SISTEM CHATBOT LAYANAN INFORMASI TUGAS AKHIR MENGGUNAKAN METODE *LONG SHORT-TERM MEMORY* (LSTM)

LAPORAN TUGAS AKHIR

Laporan ini Disusun untuk Memenuhi Salah Satu Syarat Memperoleh
Gelar Sarjana Strata 1 (S1) pada Program Studi Teknik Informatika
Fakultas Teknologi Industri Universitas Islam Sultan Agung Semarang

Disusun Oleh:

LINGGA FAHADRA
NIM 32601900014

FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS ISLAM SULTAN AGUNG
SEMARANG
2023

FINAL PROJECT

CHATBOT SYSTEM FOR THESIS INFORMATION SERVICES USING LONG SHORT-TERM MEMORY (LSTM)

Proposed to complete the requirement to obtain a bachelor's degree (S1)
at Informatics Engineering Departement of Industrial Technology Faculty
Sultan Agung Islamic University

Arrangged By:

LINGGA FAHADRA
NIM 32601900014

MAJORING OF INFORMATICS ENGINEERING
INDUSTRIAL TECHNOLOGY FACULTY
SULTAN AGUNG ISLAMIC UNIVERSITY
SEMARANG
2023

LEMBAR PENGESAHAN PEMBIMBING

Laporan Tugas Akhir dengan judul "Sistem *Chatbot* Layanan Informasi Tugas Akhir Menggunakan Metode *Long Short-Term Memory* (LSTM)" ini disusun oleh :

Nama : Lingga Fahadra

NIM : 32601900014

Program Studi : Teknik Informatika

Telah disahkan oleh dosen pembimbing pada :

Hari :

Tanggal :

Mengesahkan,

Pembimbing I	Pembimbing II
Badie'ah, ST.M.Kom	Imam Much Ibnu S, ST.MSc, Ph.D
NIDN. 0619018701	NIDN. 0613037301

Mengetahui,

Ketua Program Studi Teknik Informatika
Fakultas Teknologi Industri
Universitas Islam Sultan Agung

Ir. Sri Mulyono, M.Eng
NIDN. 0626066601

LEMBAR PENGESAHAN PENGUJI

Proposal tugas akhir dengan judul "Sistem *Chatbot* Layanan Informasi Tugas Akhir Menggunakan Metode *Long Short-Term Memory* (LSTM)" ini telah dipertahankan di depan tim penguji proposal Tugas Akhir pada:

Hari : Jumat

Tanggal : 4 Agustus 2023

TIM PENGUJI

Ketua Penguji	Anggota I
Ir. Sri Mulyono, M.Eng	Ghufron, ST, M.Kom
NIDN. 0626066601	NIDN. 0602079005

SURAT PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan dibawah ini :

Nama	:	Lingga Fahadra
NIM	:	32601900014
Judul Tugas Akhir	:	Sistem *Chatbot* Layanan Informasi Tugas Akhir Menggunakan Metode *Long Short-Term Memory* (LSTM)

Dengan bahwa ini saya menyatakan bahwa judul dan isi Tugas Akhir yang saya buat dalam rangka menyelesaikan Pendidikan Strata Satu (S1) Teknik Informatika tersebut adalah asli dan belum pernah diangkat, ditulis ataupun dipublikasikan oleh siapapun baik keseluruhan maupun sebagian, kecuali yang secara tertulis diacu dalam naskah ini dan disebutkan dalam daftar pustaka, dan apbila di kemudian hari ternyata terbukti bahwa judul Tugas Akhir tersebut pernah diangkat, ditulis ataupun dipublikasikan, maka saya bersedia dikenakan sanksi akademis. Demikian surat pernyataan ini saya buat dengan sadar dan penuh tanggung jawab.

Semarang, 23 Agustus 2023

Yang Menyatakan,

Lingga Fahadra

PERNYATAAN PERSETUJUAN PUBLIKASI KARYA ILMIAH

Saya yang bertanda tangan dibawah ini :

Nama	:	Lingga Fahadra
NIM	:	32601900014
Program Studi	:	Teknik Informatika
Fakultas	:	Teknologi industri
Alamat Asal	:	Desa Donorojo Kecamatan Jayaloka Kabupaten Musi Rawas

Dengan ini menyatakan Karya Ilmiah berupa Tugas akhir dengan Judul : **Sistem Chatbot Layanan Informasi Tugas Akhir Menggunakan Metode Long Short-Term Memory (LSTM)**

Menyetujui menjadi hak milik Universitas Islam Sultan Agung serta memberikan Hak bebas Royalti Non-Eksklusif untuk disimpan, dialihmediakan, dikelola dan pangkalan data dan dipublikasikan diinternet dan media lain untuk kepentingan akademis selama tetap menyantumkan nama penulis sebagai pemilik hak cipta. Pernyataan ini saya buat dengan sungguh-sungguh. Apabila dikemudian hari terbukti ada pelanggaran Hak Cipta/Plagiarisme dalam karya ilmiah ini, maka segala bentuk tuntutan hukum yang timbul akan saya tanggung secara pribadi tanpa melibatkan Universitas Islam Sultan agung.

Semarang, 23 Agustus 2023

Yang menyatakan,

Lingga Fahadra

KATA PENGANTAR

Dengan mengucap syukur alhamdulillah atas kehadirat Allah SWT yang telah memberikan rahmat dan karunianya kepada penulis, sehingga dapat menyelesaikan Tugas Akhir dengan judul "Sistem *Chatbot* Layanan Informasi Tugas Akhir Menggunakan Metode *Long Short-Term Memory* (LSTM)" ini untuk memenuhi salah satu syarat menyelesaikan studi serta dalam rangka memperoleh gelar sarjana (S-1) pada Program Studi Teknik Informatika Fakultas Teknologi Industri Universitas Islam Sultan Agung Semarang.

Tugas Akhir ini disusun dan dibuat dengan adanya bantuan dari berbagai pihak, materi maupun teknis, oleh karena itu saya selaku penulis mengucapkan terima kasih kepada:

1. Rektor UNISSULA Bapak Prof. Dr. H. Gunarto, S.H., M.H yang mengizinkan penulis menimba ilmu di kampus ini.
2. Dekan Fakultas Teknologi Industri Ibu Dr. Novi Marlyana, S.T., M.T.
3. Dosen pembimbing I penulis Badie'ah, ST.M.Kom yang telah meluangkan waktu dan memberi ilmu.
4. Dosen pembimbing II penulis Imam Much Ibnu S, ST.MSc, Ph.D yang memberikan banyak nasehat dan saran.
5. Orang tua penulis yang telah mengizinkan untuk menyelesaikan laporan ini,
6. Dan kepada semua pihak yang tidak dapat saya sebutkan satu persatu.

Dengan segala kerendahan hati, penulis menyadari masih terdapat banyak kekurangan dari segi kualitas atau kuantitas maupun dari ilmu pengetahuan dalam penyusunan laporan, sehingga penulis mengharapkan adanya saran dan kritikan yang bersifat membangun demi kesempurnaan laporan ini dan masa mendatang.

Semarang, 4 Agustus 2023

Lingga Fahadra

DAFTAR ISI

COVER i
LEMBAR PENGESAHAN PEMBIMBING iii
LEMBAR PENGESAHAN PENGUJI iv
SURAT PERNYATAAN KEASLIAN TUGAS AKHIR v
PERNYATAAN PERSETUJUAN PUBLIKASI KARYA ILMIAH vi
KATA PENGANTAR vii
DAFTAR ISI viii
DAFTAR TABEL x
DAFTAR GAMBAR xi
ABSTRAK xii
BAB I 1
PENDAHULUAN 1
1.1 Latar Belakang 1
1.2 Rumusan Masalah 2
1.3 Pembatasan Masalah 3
1.4 Tujuan Tugas Akhir 3
1.5 Manfaat 3
1.6 Sistematika Penulisan 4
BAB II 5
TINJAUAN PUSTAKA DAN DASAR TEORI 5
2.1 Tinjauan Pustaka 5
2.2 Dasar Teori 7
2.2.1 *Natural Language Processing* 7
2.2.2 *Chatbot* 7
2.2.3 *Deep Learning* 8
2.2.4 *Remove punctuations* 8
2.2.5 *Tokenization* 8
2.2.6 *Padding Text* 8
2.2.7 *Encoding* 9
2.2.8 *Vocabulary* 9
2.2.9 *Long Short-Term Memory* 9
BAB III 14
METODOLOGI PENELITIAN 14
3.1 Metode Penelitian 14
3.1.1 Studi *Literature* 14

3.1.2 Pengumpulan Data 14
3.1.3 Metode Perancangan Sistem 14
3.1.4 Tahapan Perancangan Sistem 15
3.1.5 Evaluasi Model 18
3.1.6 Pengujian *Black Box* 20
3.2 Analisis Kebutuhan 20
3.3 Analisis Sistem 23
3.4 Perancangan User Interface 24
3.4.1 Halaman Awal *Chatbot* 24
3.4.2 Halaman Percakapan *Chatbot* 25
BAB IV 27
HASIL DAN ANALISIS PENELITIAN 27
4.1 Hasil Penelitian 27
4.2 Hasil Implementasi Antarmuka 38
4.2.1 Halaman Awal *chatbot* 38
4.2.2 Halaman *Chatbot* 38
4.3 Hasil Pengujian Sistem 44
BAB V 46
KESIMPULAN DAN SARAN 46
5.1 Kesimpulan 46
5.2 Saran 46
DAFTAR PUSTAKA 47

DAFTAR TABEL

Table 4. 1 Konfigurasi 35
Table 4. 2 Hasil Confusion Matrix Konfigurasi..... 37
Table 4. 3 Pengujian Black Box..... 45

DAFTAR GAMBAR

Gambar 2. 1 Jaringan LSTM (Trivusi, 2022) 10
Gambar 2. 2 Forget Gate (Trivusi, 2022)..... 10
Gambar 2. 3 Input Gate (Trivusi, 2022)..... 11
Gambar 2. 4 Output Gate (Trivusi, 2022)..... 13
Gambar 3. 1 Metode Prototype (Renaningtias & Apriliani, 2021)..... 15
Gambar 3. 2 Diagram Alur Perancangan Sistem 16
Gambar 3. 3 Tabel Confusion Matrix (Nugroho, 2019) 19
Gambar 3. 4 Flowchart sistem Chatbot..... 24
Gambar 3. 5 Halaman Awal Chatbot..... 25
Gambar 3. 6 Halaman Chatting 25
Gambar 4. 1 Konversi Dataframe 28
Gambar 4. 2 Proses Remove Punctuation..... 28
Gambar 4. 3 hasil Remove Punctuation..... 29
Gambar 4. 4 Proses Tokenization 29
Gambar 4. 5 Proses Padding text 30
Gambar 4. 6 Proses Encoding..... 30
Gambar 4. 7 Proses Vocabulary..... 31
Gambar 4. 8 Proses Train Test Split 31
Gambar 4. 9 Proses Pembentukan Model 32
Gambar 4. 10 Struktur Model 33
Gambar 4. 11 Training Model..... 34
Gambar 4. 12 Grafik Training Model..... 35
Gambar 4. 13 Confusion Matrix Konfigurasi 1 36
Gambar 4. 14 Penghitungan Confusion Matrix 37
Gambar 4. 15 Halaman Awal..... 38
Gambar 4. 16 Input pertanyaan..... 39
Gambar 4. 17 Demo Aplikasi Chatbot ke-1..... 40
Gambar 4. 18 Demo Aplikasi Chatbot ke-2..... 41
Gambar 4. 19 Demo Aplikasi Chatbot ke-3..... 42
Gambar 4. 20 Demo Aplikasi Chatbot ke-4..... 43
Gambar 4. 21 Demo Aplikasi Chatbot ke-5..... 44

ABSTRAK

Teknologi berkembang pesat, termasuk kecerdasan buatan seperti *chatbot*. *Chatbot* memiliki manfaat mengurangi biaya dan memberikan akses instan 24/7. Meski demikian, Program Studi Teknik Informatika masih menjalankan kegiatan secara konvensional, khususnya dalam hal informasi Tugas Akhir. Pada dasarnya Program Studi Teknik Informatika sudah memiliki buku panduan Tugas Akhir, namun jika ingin mencari informasi harus membutuhkan waktu yang sangat lama untuk membacanya. Oleh karena itu Penelitian ini bertujuan menerapkan algoritma *Long Short-Term Memory* (LSTM) pada *chatbot* informasi tugas akhir perguruan tinggi. Tahapannya: pengumpulan dan eksplorasi data, persiapan data, pembangunan dan pelatihan model LSTM, serta pengujian dengan *confusion matrix*. Hasil pengujian menunjukkan *accuracy* 94,41%, *precision* 95,63%, dan *recall* 94,41%. Ini menandakan model LSTM efektif merespons pertanyaan tugas akhir. Hasil ini mendukung penggunaan chatbot LSTM sebagai layanan informasi di perguruan tinggi, membantu mahasiswa mengakses informasi tugas akhir lebih mudah dan efisien.

Kata Kunci: *Chatbot, Long Short-Term Memory,* LSTM, *Deep Learning*

ABSTRACT

Technology is developing rapidly, including artificial intelligence such as chatbots. Chatbots have the benefit of reducing costs and providing instant access 24/7. However, the Informatics Engineering Study Program still carries out conventional activities, especially in terms of final project information. Basically, the Informatics Engineering Study Program already has a Final Assignment manual, but if you want to find information, you have to take a very long time to read it. There-fore this study aims to apply the Long Short-Term Memory (LSTM) algorithm to information chatbots on college final assignments. The stages are: data collection and exploration, data preparation, LSTM model development and training, and testing with the confusion matrix. The test results show 94.41% accuracy, 95.63% precision and 94.41% recall. This indicates that the LSTM model is effective in responding to final project questions. These results support the use of the LSTM chatbot as an information service in higher education, helping students access final project information more easily and efficiently.

Keywords: *Chatbot, Long Short-Term Memory,* LSTM, *Deep Learning*

BAB I
PENDAHULUAN

1.1 Latar Belakang

Dengan kemajuan kemampuan teknologi masyarakat modern, banyak aspek dari cara kita hidup, bekerja, dan belajar telah diubah. Penerapan teknologi menjadi semakin penting untuk memajukan kemajuan manusia dan mengatasi setiap masalah yang mungkin timbul. Dalam situasi ini, kecerdasan buatan (AI) telah menjadi inovasi yang paling menonjol, salah satunya adalah *Natural Language Processing* (NLP) yang merupakan aplikasi kecerdasan buatan yang saat ini mendapatkan daya tarik. NLP adalah alat yang dapat digunakan komputer untuk memahami bahasa lisan yang digunakan manusia dalam kehidupan sehari-hari (A. Hikmah dkk., 2023). NLP sering digunakan dalam aplikasi pembelajaran bahasa seperti *Google Translate*, asisten pribadi seperti *Siri*, dan alat untuk menilai kemahiran bahasa seperti *Grammarly*, *Chatgpt*, dan *Chatbot*.

Chatbot adalah program komputer yang dirancang untuk melakukan percakapan dengan manusia melalui antarmuka *chat* atau pesan teks (Guntoro dkk., 2020). *Chatbot* mampu memberikan informasi, menjawab pertanyaan, meluncurkan tugas khusus, dan bahkan melayani tujuan administratif. Pengembangan *chatbot* dapat melibatkan berbagai pendekatan teknologi, termasuk *Artificial Intelligence Markup Language* (AIML), *Natural Language Processing* (NLP), dan *Machine Learning* (ML).

Seiring berjalannya waktu, jumlah penelitian tentang *chatbot* terus bertambah. Contoh kasusnya adalah studi yang memanfaatkan *chatbot* menggunakan algoritma *Bidirectional Long Short-Term Memory* untuk media interaktif yang bisa tanya-jawab mengenai ayat, tafsir, dan *asbabun nudzul* dari Al-Quran (Kurniawan, 2021). Selain itu, telah dilakukan penelitian menggunakan *chatbot* untuk memudahkan masyarakat umum dalam mencari informasi tentang sistem kesehatan di Kota Malang (Hormansyah & Utama, 2018).

Penggunaan *chatbot* di Indonesia masih tergolong rendah. Berdasarkan survei yang dilakukan oleh Pancake terhadap UMKM di Indonesia, hanya 15% UMKM

yang telah menerapkan *chatbot* (Amalia, 2021). Hal ini mengindikasikan bahwa masih banyak organisasi dan perusahaan yang belum memanfaatkan *chatbot* untuk memperluas bisnis mereka, termasuk Program Studi Teknik Informatika UNISSULA. Program Studi Teknik Informatika UNISSULA masih menjalankan kegiatan secara konvensional, khususnya dalam hal informasi Tugas Akhir. Bagi mahasiswa yang mengambil Mata Kuliah tugas akhir di program studi teknik informatika UNISSULA harus mengikuti banyak prosedur administratif pada setiap tahapan tugas akhir. Pada dasarnya, program studi teknik informatika sudah memiliki panduan tugas akhir yang cukup lengkap, sehingga mahasiswa cukup membaca buku panduan untuk mendapatkan berbagai informasi. Namun, buku panduan memiliki informasi yang sangat kompleks sehingga untuk mendapatkan informasi yang spesifik, mahasiswa membutuhkan waktu yang cukup lama untuk memilah informasi yang dibutuhkan.

Dari permasalahan yang dijelaskan diatas, maka penelitian ini akan mengembangkan sebuah sistem *chatbot* menggunakan metode *Long Short-Term Memory* sebagai layanan tanya jawab terkait informasi Tugas Akhir. Selain itu, penelitian ini juga akan berfokus pada studi kasus yaitu mengambil *dataset* dari panduan Tugas Akhir Prodi Teknik Informatika UNISSULA dan framework pemrograman yang digunakan adalah *Keras* dan *Tensorflow*. Penelitian ini menggunakan algoritma *Long Short-Term Memory*, yang merupakan metode dari *Deep Learning* yang bekerja dengan baik untuk urutan data, dan akan melakukan pengukuran untuk mengetahui kinerja sistem berdasarkan nilai akurasi, presisi, dan *recall*. Dengan dikembangkannya sistem ini diharapkan akan tercipta sistem *chatbot* yang akan memudahkan pengguna khususnya mahasiswa Teknik Informatika UNISSULA untuk mendapatkan informasi terkait Tugas Akhir.

1.2 Rumusan Masalah

Berdasarkan latar belakang permasalahan, maka dapat diidentifikasikan permasalahan yang akan dibahas adalah sebagai berikut:

1. Bagaimana mengembangkan sistem *chatbot* menggunakan algoritma LSTM dengan sampel data dari *dataset* buku panduan Tugas Akhir Prodi

Teknik Informatika UNISSULA?

2. Bagaimana performa dari hasil sistem *chatbot* menggunakan algoritma LSTM dengan sampel data dari *dataset* buku panduan Tugas Akhir Prodi Teknik Informatika UNISSULA?

1.3 Pembatasan Masalah

Adapun batasan masalah dari penulisan proposal ini yaitu sebagai berikut:

1. Sistem *chatbot* yang dibangun pada penelitian ini hanya mencakup hal-hal yang terkait dengan panduan Tugas Akhir.
2. *Dataset* yang digunakan pada penelitian ini dibuat berdasarkan pada sumber panduan Tugas Akhir Prodi Teknik Informatika UNISSULA.
3. *Datasets* terdiri dari data pertanyaan dan jawaban berdasarkan pada sumber panduan Tugas Akhir Prodi Teknik Informatika UNISSULA.
4. *Dataset* pertanyaan hanya mencakup bagian Bab I dan Bab II pada panduan Tugas Akhir Prodi Teknik Informatika UNISSULA.
5. Data uji hanya mengacu pada *dataset* yang telah dilatih.

1.4 Tujuan Tugas Akhir

Adapun Tujuan Tugas Akhir dari penulisan proposal ini yaitu sebagai berikut:

1. Mengembangkan sistem *chatbot* sebagai layanan informasi Tugas Akhir berbasis algoritma LSTM dengan sampel data dari *dataset* buku panduan Tugas Akhir Prodi Teknik Informatika UNISSULA.
2. Mengetahui kinerja dari hasil sistem *chatbot* berdasarkan nilai akurasi, *precision*, dan *recall* sebagai layanan informasi Tugas Akhir.

1.5 Manfaat

Manfaat dari Tugas Akhir ini dalah sebagai berikut:

1. Dapat membantu Mahasiswa dalam mengakses informasi tentang tugas akhir mereka kapan saja tanpa terbatas oleh jam operasional kantor.
2. Mampu memberikan respon *instan* dan cepat terhadap pertanyaan mahasiswa tentang tugas akhir.

1.6 Sistematika Penulisan

Sistematika penulisan yang digunakan oleh penulis dalam sebuah pembuatan laporan tugas akhir adalah sebagai berikut:

BAB 1 : PENDAHULUAN

Pada bab ini penulis mengutarakan latar belakang pemilihan judul,rumusan masalah, batasan masalah, tujuan penelitian, metodologi penelitian, serta sistematika penulisan.

BAB 2 : TINJAUAN PUSTAKA DAN DASAR TEORI

Pada bab ini memuat penelitian-penelitian sebelumnya dan dasar teori yang berguna untuk membantu sebuah penulis untuk memahami bagaimana teori yang berhubungan dengan *Long-Short Term Memory* (LSTM) untuk penelitian ini.

BAB 3 : METODE PENELITIAN

Pada bab ini mengungkapkan proses tahapan-tahapan penelitian dimulai dari mendapatkan data hingga proses klasifikasi data yang ada.

BAB 4 : HASIL DAN ANALISA PENELITIAN

Pada bab ini penulis mengungkapkan hasil penelitian yaitu hasil klasifikasi menggunakan *Long-Short Term Memory* (LSTM) beserta pengujian klasifikasi.

BAB 5 : KESIMPULAN DAN SARAN

Pada bab ini penulis memaparkan kesimpulan proses penelitian dari awal hingga akhir.

BAB II
TINJAUAN PUSTAKA DAN DASAR TEORI

2.1 Tinjauan Pustaka

Chatbot adalah program atau aplikasi yang dirancang untuk melakukan percakapan *online* dengan manusia. Hasilnya, ini dapat membantu proyek yang berpusat pada manusia dalam menjawab pertanyaan dari orang lain. *Chatbot* beroperasi dengan terlebih dahulu menerima input teks dan kemudian memberikan respons dalam bentuk teks tertulis berdasarkan operasi yang telah ditentukan sebelumnya (Cahn, 2017). Kegiatan ini berbeda dengan percakapan biasa, di mana *chatbot* dirancang agar dapat mensimulasikan pembicaraan layaknya manusia. namun sebenarnya tidak ada keterlibatan manusia dalam memberikan respon percakapan tersebut (Stauffer, 2021). Keberadaan *chatbot* dapat menghemat dalam penggunaan waktu dan bisa mengurangi dan bisa lebih menghemat biaya operasional. Menurut survei yang dilakukan Accenture, *chatbots* dapat membantu organisasi mengurangi biaya operasional hingga 30% dan memberikan kemudahan kepada pengguna karena mereka memiliki akses *instan* siang atau malam (Accenture, 2018).

Metode yang digunakan dalam pengembangan *chatbot* juga semakin bervariasi. Beberapa teknik yang umum digunakan adalah *Natural Language Processing*, *Bidirectional Long Short-Term Memory*, dan *Recurrent Neural Network with Long Short-Term Memory* (RNN-LSTM). Metode *Long Short-Term Memory* memungkinkan pemrosesan bahasa alami yang bertujuan untuk memfasilitasi komunikasi antara pengguna dengan komputer menggunakan bahasa alami. Metode *Long Short-Term Memory* (LSTM) sering digunakan dalam evaluasi *chatbot* karena dapat menggabungkan *input* dan *output* ke dalam urutan data, yang merupakan produk sampingan dari metode *Recurrent Neural Network* (RNN).

Karena pemrosesan bahasa alami (NLP) digunakan untuk memproses bahasa, itu akan digunakan dalam penelitian ini. NLP juga dapat menafsirkan dan mengevaluasi data tekstual, serta mengantisipasi dan membuat teks baru. Dari hasil penelitian (Mulyatun dkk., 2021) tentang *chatbot* sebagai alat bantu *customer*

service dengan menggunakan pendekatan NLP mendapatkan kesimpulan bahwa pendekatan *natural language* dapat meningkatkan efisiensi dan produktivitas dalam pelayanan *customer service*, karena dapat memproses dan memberikan jawaban secara otomatis tanpa harus melibatkan tenaga kerja manusia.

Kemudian dari hasil penelitian (N. Hikmah dkk., 2022) tentang sistem *chatbot* sebagai *virtual assistant* yang menggunakan Metode TF-IDF yang mana dalam penelitiannya mendapatkan kesimpulan bahwa metode *Term Frequency* dan *Inverse Document Frequency* berhasil diterapkan pada *chatbot* dan menghasilkan tingkat akurasi sebesar 85,7%. Pengujian UAT pada tahap pertama mencapai 84,1%, dan pada tahap kedua mencapai 82,1%, sehingga dapat dikatakan *chatbot* membantu peran operator sebagai *customer service* dengan memanfaatkan metode tf-idf, vsm, dan *cosine similarity*.

Selain metode *Term Frequency* dan *Inverse Document Frequency* Sistem *chatbot* juga dapat menggunakan pendekatan *Long Short-Term Memory*. Pendekatan ini termasuk yang terbaik untuk menangani dan memodelkan masalah dengan urutan data, seperti teks atau data berurutan lainnya. dari hasil penelitian (Wintoro dkk., 2022) tentang *chatbot* Informasi Akademik yang menggunakan metode LSTM mendapatkan kesimpulan bahwa algoritma *Long Short-Term Memory* (LSTM) dapat diimplementasikan pada *chatbot* informasi akademik dengan hasil *training* yang cukup baik dan model diuji dengan metode *user validation* yang mendapatkan akurasi sebesar 99%, dan mendapatkan hasil akurasi yang cukup baik pada tahap pengujian.

Dari ketiga hasil penelitian diatas dapat diambil kesimpulan bahwa metode pendekatan *Natural Language Processing* cocok untuk dikombinasikan dengan metode *Long Short-Term Memory* (LSTM) yang akan digunakan sebagai algoritma dalam pembuatan sistem *chatbot*. LSTM sendiri merupakan suatu metode dalam bidang *Deep Learning* yang cocok untuk memproses data *sequences*. Harapannya, penelitian ini dapat menghasilkan *chatbot* yang dapat memberikan kemudahan kepada pengguna, terutama mahasiswa dan mahasiswi Prodi Teknik Informatika UNISSULA, dalam mendapatkan informasi mengenai Tugas Akhir yang mereka butuhkan.

2.2 Dasar Teori

2.2.1 *Natural Language Processing*

Natural Language Processing (NLP) adalah cabang teknologi kecerdasan buatan (AI) yang berfokus pada pemrosesan bahasa alami, atau bahasa yang digunakan orang untuk berkomunikasi. Pemrosesan diperlukan terlebih dahulu agar bahasa dapat dipahami dengan baik oleh komputer agar mereka dapat memahami bahasa manusia. (Sembilu dkk., 2018).

Dalam bidang *Natural Language Processing*, terdapat berbagai topik yang dibahas, salah satunya adalah parsing. *Parsing* adalah proses membedah kalimat menjadi bagian-bagian komponennya berdasarkan kategori kata (*parts of speech*) untuk menganalisis tata bahasanya. Pada level selanjutnya (pemrosesan hilir), analisis ini bertujuan untuk mendukung pekerjaan pemrosesan yang lebih sulit.

2.2.2 *Chatbot*

Karena dapat meningkatkan kinerja mahasiswa dalam perkuliahan, layanan informasi sangat penting dalam pengaturan perkuliahan. Dengan kemajuan teknologi, muncul sistem yang membantu orang bereaksi terhadap pertanyaan, seperti sistem *chatbot*.

Chatbot sebagai perangkat lunak komputer masih terus berkembang di era teknologi modern. Perusahaan teknologi telah membuat aplikasi mereka sendiri untuk tujuan ini, seperti Apple dengan Siri dan Amazon dengan Alexa. Menurut penelitian (Muhammad dkk., 2023) tentang Sistem Percakapan Otomatis Berbasis Layanan Pesan Instan, *chatbot* berfungsi sebagai penjawab otomatis yang memberikan jawaban sesuai konteks sebagai tanggapan atas masukan pengguna, yang dapat diberikan secara lisan atau dalam bentuk teks.

Umumnya, *chatbots* biasanya memiliki obrolan singkat. Namun, ini tidak seperti pembicaraan biasa, yang mencakup kontak tatap muka dengan orang-orang. Terlepas dari kenyataan bahwa *chatbot* dibuat untuk meniru komunikasi manusia, tidak ada orang yang benar-benar berbicara. Karena *chatbot* "belajar" dan meningkatkan akurasi dengan data tambahan, mereka dianggap sebagai jenis kecerdasan buatan (AI).

2.2.3 *Deep Learning*

Subset *machine learning* yang dikenal sebagai "*deep learning*" terdiri dari beberapa lapisan yang ditumpuk satu sama lain. Algoritma atau teknik yang digunakan oleh setiap *layer* dalam *deep learning* bertugas mengklasifikasikan *input* yang dikirim ke *layer input* dan kemudian menghasilkan *output* yang dibutuhkan. Jaringan saraf berulang (RNN), memori jangka pendek panjang (LSTM), dan jaringan saraf *convolutional* (CNN) adalah beberapa teknik pembelajaran mendalam yang saat ini sedang dikembangkan (Gunawan & Nurfita, 2018).

2.2.4 *Remove punctuations*

Dalam pemrosesan teks, *remove punctuation* dari teks adalah tujuan dari prosedur *remove punctuation*. Titik, koma, tanda tanya, tanda seru, tanda kutip, dan tanda baca lainnya dianggap sebagai karakter *non alfa-numerik* yang memiliki pengaruh kecil pada cara beberapa teks diproses.

Untuk memastikan bahwa hanya kata atau karakter penting yang dipertahankan untuk analisis teks atau tugas pemrosesan bahasa alami lainnya, tanda baca harus dihapus dari teks. Sebelum *tokenization* atau langkah pra-pemrosesan lainnya, prosedur ini sering digunakan untuk mendapatkan teks yang sesuai untuk analisis selanjutnya.

2.2.5 *Tokenization*

Tokenization adalah proses pengelompokan urutan karakter dokumen ke dalam unit yang telah ditentukan. *Tokenization* bertujuan untuk memisahkan kalimat menjadi token dan menghilangkan beberapa komponen, seperti tanda baca.

Selain itu *tokenization* juga merupakan proses membagi teks atau kumpulan teks menjadi lebih kecil, disebut sebagai komponen "token". Kata, frasa, simbol, dan komponen lain dengan makna spesifik konteks tertentu dapat digunakan sebagai token. Untuk membantu komputer menafsirkan dan menganalisis teks dengan lebih baik, pemrosesan bahasa alami sering menggunakan proses *tokenization*.

2.2.6 *Padding Text*

Membuat setiap urutan dengan panjang yang sama adalah prosedur yang dikenal sebagai *padding*. *Padding* menyetarakan panjang setiap urutan dengan

menambahkan nilai 0 ke awal atau akhir setiap urutan hingga panjang maksimum urutan tercapai. Selain itu, *padding* dapat memotong urutan sehingga panjangnya sama dengan maksimum urutan.

2.2.7 *Encoding*

Encoding adalah proses mengubah data kategori seperti karakter atau teks menjadi data *numerik* sesuai dengan label data yang digunakan. *Encoding* mengubah data teks pada kolom tag data menjadi data *numerik* pada tahap ini menggunakan nilai 0 dan 1 yang digunakan oleh komputer untuk representasi *biner*.

2.2.8 *Vocabulary*

Kelompok tunggal dari semua kata yang ada dalam korpus atau kumpulan teks tertentu disebut sebagai kosa kata (*vocabulary*). Daftar kosakata adalah daftar semua kata yang ditemukan dalam kumpulan data atau teks yang digunakan untuk pelatihan atau analisis dalam konteks pemrosesan bahasa alami (NLP). Model NLP menggunakan indeks atau angka tertentu untuk setiap kata dalam kosakata selama representasi dan pemrosesan lebih lanjut.

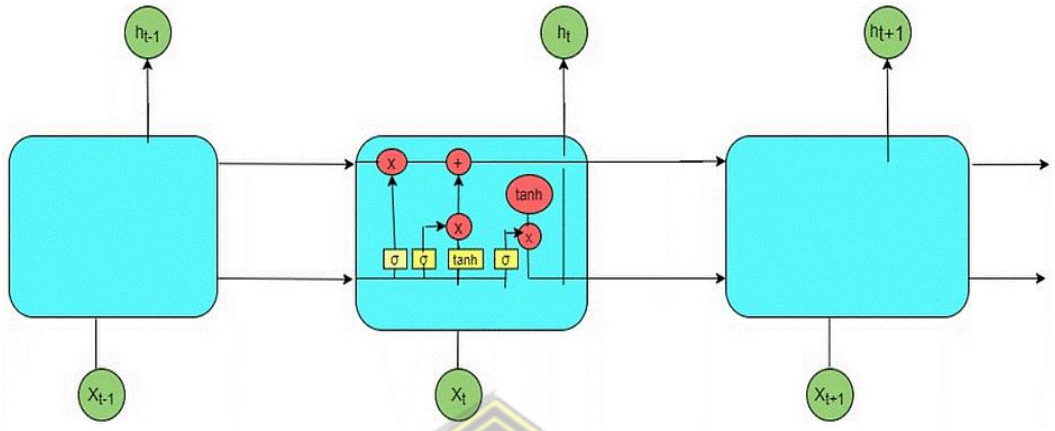
Karena menyediakan kerangka bagi model bahasa untuk memahami dan menghasilkan dokumen, kosakata sangat penting dalam NLP. Sekelompok istilah dari teks pelatihan digunakan untuk membuat kosa kata model NLP selama pelatihan. Dalam konteks model, kata-kata yang jarang muncul atau tidak pernah muncul selama pelatihan dapat diabaikan atau diperlakukan sebagai kata yang tidak dikenal.

Bergantung pada kumpulan data yang digunakan dan bagaimana *tokenization* dilakukan, ukuran kosa kata dapat berubah. Meskipun kosa kata yang lebih luas dapat membantu model dalam memahami rentang bahasa yang lebih luas, hal itu juga dapat membuat perhitungan menjadi lebih kompleks. Kosakata yang terbatas, di sisi lain, mungkin tidak dapat menangkap fitur bahasa yang lebih bernuansa secara memadai.

2.2.9 *Long Short-Term Memory*

Struktur algoritma LSTM terdiri dari jaringan saraf dan elemen bangunan memori lainnya. *cell* adalah nama untuk unit memori diskrit ini. *State* dari *cell* dan *hidden state* selanjutnya diteruskan ke *cell* berikutnya. Seperti yang terlihat pada

gambar 2.1 (Trivusi, 2022).

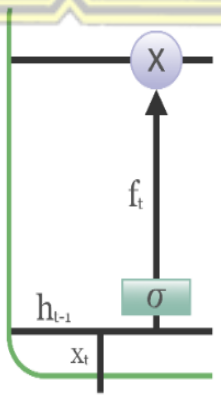


Gambar 2. 1 Jaringan LSTM (Trivusi, 2022)

Pada gambar 2.1 merupakan ilustrasi *cell* pada LSTM, *cell* kemudian akan menyimpan data yang dikumpulkan oleh algoritma LSTM, dan komponen yang dikenal sebagai *gate* akan menangani modifikasi memori. Algoritma LSTM menggunakan tiga jenis *gate*, diantaranya *Forget Gate*, *Input Gate*, dan *Output Gate* (Trivusi, 2022).

1. *Forget Gate*

Forget gate berfungsi untuk membersihkan informasi usang dari *cell*. Caranya adalah mengevaluasi *output biner* dari dua *input* x(t) dan *output cell* sebelumnya h(t-1), dikalikan dengan matriks bobot dan ditambahkan nilai bias. kemudian melewati fungsi aktivasi agar menghasilkan *output biner* yang terlihat pada gambar 2.2 (Trivusi, 2022).



Gambar 2. 2 *Forget Gate* (Trivusi, 2022)

Jika *output*-nya bernilai 0, data dianggap tidak berguna dan dapat dihapus. Di

sisi lain, jika *output*-nya adalah 1, data disimpan untuk digunakan nanti (Trivusi, 2022). Dari gambar 2.2 merupakan arsitektur dari *forget gate*, dimana pada arsitektur tersebut memiliki rumus sebagai berikut:

$$f_t = \sigma(W_f . [h_{t-1}, x_t] + b_f) \quad (1)$$

Keterangan:

x_t = nilai *input*

h_{t-1} = keluaran sebelumnya.

f_t = nilai *output* dari unit atau sel LSTM.

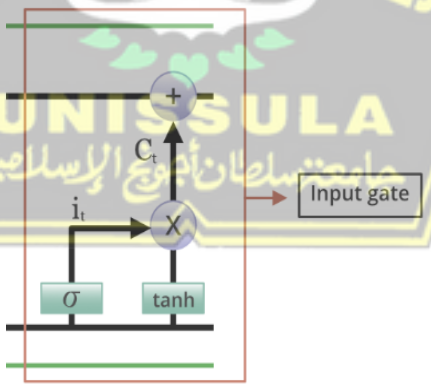
σ = fungsi aktivasi *sigmoid*.

W_f = matriks bobot.

b_f = vektor bias

2. *Input Gate*

Input gate memodifikasi keadaan *cell* dengan menambahkan informasi terkait. Tekniknya mirip dengan *forget gate*, yaitu menggunakan *input* h(t-1) dan x(t), dan dimulai dengan pengaturan informasi menggunakan fungsi *sigmoid* dan mem-*filter* nilai yang akan disimpan yang terlihat pada gambar 2.3 (Trivusi, 2022).



Gambar 2. 3 *Input Gate (Trivusi, 2022)*

Kemudian, dengan menggunakan fungsi *tanh*, sebuah vektor dihasilkan yang berisi semua nilai yang dapat dibayangkan dari h(t-1) dan x(t) dan memiliki rentang keluaran -1 hingga +1. Untuk mengekstrak informasi yang bermakna, nilai vektor dan nilai yang ditetapkan dikalikan (Trivusi, 2022). Dari gambar 2.3 merupakan arsitektur dari *input gate*, dimana pada arsitektur tersebut memiliki rumus sebagai

berikut:

$$i_t = \sigma(W_i . [h_{t-1}, x_t] + b_i) \quad (2)$$

$$C_t = \tanh\,(W_c . [h_{t-1}, x_t] + b_c) \quad (3)$$

Keterangan (2):

x_t = nilai *input* saat ini.

h_{t-1} = keluaran sebelumnya.

i_t = nilai *output* dari unit atau sel LSTM *input gate*.

σ = fungsi aktivasi *sigmoid*.

b_i = vektor bias.

W_i = matriks bobot.

Keterangan (3):

C_t = nilai ouput daru unit atau sel LSTM *cell state*.

$Tanh$ = fungsi aktivasi tangen hiperbolik.

h_{t-1} = keluaran sebelumnya.

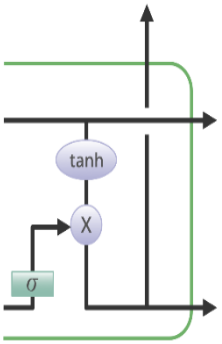
x_t = nilai *input*.

W_c = matriks bobot.

b_c = vektor bias.

3. *Output Gate*

Tugas *output gate* adalah mengambil informasi yang berarti dari *cell state* saat ini dan mengirimkannya sebagai nilai keluaran. Fungsi *tanh* pertama kali digunakan untuk membuat vektor dari sel. Data tersebut kemudian diatur menggunakan fungsi *sigmoid*, yang mem-*filter input* h_t-1 dan x_t menjadi nilai yang disimpan. Nilai vektor dan nilai yang ditetapkan kemudian dikalikan dan diberikan sebagai output dan *input* ke sel selanjutnya, yang terlihat pada gambar 2.4 (Trivusi, 2022).



Gambar 2. 4 *Output Gate (Trivusi, 2022)*

Dari gambar 2.4 merupakan arsitektur dari *output gate*, dimana pada arsitektur tersebut memiliki rumus sebagai berikut:

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \quad (4)$$

$$h_t = o_t * \tanh(C_t) \quad (5)$$

Keterangan (4):

h_{t-1} = keluaran sebelumnya.

x_t = nilai *input* saat ini.

o_t = nilai *output* dari unit atau sel LSTM *output gate*.

σ = fungsi aktivasi *sigmoid*.

b_o = *vector* bias.

Keterangan (5):

h_t = nilai *output* dari unit atau sel LSTM.

o_t = nilai *output* dari unit atau sel LSTM *output gate*.

Tanh = fungsi aktivasi tangen hiperbolik.

C_t = nilai *output* dari unit atau sel LSTM *cell state.*

BAB III
METODOLOGI PENELITIAN

3.1 Metode Penelitian

Penelitian ini menggunakan metode atau algoritma yang disebut *Long Short-Term Memory* (LSTM) dan *Natural Language Processing* (NLP). Dalam penelitian ini, kedua pendekatan tersebut digabungkan, dengan LSTM berfungsi sebagai sistem penyimpanan yang dapat memproses, meramalkan, dan mengkategorikan informasi data berdasarkan urutan waktu dan NLP berfungsi sebagai komponen dari proses penyisipan kata. Langkah-langkah yang harus dilakukan dalam penelitian ini adalah sebagai berikut:

3.1.1 Studi *Literature*

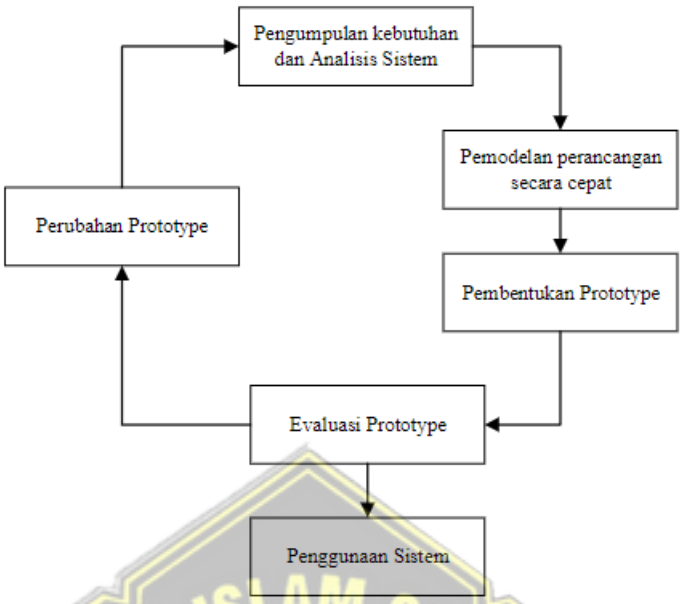
Dalam penelitian ini akan dilakukan tinjauan terhadap beberapa *e-book*, makalah, jurnal, tesis, dan skripsi terdahulu sebagai ulasan dan juga akan mengunjungi berbagai situs *web*. Tujuan dari tinjauan ini adalah untuk mempelajari teori di balik konsep seperti *Natural Language Processing* (NLP), dan metode *Long Short-Term Memory*.

3.1.2 Pengumpulan Data

Pada tahap ini sebagai persiapan awal sebelum pengembangan model. Data yang dikumpulkan diambil dari panduan Tugas Akhir Prodi Teknik Informatika UNISSULA. Dalam penelitian ini, *dataset* yang didapat berupa teks dengan format PDF yang akan dijadikan acuan dalam merancang pertanyaan dan jawaban.

3.1.3 Metode Perancangan Sistem

Pada tahap ini mengatur langkah-langkah dan proses perancangan sistem dengan menggunakan metode *Prototype*. Metode ini dibuat secara terstruktur dan memiliki beberapa tahap-tahap yang harus dilalui dalam pembuatannya, namun jika tahap final dinyatakan bahwa sistem yang telah dibuat belum sempurna maka sistem dievaluasi kembali. Alur dari metode *prototype* dapat dilihat pada gambar 3.1 (Renaningtias & Apriliani, 2021).



Gambar 3. 1 Metode *Prototype (Renaningtias & Apriliani, 2021)*

Pada gambar 3.1 merupakan alur dari model *prototype* yang digunakan sebagai acuan dalam perancangan sistem yang akan digunakan pada penelitian kali ini. Dari alur tersebut memiliki beberapa tahapan dalam perancangan sistem yang akan dijelaskan pada tahapan perancangan sistem.

3.1.4 Tahapan Perancangan Sistem

Dalam tahap perancangan sistem, akan menjelaskan mengenai tahapan-tahapan dalam metode *prototype*, yaitu sebagai berikut:

1. Pengumpulan Kebutuhan dan Analisis Sistem

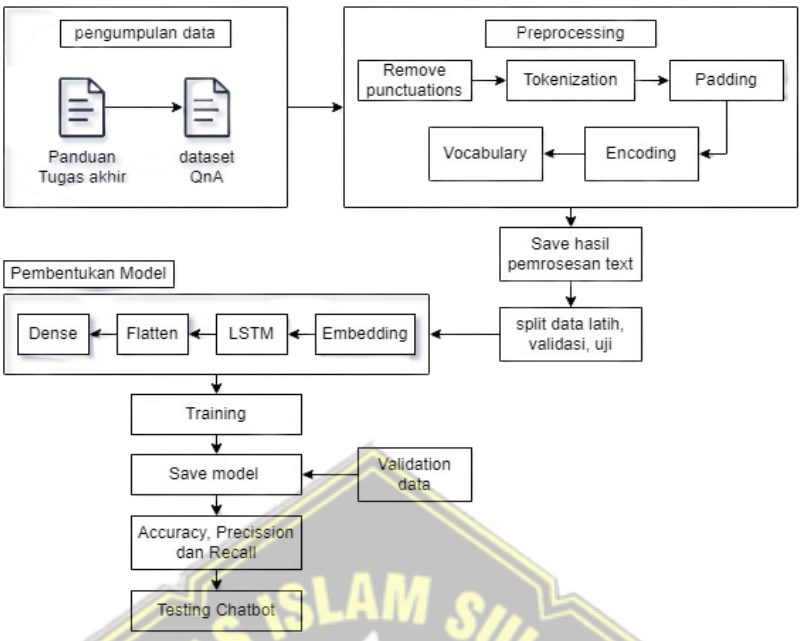
Hasil konsultasi bersama dosen pembimbing digunakan untuk menetapkan kebutuhan sistem dan tujuan. Kemudian, kebutuhan tersebut diuraikan secara rinci dan berperan sebagai spesifikasi sistem. Dalam pengumpulan kebutuhan

2. Pemodelan Perancangan Secara Cepat

Tahapan berikutnya adalah pemodelan desain secara cepat, tahapan ini berfungsi sebagai panduan untuk membuat model *prototype*. Pada tahapan ini juga bertujuan untuk memberikan gambaran model seperti apa yang akan dibangun.

3. Pembentukan *Prototype*

Pada fase ini, *prototype* dirancang menggunakan desain pemodelan yang telah diselesaikan sebelumnya. Rancangan *prototype* terlihat pada gambar 3.3.



Gambar 3. 2 Diagram Alur Perancangan Sistem

Pada gambar 3.3 merupakan diagram alur sistem, dimana pada diagram tersebut memiliki beberapa tahapan yang dilalui dari pengumpulan data hingga pembentukan model yang akan dijelaskan sebagai berikut:

1. Melakukan pengumpulan data dengan membuat data pertanyaan dan jawaban berdasarkan Panduan Tugas Akhir. Data pertanyaan dan jawaban yang dibuat hanya terkait pada bab 1 dan bab 2 dalam panduan tugas akhir. Setelah data dikumpulkan, data disimpan dalam format JSON dan dikonversi menjadi data *frame*.
2. Pada tahap *preprocessing* bertujuan untuk mengolah data yang telah konversi menjadi data frame dengan cara melakukan *Remove Punctuation*, yaitu melakukan penghapusan tanda baca atau simbol khusus pada teks. Dalam proses ini, saya menghapus karakter seperti titik, koma, tanda tanya, dan simbol lainnya yang tidak memberikan makna tambahan pada teks.
3. Selanjutnya tahap *Tokenization* yang digunakan untuk memecah teks menjadi kata per kata. Pada proses ini kalimat, yang sudah melewati pembersihan teks akan masuk ke proses *tokenization* untuk mengubah kalimat teks menjadi urutan angka yang merepresentasikan per kalimat.
4. Setelah itu masuk proses *Padding* yang bertujuan untuk mengubah setiap

urutan angka agar memiliki Panjang yang sama. Proses ini bertujuan agar saat masuk ke pelatihan model dapat bekerja secara konsisten.

5. Selanjutnya adalah tahap *Encoding*, yaitu mengubah setiap huruf atau data teks menjadi numerik atau angka.
6. Selanjutnya tahap *vocabulary*, menghitung jumlah kata unik dalam kamus yang di indeks oleh objek *tokenizer* dan menyimpannya sebagai *vocabulary*.
7. Setelah itu tahapan *save* hasil pemrosesan *text*, menyimpan hasil dari data pemrosesan *text*.
8. Selanjutnya *split* data latih, pada tahap ini data yang sudah melalui pemrosesan teks akan dibagi menjadi data latih dan data validasi untuk menghitung akurasi, presisi dan *recall*.
9. Setelah melalui pemrosesan teks dan pembagian data, lalu masuk ke pembentukan model. Dimana pada pembentukan model ada beberapa lapisan yang harus dilewati oleh data yang sudah melalui pemrosesan data yang pertama, membangun lapisan *Embedding* untuk memetakan kata-kata dalam teks ke dalam *vector* ruang.
10. Setelah itu membangun lapisan LSTM untuk memproses urutan *inputan* secara berurutan.
11. Selanjutnya *Flatten* yaitu untuk meratakan *output* dari LSTM menjadi *vector* satu dimensi.
12. Setelah itu *Dense* digunakan untuk menghasilkan *output* yang sesuai dengan jumlah kelas pada masalah klasifikasi.
13. Selanjutnya *Training*, yaitu melatih model dengan menggunakan data yang sudah melalui pemrosesan teks dan pembagian data. Data yang sudah dibagi berupa data latih dan data uji.
14. Setah itu *save* model, dimana pada tahap ini menyimpan hasil data yang sudah melalui proses *training*. Data disimpan dalam format h5.
15. Pada tahap evaluasi model, yaitu mengukur hasil *training* untuk mengetahui akurasi, presisi dan *recall*. Tahap ini digunakan untuk mengetahui hasil *training* dari data model apakah data yang telah melewati proses *training* dapat digunakan sebagai data latih dan validasi.

16. Pada tahap *Testing*, model yang sudah melewati proses pelatihan maka akan dilakukan uji coba agar dapat mengetahui hasil dari pelatihan model dapat bekerja dengan baik.

4. Evaluasi *Prototype*

Pada tahapan ini, dilakukan evaluasi terhadap *prototype* yang disesuaikan dengan kebutuhan. Jika belum sesuai dengan kebutuhan, maka dapat dilakukan perubahan *prototype* hingga sesuai dengan kebutuhan.

5. Perubahan *Prototype*

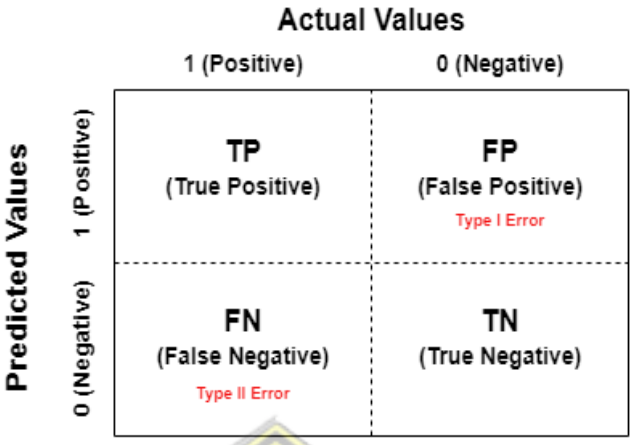
Tahapan ini digunakan apabila terdapat perubahan yang dibutuhkan untuk menyempurnakan *prototype* yang dibangun agar menghasilkan *prototype* yang sesuai dengan apa yang dibutuhkan.

6. Penggunaan Sistem

Tahap ini merupakan tahap akhir dalam metode *prototype*. Tahap ini dilakukan apabila sistem yang dibangun sudah melewati tahap evaluasi, jika sudah melewati tahap evaluasi maka sistem siap digunakan.

3.1.5 Evaluasi Model

Matriks kesalahan adalah nama lain untuk *confusion matrix*. *Confusion matrix* pada dasarnya membandingkan hasil klasifikasi yang dihasilkan oleh sistem (model) dengan hasil klasifikasi aktual. Kinerja model klasifikasi pada sekumpulan data uji yang nilai riilnya diketahui dijelaskan oleh *confusion matrix*, yang berbentuk tabel matriks (Nugroho, 2019). Matriks kebingungan memiliki empat kemungkinan kombinasi dari nilai yang diharapkan dan nilai aktual ditunjukkan pada gambar 3.4.



Gambar 3. 3 Tabel *Confusion Matrix (Nugroho, 2019)*

Pada gambar 3.4 Terdapat empat istilah sebagai representasi hasil proses klasifikasi pada *confusion matrix*. *Confusion matrix* sering digunakan untuk mengukur kinerja model yang dibuat. Berikut istilah representasi hasil dari *confusion matrix*, yaitu:

1. *True Positive* (TP) yang merupakan data positif yang diprediksi benar.
2. *True Negative* (TN) merupakan data negatif yang diprediksi benar.
3. *False Positive* (FP) merupakan data negatif namun diprediksi sebagai data positif.
4. *False Negative* (FN) merupakan data positif namun diprediksi sebagai data negatif.

Performance matrix yang umum dan sering digunakan yaitu *Accuracy, Precision* dan *Recall*

A. *Accuracy*

Seberapa akurat model dapat mengklasifikasikan objek disebut sebagai akurasi. Jadi, rasio prediksi yang benar (baik positif maupun negatif) terhadap semua data adalah ukuran akurasi-nya. Atau dikatakan, akurasi adalah sejauh mana nilai prediksi menyerupai nilai nyata (aktual). Persamaan (6) dapat digunakan untuk menentukan nilai akurasi.

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (6)$$

B. *Precision*

Tingkat akurasi antara data yang diminta dan keluaran model yang diharapkan disebut sebagai presisi. Dengan demikian, rasio perkiraan positif yang benar terhadap keseluruhan hasil yang diharapkan positif dikenal sebagai presisi. Berapa banyak data yang benar-benar positif di antara semua kelas positif yang diprediksi dengan benar. Persamaan (7) dapat digunakan untuk menentukan nilai presisi.

$$precision = \frac{TP}{TP + FP} \quad (7)$$

C. *Recall*

Keberhasilan model dalam pencarian informasi ditunjukkan dengan *recall*. Oleh karena itu, penarikan kembali adalah proporsi prediksi positif yang benar untuk semua data positif yang benar. Persamaan (8) dapat digunakan untuk mendapatkan nilai *recall*.

$$recall = \frac{TP}{TP + FN} \quad (8)$$

3.1.6 Pengujian *Black Box*

Pada tahap ini, aplikasi yang telah dibuat dilakukan pengujian dengan menggunakan metode *Black Box Testing*. Tujuannya adalah untuk mengetahui apakah aplikasi berjalan dengan baik dan sesuai dengan spesifikasi kebutuhan. Metode *Black Box Testing* merupakan metode pengujian untuk mengamati hasil masukan dan keluaran dari perangkat lunak tanpa harus mengetahui kode program dari perangkat lunak.

Black Box Testing dilakukan dengan memberikan berbagai jenis *input* yang berbeda ke aplikasi dan mengetahui *respon* atau *output* yang dihasilkan. Dengan mencoba berbagai skenario penggunaan dalam beragam situasi yang mungkin terjadi saat aplikasi ber-intraksi dengan pengguna atau sistem lainnya.

3.2 Analisis Kebutuhan

pada tahap analisis kebutuhan, penelitian ini akan menganalisis *software* apa saja yang akan dibutuhkan dalam pengembangan aplikasi ini. sehingga, mulai dari proses *input* hingga hasil akhir dapat berjalan sesuai dengan yang diharapkan. berikut adalah daftar perangkat lunak yang digunakan dalam pengembangan sistem:

1. Python 3.11.0

Python adalah bahasa pemrograman tingkat tinggi yang fleksibel dengan fokus pada keterbacaan dan kesederhanaan. Python, yang dikembangkan oleh Guido van Rossum dan pertama kali tersedia pada tahun 1991, telah berkembang menjadi salah satu bahasa pemrograman yang paling disukai dan sering digunakan di seluruh dunia. Python versi 3.11.0 digunakan sebagai bahasa pemrograman untuk penelitian ini karena memiliki *library* yang lengkap dan bersifat *open source*.

2. *Library String*

Library string merupakan pustaka bawaan di Python yang menyediakan berbagai konstanta dan fungsi yang terkait dengan manipulasi *string* atau teks. Dalam penelitian ini, library string digunakan untuk *remove punctuation* atau menghapus tanda baca umum seperti !"#$%&'()*+,-./:;<=>?@[\]^_{|}~` serta membuat teks menjadi huruf kecil.

3. *Library* NLTK (*Natural Language Tool Kit*)

Natural Language Toolkit (NLTK) adalah *library* atau *toolkit* populer dalam bahasa pemrograman Python yang digunakan untuk pemrosesan *Natural Language Processing* (NLP). dalam penelitian ini, NLTK digunakan dalam proses pengolahan teks seperti *tokenization*.

4. *Library TensorFlow*

TensorFlow adalah sebuah *framework* sumber terbuka (*open-source*) untuk kecerdasan buatan (*Artificial Intelligence*) dan pembelajaran mesin (*Machine Learning*). Dalam penelitian ini *library TensorFlow* digunakan untuk membangun model *neural network* dengan berbagai lapisan (*layers*) seperti *Input*, *Embedding*, LSTM, *Flatten*, *Dense*, dan GlobalMaxPool1D.

5. *Library Keras*

Keras adalah API *deep learning* yang ditulis dengan python dan berjalan di atas *platform machine learning TensorFlow*. *Library Keras* menjadi bagian dari *TensorFlow* mulai dari versi *TensorFlow* 2.x. Pada penelitian ini *library Keras* digunakan untuk membangun dan melatih model *neural network*.

6. *Library* NumPy

NumPy (*Numerical Python*) adalah *library* python yang digunakan untuk bekerja dengan *array* dan juga memiliki fungsi yang bekerja dalam *domain* aljabar

linier, transformasi *Fourier*, dan matriks.

7. *Library Pandas*

Library Pandas python merupakan *library* yang digunakan untuk memproses data, mulai pembersihan data, manipulasi data, hingga melakukan analisis data. Dalam penelitian ini *library Pandas* digunakan untuk membuat laporan data pada saat *testing* dalam *google colab* dan juga digunakan dalam pengolahan data menjadi data JSON yang nantinya akan ditampilkan dalam tampilan *website*.

8. *Library Pickle*

Library pickle adalah sebuah *library* di Python yang digunakan untuk serialisasi dan deserialisasi objek Python. dalam penelitian ini, *library pickle* digunakan untuk menyimpan dan mengambil objek Python dalam bentuk serialisasi.

9. *Library* Json

Library JSON (*Java Script Object Notation*) adalah sebuah *library* atau modul di Python yang digunakan untuk bekerja dengan data dalam format JSON.

10. *Library Scikit-learn*

Library scikit-learn, adalah *library* yang digunakan untuk pembelajaran mesin (*machine learning*) yang ditulis dalam bahasa pemrograman Python. Dalam penelitian ini *library scikit-learn* digunakan sebagai evaluasi performa model dengan cara mengukur model menggunakan *confusion matrix*, *accuracy score*, *precision score*, dan *recall score* untuk mengukur akurasi, presisi, *recall*, dan matriks kebingungan dari model yang telah dilatih.

11. *Library Matplotlib*

Library matplotlib adalah sebuah *library* di Python yang digunakan untuk membuat visualisasi data dalam bentuk grafik, *plot*, diagram, dan gambar lainnya. dalam penelitian ini, *library Matlplotlib* digunakan untuk mem-visualisasi data dengan grafik garis dan menampilkannya.

12. *Framework Streamlit*

Streamlit adalah sebuah *framework* berbasis Python dan bersifat *open-source* yang dibuat untuk memudahkan dalam membangun aplikasi *web* di bidang *sains* data dan *machine learning* yang interaktif (UII, 2022). Dalam penelitian,

framework streamlit digunakan sebagai pengembangan aplikasi *website* dikarenakan *core* dari sistem yang akan dikembangkan sendiri berbahasa python.

13. *Jupyter Notebook*

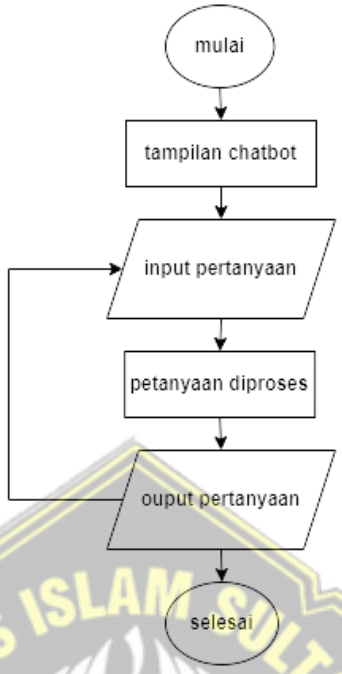
Jupyter Notebook adalah sebuah aplikasi *web open-source* yang digunakan untuk mengembangkan dan menjalankan kode dalam lingkungan komputasi interaktif. dalam penelitian ini, *jupyter notebook* digunakan untuk menulis dan menjalankan kode program, serta membuat penyimpanan catatan atau dokumentasi.

14. *Visual Studio Code*

Visual Studio Code dipilih sebagai *text editor* pada pengembangan aplikasi dalam penelitian ini, *Visual Studio Code* dipilih dikarenakan mendukung banyak Bahasa pemrograman dan *framework*, *multi-platform*, performa yang sangat cepat, mempunyai banyak *extensions* yang dapat mempermudah proses pengembangan.

3.3 Analisis Sistem

Pada tahap ini, dilakukan analisis untuk menentukan alur kerja dari sistem yang akan dibuat dalam bentuk *flowchart*. Pertama-tama *user* membuka aplikasi. Selanjutnya akan langsung bertemu dengan halaman *chatbot*. *User* dapat mengetikkan pertanyaan pada *text input*. *User* dapat menekan tombol *send* untuk mengirim pertanyaan. Setelah *user* mengirim pertanyaan, maka pertanyaan yang dikirim akan diproses oleh sistem dengan cara mencocokkan pertanyaan yang dikirim dengan pertanyaan sudah dilatih oleh model LSTM. Apabila terdapat pertanyaan yang sama maka sistem akan memberikan *output* berupa jawaban dan menampilkan di halaman *chatbot*. Gambaran *flowchart* sistem *chatbot* dapat dilihat pada Gambar 3.4.



Gambar 3. 4 *Flowchart* sistem *Chatbot*

Pada Gambar 3.4 merupakan *flowchart chatbot* yang menggambarkan proses sistem *chatbot* berjalan. Dari alur sistem *chatbot* tersebut memiliki tahapan yang akan dijelaskan sebagai berikut:

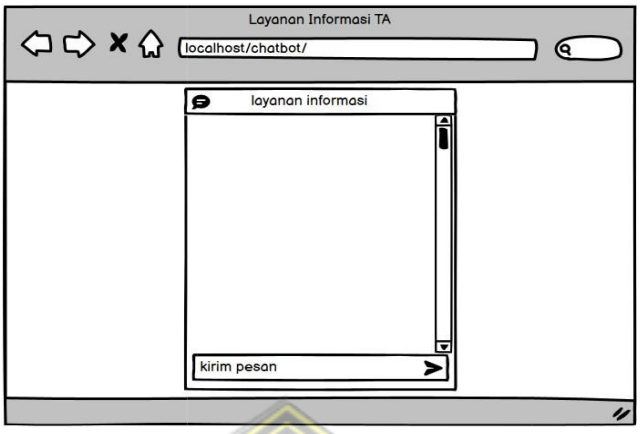
a. Pertama pengguna masuk ke tampilan *chatbot*, pada tampilan *chatbot* pengguna akan melihat bagian *appbar*, *body* dan bagian *input* text.
b. Kemudian pengguna dapat menginputkan pertanyaan di bagian *text input*, setelah pengguna menginputkan pertanyaan pengguna dapat mengirimnya denga cara menekan "*Enter*" pada *keyboard* atau juga dapat dengan cara meng-klik *icon* pesawat kertas di dalam *input text*.
c. Setelah pertanya terkirim, maka sistem akan memproses pertanyaan tersebut.
d. Jika pertanyaan yang dikirim memilik kecocokan dengan data yang sudah dilatih oleh model maka sistem akan memberikan *responses* berupa jawaban dan akan menampilkan pertanyaan dan jawab di bagian *body*.

3.4 Perancangan *User Interface*

3.4.1 Halaman Awal *Chatbot*

Halaman awal *chatbot* merupakan halaman awal sebelum memulai

percakapan antara pengguna dengan *chatbot* seperti pada Gambar 3.5.

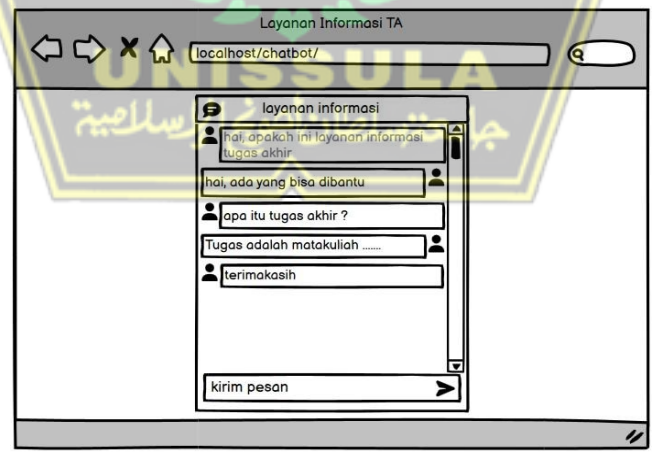


Gambar 3. 5 Halaman Awal *Chatbot*

Pada Gambar 3.5 merupakan halaman pertama pengguna pertama kali akan melihat halaman percakapan, pengguna juga melihat bagian *input text* untuk memberikan pertanyaan nantinya dan terdapat *icon* pesawat ketas sebagai tombol untuk mengirim pertanyaan nantinya.

3.4.2 Halaman Percakapan *Chatbot*

Halaman Percakapan *chatbot* merupakan halaman yang menampilkan hasil pertanyaan dari pengguna dan jawaban yang diberikan dari sistem *chatbot* tersebut, seperti terlihat pada Gambar 3.6.



Gambar 3. 6 Halaman *Chatting*

Pada gambar 3.6 merupakan halaman yang menampilkan isi percakapan antara pengguna dengan *bot*. Dimana pada tahapan ini pertanyaan yang sudah diinputkan oleh pengguna dan di proses oleh sistem, maka sistem akan memberikan

responses dengan cara memberikan jawaban dan menampilkannya di bagian *body* pada halaman *chatbot*. Penelitian ini akan mendemokan halaman *web* interaktif yang sudah dibuat kepada pengguna dan apakah pengguna sudah merasa sesuai dengan apa yang diinginkan.

BAB IV
HASIL DAN ANALISIS PENELITIAN

4.1 Hasil Penelitian

Data yang didapat merupakan informasi mengenai panduan Tugas Akhir, data tersebut akan dijadikan sebagai jawaban. Data ini merupakan data yang panting bagi mahasiswa, karena data tersebut berisikan panduan pengerjaan Tugas Akhir yang dapat membantu mahasiswa dalam proses pengerjaan Tugas Akhir. Data didapat dari Prodi Teknik Informatika UNISSULA yang diambil dari https://linktr.ee/tugasakhir dalam bentuk *file* pdf.

Setelah data panduan Tugas Akhir didapatkan, selanjutnya membuat data pertanyaan dan data jawaban. Pertanyaan dibuat berkaitan denga data panduan Tugas Akhir seperti tahap pengambilan matakuliah TA, pelaksanaan dan penyusunan laporan TA, proses bimbingan TA, pendaftaran sidang TA, pelaksanaan sidang TA, maupun penilaian TA.

Setelah mendapatkan data dan melakukan pemahaman terhadap data, selanjutnya adalah tahap persiapan data. Data yang dijadikan acuan untuk membuat jawaban dan data pertanyaan selanjutnya dibuat menjadi satu *file* JSON yang terdiri dari *intent*, *tag*, *pattern* dan *response* seperti pada gambar yang kemudian akan diproses dengan python. Jumlah data yang sudah dibuat adalah 983 data yang terdiri dari data *tag* sebagai label, *patterns* sebagai pertanyaan dan *responses* sebagai jawaban. Data tersebut dibuat dalam format JSON kemudian dikonversi menjadi *dataframe Pandas*. Berikut merupakan data JSON yang sudah dikonversi menjadi *dataframe* menjadi sebuah *list patterns* dan *tag* yang terlihat pada gambar 4.1.

```
#printing the data
data
```

1 to 25 of 983 entries Filter

index	patterns	tag
0	halo	menyapa
1	hei	menyapa
2	hai	menyapa
3	hai, apakah ini layanan informasi tugas akhir	menyapa
4	hei, apakah ini layanan informasi tugas akhir	menyapa
5	halo, apakah ini layanan informasi tugas akhir	menyapa
6	Terima kasih	berakhir
7	Terima kasih infonya	berakhir
8	bye	berakhir
9	bye bro	berakhir
10	oke bye	berakhir
11	selamat tinggal	berakhir
12	sampai jumpa lagi	berakhir
13	oke aku akan menemuimu nanti	berakhir
14	aku akan berbicara denganmu nanti	berakhir
15	akan mengirimimu pesan nanti	berakhir
16	bye bro, aku akan berbicara denganmu nanti	berakhir
17	Deskripsi dari Tugas Akhir?	definisi
18	Definisi dari TA?	definisi
19	Maksud dari Tugas Akhir	definisi
20	Apa yang dimaksud dengan Tugas Akhir?	definisi
21	Jelaskan Apa itu Tgas Akhir?	definisi
22	Apa arti yang terkandung dalam Tugas Akhir?	definisi
23	Apa yang sebenarnya dimaksudkan dengan istilah Tugas Akhir?	definisi
24	Apa itu TA atau Tugas Akhir?	definisi

Show 25 per page 1 2 10 30 40

Gambar 4. 1 Konversi *Dataframe*

Pada Gambar 4.1 merupakan gambar *table dataframe* yang dihasilkan dari *dataset* dengan format JSON. Data pertanyaan yang telah diubah menjadi *list* kemudian dibersihkan. Proses pembersihan data terlihat pada gambar 4.2.

```
[ ] #removing punctuations
    import string
    data['patterns'] = data['patterns'].apply(lambda wrd:[ltrs.lower() for ltrs in wrd if ltrs not in string.punctuation])
    data['patterns'] = data['patterns'].apply(lambda wrd: ''.join(wrd))
    data
```

Gambar 4. 2 Proses *Remove Punctuation*

Terlihat pada gambar 4.2, menggunakan *statement "import string"* untuk mengimpor modul *string* yang berisi berbagai konstanta karakter seperti tanda baca. Modul ini digunakan untuk mendeteksi dan menghapus tanda baca. Penghapusan tanda baca dilakukan pada kolom *patterns* dalam *dataframe* data. Dimana pada tahap ini menggunakan metode *apply* pada kolom *patterns* untuk menerapkan fungsi *lambda* pada setiap elemen (kata-kata) dalam kolom tersebut. Fungsi *lambda* sendiri digunakan untuk mengubah kalimat menjadi huruf kecil dan mengabaikan karakter yang termasuk kedalam tanda baca menggunakan *string punctuation*.

Setelah mengubah kalimat menjadi huruf kecil dan menghilangkan tanda baca, selanjutnya menggabungkan karakter kata menjadi satu *string* dengan

menggunakan metode *join*. Tujuan dari proses ini adalah agar dapat mempermudah dalam pemrosesan teks selanjutnya, hasil dari pembersihan kalimat tersebut dapat dilihat pada gambar 4.3.

index	patterns	tag
227	kapan ta direncanakan akan dilaksanakan	pelaksanaan
352	apa yang perlu dilakukan agar dapat sukses dalam mengajukan judul tugas akhir	cara_pengajuan
806	apa yang sebenarnya menjadi makna dari istilah proposal ta	definisi_proposal_TA
868	apa yang harus dilakukan setelah membuat proposal ta atau tugas akhir	sebelum_pengajuan
628	apa yang perlu dilakukan saat mendaftar sidang tugas akhir	sidang
873	bagaimana tindakan selanjutnya setelah menyelesaikan penyusunan proposal ta atau tugas akhir	sebelum_pengajuan
871	apa yang harus dilakukan sebelum mengajukan proposal ta	sebelum_pengajuan
413	apa saja prosedur yang harus dipatuhi oleh mahasiswa dalam menjalankan tahapan seminar ta	tahapan seminar
226	kapan waktu yang ditentukan untuk menjalankan tugas akhir	pelaksanaan
404	apa saja proses yang harus dilewati mahasiswa pada tahap seminar	tahapan seminar
42	tugas akhir atau ta mohon pencerahan agar tidak lagi bingung	definisi
498	jika tidak lulus pada seminar proposal tindakan apa yang sebaiknya diambil oleh mahasiswa	ketentuan_sempro_tidak_lulus
88	apa alasan kuat mahasiswa harus menyelesaikan tugas akhir sebagai syarat kelulusan	tujuan
90	apa yang ingin mahasiswa tunjukkan ketika mereka melaksanakan tugas akhir	tujuan
290	apa langkah selanjutnya setelah merumuskan judul tugas akhir	tahapan
436	bagaimana tata tertib seminar proposal	tata_tertib_sempro
153	apa komponen persyaratan administrasi yang harus dipenuhi mahasiswa sebelum mendaftar matakuliah tugas akhir	administrasi
210	bagaimana persyaratan akademik yang harus terpenuhi sebelum mahasiswa dapat mengikuti matakuliah tugas akhir	akademik
721	syarat harus dipenuhi untuk mengikuti wisuda	wisuda
107	bagaimana prosedur pendaftaran matakuliah tugas akhir	mengambil_matkul_TA
758	bagaimana tata cara mendaftar dan memastikan persyaratan terpenuhi untuk ikut dalam upacara wisuda	wisuda
325	apa yang harus diperhatikan saat ingin mengajukan judul tugas akhir	cara_pengajuan
92	apa manfaat utama yang bisa diperoleh mahasiswa melalui kelengkapan tugas akhir	tujuan
51	terminologi tugas akhir sulit dipahami bisa dijelaskan lebih sederhana	definisi
415	apa langkahlangkah penting dalam menjalankan tahapan seminar tugas akhir	tahapan seminar

Show 25 per page 1 2 10 30 37

Gambar 4. 3 hasil *Remove Punctuation*

Pada Gambar 4.3 merupakan hasil dari proses pembersihan data. Setelah data dibersihkan maka akan dilakukan proses *tokenization* untuk mengubah setiap kalimat menjadi urutan bilangan bulat, proses tersebut terlihat pada gambar 4.4.

```
#tokenize the data
from tensorflow.keras.preprocessing.text import Tokenizer
tokenizer = Tokenizer(num_words=2000)
tokenizer.fit_on_texts(data['patterns'])
train = tokenizer.texts_to_sequences(data['patterns'])
```

Gambar 4. 4 Proses *Tokenization*

Terlihat pada gambar 4.4, dimulai dengan mengunduh modul *Tokenizer*. Dari *TensorFlow Keras*. Selanjutnya, membuat objek *tokenizer* dengan menginisialisasi *tokenizer(num_word=2000).* Parameter num words diatur ke 2000 karena *tokenizer* akan mempertimbangkan hanya 2000 kata paling umum dalam data. Setelah itu memproses teks dalam kolom *patterns* dari *dataframe* dan menghasilkan token-token menggunakan metode *fit on text*. Setelah dilakukan proses *tokenization*, mengunduh modul *pad sequences* dari *TensorFlow Keras*. Proses tersebut dapat

dilihat pada gambar 4.5.

```
#apply padding
from tensorflow.keras.preprocessing.sequence import pad_sequences
x_train = pad_sequences(train)
```

Gambar 4. 5 Proses *Padding text*

Terlihat pada gambar 4.5, dimana dari kode tersebut bertujuan untuk menghasilkan urutan angka yang memiliki panjang yang sama. Dalam proses ini, menentukan panjang urutan angka agar memiliki panjang yang sama merupakan proses yang sangat penting karena model yang akan dilatih harus membutuhkan data yang konsisten. Selanjutnya adalah proses *Encoding*, proses tersebut dapat dilihat pada gambar 4.6.

```
#encoding the outputs
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
y_train = le.fit_transform(data['tag'])
y_test = le.transform(data['tag'])
```

Gambar 4. 6 Proses *Encoding*

Terlihat pada gambar 4.6, proses ini menggunakan modul *Label Encoder* dari *Scikit Learn*, proses ini dilakukan untuk mengubah kalimat pada kolom *tag* dari *dataframe* menjadi label dalam bentuk *numerik*. Proses ini dilakukan dengan *fitting* dan *transformasi* pada kolom tag dengan menggunakan metode *le.fit_transform*. Hasil dari proses tersebut menjadi nilai *numerik* yang merepresentasikan kategori atau label pada kolom *tag*.

Tahap berikutnya adalah proses *Vocabulary*, pada tahap ini bertujuan untuk menghitung beberapa parameter penting yang akan digunakan dalam pelatihan model. Berikut proses dari *Vocabulary* yang terlihat pada gambar 4.7.

```
#define vocabulary
vocabulary = len(tokenizer.word_index)
print("number of unique words : ",vocabulary)
output_length = le.classes_.shape[0]
print("output length: ",output_length)

number of unique words :  538
output length:  27
```

Gambar 4. 7 Proses *Vocabulary*

Pada gambar 4.7 merupakan proses dari *vocabulary*, pada bagian ini menghitung jumlah kata unik dalam data yang telah melewati proses *tokenization* menggunakan *tokenizer.word_index*. kode tersebut merupakan kamus yang memetakan kata pada *indeks numerik* berdasarkan urutan frekuensi kata. Dari proses *vocabulary* tersebut menghasilkan 538 kata unik. Selain menghitung jumlah kata unik dalam kode tersebut juga mendefinisikan panjang *output* dari label-label yang sudah dirubah menjadi angka. Proses ini menggunakan *le.classes_.shape[0]* yang menghasilkan *output lenght* 27. Tujuan dari proses ini adalah untuk mengetahui jumlah dari proses *LabelEncoder* yang cocok untuk model klasifikasi *multi*-kelas.

Selanjutnya dilakukan proses pembagian data, proses ini bertujuan untuk membagi data menjadi data latihan dan data uji menggunakan fungsi *train test split* dari *library Scikit Learn*. Pada tahap ini membagi data menjadi dua *subset*, satu untuk melatih model dan yang lainnya untuk menguji kinerja model. Tujuan dari proses ini adalah untuk memvalidasi bagaimana model yang dilatih pada data latih dapat tampil pada data yang belum pernah dilihat sebelumnya agar dapat menghindari *overfitting*. Kode proses terlihat pada gambar 4.8.

```
[ ] x_train, x_test, y_train, y_test = train_test_split(x_train, y_train, test_size=0.2, random_state=42)
```

Gambar 4. 8 Proses *Train Test Split*

Pada gambar 4.8 merupakan proses pemisahan data yang mengambil beberapa argumen penting yaitu:

a. *x_train*: data masukan yang telah dihasilkan dari *tokenization* dan *padding* digunakan untuk melatih model

b. *y_train*: data keluaran atau label yang telah diubah menjadi angka melalui proses *encoding* sebagai target yang cocok dengan data latih.
c. *test size*: merupakan ukuran proporsi data yang akan digunakan sebagai data uji. Pada kode tersebut 0,2 berarti 20% dari total data yang akan digunakan sebagai data uji, sisanya 80% sebagai data latih.
d. *Random state*: merupakan biji acak untuk menghasilkan pemisahan data yang konsisten.

Setelah melakukan pemisahan, hasilnya akan dipecahkan menjadi empat variabel yaitu, *x_train, x_test, y_train,* dan *y_test*. Variabel *x_train* dan *y_train* akan berisi data untuk melatih model, sedangkan *x_test* dan *y_test* berisi data yang digunakan sebagai pengujian model.

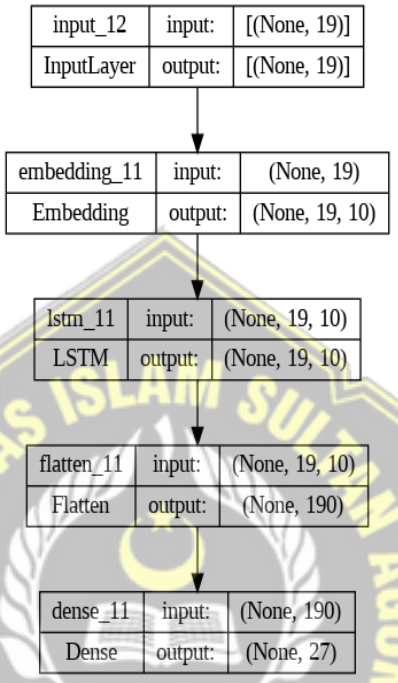
```
from nltk.util import flatten
#creating the model

i = Input(shape=(input_shape,))
x = Embedding(vocabulary+1,10)(i)
x = LSTM(10,return_sequences=True)(x)
x = Flatten()(x)
x = Dense(output_length,activation="softmax")(x)
model  = Model(i,x)
```

Gambar 4. 9 Proses Pembentukan Model

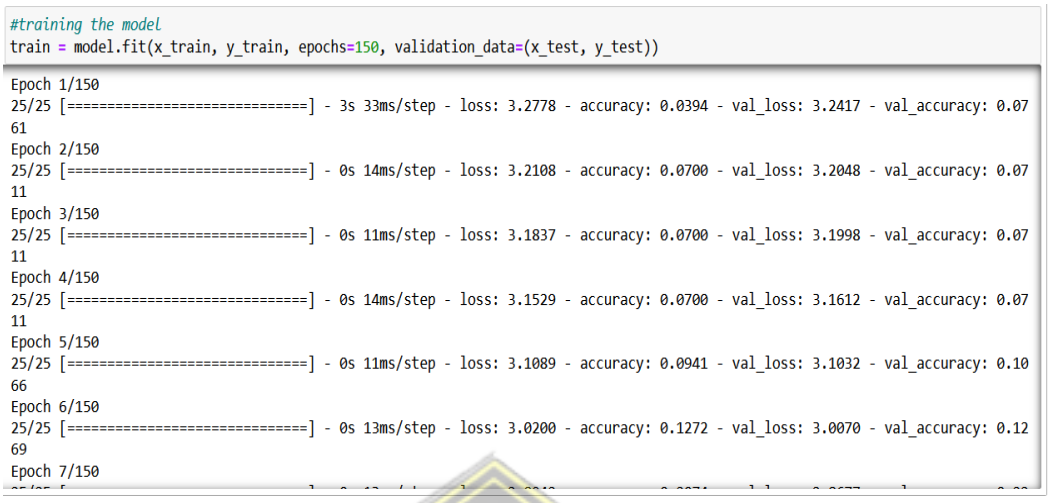
Pada gambar 4.9, merupakan struktur model yang dipakai pada penelitian ini, pada struktur tersebut memiliki beberapa lapisan yaitu lapisan pertama: *Input Layer* menggunakan *input shape* sebagai dimensi *input* yang akan diberikan ke dalam model sebesar 19. Lapisan kedua *Embedding Layer* mengkonversi kata-kata yang direpresentasikan dalam bentuk *indeks* (angka) menjadi vektor-vektor dengan dimensi lebih rendah. Parameter pertama (*vocabulary+1*) adalah jumlah kata unik dalam kamus yang telah di *indeks*. Parameter kedua (10) adalah dimensi vektor yang akan dihasilkan untuk setiap kata. Lapisan ketiga LSTM *Layer* dengan 10 unit. LSTM memahami hubungan antar kata dalam urutan (*sequence*) *input*. Parameter *return sequences = True* mengindikasikan bahwa LSTM mengembalikan *output* untuk setiap *timestep* dalam *input sequence*, yang akan digunakan oleh lapisan berikutnya. Lapisan keempat *Flatten Layer* meratakan (*flatten*) hasilnya menjadi bentuk satu dimensi. Karena lapisan *Dense* yang akan mengikuti memerlukan *input*

dalam bentuk satu dimensi. Lapisan kelima *Dense Layer* memiliki *output-length unit*. Parameter *activation = soft max*, yang menghasilkan probabilitas *output* untuk setiap kelas. Berikut hasil dari proses pembentukan model yang dapat dilihat pada gambar 4.10.



Gambar 4. 10 Struktur Model

Pada gambar 4.10 merupakan hasil dari proses struktur model, dari hasil ini dapat disimpulkan bahwa lapisan pertama yaitu *Input Layer* mengindikasi bahwa ukuran jumlah sampel dalam setiap *batch* belum ditentukan dan 19 adalah panjang urutan data. Selanjutnya lapisan kedua yaitu *Embedding Layer* yang menunjukan bentuk keluaran dari lapisan ini. Dimana masing-masing kata dalam urutan telah di ubah menjadi vektor *embedding* dengan panjang 10. Lapisan ketiga yaitu LSTM *Layer* yang menunjukan bahwa pada model yang rancang memiliki *output* sepanjang 19 urutan dengan 10 fitur pada setiap urutan. Selanjutnya lapisan keempat yaitu *Flatten Layer* yang menunjukan bahwa pada sistem hasil model yang dirancang memiliki vektor dengan panjang 190 yang dihasilkan oleh *flatten layer*, dan lapisan yang terakhir adalah *Dense Layer* menunjukan bahwa *output* yang dihasilkan memiliki panjang 27, sesuai dengan kategori yang dimiliki.



```
#training the model
train = model.fit(x_train, y_train, epochs=150, validation_data=(x_test, y_test))
```

```
Epoch 1/150
25/25 [==============================] - 3s 33ms/step - loss: 3.2778 - accuracy: 0.0394 - val_loss: 3.2417 - val_accuracy: 0.07
61
Epoch 2/150
25/25 [==============================] - 0s 14ms/step - loss: 3.2108 - accuracy: 0.0700 - val_loss: 3.2048 - val_accuracy: 0.07
11
Epoch 3/150
25/25 [==============================] - 0s 11ms/step - loss: 3.1837 - accuracy: 0.0700 - val_loss: 3.1998 - val_accuracy: 0.07
11
Epoch 4/150
25/25 [==============================] - 0s 14ms/step - loss: 3.1529 - accuracy: 0.0700 - val_loss: 3.1612 - val_accuracy: 0.07
11
Epoch 5/150
25/25 [==============================] - 0s 11ms/step - loss: 3.1089 - accuracy: 0.0941 - val_loss: 3.1032 - val_accuracy: 0.10
66
Epoch 6/150
25/25 [==============================] - 0s 13ms/step - loss: 3.0200 - accuracy: 0.1272 - val_loss: 3.0070 - val_accuracy: 0.12
69
Epoch 7/150
```

Gambar 4. 11 *Training* Model

Pada gambar 4.11 merupakan proses dari *training* model, pada proses ini data yang digunakan dalam proses *training* model adalah data latih dan data uji yang sudah diproses sebelumnya. Pada tahap ini menggunakan metode *".fit()"* untuk memulai pelatihan, dalam proses ini terdapat argumen penting yaitu:

a. *x_train*: data masukan atau fitur yang akan digunakan untuk melatih model.
b. *y_train*: data keluaran atau label yang sesuai dengan data masukan.
c. *epochs*: merupakan jumlah kali iterasi atau pengulangan data latih selama pelatihan. Pada proses ini, *epoch* yang digunakan sebesar 150.
d. *validation_data*: merupakan data uji yang digunakan untuk memvalidasi kinerja model selama pelatihan. Pada proses ini, data *x_test* dan *y_test* digunakan sebagai data validasi.

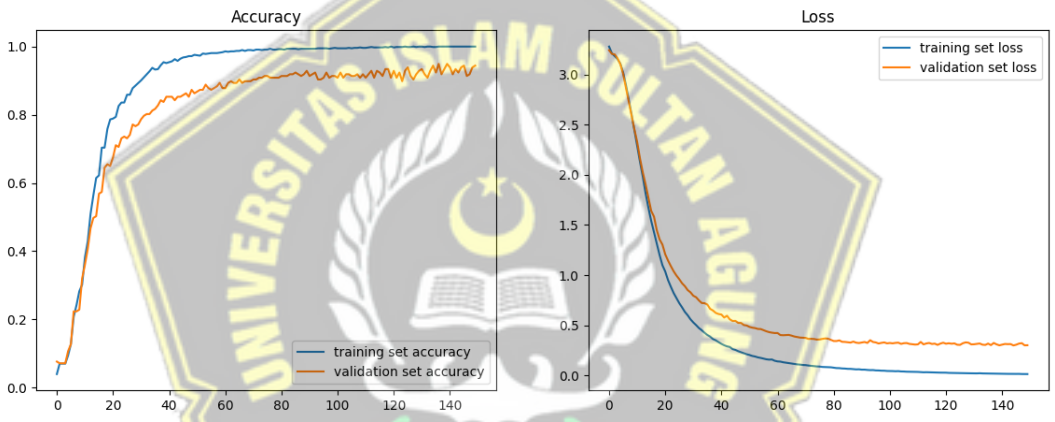
Selama proses *training* model, model akan mengiterasi melalui data latih dalam jumlah yang sudah ditentukan. Pada setiap *epoch*, model akan menghitung *loss* berdasarkan data latih dan juga validasi *loss* berdasarkan data uji. Setelah pelatihan selesai, metode *".fit()"* akan mengembalikan objek *History* yang berisi informasi tentang kinerja model selama pelatihan.

Untuk mengetahui sebuah model bekerja dengan baik atau tidak, perlu dilakukan evaluasi pada model tersebut. Proses evaluasi model menggunakan *confusion matrix* untuk mendapatkan nilai *accuracy*, *precision*, dan *recall*.

Table 4. 1 Konfigurasi

Epoch	*Optimizer*	*Learning Rate*
150	*Adam*	0,001

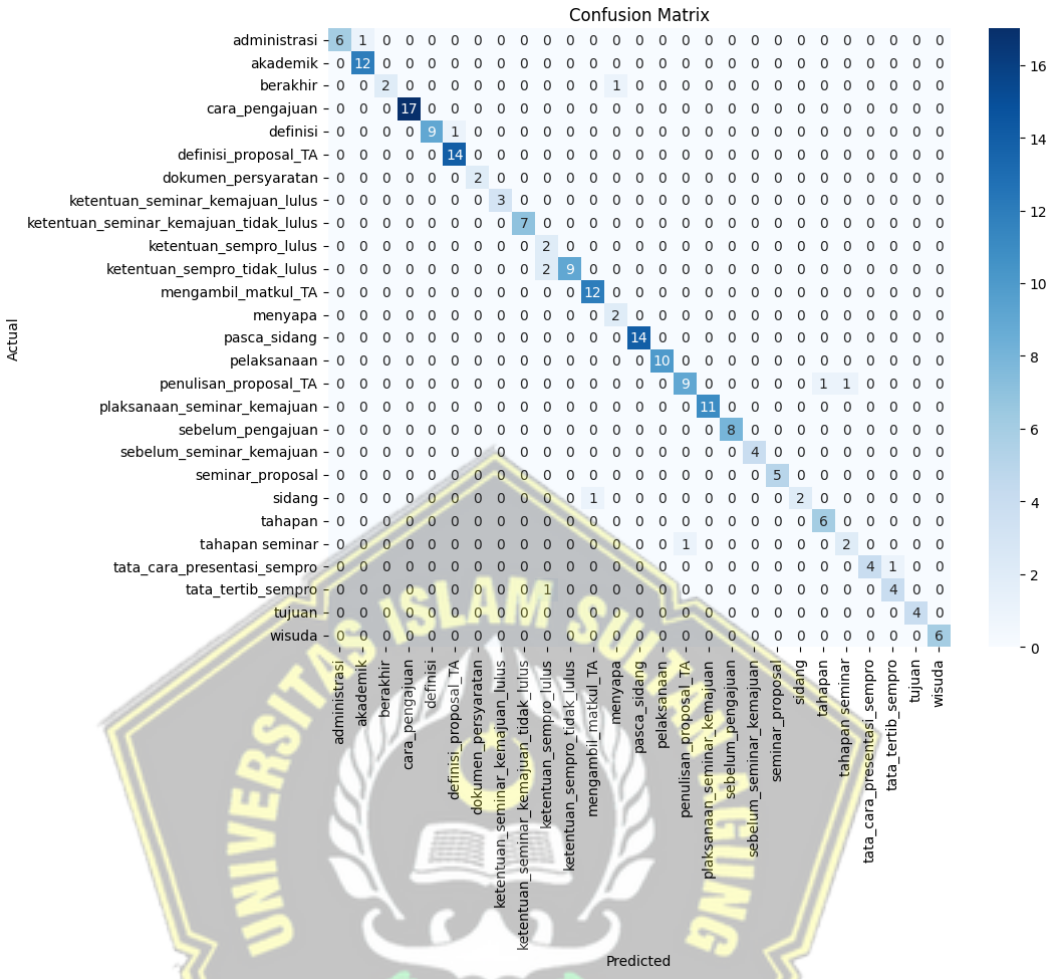
Table 4.1 merupakan komposisi percobaan pada konfigurasi dengan *epoch* sebesar 150 dan menggunakan *optimizer adam*. Dengan nilai *learning rate* sebesar 0,001. Pada pelatihan model konfigurasi mendapatkan *accuracy* sebesar 1, *loss* sebesar 0,0134, validasi *accuracy* sebesar 0,94, dan validasi *loss* sebesar 0,30. Grafik hasil *accuracy* dan *loss* serta *validasi accuracy* dan *validasi loss* pada setiap *epoch* dapat dilihat pada gambar 4.12.



Gambar 4. 12 Grafik *Training* Model

Pada gambar 4.12 merupakan hasil dari proses *training* model yang digambarkan dalam bentuk grafik. Dari grafik *accuracy* tersebut dapat dilihat bahwa pada garis berwarna biru menujukan *accuracy* terlihat bagus, sedangkan untuk garis berwarna *orange* merupakan *validasi accuracy* yang menunjukan sedikit dibawah garis *accuracy*. Sedangkan untuk grafik *loss*, garis biru merupakan *loss* yang menujukan hasil yang baik dan untuk garis dengan warna *orange* merupakan *validasi loss* yang menunjukan sedikit diatas garis *loss*. Dapat disimpulkan dari hasil grafik tersebut, proses *training* model mendapatkan hasil yang bagus dan tidak *overfitting*.

Selanjutnya adalah tahap pengukuran yang menggunakan metode *confusion matrix* yang dapat dilihat pada gambar 4.13.



Gambar 4. 13 *Confusion Matrix* Konfigurasi 1

Berdasarkan pada gambar 4.13, merupakan visualisasi nilai matriks yang akan digunakan untuk menghitung *accuracy*, *precision*, dan *recall*. Pada gambar tersebut terdapat keterangan *Predicted* sebagai label pada sumbu x dan *Actual* sebagai label pada sumbu y. Karena data yang ditampilkan sangat banyak, maka proses penghitungan dilakukan menggunakan *library scikit learn* seperti pada Gambar 4.14.

```
[321] # Menghitung dan mencetak akurasi, presisi, dan recall
      akurasi = accuracy_score(y_test, y_pred)
      presisi = precision_score(y_test, y_pred, average='weighted', zero_division=1)
      recall = recall_score(y_test, y_pred, average='weighted', zero_division=1)

      print(f"Akurasi pada data uji: {akurasi}")
      print(f"Presisi pada data uji: {presisi}")
      print(f"Recall pada data uji: {recall}")

      Akurasi pada data uji: 0.9441624365482234
      Presisi pada data uji: 0.9563284431304737
      Recall pada data uji: 0.9441624365482234
```

Gambar 4. 14 Penghitungan *Confusion Matrix*

Pada gambar 4.14 merupakan proses evaluasi kinerja model seperti *accuracy*, *precision*, dan *recall* pada data uji. Pada proses ini menghitung metrik evaluasi menggunakan fungsi *accuracy_score*, *precision_score*, dan *recall_score*. Setiap proses penghitungan akan dijelaskan sebagai berikut:

a. *accuracy_score*: untuk menentukan nilai akurasi dengan cara, *y_test* adalah label yang sebenarnya dari data uji, dan *y_pred* adalah prediksi yang dihasilkan oleh model pada data uji.
b. *precision_score*: untuk menentukan nilai presisi dengan cara menghitung presisi rata-rata berbobot berdasarkan kelas, dan *zero_devision*=1 bertujuan untuk menghindari pesan *error* jika presisi dihitung dengan pembagi nol.
c. *recall_score*: untuk menentukan nilai *recall* dengan cara menghitung *recall* rata-rata berbobot berdasarkan kelas, dan *zero_devision*=1 bertujuan untuk menghindari pesan *error* jika presisi dihitung dengan pembagi nol.

Dari penghitungan matriks evaluasi, menujukan hasil perhitungan yang dituliskan pada table 4.2.

Table 4. 2 Hasil *Confusion Matrix* Konfigurasi

Konfigurasi	*Accuracy*	*Precision*	*Recall*
Konfigurasi 1	0.94	0.95	0.94

Pada tabel 4.2 merupakan hasil dari evaluasi model dengan menggunakan metode *confusion matrix*, maka dapat disimpulkan bahwa model memiliki akurasi yang tinggi, yang mengindikasikan kinerja keseluruhan yang baik. Selain itu, nilai *precision* yang tinggi menunjukan bahwa model memiliki kemampuan yang baik untuk mengidentifikasi data positif dengan benar. Selanjutnya nilai *recall* yang

sama dengan nilai akurasi menandakan bahwa tidak ada prediksi kelas positif yang terabaikan.

4.2 Hasil Implementasi Antarmuka

Setelah tahap pemodelan selesai, selanjutnya tahap implementasi sistem pada *platform website*. Antarmuka yang dikembangkan merujuk pada desain *wireframe* sistem pada bab III.

4.2.1 Halaman Awal *chatbot*

Implementasi dari desain antarmuka Halaman Awal *Chatbot*. Pada bagian *appbar* terdapat *title* dari aplikasi. Bagian *body* terdapat halaman percakapan, kemudian terdapat *input text* untuk memberikan pertanyaan dan terdapat *icon* pesawat sebagai tombol kirim. Halaman Awal *Chatbot* dapat dilihat pada gambar 4.15.

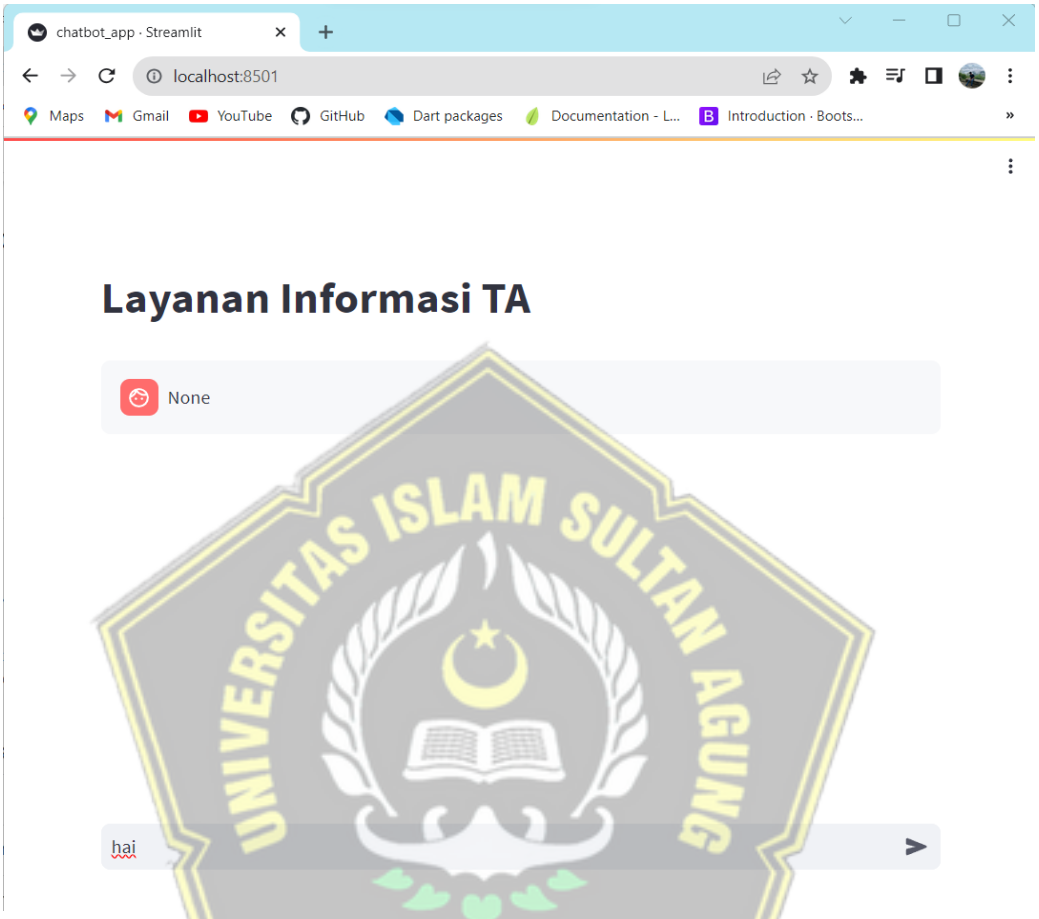
Gambar 4. 15 Halaman Awal

Pada gambar 4.15 merupakan tampilan awal yang akan ditampilkan pengguna saat pertama kali masuk. Pada tampilan awal ini pengguna dapat memberikan pertanyaan pada kotak *input text*.

4.2.2 Halaman *Chatbot*

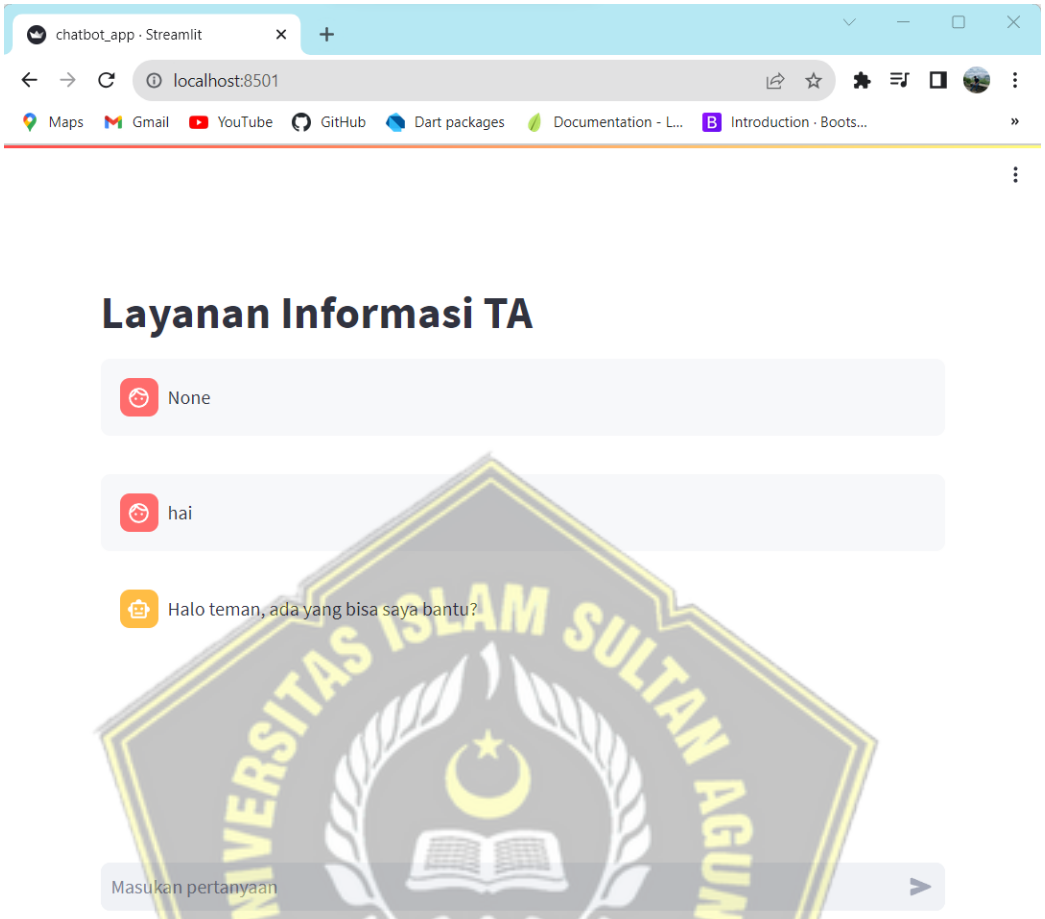
Implementasi dari desain antarmuka Halaman *Chatbot*. Tampilan menu dibuat sama dengan halaman awal *chatbot*. Pada halaman *chatbot* ini bertujuan

untuk lebih interaktif, berikut demo aplikasi dengan mengajukan beberapa pertanya terlihat pada gambar 4.16.



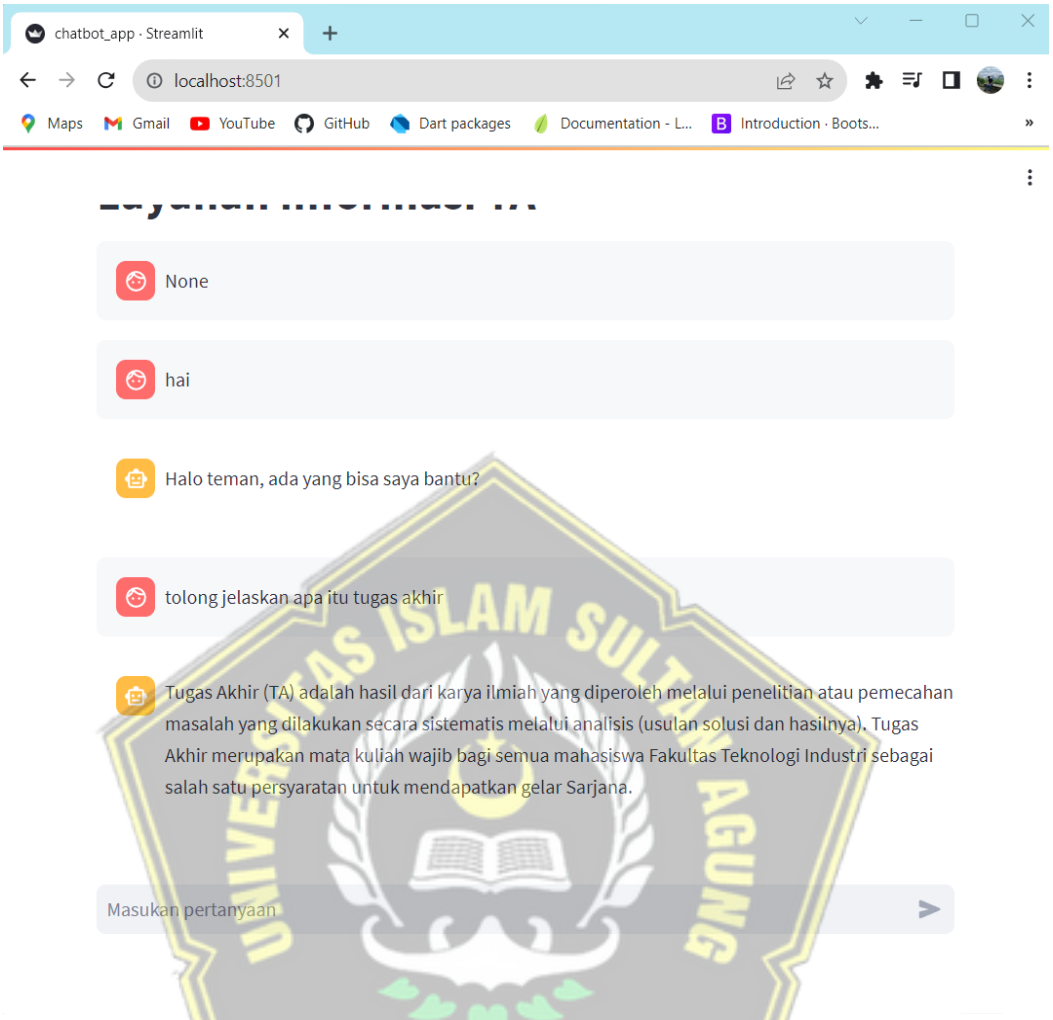
Gambar 4. 16 Input pertanyaan

Pada gambar 4. 16 merupakan demo aplikasi yang diawali dengan menyapa sistem *chatbot* dengan cara menginputkan pertanyaan "hai", dari proses tersebut sistem *chatbot* memberikan *responses* yang terlihat pada gambar 4.17.



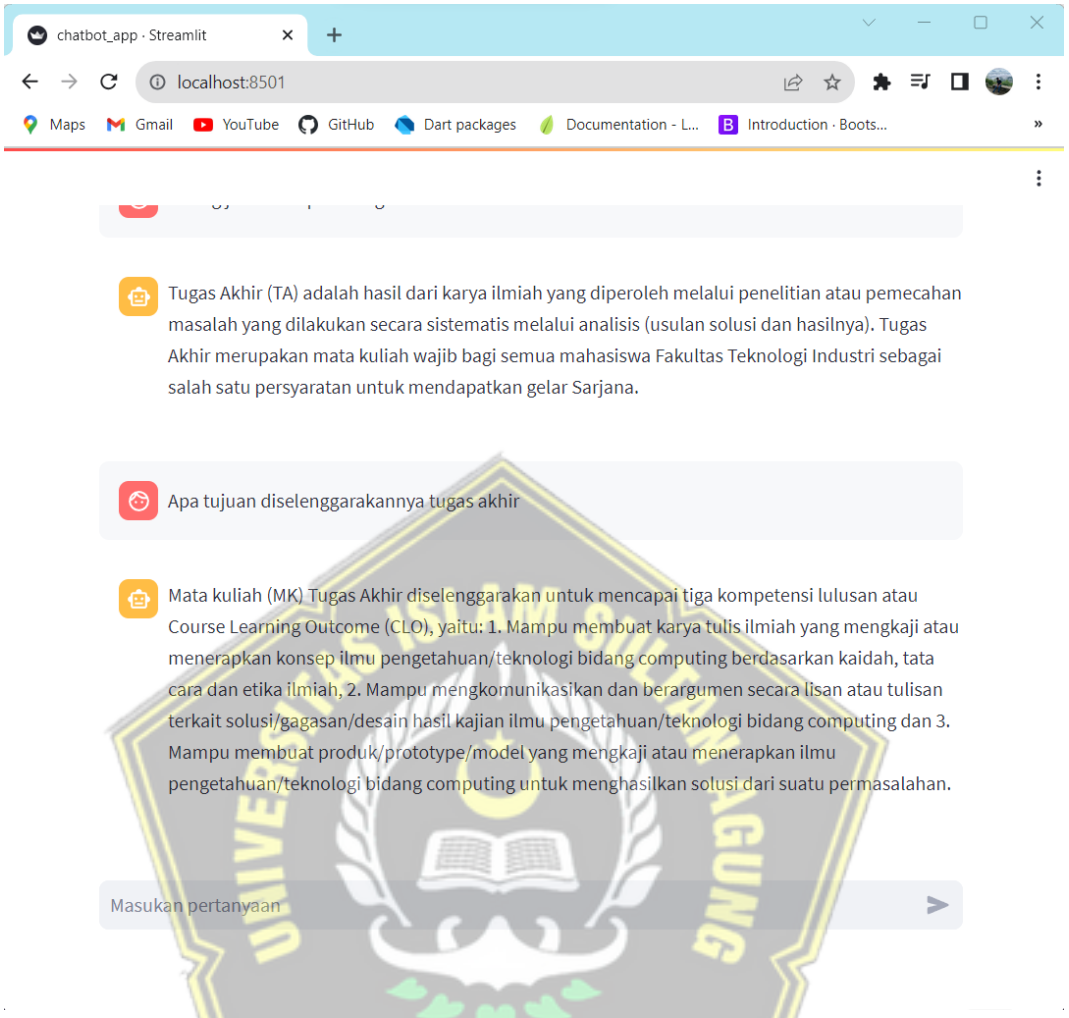
Gambar 4. 17 Demo Aplikasi *Chatbot* ke-1

Terlihat pada gambar 4.17, *chatbot* mampu memberikan *responses* yang baik dan akurat sesuai dengan pertanyaan yang diberikan. Berikutnya mencoba menginputkan pertanyaan seputar tugas akhir, demo aplikasi tersebut seperti pada gambar 4.18.



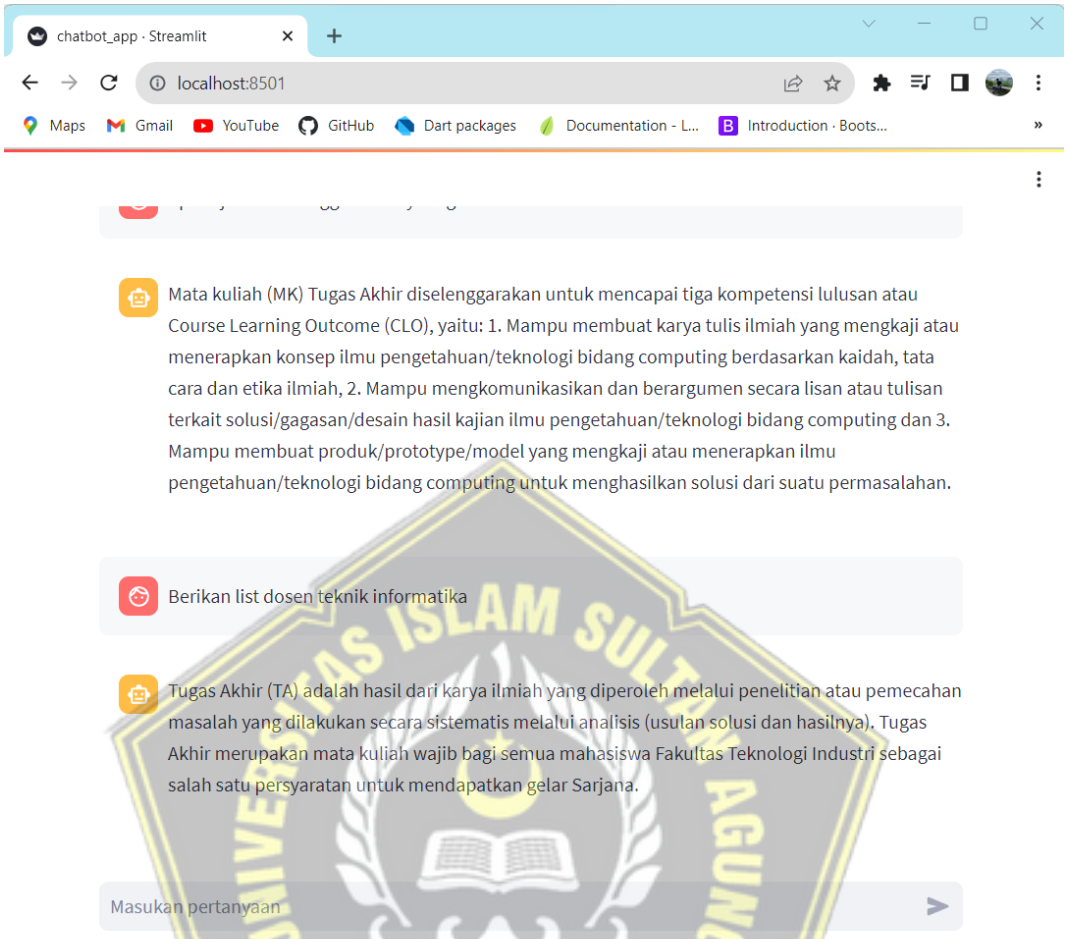
Gambar 4. 18 Demo Aplikasi *Chatbot* ke-2

Terlihat pada gambar 4.18, menampilkan hasil dari pertanyaan seputar tugas akhir yang diberikan. Dari *responses* yang diberikan oleh *chatbot* menandakan bahwasanya *chatbot* bekerja dengan baik dan memberikan jawaban yang akurat. Berikutnya mencoba memberikan pertanyaan lagi seputar tugas akhir yang terlihat pada gambar 4.19.



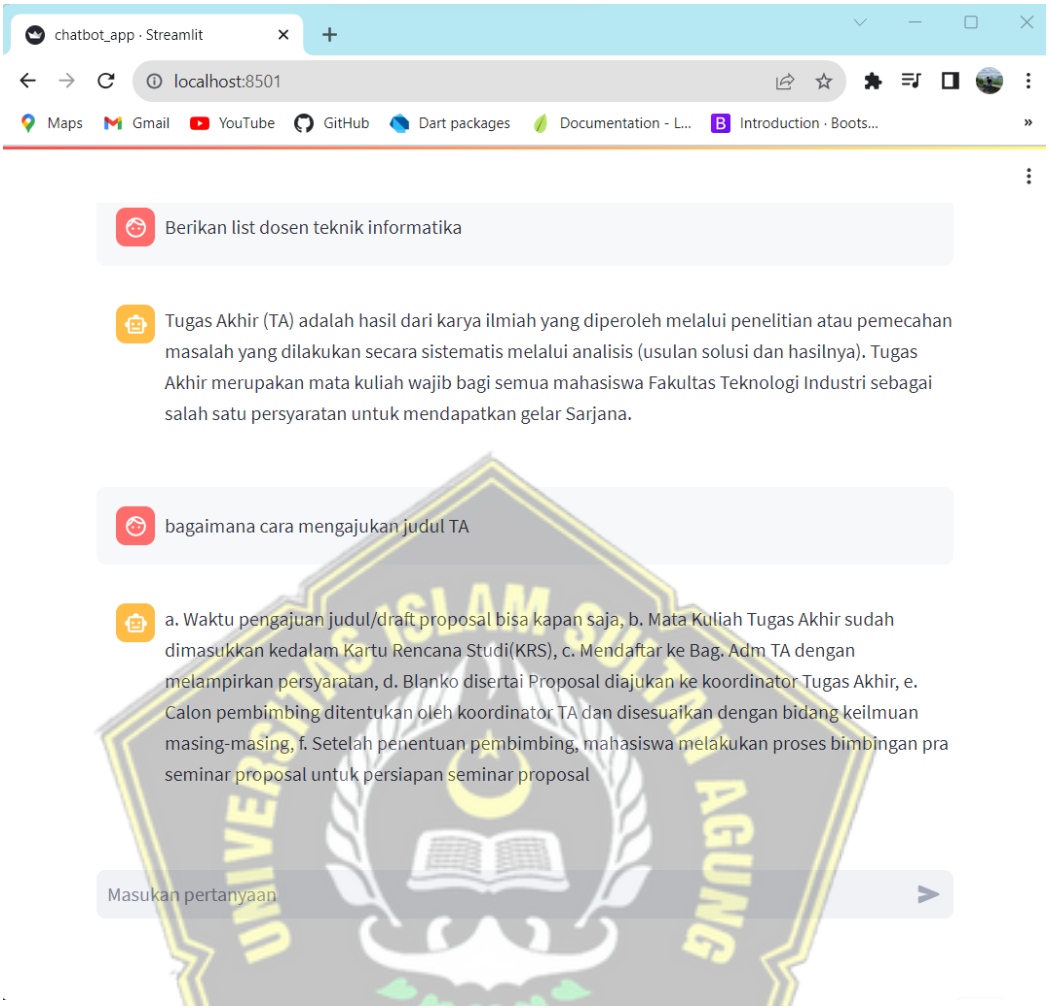
Gambar 4. 19 Demo Aplikasi *Chatbot* ke-3

Terlihat pada gambar 4.19, ternyata *chatbot* masih memberikan *responses* yang baik dan jawaban sesuai dengan pertanyaan yang diberikan. Selanjutnya mencoba memberikan pertanyaan pada sistem *chatbot* tetapi topiknya diluar dari tugas akhir, seperti pada gambar 4.20.



Gambar 4. 20 Demo Aplikasi *Chatbot* ke-4

Terlihat pada gambar 4.20, *chatbot* masih memberikan *responses* berupa jawaban akan tetapi jawaban yang diberikan tidak sesuai dengan pertanyaan. Itu dikarenakan pertanyaan yang diberikan pada sistem *chatbot* tidak terkait dengan tugas akhir. Demo Aplikasi terakhir dengan memberikan pertanyaan pada *chatbot* seputar tugas akhir, seperti pada gambar 4.21.



Gambar 4. 21 Demo Aplikasi *Chatbot* ke-5

Terlihat pada gambar 4.21, *chatbot* masih dapat memberikan jawaban yang sesuai dengan apa yang ditanyakan. Maka dari beberapa percobaan tersebut dapat disimpulkan bawah sistem *chatbot* dapat bekerja dengan baik dan mampu memberikan *responses* yang akurat apabila pertanyaan yang diberikan masih berkaitan dengan Tugas Akhir.

4.3 Hasil Pengujian Sistem

Pada tahap pengujian sistem *chatbot* layanan informasi tugas akhir dilakukan pengujian *black box*. Pengujian *black-box* adalah untuk meng-*input* dan menguji apakah fungsi-fungsi yang terdapat pada sistem yang sedang berjalan sudah sesuai dengan tujuan, dan tercermin dalam hasil keluaran. Adapun rencana yang akan dibuat yaitu *input*, hasil yang diharapkan, *output* dan kesimpulan, yang akan diuji

adalah *text input*, dan hasil percakapan yang terlihat pada *table* 4.3 Merupakan hasil pengujian dari *black box*.

Table 4. 3 Pengujian *Black Box*

Input	**Hasil yang diharapkan**	***Output***	**Kesimpulan**
Masukan pertanyaan "hai"	Menampilkan input pertanyaan dan jawaban di bagian *body*	Begitu mengirim pertanyaan keluar jawaban	*Responses* Akurat
Masukan pertanyaan "Tolong jelaskan apa itu tugas akhir"	Menampilkan input pertanyaan dan jawaban di bagian *body*	Begitu mengirim pertanyaan keluar jawaban	*Responses* Akurat
Masukan pertanyaan "Apa tujuannya diselenggarakannya tugas akhir"	Menampilkan input pertanyaan dan jawaban di bagian *body*	Begitu mengirim pertanyaan keluar jawaban	*Responses* Akurat
Masukan pertanyaan "berikan list dosen teknik informatika"	Menampilkan input pertanyaan dan jawaban di bagian *body*	Begitu mengirim pertanyaan keluar jawaban	*Responses* Tidak Akurat
Masukan pertanyaan "bagaimana cara mengajukan proposal ta"	Menampilkan input pertanyaan dan jawaban di bagian *body*	Begitu mengirim pertanyaan keluar jawaban	*Responses* Akurat

Pada tabel 4.3, merupakan hasil pengujian *black box* yang telah dilakukan, empat pertanyaan yang diberikan dapat terjawab dengan baik dan memberika jawab yang akurat, akan tetapi salah satu pertanyaan yang diberikan menghasilkan *responses* atau jawaban yang tidak akurat. Dikarenakan pertanyaan yang diberikan tidak terdapat dalam data model yang sudah dilatih. Maka dari itu dapat disimpulkan *Chatbot* Layanan Informasi Tugas Akhir akan memberikan *responses* yang akurat apabila pertanyaan yang diberikan masih terkait dengan Tugas Akhir.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil dari penelitian ini, dapat disimpulkan tentang sistem *chatbot* layanan informasi menggunakan metode *Long Short-Term Memory* (LSTM) dapat efektif dalam memberikan informasi seputar tugas akhir. Hasil pengujian mengonfirmasi kemampuan model LSTM dengan tingkat akurasi 94.41%, presisi 95.63%, dan *recall* 94.41%. Dari hasil kemiripan pertanyaan dengan data pertanyaan yang dilatih dapat dijadikan parameter yang cocok dengan jawaban yang tersedia. semakin banyak data pertanyaan yang dilatih maka semakin tinggi juga hasil perhitungan akurasi serta jawaban yang diberikan.

5.2 Saran

Saran yang dapat diterapkan untuk pengembangan sistem lebih lanjut di masa mendatang yaitu: Pada sistem *chatbot* layanan informasi tugas akhir menggunakan metode *long short-term memory* adalah dataset yang digunakan sebagai bahan latih pada model nantinya dibuat secara otomatis dengan cara meng-*generate* dokumen maupun data lainnya yang akan dijadikan sebagai jawaban. Dari hasil *generate* itu sendiri juga dapat otomatis menghasilkan pertanyaan yang kemungkinan diajukan oleh pengguna.

DAFTAR PUSTAKA

Accenture. (2018). *Chatbots Are Here To Stay*. Retrieved Juni 8, 2023 from accenture: https://www.accenture.com/_acnmedia/pdf-77/accenture-research-conversational-ai-platforms.pdf

Amalia, H. A. (2021, Oktober 1). *Survei Pancake: Baru 15% Pelaku UMKM Indonesia Pakai Chatbot*. Dipetik Juni 23, 2023, dari beritasatu: https://www.beritasatu.com/digital/835281/survei-pancake-baru-15-pelaku-umkm-indonesia-pakai-chatbot

Gunawan, A., & Nurfita, R. D. (2018). Implementasi Deep Learning Berbasis Tensorflow Untuk Pengenalan Sidik Jari. *Jurnal Teknik Elektro*, 18.

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

Nugroho, K. S. (2019, November 13). *Confusion Matrix untuk Evaluasi Model pada Supervised Learning*. Retrieved Agustus 2, 2023 from medium: https://ksnugroho.medium.com/confusion-matrix-untuk-evaluasi-model-pada-unsupervised-machine-learning-bc4b1ae9ae3f

Stauffer, R. (2021, April 20). *Understanding & Implementing Chatbots at Your Agency*. Retrieved Mei 25, 2023 from hawksoft: https://blog.hawksoft.com/chatbots

Trivusi. (2022, September 17). *Mengenal Algoritma Long Short Term Memory (LSTM)*. Retrieved Juli 13, 2023 from trivusi: https://www.trivusi.web.id/2022/07/algoritma-lstm.html

UII, J. I. (2022). *MEMBUAT APLIKASI WEB SAINS DATA DENGAN MUDAH MENGGUNAKAN STREAMLIT*. Retrieved Agustus 7, 2023 from informatics: https://informatics.uii.ac.id/2021/03/15/streamlit-membuat-aplikasi-web-sains-data/#:~:text=Streamlit%20adalah%20sebuah%20framework%20berbasis,dan%20machine%20learning%20yang%20interaktif%20.

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

Guntoro, G., Loneli Costaner, & Lisnawita, L. (2020). Aplikasi Chatbot untuk Layanan Informasi dan Akademik Kampus Berbasis Artificial Intelligence Markup Language (AIML). *Digital Zone: Jurnal Teknologi Informasi dan Komunikasi*, *11*(2), 291–300. https://doi.org/10.31849/digitalzone.v11i2.5049

Hikmah, A., Azmi, F., & Nugrahaeni, R. A. (2023). Implementasi Natural Language Processing Pada Chatbot Untuk Layanan Akademik. *eProceedings of Engineering*, *10*(1).

Hikmah, N., Ariyanti, D., & Pratama, F. A. (2022). Implementasi Chatbot Sebagai Virtual Assistant di Universitas Panca Marga Probolinggo menggunakan Metode TF-IDF. *JTIM: Jurnal Teknologi Informasi dan Multimedia*, *4*(2), 133–148.

Hormansyah, D. S., & Utama, Y. P. (2018). APLIKASI CHATBOT BERBASIS WEB PADA SISTEM INFORMASI LAYANAN PUBLIK KESEHATAN DI MALANG DENGAN MENGGUNAKAN METODE TF-IDF. *Jurnal Informatika Polinema*, *4*(3), 224. https://doi.org/10.33795/jip.v4i3.211

Muhammad, R., Ardimansyah, M. I., & Afini, A. (2023). *MENGEMBANGKAN SISTEM PERCAKAPAN OTOMATIS BERBASIS LAYANAN PESAN INSTAN*. Uwais Inspirasi Indonesia.

Mulyatun, S., Utama, H., & Mustopa, A. (2021). PENDEKATAN NATURAL LANGUAGE PROCESSING PADA APLIKASI CHATBOT SEBAGAI ALAT BANTU CUSTOMER SERVICE. *Journal of Information System Management (JOISM)*, *3*(1), 12–17. https://doi.org/10.24076/JOISM.2021v3i1.404

Renaningtias, N., & Apriliani, D. (2021). Penerapan Metode Prototype Pada Pengembangan Sistem Informasi Tugas Akhir Mahasiswa. *Rekursif: Jurnal Informatika*, *9*(1). https://doi.org/10.33369/rekursif.v9i1.15772

Sembilu, N., Samopa, F., & Er, M. (2018). Perbandingan Algoritma Kemiripan Teks untuk Perbaikan dan Saran Penulisan Frasa Dalam Bahasa ALami. *SISFO*, *8*(1), 19–32.

Wintoro, P. B., Hermawan, H., Muda, M. A., & Mulyani, Y. (2022). Implementasi Long Short-Term Memory pada Chatbot Informasi Akademik Teknik Informatika Unila. *EXPERT: Jurnal Manajemen Sistem Informasi dan Teknologi*, *12*(1), 68. https://doi.org/10.36448/expert.v12i1.2593