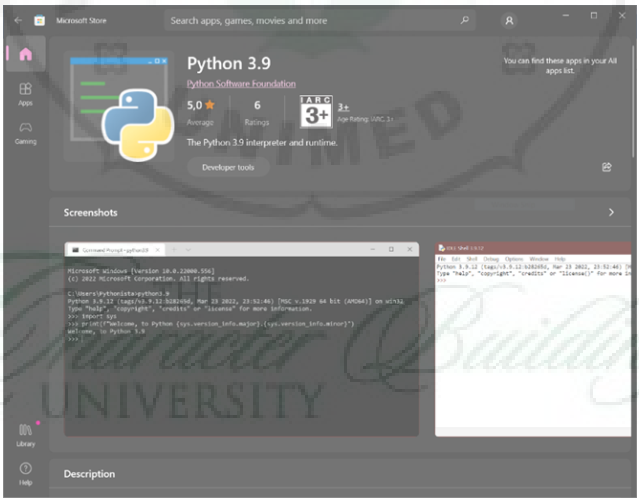
h. Tampilan awal visual studio code.



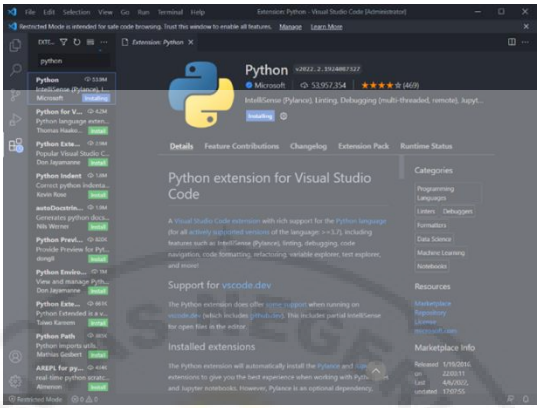
Gambar 63. Tampilan Visual Studio Code

B. Konfigurasi Python

1. Melakukan instalasi bahasa pemrograman Python 3.9 di sistem operasi melalui Microsoft Store (Gambar 64). Kemudian pada Visual Studio Code, lakukan isntalasi Python 3.9 melalui pencarian di menu Extensions (Gambar 65).



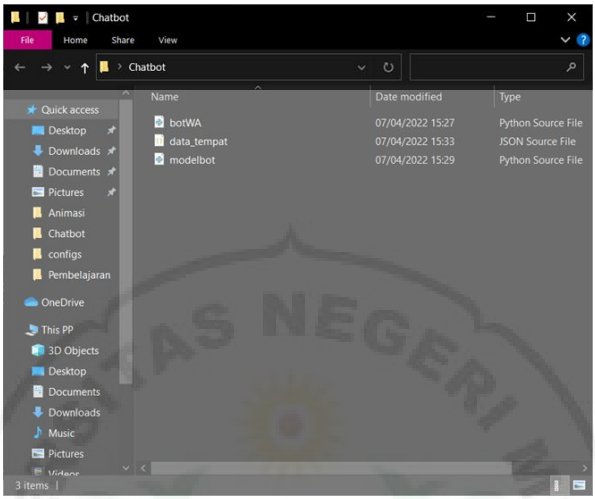
Gambar 64. Instalasi Python melalui Microsoft Store



Gambar 65. Instalasi Python melalui Extension VS Code

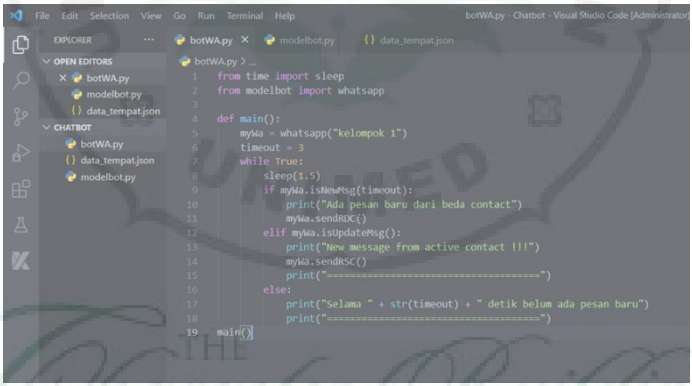
2. Membuat file program dengan extensi python, dengan nama botWA.py sebagai main program dan modelbot.py sebagai model chatbotnya. Membuat file data tempat dengan nama data tempat.json, yang akan digunakan sebagai penyimpanan informasi data tempat wisata di sumatera utara yang berupa deskripsi tempat, link gambar, dan link lokasi. Nantinya file .json ini akan dipanggil melalui modelbot.py. Simpan ketiga file ke dalam folder yang sama. Seperti tampak pada Gambar 66.

Gambar 66. Membuat File Python dan Json Baru



Gambar 67. Menyimpan File Python dan Json Dalam Satu Folder

3. Memasukkan kode program pada file **botWA.py**



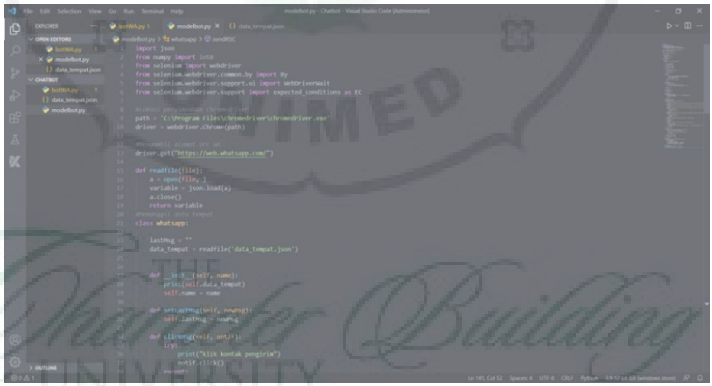
Gambar 68. Script Program pada File **botWA.py**

Kode Program:

```
from time import sleep
from modelbot import whatsapp
def main():
    myWa = whatsapp("kelompok 1")
    timeout = 3
    while True:
```

```
        sleep(1.5)
        if myWa.isNewMsg(timeout):
            print("Ada pesan baru dari beda contact")
            myWa.sendRDC()
        elif myWa.isUpdateMsg():
            print("New message from active contact !!!")
            myWa.sendRSC()

print("====================================="
)
        else:
            print("Selama " + str(timeout) + " detik belum ada
pesan baru")

print("====================================="
)
main()
```

4. Memasukkan kode program pada file **modelbot.py**



Gambar 69. Script Program pada File **modelbot.py**

Kode Program Keseluruhan:

```
import json
from numpy import int0
from selenium import webdriver
from selenium.webdriver.common.by import By
```

```
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC

#Lokasi penyimpanan Chromedriver
path = 'C:\Program Files\chromedriver\chromedriver.exe'
driver = webdriver.Chrome(path)
#Mengambil alamat Url WA
driver.get("https://web.whatsapp.com/")
def readfile(file):
  a = open(file, )
  variable = json.load(a)
  a.close()
  return variable
#Memanggil data tempat
class whatsapp:
  lastMsg = ""
  data_tempat = readfile('data_tempat.json')
  def __init__(self, name):
    print(self.data_tempat)
    self.name = name
  def setLastMsg(self, newMsg):
    self.lastMsg = newMsg
  def clickMSg(self, notif):
    try:
      print("klik kontak pengirim")
      notif.click()
    except:
      print("Gagal Klik")
  def getMsg(self):
    msg = ""
    try:
      for msgIn in driver.find_elements_by_class_name("message-in"):
```

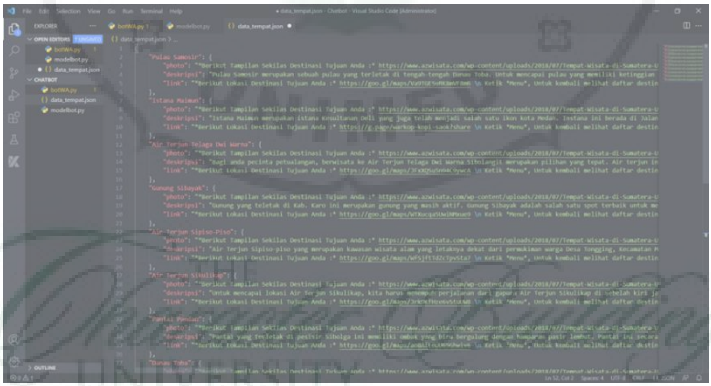
```
            msgIn                                                    =
msgIn.find_elements_by_class_name("_1Gy50")
            msgIn = msgIn[len(msgIn)-1]
            msg = msgIn.text
        except:
            print("Can't found last message")

        return msg
    def isNewMsg(self, to):
        print("Mengecek pesan baru . . .")
        try:
            element = WebDriverWait(driver, to).until(

EC.presence_of_element_located((By.CLASS_NAME,
"_23LrM"))
            )
        except:
            return False
        return True
    def isEmptyMsg(self, str):
        if(len(str) == 0):
            return True
        return False
    def isUpdateMsg(self):
        updateMsg = self.getMsg()
        if (self.lastMsg != updateMsg) and (not
self.isEmptyMsg(updateMsg)):
            self.lastMsg = updateMsg
            return True
        return False
    def processMsg(self, msgIn):
        msg = msgIn.lower()
        responseMsg = "Mohon maaf, coba dengan kata *Hai*"

        if msg.find("hello") != -1:
```

```
        responseMsg = "Hai, Selamat datang di *BOT Pariwisata Sumatera Utara*. \n Kami akan memberikan rekomendasi destinasi wisata yang populer di *Sumatera Utara* untuk Anda. \n Jika Anda tertarik silakan balas pesan ini dengan *Ya* "

      if msg.find("hai") != -1:
         responseMsg = "Hello, Selamat datang di *BOT Pariwisata Sumatera Utara*. \n Kami akan memberikan rekomendasi destinasi wisata yang populer di *Sumatera Utara* untuk Anda. \n Jika Anda tertarik silakan balas pesan ini dengan *Ya* "

      if msg.find("ya") != -1:
         responseMsg = "*Berikut destinasi wisata yang bisa kami rekomendasikan untuk Anda.* \n Mohon balas dengan mengetikkan nama lokasi pilihan Anda sesuai daftar yang tertera pada list di bawah ini (contohnya ketik: *Pantai Pandan*) \n"
         num = 0
         for i in self.data_tempat:
            num = num + 1
            responseMsg += f"{num}. {i} \n"

      if msg.find("menu") != -1:
         responseMsg = "*Berikut destinasi wisata yang bisa kami rekomendasikan untuk Anda.* \n Mohon balas dengan mengetikkan nama lokasi pilihan Anda sesuai daftar yang tertera pada list di bawah ini (contohnya ketik: *Pantai Pandan*) \n"
         num = 0
         for i in self.data_tempat:
            num = num + 1
            responseMsg += f"{num}. {i} \n"

      if msg.find("selesai") != -1:
```

```
        responseMsg = "Terima kasih sudah mengunjungi
BOT kami, Selamat Berwisata!"

      if msg.find("ok") != -1:
         responseMsg = ":D"

      if msg.find("?") != -1:
         responseMsg += "Mohon maaf. Layanan belum bisa
memproses pertanyaan anda! "

      for i in self.data_tempat:
         if msg.find(f"{i.lower()}") != -1:
            responseMsg = f"{i} \n
{self.data_tempat[i]['photo']} \n
{self.data_tempat[i]['deskripsi']} \n
{self.data_tempat[i]['link']}"

      return responseMsg

   def postMsg(self, msg):
      try:
         textInput =
driver.find_elements_by_class_name("_13NKt")
         textInput[len(textInput)-1].send_keys(msg)
         try:
            send =
driver.find_element_by_class_name("_4sWnG")
            send.click()
         except:
            print("Gagal Klik")
      except:
         print("tidak bisa input")
      print("Response me : " + msg)

   #mengirimi jawaban ke kontak yang berbeda
   def sendRDC(self):
```

```
        contacts                                        =
driver.find_elements_by_class_name('_23LrM')
        for notif in contacts:
            self.clickMSg(notif)
            self.lastMsg = self.getMsg()
            print("Pesan terakhir adalah : " + self.lastMsg)
            self.postMsg(self.processMsg(self.lastMsg))

print("======================================"
)

    #mengirimi jawaban ke kontak yang sama atau active
    def sendRSC(self):
        print("Pesan terakhir adalah : " + self.lastMsg)
        self.postMsg(self.processMsg(self.lastMsg))
```

5. Mengisi data ke dalam file **data_tempat.json** yang berupa 10 daftar tempat wisata beserta deskripsi tempat wisata, link gambar, dan link google maps lokasi wisata.



Gambar 70. Script Program pada File **data_tempat.json**

Kode program keseluruhan:

```
from time import sleep
from modelbot import whatsapp
def main():
    myWa = whatsapp("kelompok 1")
```

```
    timeout = 3
    while True:
        sleep(1.5)
        if myWa.isNewMsg(timeout):
            print("Ada pesan baru dari beda contact")
            myWa.sendRDC()
        elif myWa.isUpdateMsg():
            print("New message from active contact !!!")
            myWa.sendRSC()

print("====================================")
        else:
            print("Selama " + str(timeout) + " detik belum ada pesan baru")

print("====================================")
main(){
   "Pulau Samosir": {
      "photo": "*Berikut Tampilan Sekilas Destinasi Tujuan Anda :* https://www.azwisata.com/wp-content/uploads/2018/07/Tempat-Wisata-di-Sumatera-Utara-Pulau-Samosir.jpg",
      "deskripsi": "Pulau Samosir merupakan sebuah pulau yang terletak di tengah-tengah Danau Toba. Untuk mencapai pulau yang memiliki ketinggian 1,000 meter di atas permukaan laut ini, Anda dapat menggunakan kapal ferry atau menempuh jalur darat.\n\nDi pulau ini, Anda akan menjumpai pemandangan alam yang menarik dan indah. Selain itu, Anda juga bisa memperoleh pengalaman baru dengan menikmati keunikan budaya, sejarah juga kuliner khas Batak.\n\nDi pulau ini juga terdapat sebuah pemandian air panas yang siap memanjakan Anda.",
      "link": "*Berikut Lokasi Destinasi Tujuan Anda :* https://goo.gl/maps/Va9TGESuRK8mVF8m6 \n Ketik
```

Menu, Untuk kembali melihat daftar destinasi wisata. \n ketik *Selesai*, untuk mengakhiri BOT ini. "
},
"Istana Maimun": {
"photo": "*Berikut Tampilan Sekilas Destinasi Tujuan Anda :* https://www.azwisata.com/wp-content/uploads/2018/07/Tempat-Wisata-di-Sumatera-Utara-Istana-Maimun.jpg",
"deskripsi": "Istana Maimun merupakan istana Kesultanan Deli yang juga telah menjadi salah satu ikon kota Medan. Instana ini berada di Jalan Brigadir Jenderal Katamso, Kelurahan Sukaraja, Kecamatan Medan Maimun.\n\n Jika anda gemar berwisata ketempat tempat bersejarah, maka Istana Maimun adalah tempat yang pas untuk anda datangi.\n\n Selain dari sejarahnya, Istana ini juga memiliki desain interior yang unik, dimana paduan unsur warisan kebudayaan Melayu tergabung dengan gaya Islam, Spanyol, India serta Italia yang dapat membuat para wisatawan terpukau.",
"link": "*Berikut Lokasi Destinasi Tujuan Anda :* https://g.page/warkop-kopi-saok?share \n Ketik *Menu*, Untuk kembali melihat daftar destinasi wisata. \n ketik *Selesai*, untuk mengakhiri BOT ini. "
},
"Air Terjun Telaga Dwi Warna": {
"photo": "*Berikut Tampilan Sekilas Destinasi Tujuan Anda :* https://www.azwisata.com/wp-content/uploads/2018/07/Tempat-Wisata-di-Sumatera-Utara-Air-Terjun-Telaga-Dwi-Warna-Sibolangit.jpg",
"deskripsi": "Bagi anda pecinta petualangan, berwisata ke Air Terjun Telaga Dwi Warna Sibolangit merupakan pilihan yang tepat. Air terjun ini terletak di dalam hutan sibolangit.\n\n Air terjun ini letaknya di Kab. Deli Serdang, atau sekitar 2 jam perjalanan menggunakan kendaraan darat dari Kota Medan. Setelah sampai di hutan Sibolangit, anda haru berjalan kaki melintasi hutan yang

masih cukup lebat.\n\n Perjalanan ini bisa memakan waktu hingga 3 jam atau lebih tergantung dari kecepatan perjalanan anda. Perjalanan ini akan terasa sangat melelahkan tetapi menyenangkan.\n\n Setelah sampai di Air Terjun Telaga Dwi Warna Sibolangit, segala perjuangan anda akan terbayar oleh keindahan air terjun yang mempunyai dua warna ini, yaitu biru untuk air dingin dan putih untuk air hangat.",
"link": "*Berikut Lokasi Destinasi Tujuan Anda :* https://goo.gl/maps/JFxXQSu5n94C9ywcA \n Ketik *Menu*, Untuk kembali melihat daftar destinasi wisata. \n ketik *Selesai*, untuk mengakhiri BOT ini. "
},
"Gunung Sibayak": {
"photo": "*Berikut Tampilan Sekilas Destinasi Tujuan Anda :* https://www.azwisata.com/wp-content/uploads/2018/07/Tempat-Wisata-di-Sumatera-Utara-Gunung-Sibayak.jpg",
"deskripsi": "Gunung yang teletak di Kab. Karo ini merupakan gunung yang masih aktif. Gunung Sibayak adalah salah satu spot terbaik untuk melihat kota Medan dari ketinggian.\n\n Selain itu tempat ini juga sering digunakan untuk melihat matahari terbit. Anda cukup membayar retribusi sebesar Rp 4 ribu untuk mendaki gunung ini.",
"link": "*Berikut Lokasi Destinasi Tujuan Anda :* https://goo.gl/maps/WTXucqa5UwiNMxue9 \n Ketik *Menu*, Untuk kembali melihat daftar destinasi wisata. \n ketik *Selesai*, untuk mengakhiri BOT ini. "
},
"Air Terjun Sipiso-Piso": {
"photo": "*Berikut Tampilan Sekilas Destinasi Tujuan Anda :* https://www.azwisata.com/wp-content/uploads/2018/07/Tempat-Wisata-di-Sumatera-Utara-Air-Terjun-Sipiso-piso.jpg",

"deskripsi": "Air Terjun Sipiso-piso yang merupakan kawasan wisata alam yang letaknya dekat dari permukiman warga Desa Tongging, Kecamatan Merek, Kabupaten Karo.\n\n Air terjun ini memiliki ketinggian kurang lebih 800 meter dari permukaan laut (dpl) serta dikelilingi oleh bukit yang hijau yang ditumbuhi hutan pinus menambah suasana nyaman di tempat ini.",

"link": "*Berikut Lokasi Destinasi Tujuan Anda :* https://goo.gl/maps/WfSjftTdZcTpvSta7 \n Ketik *Menu*, Untuk kembali melihat daftar destinasi wisata. \n ketik *Selesai*, untuk mengakhiri BOT ini. "

},

"Air Terjun Sikulikap": {

"photo": "*Berikut Tampilan Sekilas Destinasi Tujuan Anda :* https://www.azwisata.com/wp-content/uploads/2018/07/Tempat-Wisata-di-Sumatera-Utara-Air-Terjun-Sikulikap.jpg",

"deskripsi": "Untuk mencapai lokasi Air Terjun Sikulikap, kita harus menempuh perjalanan dari gapura Air Terjun Sikulikap di sebelah kiri jalan dekat Gerbang Selamat Datang, di Kabupaten Karo.\n\n Turun lewat anak tangga tak jauh dari gapura tersebut, butuh waktu sekitar 20 menit. Setelah melewati hutan, barulah sampai di Air Terjun Sikulikap. Air terjun dengan ketinggian 30 meter ini memiliki banyak tebing di sekeilingnya. Banyak yang menggunakan tebing-tebing tersebut untuk olahraga panjat tebing.",

"link": "*Berikut Lokasi Destinasi Tujuan Anda :* https://goo.gl/maps/Jrk9KfHze6vStuUW8 \n Ketik *Menu*, Untuk kembali melihat daftar destinasi wisata. \n ketik *Selesai*, untuk mengakhiri BOT ini. "

},

"Pantai Pandan": {

"photo": "*Berikut Tampilan Sekilas Destinasi Tujuan Anda :* https://www.azwisata.com/wp-

content/uploads/2018/07/Tempat-Wisata-di-Sumatera-Utara-Pantai-Pandan.jpg",

"deskripsi": "Pantai yang terletak di pesisir Sibolga ini memiliki ombak yang biru bergulung dengan hamparan pasir lembut. Pantai ini secara administratif berada dalam kawasan Desa Pandan, Kec. Tapanuli Tengah, Kab. Tapanuli Tengah.\n\n Pemandangan yang cukup indah membuar para pengunjung akan betah beralama-lama dan kembali lagi untuk menikmati keindahan laut yang cukup mempesona. Pastinya pantai ini akan sangat menarik untuk di jadikan tempat surfing para peselancar.",

"link": "*Berikut Lokasi Destinasi Tujuan Anda :* https://goo.gl/maps/ao8AfLoUU65Ghwiv6 \n Ketik *Menu*, Untuk kembali melihat daftar destinasi wisata. \n ketik *Selesai*, untuk mengakhiri BOT ini. "

},

"Danau Toba": {

"photo": "*Berikut Tampilan Sekilas Destinasi Tujuan Anda :* https://www.azwisata.com/wp-content/uploads/2018/07/Tempat-Wisata-di-Sumatera-Utara-Danau-Toba.jpg",

"deskripsi": "Danau Toba adalah danau yang paling terkenal di Indonesia karena ukurannya yang besar dan keunikan yang berupa adanya sebuah pulau di tengah Danau Toba. Dengan panjang sekitar 100 KM dan lebar sektiar 30 KM, Danau Toba adalah danau vulkanik yang terbesar di Asia Tenggara. \n Danau Toba adalah tempat wisata di Sumatera Utara yang paling terkenal, sehingga tidak perlu kuatir apabila anda ingin menginap di Danau Toba karena di sepanjang tepian Danau Toba banyak terdapat penginapan.",

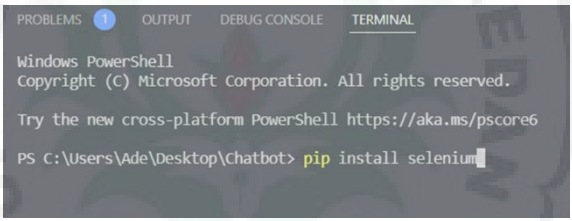
"link": "*Berikut Lokasi Destinasi Tujuan Anda :* https://goo.gl/maps/VjTiySV8bNiWz6GMA \n Ketik *Menu*, Untuk kembali melihat daftar destinasi wisata. \n ketik *Selesai*, untuk mengakhiri BOT ini. "

},

```
  "Bukit Lawang": {
    "photo": "*Berikut Tampilan Sekilas Destinasi Tujuan Anda :* https://www.azwisata.com/wp-content/uploads/2018/07/Tempat-Wisata-di-Sumatera-Utara-Bukit-Lawang.jpg",
    "deskripsi": "Bukit Lawang Merupakan sebuah tempat wisata di Sumatera Utara yang menyuguhkan Anda akan kegiatan penjelajahan hutan serta menyaksikan dari dekat kehidupan satwa di alam bebas. Bukit Lawang adalah sebuah perwujudan hutan tropis, terletak dekat pintu masuk Taman Nasional Gunung Leuser. \n Kawasan wisata Bukit Lawang disukai banyak kalangan sebagai salah satu destinasi wisata alam yang menarik di Sumatera Utara. Tersedia banyak penginapan dan losmen. Kegiatan yang dapat dilakukan diantaranya trekking hingga rafting.",
    "link": "*Berikut Lokasi Destinasi Tujuan Anda :* https://goo.gl/maps/XC2HjUttbUry2Ks27 \n Ketik *Menu*, Untuk kembali melihat daftar destinasi wisata. \n ketik *Selesai*, untuk mengakhiri BOT ini. "
  },
  "Sipinsur": {
    "photo": "*Berikut Tampilan Sekilas Destinasi Tujuan Anda :* https://www.azwisata.com/wp-content/uploads/2018/07/Tempat-Wisata-di-Sumatera-Utara-Sipinsur.jpg",
    "deskripsi": "Taman sipinsur ini terletak di desa Parulohan, Kab. Humbang Hasundutan. Di sini Anda dapat menikmati berdiri di atas tanah seluas 2 hektar dengan tinggi kira-kira 1.213 meter dari permukaan air laut sambil menyaksikan pemandangan alam yang menawan. Dua pemandangan yang memukau dari sini yaitu Danau Toba dan Pulau Samosir. \n Di bawah lokasi Sipinsur juga tampak Pulau Sibandang dan Kota Muara. Para wisatawan akan melihat keindahan alam mulai dari laut, bukit, dan pegunungan di waktu bersamaan karena dapat dilihat dari
```

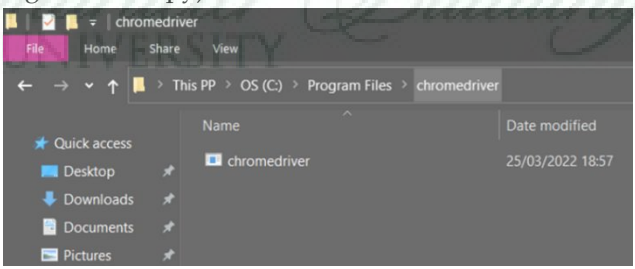
```
ketinggian. Hal ini lah yang akan membuat Sipinsur menjadi wisata favorit keluarga untuk dikunjungi.",
        "link": "*Berikut Lokasi Destinasi Tujuan Anda :* https://goo.gl/maps/sSEGj4SJk5uvnFj69 \n Ketik *Menu*, Untuk kembali melihat daftar destinasi wisata. \n ketik *Selesai*, untuk mengakhiri BOT ini. "
    }
}
```

6. Melakukan instalasi library selenium melalui terminal pada visual code studio, dengan perintah **pip install selenium**. Tekan "enter", selanjutnya tunggu proses instalasi selenium selesai.



Gambar 71. Instalasi Selenium Melalui Terminal VS Code

7. Mengunduh chromedriver melalui: https://chromedriver.chromium.org/downloads unduh versi chromedriver sesuai dengan versi browser (Google Chrome) yang digunakan. Kemudian letakkan chromedriver.exe di C:\Program Files\chromedriver. (Lokasi penyimpanan sesuai yang dideskripsikan pada coding modelbot.py).

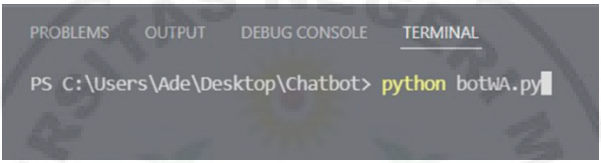


Gambar 72. Lokasi Penyimpanan Chromedriver

```
#Lokasi penyimpanan Chromedriver
path = 'C:\Program Files\chromedriver\chromedriver.exe'
driver = webdriver.Chrome(path)
```

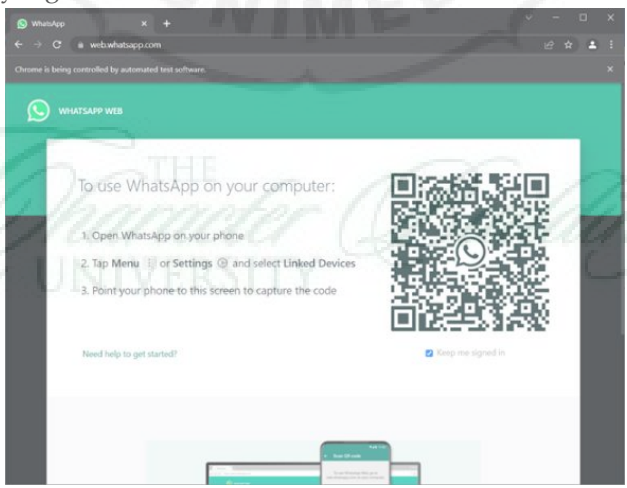
Gambar 73. Deskripsikan Lokasi Penyimpanan Chromedriver pada modelbot.py

8. Menjalankan *main program* melalui Terminal di Visual Studio Code dengan perintah seperti Gambar 74.



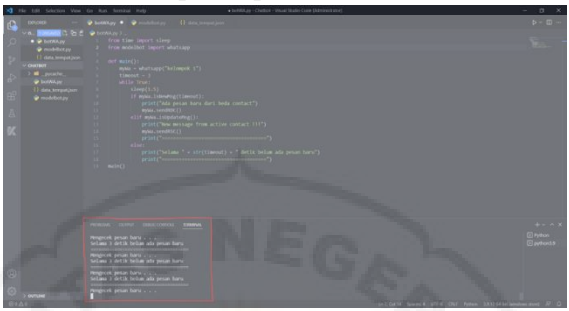
Gambar 74. Running Program botWA.py Melalui Terminal VS Code

9. Selanjutnya akan muncul tab Google Chrome yang berarti Chromedriver membuka halaman URL: https://web.whatsapp.com/ (dipanggil melalui modelbot.py). Langkah selanjutnya adalah melakukan *Login* melalui akun WhatsApp dengan melakukan scaning kode QR yang tertera.



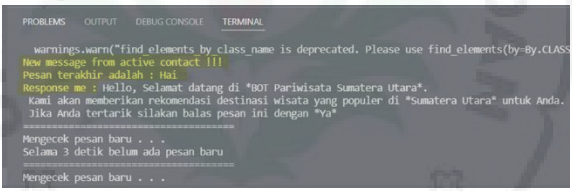
Gambar 75. Scan Kode QR untuk *Login* WhatsApp

10. Jika bot sudah berjalan, maka tampilan terminal pada visual studio code akan tampak seperti Gambar 76.



Gambar 76. Tampilan Terminal Saat Program **botWA.py** Berjalan

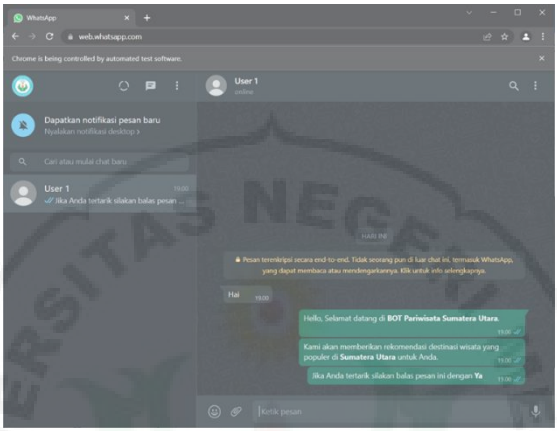
11. Jika terdeteksi pesan baru, maka akan ada pesan melalui terminal seperti Gambar 77.



Gambar 77. Tampilan Terminal Jika Terdeteksi Pesan Masuk

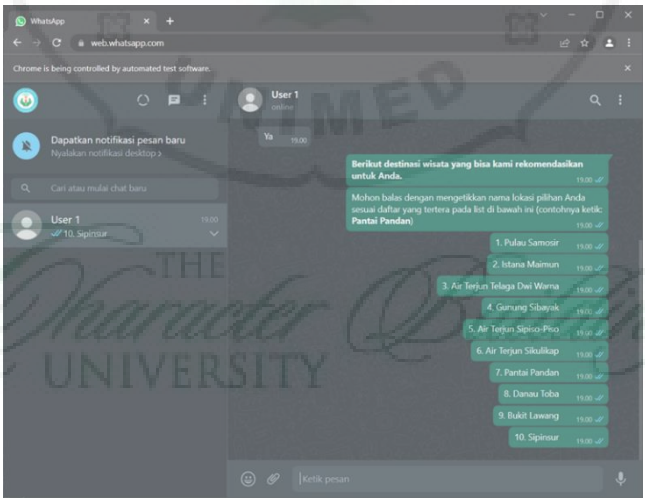
C. Konfigurasi WhatsApp

1. Tampilan melalui akun yang digunakan sebagai Bot seperti Gambar 78.



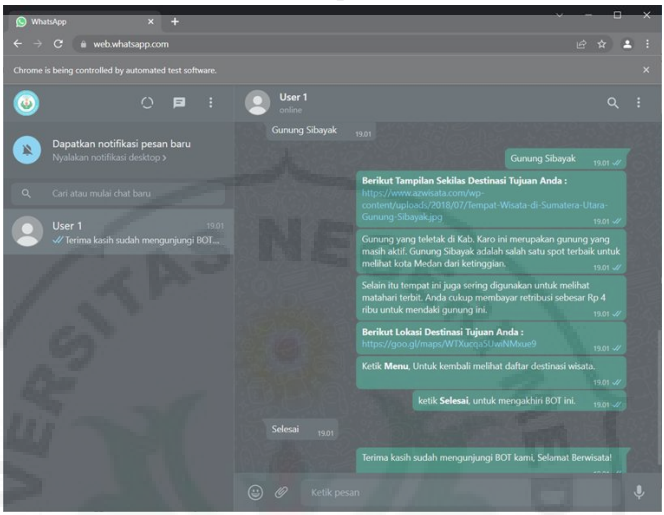
Gambar 78. Tampilan Melalui Akun WA yang Digunakan Sebagai Bot

2. Tampilan dari bot jika pengguna membalas pesan dengan "Ya".



Gambar 79. Tampilan Jawaban Bot Mengenai Objek Wisata

3. Apabila pengguna menjawab “Gunung Sibayak”, maka bot akan memberikan informasi seperti Gambar 80.



Gambar 80. Tampilan Jawaban Bot Mengenai Informasi Wisata

4. Tampilan jawaban chatbot pada whatsapp pengguna. Gambar 81 dimana tampilan pengguna ketika menyapa chatbot dengan kata “Hai”.

Gambar 81. Tampilan WhatsApp Pengguna Ketika Mengetikkan “Hai”

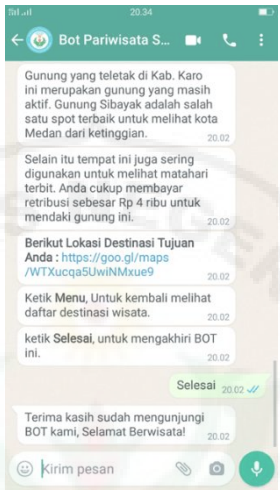
5. Tampilan jawaban chatbot pada whatsapp pengguna ketika pengguna memilih "ya", untuk melanjutkan percakapan dapat dilihat pada Gambar 82.

Gambar 82. Tampilan WhatsApp Pengguna Ketika Memilih "ya"

6. Tampilan jawaban chatbot pada whatsapp pengguna ketika memilih "Gunung Sibayak", dapat dilihat pada Gambar 83.

Gambar 83. Tampilan WhatsApp Pengguna Ketika Memilih "Gunung Sibayak"

7. Tampilan jawaban chatbot pada whatsapp pengguna ketika memilih "Selesai" untuk mengakhiri pecakapan dengan chatbot.



Gambar 84. Tampilan WhatsApp Pengguna Ketika Memilih "Selesai"

DAFTAR PUSTAKA

Abu Shawar, B., & Atwell, E. (2005, January 01). A chatbot system as a tool to animate a corpus. Retrieved July 18, 2020, from http://eprints.whiterose.ac.uk/81677/

Asian, Jelita. (2007). Effective techniques for Indonesian text retrieval.

Bansal, H., & Khan, R. (2018). A review paper on human computer interaction. International Journal of Advanced Research in Computer Science and Software Engineering, 8(53), http://dx.doi.org/10.23956/ijarcsse.v8i4.630.

Bayu, S., & Fery, W. 2016. *Chatbot Using a Knowledge in Database*. 7th International Conference on Intelligent Systems, Modelling and Simulation.

Cahn, J. (2017). CHATBOT: Architecture, design, & development. University of Pennsylvania School of Engineering and Applied Science Department of Computer and Information Science.

Costa, P. C. F. da. (2018). Conversing with personal digital assistants: On gender and artificial intelligence. Journal of Science and Technology of the Arts, 10(3), http://dx.doi.org/10.7559/citarj.v10i3.563,2-59–79.

Fernandes, A. (2018). Nlp, nlu, nlg and how chatbots work. Medium website: https:// chatbotslife.com/nlp-nlu-nlg-and-how-chatbots-work-dd7861dfc9df. (Retrieved 8 November 2019).

Fernando A. Mikic, Juan C. Buirguillo, Daniel A. Rodriguez, Eduardo Rodriguez, & Martin Llamas. 2008. *T-BOT and Q-BOT: A Couple of AIML-Based Bots for Tutoring Courses and Evaluating Students*. 38th ASEE/IEEE Frontiers in Education Conference S3A-7.

Go, E., & Sundar, S. S. (2019). Humanizing chatbots: The effects of visual, identity and conversational cues on humanness perceptions. Computers in Human Behavior, 97, 304–316. http://dx.doi.org/10.1016/j.chb.2019.01.020.

Julius, Andrew. 2020. *Pemanfaatan Artificial Intelligence Markup Language (AIML) dan Latent Semantic Analysis (LSA) dalam Pengembangan Chatbot Universitas Sumatera Utara*. Skripsi. Medan: Universitas Sumatera Utara.

Khairani, Fitri. 2021. *Aplikasi Chatbot Tanya Jawab Tentang Kesehatan Menggunakan Algoritma Enhanced Confix Stripping dan Algoritma Knuth Morris Pratt.* Skripsi. Medan: Universitas Sumatera Utara.

Khanna, A., Pandey, B., Vashishta, K., Kalia, K., Bhale, P., & Das, T. (2015). A study of today's A.I. through chatbots and rediscovery of machine intelligence. International Journal of U- and e-Service, Science and Technology, 8, 277–284. http://dx.doi.org/10.14257/ijunesst.2015.8.7.28.

Kurniawan, Boby. 2021. *Chatbot Eksplorasi Ayat Al-Quran Menggunakan Algoritma Bidirectional Long Short-Term Memory (Bi-LSTM) dan Fuzzy String Matching*. Skripsi. Medan: Universitas Sumatera Utara.

M. Naveen, P., C., Linga, A., Venkatesh, K., Sumangali. 2016. *Android Based Educational Chatbot for Visually Impaired People*. IEEE InternationalConference on Computational Intelligence and Computing Research (ICCIC).

Mayrisa. 2019. *USU Pintar: Pencarian Informasi di Website USU Berbasis Chatbot*. Skripsi. Medan: Universitas Negeri Medan.

Noy, N.F. & McGuinness, D.L. (2001). Ontology Development 101: A Guide to Creating Your First Ontology. Stanford University.

P, Elisabet Nila S. C. 2013. *Rancang Bangun Aplikasi Chatbot Informasi Objek Wisata Kota Bandung dengan Pendekatan Natural Language Processing*. Skripsi. Universitas Komputer Indonesia.

Palanica, A., Flaschner, P., Thommandram, A., Li, M., & Fossat, Y. (2019). Physicians' Perceptions of Chatbots in Health Care: Cross-Sectional Web-Based Survey. Journal of Medical Internet Research, 21(4), e12887. https://doi.org/10.2196/12887

Prima, Elsa Adam Alvin. 2021. *Aplikasi Chatbot Informasi Lokasi Wisata dan Kuliner Kota Batam*. Skripsi. Batam: Universitas Putra Batam.

R. Ranoliya, B., Raghuwanshi, N., & Singh, S. (2017). Chatbot for university related FAQs. In 2017 international conference on advances in computing, communications, and informatics (pp. 1525–1530). Udupi. https://doi.org/10.1109/ICACCI.2017.8126057.

Reshmi, S. & Balakrishnan, K. (2018). Empowering Chatbots with Business Intelligence by Big Data Integration. International Journal of Advanced Research in Computer Science, 9(1), 627–631. https://doi.org/10.26483/ijarcs.v9i1.5398

Sari, Dewi Wiranda. 2018. *Implementasi Natural Language Processing pada Chatbot Peribahasa Indonesia*. Skripsi. Medan: Universitas Sumatera Utara.

Sebastian, A., & George, J. (2016). *Fuzzy Pattern Matching Algorithm for Location Based Approximate Strings*. 7(7), 583–587.

Shawar, B., & Atwell, E. (2010). Chatbots: Can they serve as natural language interfaces to QA corpus? https://doi.org/10.2316/P.2010.689-050.

Sihotang, M. Tohir. 2019. *Chatbot Tanya Jawab Islamic App Menggunakan Algoritma Enhanced Confix Stripping dan Algoritma Fuzzy String Matching*. Skripsi. Medan: Universitas Sumatera Utara.

Susanto. 2014. *Perancangan Aplikasi Kamus Istilah Latin Hewan dan Tumbuhan dengan Menerapkan Algoritma Boyer-Moore Berbasis Mobile*. Jurnal Informasi dan Teknologi Ilmiah (INTI) 4(3): 88-90.

Wallace, R. S. (2009). The anatomy of a.l.I.C.e. In R. Epstein, G. Roberts, & G. Beber (Eds.), Parsing the turing test: philosophical and methodological issues in the quest for the thinking computer (pp. 181–210). Dordrecht: Springer Netherlands, http://dx.doi.org/10.1007/978-1-4020-6710-5_13.

Wright, D. R. (2005). Finite state machines. Carolina State University, 203.

Yi, S., & Jung, K. (2017). A chatbot by combining finite state machine, information retrieval, and bot-initiative strategy. Proc. Alexa Prize.