PERANCANGAN *CHATBOT* BERBASIS *ARTIFICIAL NEURAL NETWORK* UNTUK MENINGKATKAN EFISIENSI LAYANAN INFORMASI PPDB (Studi Kasus: SMK KESATRIAN PURWOKERTO)

Skripsi

Disusun oleh

Hanan Abdul Ghani

21SA1035

PROGRAM STUDI INFORMATIKA

FAKULTAS ILMU KOMPUTER

UNIVERSITAS AMIKOM PURWOKERTO

2025

BAB I

PENDAHULUAN

A. Latar Belakang Masalah

Dalam era digital saat ini, institusi pendidikan dituntut untuk memberikan layanan informasi yang cepat dan akurat, terutama terkait proses Penerimaan Peserta Didik Baru (PPDB). Informasi mengenai persyaratan administrasi, jadwal pendaftaran, biaya pendidikan, serta pengumuman penting lainnya perlu disajikan secara jelas dan mudah diakses oleh calon siswa, orang tua, dan staf sekolah (Yudahana dkk., 2023). Namun seringkali akses terhadap informasi ini masih menghadapi berbagai kendala, seperti keterbatasan waktu operasional kantor, ketersediaan staf untuk menjawab pertanyaan (Ivan dkk., 2022). Hal ini menimbulkan keterlambatan dalam penyampaian informasi yang pada akhirnya mempengaruhi kenyamanan dan efisiensi dalam proses belajar mengajar.

SMK Kesatrian Purwokerto, yang berdiri sejak tahun 1996, merupakan Sekolah Menengah Kejuruan di bawah naungan Yayasan Perguruan Islam Republik Indonesia yang berlokasi di Desa Sokanegara, Kecamatan Purwokerto, Kabupaten Banyumas. Saat ini, sekolah ini memiliki sebanyak 2.025 siswa. SMK Kesatrian Purwokerto menawarkan berbagai jurusan, antara lain Teknik Kendaraan Ringan Otomotif, Teknik dan Bisnis Sepeda Motor, Teknik Audio Video, Teknik Komputer, dan Desain Komunikasi Visual (DKV). Untuk menunjang minat dan bakat siswa, sekolah menyediakan berbagai fasilitas seperti bengkel berstandar industri,

laboratorium komputer, dan lapangan olahraga. SMK Kesatrian Purwokerto juga memiliki program unggulan berupa kelas industri di setiap jurusannya. Program ini terdiri dari siswa-siswi terpilih yang diarahkan untuk mengikuti magang di perusahaan mitra sekolah. Program ini dirancang sebagai langkah awal bagi siswa yang ingin langsung bekerja setelah lulus, memberikan prospek karier yang lebih jelas dan kesiapan menghadapi dunia kerja. Sebagai sekolah yang berkomitmen untuk memberikan pendidikan berkualitas, SMK Kesatrian Purwokerto senantiasa berupaya meningkatkan layanannya. Salah satu aspek penting yang perlu diperhatikan adalah penyediaan informasi yang akurat, mudah diakses, dan *up-to-date* bagi calon siswa.

Berdasarkan wawancara dengan Bapak Agung Sulistiono, S.T., selaku staf IT dan admin di SMK Kesatrian Purwokerto, Beliau menyampaikan bahwa informasi PPDB dapat diperoleh melalui laman resmi PPDB sekolah, kegiatan promosi ke SMP di sekitar, atau dengan mengunjungi sekolah secara langsung. Meskipun sekolah telah menyediakan platform *WhatsApp* sebagai media alternatif untuk memperoleh informasi, beliau mengatakan pada platform *WhatsApp* ini memiliki kendala seperti calon siswa yang kerap mengirimkan pertanyaan di luar jam kerja sehingga admin akan membalas pesan tersebut di keesokan harinya. Hal ini dapat memakan waktu dalam merespon pertanyaan terutama saat volume pertanyaan sedang tinggi.

Permasalahan serupa pernah menjadi topik penelitian sebelumnya, seperti penelitian oleh Nugraha & Sebastian (2021) meneliti keterbatasan layanan *customer service* otomatis, mengembangkan *chatbot* layanan

akademik berbasis *K-Nearest Neighbor* (KNN) dengan akurasi 53,48% (K=3) dari 86 pertanyaan. Mustakim & Hayati (2021) menangani pertanyaan berulang dengan chatbot berbasis *Artificial Neural Network* (ANN), mencapai akurasi 97,27% dari 110 percakapan. Hikmah dkk. (2023) meningkatkan efisiensi pelayanan informasi akademik di Telkom *University*, menggunakan ANN dengan akurasi 100% untuk 54 pertanyaan acak dan tingkat kepuasan pengguna 93%. Penelitian-penelitian tersebut menunjukkan ANN sangat cocok untuk *chatbot* layanan akademik karena keunggulan *machine learning* yang memungkinkan sistem belajar dan berkembang dari data serta pengalaman.

Machine learning adalah cabang dari kecerdasan buatan (AI) yang berfokus pada pengembangan algoritma dan teknik yang memungkinkan komputer untuk belajar dari data dan pengalaman tanpa perlu diprogram secara eksplisit. Konsep dasar dari machine learning adalah bahwa sistem dapat meningkatkan kinerjanya dalam menyelesaikan tugas tertentu seiring dengan bertambahnya data dan pengalaman yang diperoleh (Ling, 2023).

Kemampuan *machine learning* dalam belajar dari data semakin diperkuat oleh munculnya *Artificial Neural Network* (ANN). *Artificial Neural Network* (ANN) adalah model komputasi yang dirancang untuk meniru cara kerja otak manusia dalam mengolah informasi. ANN terdiri dari kumpulan "neuron" atau elemen komputasi sederhana yang saling terhubung untuk membentuk sistem yang mampu mempelajari pola, mengklasifikasikan, dan memprediksi data. ANN menggunakan bobot koneksi antar neuron untuk

menyimpan informasi yang diperoleh dari proses pembelajaran. Dengan kemampuan ini, ANN sangat berguna dalam penambangan data, terutama karena ketahanannya dalam mengelola data yang mengandung noise atau ketidakpastian (Purwono dkk., 2022).

Chatbot merupakan produk hasil keluaran dari *machine learning*. *Chatbot* adalah program komputer yang menyimulasikan percakapan manusia dengan pengguna akhir. *Chatbot* adalah program komputer yang mensimulasikan percakapan manusia dalam format yang alami, baik dalam bentuk teks maupun suara, dengan memanfaatkan teknik kecerdasan buatan seperti *Natural Language Processing* (NLP), pemrosesan gambar dan video, serta analisis audio (Zuraiyah dkk., 2019). Salah satu perkembangan teknologi yang mendukung kemampuan dalam mengolah data teks adalah *text mining*, yang memungkinkan *chatbot* menganalisis dan memahami input dari pengguna dengan lebih efektif (Nurul Puteri dkk., 2022).

Secara keseluruhan, penulis bermaksud untuk membuat *chatbot* berbasis *machine learning* untuk *website* PPDB SMK Kesatrian Purwokerto menggunakan Algoritma *Artificial Neural Network* (ANN) guna membantu pengguna dalam mendapatkan informasi terkait PPDB.

B. Rumusan Masalah

Bagaimana merancang *chatbot* berbasis *machine learning* pada *website* PPDB sekolah menggunakan algoritma *Artificial Neural Network* guna membantu calon siswa dalam mendapatkan informasi terkait PPDB.

C. Batasan Masalah

Berdasarkan latar belakang masalah, penelitian ini dibatasi oleh hal-hal sebagai berikut :

1. ini dirancang hanya untuk menjawab pertanyaan terkait informasi PPDB sekolah atau informasi pendaftaran.
2. hanya akan mendukung bahasa indonesia.
3. Pengembangan *chatbot* pada penelitian ini dilakukan dengan bahasa pemrograman Python.

D. Tujuan Penelitian

Berdasarkan latar belakang dan rumusan masalah. Penelitian ini bertujuan untuk Merancang *chatbot* berbasis *machine learning* pada *website* PPDB sekolah menggunakan algoritma *Artificial Neural Network* guna membantu calon siswa dalam mendapatkan informasi terkait PPDB.

E. Manfaat Penelitian

1. Manfaat Teoritik
    a. Memberikan kontribusi dalam bidang ilmu pengetahuan dan teknologi yang berbasis kecerdasan buatan (AI), khususnya dalam penerapan *Machine Learning*

b. Sebagai acuan bagi penelitian mendatang dalam bidang kecerdasan buatan (AI), khususnya dalam penerapan machine learning berbasis web.

2. Manfaat Aplikatif

a. Mempermudah admin sekolah SMK Kesatrian Purwokerto dalam memberikan layanan yang lebih efisien kepada calon siswa

b. Membuat waktu dalam memperoleh informasi sekolah menjadi lebih efisien bagi calon siswa.

BAB II

TINJAUAN PUSTAKA

A. Landasan Teori

1. Implementasi

Dalam Kamus Besar Bahasa Indonesia (KBBI), implementasi memiliki makna pelaksanaan atau penerapan. Implementasi bukan hanya sekadar sebuah aktivitas, melainkan sebuah kegiatan yang dirancang dengan baik dan dilaksanakan dengan serius sesuai pedoman atau norma tertentu untuk mencapai tujuan yang telah ditetapkan. Oleh karena itu, implementasi tidak terjadi secara terpisah, melainkan dipengaruhi oleh elemen-elemen lain yang terkait(Rosad, 2019).

Keberhasilan implementasi adalah sebuah proses yang melibatkan berbagai elemen baru dan implementasi yang berhasil bergantung pada perencanaan langkah-langkah yang tepat, khususnya terkait proses pengembangannya(Suprapto & Malik, 2019).

2. Penerimaan Peserta Didik Baru (PPDB)

Penerimaan perserta didik baru (PPDB) merupakan kegiatan wajib yang dilakukan oleh lembaga pendidikan untuk menerima peserta didik baru, baik formal maupun non formal. PPDB adalah proses seleksi calon siswa yang akan menjadi bagian dari sekolah. Kegiatan ini merupakan rutinitas tahunan yang dilakukan setiap awal tahun ajaran. Penyelenggaraan PPDB harus dilakukan sesuai dengan standar yang telah ditetapkan oleh pemerintah (Rohmah dkk., 2021).

Tujuan dari PPDB menurut Pasal 2 ayat (1) menyatakan bahwa tujuan dari PPDB adalah memastikan proses penerimaan peserta didik baru dilakukan secara objektif, transparan, akuntabel, tanpa diskriminasi, dan berkeadilan, sehingga dapat mendukung peningkatan akses terhadap layanan pendidikan .

3. Machine Learning

Machine learning (ML) adalah mesin yang dikembangkan untuk bisa belajar dengan sendirinya tanpa arahan dari penggunanya. Machine learning, atau pembelajaran mesin, adalah teknologi yang sangat bermanfaat dalam menyelesaikan berbagai masalah dan mempermudah pelaksanaan berbagai tugas(Telaumbanua dkk., 2020). Menurut (Kurniyawan, 2022) *Machine Learning* adalah ilmu yang mempelajari tentang algoritma komputer yang bisa mengenali pola-pola di dalam data, dengan tujuan untuk mengubah beragam macam data menjadi suatu tindakan yang nyata dengan sesedikit mungkin campur tangan manusia.

Dari pengertian di atas, *machine learning* (ML) dapat diartikan sebagai cabang ilmu komputer yang mengembangkan kemampuan mesin untuk belajar secara mandiri dari data tanpa arahan langsung dari manusia. Teknologi ini dirancang untuk mengenali pola-pola dalam data dan mengolahnya menjadi tindakan nyata dengan minim campur tangan manusia.

4. *Chatbot*

Chatbot adalah program komputer yang dirancang untuk menirukan percakapan manusia dengan pengguna, baik tertulis maupun lisan. *Chatbot* merupakan aplikasi yang menggunakan kecerdasan buatan untuk berkomunikasi secara otomatis dengan pengguna melalui antarmuka percakapan. Dengan kemampuannya dalam menjawab pertanyaan, menyampaikan informasi, dan memberikan solusi, *chatbot* dapat berfungsi sebagai asisten virtual yang efisien (Lubis & Sumartono, 2023)

Chatbot adalah aplikasi yang didasarkan pada ilmu Natural Language Processing (NLP) dan berbasis kecerdasan buatan (AI), yang berfungsi sebagai sumber informasi yang dapat diakses oleh pengguna aplikasi(Sujacka Retno dkk., 2023).

5. *Text Mining*

Text mining dalah suatu bidang khusus *data mining*. *Text mining* adalah proses ekstraksi informasi dari data berbentuk teks, dengan sumber informasi yang biasanya berasal dari dokumen. Tujuan utamanya adalah mengidentifikasi kata-kata yang mewakili isi dokumen, sehingga analisis keterkaitan antar dokumen dapat dilakukan (Purnajaya dkk., 2022).

Text mining mencakup berbagai teknik dan metode yang digunakan untuk menganalisis, mengklasifikasikan, dan mengekstrak informasi dari teks. Pada *text mining* terdapat tahapan t*ext Preprocessing* yang Bertujuan untuk mengonversi data teks yang awalnya tidak terstruktur menjadi data teks yang lebih terorganisasi. Data yang telah

terstruktur ini kemudian dapat dimanfaatkan untuk berbagai keperluan, seperti analisis mendalam, pencarian pola, pengelompokan informasi, atau sebagai dasar dalam pengambilan keputusan berbasis data (Ramadhani dkk., 2022). Berikut tahapan dari *text prossesing*.

a. *Case Folding*

Case folding adalah mengubah semua huruf dalam dokumen menjadi huruf kecil, hanya huruf 'a' sampai dengan 'z' yang diterima. Karakter selain huruf dihilangkan dan dianggap delimeter.

b. Tokenisasi

Tokenisasi adalah proses memecah teks menjadi unit-unit kecil yang disebut token, seperti kata, frasa, atau simbol. Proses ini merupakan langkah awal dalam pengolahan teks untuk memudahkan analisis lebih lanjut, seperti pencarian pola atau penghitungan frekuensi kata.

c. *Stopword Removal*

Stopword removal adalah proses menghapus kata-kata umum yang tidak memiliki makna signifikan dalam analisis teks, seperti "dan," "atau," "yang," atau "itu." Tujuannya adalah untuk mengurangi kata-kata yang tidak relevan sehingga fokus analisis dapat diarahkan pada kata-kata yang lebih bermakna.

d. *Stemming*

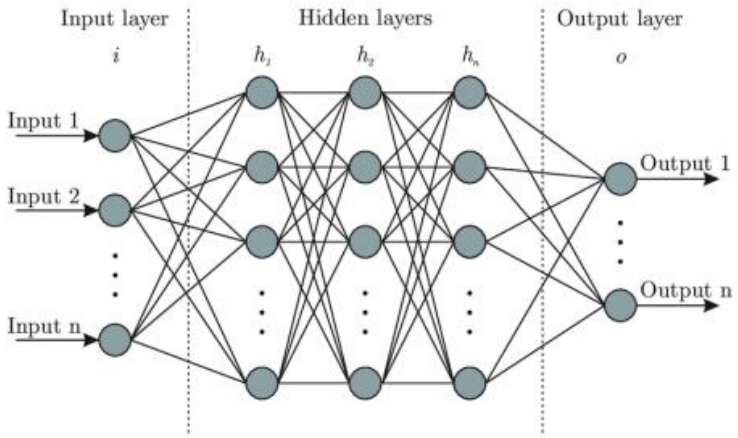
Stemming adalah proses mengubah kata turunan ke bentuk dasarnya (akar kata) dengan menghapus imbuhan seperti awalan,

akhiran, atau sisipan. Tujuannya adalah untuk menyederhanakan analisis teks dengan mengelompokkan kata-kata yang memiliki makna serupa. Contohnya, kata "berlari" dan "lari-lari" akan direduksi menjadi "lari."

6. *Artificial Neural Network* (ANN)

Artificial Neural Network merupakan model algoritma yang mencoba meniru otak manusia yang mampu memberikan stimulasi/rangsangan, melakukan proses, dan memberikan output untuk menemukan hubungan antara kumpulan data. Pemodelan berbasis *Artificial Neural Network* (ANN) adalah pendekatan yang digunakan untuk menyelesaikan masalah pengenalan pola dan teknik *data mining*. Jaringan ini terdiri dari sejumlah neuron buatan yang dikenal sebagai elemen pemrosesan (PE), unit, atau node (Iskandar & Sriharyani, 2021).

Struktur jaringan dan algoritma pelatihan memainkan peran penting dalam menentukan model-model ANN. Struktur ini berfungsi untuk menggambarkan bagaimana sinyal atau data bergerak melalui jaringan. Sementara itu, algoritma pembelajaran menjelaskan cara-cara mengubah bobot koneksi agar pasangan input-output yang diinginkan dapat tercapai (Syukri & Samsuddin, 2019).



gambar 2. 1 Arsitektur *Artificial Neural Network* (ANN)

pada gambar 2.1 merupakan bentuk dari arsitektur *Artificial Neural Network* (ANN), yang terdiri dari :

a. Input Layer

Input layer adalah lapisan yang terdiri dari unit-unit (neuron) yang menerima sinyal input langsung dari sumber eksternal dan mengirimkan informasi tersebut ke setiap neuron di hidden layer melalui bobot yang menghubungkan kedua lapisan tersebut.

b. *Hidden Layer*

Hidden layer adalah lapisan yang terdiri dari neuron-neuron tersembunyi yang terletak di antara input *layer* dan *output layer*, di mana *output*-nya tidak terlihat secara langsung. Penambahan *hidden layer* dapat memperbaiki kemampuan jaringan dalam mengenali pola.

c. Output Layer

Output layer adalah lapisan yang memiliki unit *output* yang keluarannya memiliki solusi dari algoritma *Artificial Neural Network* pada permasalahan yang di berikan.

7. Python

Python merupakan bahasa pemrograman yang biasa dipakai untuk membangun situs, software/aplikasi, dan melakukan analisis data. Python memiliki kemampuan untuk membangun software berbasis kecerdasan buatan (*artificial intelligence*) seperti pengolahan data, *machine learning*, *deep learning*, dan *data science*. Python adalah bahasa pemrograman multi-paradigma yang menggabungkan pemrograman berorientasi objek, imperatif, dan fungsional. (Enterprise, 2019).

Dari penjelasan ditas bahasa pemrograman python memiliki Kelebihannya tersendiri antara lain :

a. Proses pengembangan perangkat lunak menjadi lebih efisien dengan pengurangan baris kode
b. Mendukung multi platform
c. Dilengkapi dengan fitur alokasi memori otomatis.
d. Python mendukung paradigma pemrograman berorientasi objek.

B. Penelitian Sebelumnya

1. Kristian Adi Nugraha & Danny Sebastian (2021) melakukan penelitian dengan judul "*Chatbot* Layanan Akademik Menggunakan *K-Nearest Neighbor*". Institusi kecil hingga menengah seringkali menghadapi kendala dalam hal menjawab pertanyaan berulang. Penelitian ini berfokus pada mengembangkan *chatbot* dengan memanfaatkan algoritma *K-Nearest Neighbor* (K-NN) untuk menjawab pertanyaan terkait kegiatan akademik secara otomatis, khususnya di lingkungan Fakultas Teknologi

Informasi Universitas Kristen Duta Wacana. Sistem ini dibangun menggunakan bahasa pemrograman Python dan mengadopsi teknik Natural Language Processing seperti tokenisasi, stemming, dan penghilangan kata-kata *stop*. Hasil pengujian menunjukkan akurasi tertinggi sebesar 53,48% pada nilai K=3. Kendati demikian, sistem masih menghadapi tantangan dalam mengklasifikasi pertanyaan yang memiliki struktur kata serupa atau menggunakan kata-kata tidak baku.

2. Feri Mustakin, dkk (2021) melakukan penelitian dengan judul "Algoritma *Artificial Neural Network* pada Text-based *Chatbot Frequently Asked Question* (FAQ) Web Kuliah Universitas Nasional". Kurangnya literasi mahasiswa terhadap penggunaan web kuliah Universitas Nasional menyebabkan pertanyaan berulang terkait hal seperti pengumpulan tugas dan lupa kata sandi. Penelitian ini mengembangkan *chatbot* berbasis teks menggunakan algoritma *Artificial Neural Network* (ANN) dengan dataset 25 pertanyaan FAQ yang dikelompokkan menjadi 16 label. Setelah pelatihan dengan 1000 epoch dan teknik Natural Language Processing (NLP), pengujian menunjukkan akurasi tinggi sebesar 97,27%, sehingga efektif dalam menjawab pertanyaan mahasiswa secara otomatis.
3. Muhamad Sidik, dkk (2021)melakukan penelitian dengan judul "Pembuatan Aplikasi *Chatbot* Kolektor Dengan Metode *Extreme Programming* Dan Strategi *Forward Chaining*." Penelitian ini mengatasi kendala sistem SMSCenter di PT. Indomobil Finance Indonesia, seperti

ketergantungan pada kartu SIM dan biaya tinggi, dengan mengembangkan *Chatbot* Kolektor berbasis LINE menggunakan metode Extreme Programming dan strategi Forward Chaining. Dataset mencakup data kolektor dan kendaraan, seperti nomor polisi, mesin, rangka, serta informasi blacklist pelanggan. Hasil pengujian menunjukkan *chatbot* memiliki tingkat keberhasilan 95% dan waktu respon rata-rata 3,42 detik, jauh lebih cepat dari SMSCenter (24,25 detik), sehingga meningkatkan efisiensi layanan dan mengurangi biaya operasional.

4. Fahmi Yusron, dkk (2024) melakukan penelitian dengan judul "*Chatbot* Informasi Penerimaan Mahasiswa Baru Menggunakan Metode *FastText* dan LSTM" Layanan informasi penerimaan mahasiswa baru (PMB) di Fakultas Sains dan Informatika Universitas Jenderal Achmad Yani dinilai kurang efisien karena masih dilakukan secara manual, menyebabkan pengulangan jawaban atas pertanyaan serupa. Artikel ini membahas pengembangan *chatbot* berbasis metode *FastText* untuk representasi kata dan *Long Short-Term Memory* (LSTM) untuk klasifikasi teks, guna meningkatkan efisiensi dan konsistensi layanan. Dengan data dari kuesioner mahasiswa, model yang dihasilkan memiliki akurasi tinggi (89–90%) dan mampu menjawab berbagai pertanyaan terkait PMB dengan *respon* relevan dan informatif.
5. Mohammad Ovi Sanjaya, dkk (2023) melakukan penelitian dengan judul "*Virtual Assistant for Thesis Technical Guide Using Artificial Neural Network*" Permasalahan dalam memberikan panduan teknis skripsi secara

cepat dan akurat mendorong pengembangan *chatbot* berbasis *Artificial Neural Network* (ANN). *Chatbot* ini menggunakan model ANN dengan fungsi aktivasi ReLU dan Softmax serta dioptimalkan menggunakan Stochastic Gradient Descent (SGD). Dengan basis Panduan Teknis Skripsi 2022, *chatbot* mencapai akurasi 99,49% dan skor F1 sebesar 91%. Diuji dengan confusion matrix dan diimplementasikan pada Telegram, *chatbot* ini efektif memberikan panduan teknis berbasis teks

Tabel 2. 1 Tabel Penelitian Sebelumnya

No	Nama dan Tahun Penelitian	Judul Penelitian	Hasil	Perbandingan	
				Persamaan	Perbedaan
1	Kristian Adi Nugraha dan Danny Sebastian, 2021	*Chatbot* Layanan Akademik Menggunakan K-Nearest Neighbor	Menghasilkan *chatbot* dengan algoritma KKN untuk menjawab pertanyaan seputar kegiatan akademik mendapatkan akurasi sebesar 53,48%	1. *Chatbot* berbasis *machine learning.*	Penelitian ini menggunakan algoritma ANN dikarenakan algoritma lebih cocok untuk dataset yang kompleks dibanding dengan algoritma KNN yang cocok pada dataset kecil yang membutuhkan interpretasi mudah.
2	Feri Mustakin, Fauziah, Nur Hayati, 2021	Algoritma *Artificial Neural Network* pada *Text-based Frequently Asked Question* (FAQ) Web Kuliah Universitas	Menghasilkan dengan algoritma ANN untuk membantu dalam menjawab pertanyaan dalam FAQ dalam bentuk GUI mendapatkan akurasi 97,27%.	1. Menggunakan algoritma ANN 2. Berbasis *machine learning*	Perbedaan pada *output chatbot* • Pada penelitian sebelumnya *chatbot* bentuk GUI • Pada penelitian ini berbentuk *website*
3	Muhamad Sidik, Bambang Gunawan, dan Dina Anggraini, 2021	Pembuatan Aplikasi *Chatbot* Kolektor Dengan Metode *Extreme* Programming Dan Strategi *Forward Chaining*	Menghasilkan dengan algoritma *Forward Chaining* berbasis LINE sebagai media layanan informasi secara otomatis kepada dengan akurasi 95%.	1. Penelitian sebelumnya sama-sama meneliti terkait *chatbot*	Penelitian ini menggunakan algoritma ANN dikarenakan ideal untuk *chatbot* interaktif berbasis NLP dengan fleksibilitas tinggi dari pada algoritma *Forward Chaining* berbasis logika *(rule-based)*

4	Fahmi Yusron F, Agus Komarudin, dan Melina, 2024	*Chatbot* Informasi Penerimaan Mahasiswa Baru Menggunakan Metode *FastText* dan LSTM	Menghasilkan untuk menjawab pertanyaan PMB seperti biaya, jadwal pendaftaran, beasiswa, dll. Menggunakan metode FastTex dan LSTM untuk klasifikasi teks mendapatkan akurasi 89–90%	1. Penelitian pengembangan *chatbot*	Pada penelitian ini menggunakan algoritma ANN Dikarenakan lebih sederhana dibandingkan LSTM, sehingga lebih cepat dilatih dan membutuhkan lebih sedikit sumber daya komputasi
5	Mohammad Ovi Sanjaya, Saiful Bukhori, dan Muhammad ‘Ariful Furqon, 2023	*Virtual Assistant for Thesis Technical Guide Using Artificial Neural Network*	Menghasilkan untuk panduan teknis tesis. Menggunakan algoritma ANN dan menggunakan metode optimasi *Stochastic Gradient Descent* (SGD*)* mendapatkan akurasi 91%.	1. Menggunakan algoritma ANN	Perbedaan pada *output chatbot* • Penelitian sebelumnya menggunakan telegram • Penelitian ini *chatbot* berbasis *website*

BAB III
METODE PENELITIAN

A. Tempat dan Waktu Penelitian

1. Tempat Penelitian

   Pada penelitian ini dilaksanakan di SMK Kesatrian Purwokerto Desa Sokanegara, Kecamatan Purwokerto, Kabupaten Banyumas, Jawa Tengah 53115.

2. Waktu Penelitian

   Waktu Penelitian dilakukan selama 4 bulan mulai september 2024 hingga 14 januari 2025.

B. Metode Pengumpulan Data

Bagian ini memuat penjelasan secara lengkap dan terinci tentang cara-cara yang digunakan dalam proses pengumpulan data untuk jenis data yang diperlukan. Misalnya melalui observasi, wawancara, eksperimen, atau kuesioner. Jika metode kuesioner digunakan, maka blangko angket kuesioner harus dilampirkan dalam laporan.

Pada penelitian ini diperlukan serangkaian kegiatan untuk mendapatkan data yang dibutuhkan pada penelitian. Dalam melakukan penelitian penulis menggunakan metode pengumpulan data sebagai berikut:

1. Wawancara

   Menurut (Fadhallah, 2021) wawancara merupakan bentuk komunikasi antara dua atau lebih pihak yang biasanya dilakukan secara langsung. Dalam wawancara, satu pihak bertindak sebagai pewawancara

(interviewer) dan pihak lainnya sebagai yang diwawancarai (interviewee) dengan tujuan tertentu, seperti memperoleh informasi atau mengumpulkan data. Pewawancara mengajukan sejumlah pertanyaan kepada yang diwawancarai untuk mendapatkan jawaban yang diperlukan.

Peneliti melakukan wawancara dengan staf IT sekaligus Admin PPDB SMK Kesatrian Purwokerto yaitu bapak Agung Sulistiono, S.T.

2. Observasi

Observasi adalah teknik pengumpulan data yang unik karena melibatkan pengamatan langsung terhadap suatu fenomena. Berbeda dengan wawancara atau kuesioner yang mengandalkan laporan subjektif, observasi memungkinkan peneliti untuk mengamati perilaku dan kejadian secara objektif. Metode ini sangat relevan untuk penelitian yang berkaitan dengan perilaku manusia, proses kerja, atau gejala alam, terutama ketika jumlah subjek penelitian relatif kecil (Sugiyono, 2018).

Penulis menggunakan teknik observasi untuk mengumpulkan data mengenai pertanyaan-pertanyaan yang sering ditanyakan oleh calon siswa mengenai informasi PPDB dan informasi sekolah untuk dijadikan *dataset* pada *chatbot*.

3. Studi Pustaka

Studi kepustakaan merupakan langkah penting dalam penelitian yang melibatkan pengkajian mendalam terhadap teori-teori relevan, mutakhir, dan asli yang berkaitan dengan objek penelitian. Teori-teori ini

berperan sebagai landasan berpikir dalam menganalisis data dan menarik kesimpulan (Sugiyono, 2018).

Penulis melakukan kajian pustaka komprehensif dengan merujuk pada berbagai sumber seperti jurnal ilmiah, buku, skripsi, dan ebook untuk memperkaya landasan teori dan metodologi penelitian.

C. Alat dan Bahan Penelitian

Dalam penelitian ini memerlukan alat dan bahan. Berikut adalah alat dan bahan yang digunakan:

1. Alat Penelitian
   a. Komputer PC (Personal Computer)

      Spesifikasi komputer pc yang di gunakan penulis sebagai berikut :

      1) Laptop : Acer aspire 4741
      2) *Processor* : Intel Core i3 i3-350M 2,26 GHz
      3) RAM : 6 GB
      4) *Hardisk* : 500 GB

   b. Perangkat Lunak (Software)

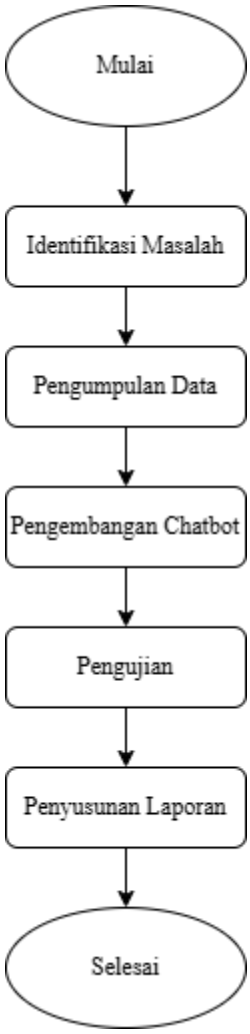
      1) Sistem Operasi Windows 10
      2) Microsoft Word 2019
      3) Chrome Browser
      4) Visual Studio Code
      5) Python

2. Bahan

   Bahan penelitian ini berupa *dataset* yang berasal dari hasil wawancara dengan Bapak Agung Sulistiono, S.T. sebagai narasumber ahli, serta tanggapan responden terhadap kuesioner yang berisi pertanyaan-pertanyaan umum seputar pelaksanaan PPDB.

a. Konsep Penelitian

1. Kerangka Berpikir



Gambar 3. 1 Alur kerangka berpikir

Penelitian ini menggunakan alur kerangka berpikir sebagai panduan untuk menyelesaikan setiap proses yang ada, sehingga dapat membantu dalam perancangan *chatbot* agar berfungsi dengan baik sesuai dengan tahapan yang telah dirancang. Berikut ini adalah penjelasan mengenai alur kerangka berpikir dalam penelitian ini:

a. Identifikasi Masalah

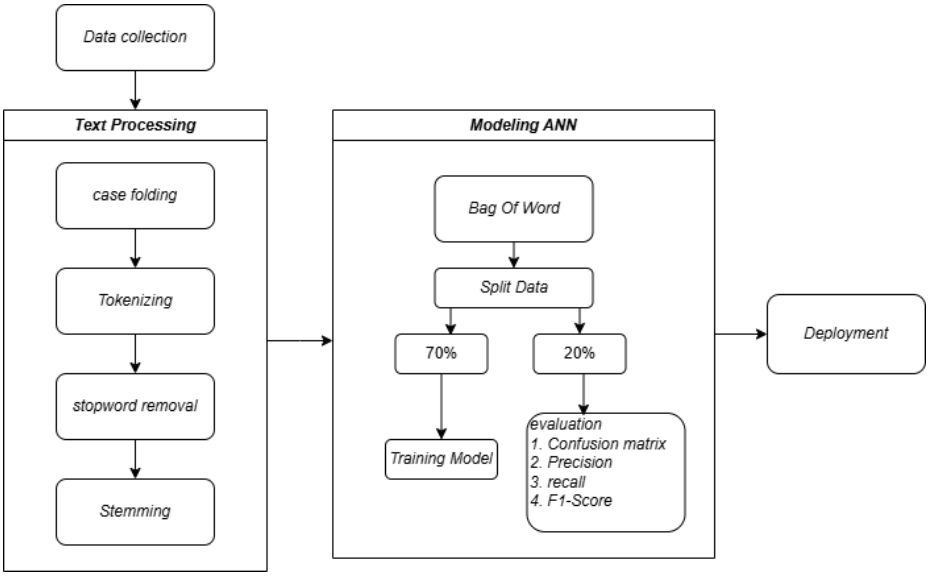
Proses identifikasi masalah adalah langkah awal dalam penelitian yang bertujuan untuk mengidentifikasi permasalahan pada objek penelitian SMK Kesatrian Purwokerto. Tahap ini memiliki peran penting karena hasilnya akan digunakan untuk merumuskan permasalahan yang ada di lokasi penelitian, yang nantinya menjadi dasar dalam merancang latar belakang penelitian tersebut.

b. Pengumpulan Data

Pada tahap ini, peneliti mengumpulkan berbagai data yang relevan dengan objek permasalahan untuk melengkapi bahan penelitian. Dalam proses pengumpulan data, peneliti menggunakan beberapa metode, antara lain wawancara, pengamatan (observasi), dan Studi pustaka.

c. Tahapan Pengembangan *Chatbot*

Dalam pengembangan *chatbot*, terdapat tahapan-tahapan yang perlu dilakukan untuk membuat *chatbot* yang akan dikembangkan. Setelah mengumpulkan data dan memilih pertanyaan-pertanyaan yang sering diajukan oleh calon siswa, data tersebut akan diolah dan dilatih.



Gambar 3. 2 Alur Proses Pengembangan Chatbot

1) *Data Collection*

Data Collection dikumpulkan untuk keperluan *Text Processing* dengan tujuan menyediakan bahan baku yang memadai untuk menjalankan berbagai analisis bahasa. Sumber dataset yang digunakan adalah pertanyaan-pertanyaan yang sering di ajukan tentang PPDB dan akan disimpan dalam format JSON. Dataset memiliki struktur diantaranya :

- *Intents*, Kumpulan semua data input dan output yang digunakan untuk melatih *chatbot*
- *Tag*, digunakan untuk mengelompokkan data teks yang serupa dan menjadikannya sebagai output target dalam melatih jaringan neural.
- *Patterns*, merupakan bagian yang berisi pola pertanyaan yang diinginkan pengguna.

- *Response*, berisikan data pola *output* atau jawaban dari pertanyaan yang akan dikirimkan pada pengguna.

2) *Text Processing*

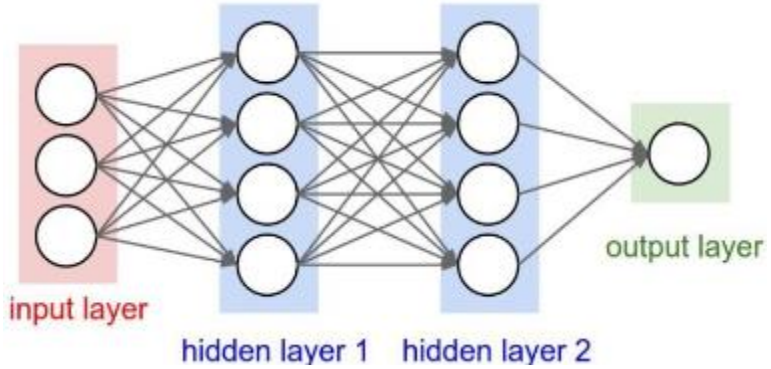
Text processing merupakan tahapan awal dalam pengembangan *chatbot* yang bertujuan untuk mempersiapkan data teks agar lebih mudah diproses oleh model.

- *case folding*, yaitu mengubah seluruh huruf dalam teks menjadi huruf kecil untuk menghindari perbedaan akibat kapitalisasi.
- *tokenisasi*, di mana teks dibagi menjadi potongan-potongan kecil berupa kata atau token.
- *stopword removal* dilakukan untuk menghapus kata-kata umum yang tidak memiliki makna signifikan, seperti "dan," "ke," atau "yang."
- *stemming* bertujuan mengubah kata menjadi bentuk dasar atau akarnya, sehingga kata seperti "berlari" akan dikembalikan menjadi "lari." Semua tahapan ini memastikan data teks siap digunakan untuk pembuatan model yang lebih akurat dan efisien.

3) *Modelling* ANN

Implementasi model *Artificial Neural Network* (ANN) untuk pemrosesan teks melibatkan perancangan arsitektur jaringan saraf tiruan jenis *forward neural network* (FNN) model *Multilayer perceptrons* (MLP) dengan menggunakan dua lapisan *hidden layer* dalam menghasilkan serangkaian *output* dari *input* yang diberikan.

Arsitektur *Multilayer Perceptron* (MLP) adalah salah satu bentuk populer dari jaringan saraf tiruan *Artificial Neural Network* yang sering digunakan dalam pemrosesan teks (Sai dkk., 2023). MLP diklasifikasikan sebagai jenis ANN yang diawasi (*supervised*), di mana proses pelatihannya menggunakan metode *backpropagation*. Arsitektur jaringan saraf MLP melibatkan penentuan jumlah neuron di setiap lapisan serta fungsi transfer yang digunakan pada lapisan-lapisan tersebut (Paluang dkk., 2024).



Gambar 3. 3 Arsitektur *Multilayer perceptrons* (MLP)

Dari gambar 3.3 menunjukan arsitektur MLP *input layer* untuk menerima data, dua *hidden layers* untuk memproses pola kompleks. Setiap lapisan dalam jaringan saraf tiruan menggunakan fungsi aktivasi untuk menentukan keluaran neuron. Penelitian ini menggunakan fungsi ReLU, yang mengubah nilai negatif menjadi nol dan fungsi *softmax*, yang menghitung probabilitas untuk klasifikasi multi-kelas (Mustakim dkk., 2021). Berikut rumusan dari fungsi aktivitas ReLU (1) dan fungsi aktivitas softmax (2):

$$f(x) = \max(0, x) \qquad (1)$$

$$atau$$

$$f(x) = \begin{cases} 0\ untuk\ x\ \leq\ 0 \\ x\ untuk\ x\ >\ 0 \end{cases}$$

$$f(Xi) = \frac{Exp(Xi)}{\sum_{j=0}^{k} Exp(Xj)}\ , nilai\ i \qquad (2)$$

$$= 0,1,2, \ldots, k$$

ReLU sederhana dan efisien, sedangkan softmax menghasilkan output berupa probabilitas antara 0 dan 1. Selain itu, MLP dapat memberikan fleksibilitas untuk memahami pola dalam data teks dan menghasilkan respons berdasarkan pola yang dipelajari, menjadikannya efektif dalam berbagai aplikasi seperti klasifikasi sentimen atau generasi teks (Bhashkar, 2019).

Model yang telah dilatih akan mampu memahami pola kompleks dalam data teks dan melakukan tugas-tugas seperti klasifikasi sentimen, terjemahan mesin, atau generasi teks.

4) *Bag of Words*

Konsep *bag of words* mereduksi teks menjadi sekumpulan kata dan menghitung jumlah kemunculan masing-masing kata dalam dokumen, menghasilkan vektor numerik yang mewakili teks tersebut (Arbizal dkk., 2024). Proses ini menginisialisasi sistem dengan data pelatihan. Setiap kata dalam data ini akan dipetakan ke sebuah vektor biner, di mana nilai 1 menunjukkan keberadaan kata dalam kamus kata, dan 0 menunjukkan ketidakhadirannya seperti yang dijelaskan pada tabel.

Tabel 3. 1 Proses *Bag of Words*

Kata	Bag of Words				
	Saya	Ingin	Sekali	Daftar	Sekolah
Saya	1	0	0	0	0
Ingin	0	1	0	0	0
Daftar	0	0	0	1	0
Sekolah	0	0	0	0	1

5) Split Data

Split data adalah teknik yang digunakan untuk membagi dataset menjadi beberapa bagian dan merupakan salah satu faktor yang mempengaruhi performa model klasifikasi dalam algoritma pembelajaran mesin (Nurhopipah & Hasanah, 2020). Pada tahapan ini memisahkan data menjadi 70% - 20%, 70% untuk data *Training* dan 20 % untuk data evaluasi. Tujuan melakukan pemisahan data ini untuk menghindari *overfitting*

6) Training Model

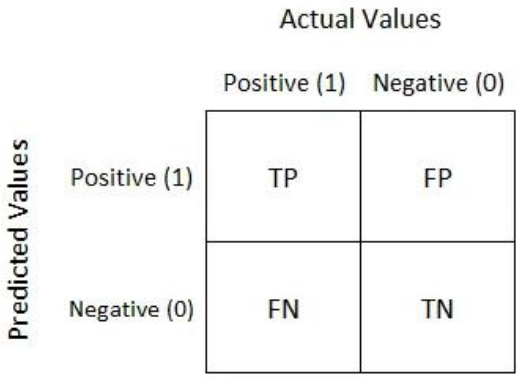
Proses training model bertujuan untuk mengoptimalkan performa *Artificial Neural Network* (ANN) dengan mencari kombinasi bobot dan bias terbaik yang meminimalkan *error* (*loss*) dan meningkatkan tingkat akurasi. Selama training, model belajar dari data latih melalui iterasi berulang, di mana algoritma optimasi, *Stochastic Gradient Descent* (SGD) digunakan untuk menyesuaikan parameter model berdasarkan perhitungan error. Proses pelatihan data

dilakukan dengan *stoping* kriteria berdasarkan jumlah epoch dan batch size.

7) Evaluasi

Dari *training* data akan dievaluasi menggunakan *confusion matrix.* Confusion matrix merupakan representasi kinerja model klasifikasi pada dataset Berhasilasi dengan nilai yang telah diketahui sebelumnya. *Confusion matrix* mencakup empat metrik evaluasi utama *recall, precision, accuracy*, dan *F1-score*. Metrik-metrik ini digunakan untuk mengukur sejauh mana model mampu memprediksi kelas pada data uji dengan benar. Selain itu, *confusion matrix* juga membantu mengidentifikasi kesalahan klasifikasi serta membandingkan performa antar model (Faurina dkk., 2023).

Confusion matrix terdiri dari empat elemen utama dalam klasifikasi, yaitu *False Positive* (FP), *True Positive* (TP), *True Negative* (TN), dan *False Negative* (FN). *True Positive* (TP) mengacu pada data positif yang berhasil terdeteksi dengan benar, sedangkan *True Negative* (TN) adalah data negatif yang diidentifikasi secara akurat. Sebaliknya, *False Positive* (FP) terjadi ketika data negatif salah diklasifikasikan sebagai positif, dan *False Negative* (FN) adalah data positif yang tidak dikenali dan dianggap sebagai negatif .



Gambar 3. 4 Confusion Matrix

Precision mengukur seberapa sering prediksi positif yang dibuat oleh model sebenarnya benar (Krstinic dkk., 2020) (6) .

$$\frac{\text{TP}}{(\text{TP}+\text{FP})} * 100\% \qquad (6)$$

Recall mengukur seberapa lengkap model dapat menemukan semua contoh positif yang sebenarnya ada dalam data (Krstinic dkk., 2020) (7).

$$\frac{TP}{(TP+FN)} * 100\% \qquad (7)$$

Akurasi mengukur seberapa sering model kita membuat prediksi yang benar secara keseluruhan (Krstinic dkk., 2020) (8).

$$\frac{TP+TN}{(TP+FP+FN+TN\ )} * 100\% \qquad (8)$$

F1-score adalah metrik yang menggabungkan *precision* dan *recall* untuk memberikan penilaian yang lebih seimbang tentang kinerja model (Krstinic dkk., 2020) (9).

$$2 * \frac{Recall * Precission}{Recall+Precissio} * 100\% \qquad (9)$$

Evaluasi model *chatbot* dalam penelitian ini menggunakan akurasi, presisi, *recall*, dan *F1-score* untuk memberikan analisis kinerja model yang menyeluruh. Akurasi menilai seberapa sering prediksi benar secara keseluruhan, presisi memastikan akurasi prediksi kelas positif untuk meminimalkan *false positives*, dan *recall* mengukur kemampuan model mendeteksi semua kasus positif guna mengurangi *false negatives*. *F1-score*, sebagai rata-rata harmonis presisi dan *recall* (Faurina dkk., 2023). Evaluasi ini memastikan *chatbot* dapat memberikan informasi yang relevan, benar, dan andal kepada pengguna.

8) Deployment

Proses *deployment chatbot* untuk layanan penerimaan peserta didik baru (PPDB) pada SMK Kesatrian Purwokerto dilakukan dalam bentuk *website* dengan menggunakan *framework* Flask berbasis Python. *Framework* Flask tergolong sebagai *micro-framework* karena tidak membutuhkan banyak *library* atau alat tambahan. *Framework* ini mengandalkan Werkzeug dan Jinja *Template Engine* sebagai dependensinya, serta memungkinkan pengembangan aplikasi dengan *sintaks* yang mudah dan sederhana (Larasati & Susetyo, 2024).

d. Pengujian

Pengujian dilakukan dengan pendekatan *Blackbox Testing*, yaitu metode pengujian aplikasi yang tidak memerlukan pemahaman mendalam tentang detail teknis aplikasi, seperti *source code* (Sasongko

dkk., t.t.). Pada *blackbox* terdapat 2 metode lainnya yaitu *Alpha Testing* dan *Beta Testing.*

Alpha testing dan beta testing adalah dua tahap pengujian penting dalam pengembangan aplikasi. *Alpha testing* bertujuan untuk memastikan bahwa aplikasi dapat berfungsi dengan baik tanpa *error* atau *bug*, dilakukan di lingkungan pengembangan sebelum aplikasi dirilis. Setelah itu, *beta testing* dilakukan di lingkungan nyata dengan melibatkan pengguna akhir, biasanya melalui penyebaran kuesioner untuk mengumpulkan masukan dan menyimpulkan penilaian terhadap aplikasi yang telah dikembangkan (Yulia Puspaningrum dkk., 2024). Pengujian *beta testing* diperlukan nya sejumlah responden untuk mencoba dan menguji apakah *chatbot* sudah sesuai dengan apa yang diharapkan oleh pengguna atau belum.

e. Penyusunan Laporan

Setelah seluruh proses penelitian tuntas, langkah berikutnya adalah menyusun laporan ilmiah. Laporan ini akan mencakup saran dan kesimpulan yang menunjukkan potensi pengembangan lebih lanjut dari penelitian ini.

BAB IV
HASIL DAN PEMBAHASAN

A. Identifikasi Masalah

Dalam Proses identifikasi masalah dilakukan untuk menentukan permasalahan utama yang dihadapi oleh SMK Kesatrian Purwokerto dalam penyampaian informasi Penerimaan Peserta Didik Baru (PPDB). Berdasarkan wawancara dengan Bapak Agung Sulistiono, S.T., selaku staf IT dan admin PPDB sekolah, ditemukan beberapa kendala:

1. Keterbatasan Layanan di Luar Jam Kerja
2. Volume Pertanyaan yang Tinggi

Hasil identifikasi ini menjadi dasar untuk merancang *chatbot* yang mampu memberikan layanan informasi secara otomatis dan efisien, terutama terkait informasi PPDB seperti jadwal pendaftaran, persyaratan, biaya, dan lainnya.

B. Pengumpulan Data

Dalam tahapan ini pengumpulan data dilakukan melalui observasi dan wawancara untuk mencari tahu bagaimana admin dalam merespon pertanyaan terkait penerimaan peserta didik baru (PPDB) pada SMK Kesatrian Purwokerto. Dari hasil wawancara dan observasi menyimpulkan bahwa:

1. Pertanyaan yang masuk akan di balas satu persatu sesuai pesan yang masuk terlebih dahulu.

2. Batas waktu admin menjawab hanya sampai jam kerja selesai. Apabila ada pesan yang masuk di luar jam kerja maka akan dibalas keesokan harinya.
3. Pertanyaan yang sering ditanyakan oleh calon siswa, seperti informasi jurusan yang dibuka, jadwal pendaftaran, Proses pendaftaran secara online,dan nilai rata-rata yang digunakan.

C. Pengembangan *Chatbot*

1. Data Collection

Data yang sudah kita kumpulkan sebelumnya, terutama pertanyaan-pertanyaan yang sering muncul, akan diubah ke dalam format data JSON. Data ini didapatkan dari berbagai sumber, seperti brosur dan *website* PPDB SMK Kesatrian Purwokerto, serta dari hasil penelitian lainnya.

Proses selanjutnya melibatkan pembuatan dataset, dimana data dikumpulkan secara manual dan diubah menjadi format JSON. Dataset ini mencakup pertanyaan-pertanyaan yang dibutuhkan untuk sistem membaca dan memproses setiap pola pertanyaan yang muncul, bersama dengan jawaban yang sesuai. Dataset ini terdiri dari :

a. *Tag*, adalah kategori atau penanda untuk kumpulan pola-pola pertanyaan.
b. *Pattern*, berisi bagian pola pertanyaan yang diinginkan pengguna
c. *Response*, yang berisi jawaban yang akan dihasilkan oleh sistem berdasarkan kombinasi dari tag dan pola yang telah ditentukan.

Pada penelitian ini, terdapat 295 dataset pola pertanyaan, 38 tag, dan 44 respon. Pada gambar 4.1 merupakan struktur dari dataset.

Gambar 4. 1 Struktur dataset JSO

Pada gambar 4.1 merupakan struktur dari dataset yang digunakan terdapat 3 bagian yaitu *tag*, *patterns*, dan *respon es*. Sebagai contoh, tag "*greeting*" merupakan label yang mewakili maksud sapaan. Pada tag ini, terdapat beberapa pola pertanyaan pada bagian patterns, seperti "Hai", "Hi", "Apa kabar?", dan "Selamat siang". Jika *chatbot* mendeteksi salah satu pola tersebut, maka *respon* yang diberikan adalah "Halo! Ada yang bisa saya bantu?".

Namun, pola-pola yang ada pada bagian *patterns* mungkin masih terbatas dalam mencakup keragaman pola pertanyaan pengguna. Untuk meningkatkan performa dan kemampuan *chatbot* dalam memahami berbagai variasi input, diperlukan proses *augmentasi* data. Proses ini

bertujuan untuk memperluas pola-pola yang ada dengan menciptakan variasi data secara sistematis, sehingga model menjadi lebih robust terhadap berbagai bentuk input (Harahap & Muslim, 2020).

```python
# Fungsi augmentasi
def augment_data(patterns, num_augment=3):
    augmented_patterns = []
    for pattern in patterns:
        for _ in range(num_augment):
            # Pilih augmentasi secara acak
            choice = random.choice(['synonym', 'insertion', 'deletion'])

            if choice == 'synonym':
                augmented_pattern = replace_with_synonym(pattern)
            elif choice == 'insertion':
                augmented_pattern = insert_word_randomly(pattern)
            elif choice == 'deletion':
                augmented_pattern = delete_word_randomly(pattern)
            else:
                augmented_pattern = pattern

            augmented_patterns.append(augmented_pattern)
    return augmented_patterns
```

Gambar 4. 2 *Source Code Augmentasi Data*

Gambar 4.2 merupakan *augmentasi* data untuk memperluas variasi pola input (*patterns*) dalam dataset *chatbot* menggunakan tiga teknik penggantian sinonim, penyisipan kata, dan penghapusan kata. Prosesnya iteratif, di mana setiap pola dalam dataset diproses melalui fungsi *augment_data* yang memilih salah satu teknik augmentasi secara acak dan menghasilkan pola baru sejumlah yang diinginkan (*num_augment*). Pola hasil augmentasi kemudian digabungkan dengan pola asli, menciptakan dataset yang lebih beragam dan *robust* untuk melatih model *chatbot* agar dapat memahami input yang lebih bervariasi dengan tetap mempertahankan makna.

```
Tag: prestasi_sekolah, Jumlah Pola: 15
Patterns:
- Apa saja prestasi yang pernah diraih sekolah ini?
- Prestasi sekolah ini apa saja?
- Prestasi akademik sekolah?
- Prestasi non-akademik sekolah?
- Apa saja penghargaan yang pernah diterima sekolah?
- Apa saja prestasi yang pernah diraih ini?
- Apa saja prestasi yang saja pernah diraih sekolah ini?
- sekolah ini apa saja?
- Prestasi saja? sekolah ini apa saja?
- Prestasi akademik sekolah?
- akademik sekolah?
- Prestasi non-akademik sekolah?
- Prestasi non-akademik
- Apa saja penghargaan yang Apa pernah diterima sekolah?
- Apa saja penghargaan pernah yang pernah diterima sekolah?
```

Gambar 4. 3 Hasil Augmentasi Data

Hasil augmentasi data pada tag "prestasi_sekolah" menunjukkan, Beberapa pola baru dihasilkan melalui penggantian sinonim (misalnya, "prestasi akademik" menjadi "akademik sekolah") dan penyisipan kata (misalnya, "prestasi yang pernah diraih ini" atau "prestasi saja? sekolah ini apa saja?"), sementara penghapusan kata juga terlihat pada pola seperti "prestasi non-akademik" yang menjadi lebih singkat. Teknik augmentasi juga telah menambah dataset menjadi lebih banyak dengan pola-pola baru yang ditambahkan, dari awal jumlah dataset ada 295 setelah melakukan augmentasi data jumlah dataset menjadi 882 data pola pertanyaan. Hasil jumlah data setelah augmentasi dapat dilihat pada gambar 4.4.

```
Jumlah tags: 38
Jumlah responses: 44
Jumlah pattren: 294
```

Gambar 4. 4 Hasil Sebelum Augmentasi Data

```
Jumlah kata unik yang telah dilemmatize: 166
Jumlah kelas: 38
Jumlah dokumen: 882
```

Gambar 4. 5 Hasil Setelah Augmentasi Data

2. *Text processing*

a. *case folding*

Case folding adalah teknik untuk menyeragamkan bentuk huruf dalam teks dengan mengubah semua huruf menjadi huruf kecil, mulai dari 'a' hingga 'z', dan menghilangkan karakter lainnya (Nugroho, 2019). Ini adalah langkah awal untuk *Text processing* menyamakan bentuk huruf dalam dokumen sehingga bentuk nya standar menjadi huruf kecil semua .

```
def casefolding_text(text):
    return text.lower()
```

Gambar 4. 6 Kode Program *Case Folding*

Pada gambar 4.2 merupakan *function* untuk *case folding* yang memiliki parameter (Text) yang dimana akan mengembalikan teks yang sudah dirubah menjadi huruf kecil menggunakan method .lower(). Untuk hasil *case folding* seperti pada tabel 4.1.

Tabel 4. 1 Proses *Case Folding*

Sebelum *Case Folding*	**Setelah *Case Folding***
Kapan jadwal pendaftaran dibuka?	kapan jadwal pendaftaran dibuka?
Tanggal berapa PPDB dimulai?	tanggal berapa ppdb dimulai?
Di mana lokasi sekolah?	di mana lokasi sekolah?

b. Tokenizing

Tokenisasi adalah proses penguraian kalimat menjadi unit-unit kata yang terpisah untuk memudahkan pengolahan dan analisis teks lebih lanjut. Pada tahapan ini seluruh kalimat pada dataset akan dipisah menjadi kata perkata.

```
from nltk.tokenize import word_tokenize

def tokenize_text(text):
    return word_tokenize(text)
```

Gambar 4. 7 Kode program *Tokenizing*

Pada gambar 4.3 merupakan Fungsi tokenize text bertujuan untuk memecah teks input menjadi kata-kata individual (token) menggunakan fungsi word_tokenize dari library nltk, kemudian mengembalikan hasilnya dalam bentuk list yang berisi kata-kata tersebut, untuk kemudian diproses lebih lanjut dalam analisis teks. hasil dari Tokenizing pada gambar 4.4.

```
Hasil tokenisasi: ['kapan', 'jadwal', 'pendaftaran', 'dibuka', '?']
Hasil tokenisasi: ['tanggal', 'berapa', 'ppdb', 'dimulai', '?']
Hasil tokenisasi: ['kapan', 'pendaftaran', 'di', 'tutup', '?']
Hasil tokenisasi: ['kapan', 'penututupan', 'pendaftaran', 'nya', '?']
Hasil tokenisasi: ['jadwal', 'ppdb']
Hasil tokenisasi: ['kapan', 'dimulainya', 'pendaftaran', '?']
Hasil tokenisasi: ['pendaftaran', 'dibuka', 'sampai', 'kapan', '?']
Hasil tokenisasi: ['jadwal', 'penerimaan', 'siswa', 'baru', '?']
Hasil tokenisasi: ['jam', 'berapa', 'pendaftaran', 'dibuka', '?']
Hasil tokenisasi: ['jam', 'berapa', 'saya', 'bisa', 'mulai', 'daftar', '?']
Hasil tokenisasi: ['pendaftaran', 'di', 'tutup', 'jam', 'berapa', '?']
Hasil tokenisasi: ['penutupan', 'pendaftaran', 'di', 'jam', 'berapa', '?']
Hasil tokenisasi: ['jam', 'buka', 'pendaftaran']
Hasil tokenisasi: ['jam', 'tutup', 'pendaftaran']
Hasil tokenisasi: ['apa', 'syarat', 'pendaftaran', 'sekolah', '?']
Hasil tokenisasi: ['dokumen', 'apa', 'saja', 'yang', 'dibutuhkan', '?']
Hasil tokenisasi: ['syarat', 'pendaftaran', 'apa', 'saja', '?']
Hasil tokenisasi: ['apa', 'yang', 'harus', 'disiapkan', 'untuk', 'mendaftar', '?']
Hasil tokenisasi: ['berkas', 'pendaftaran']
Hasil tokenisasi: ['apa', 'saja', 'persyaratan', 'untuk', 'mendaftar', '?']
Hasil tokenisasi: ['dokumen', 'apa', 'yang', 'perlu', 'dibawa', 'saat', 'pendaftaran', '
```

Gambar 4. 8 Hasil Tokenizing

Hasil tokenisasi pada gambar 4.2 menunjukkan proses memecah setiap kalimat menjadi unit-unit kecil yang disebut token, di mana setiap kata dan tanda baca seperti "?" dipisahkan sebagai token individual. Contohnya, kalimat "kapan jadwal pendaftaran dibuka?" diubah menjadi ['kapan', 'jadwal', 'pendaftaran', 'dibuka', '?'], dan pola serupa terlihat pada kalimat lainnya.

c. *Stopword Removal*

Tahapan selanjutnya melakukan *Stopword removal* untuk menghapus kata kata yang dianggap tidak penting dan mengabaikan tanda baca seperti “ ? ”, ” ! ”, ” . ”, “ , ”. berikut kode program dari *Stopword removal* pada gambar 4.5.

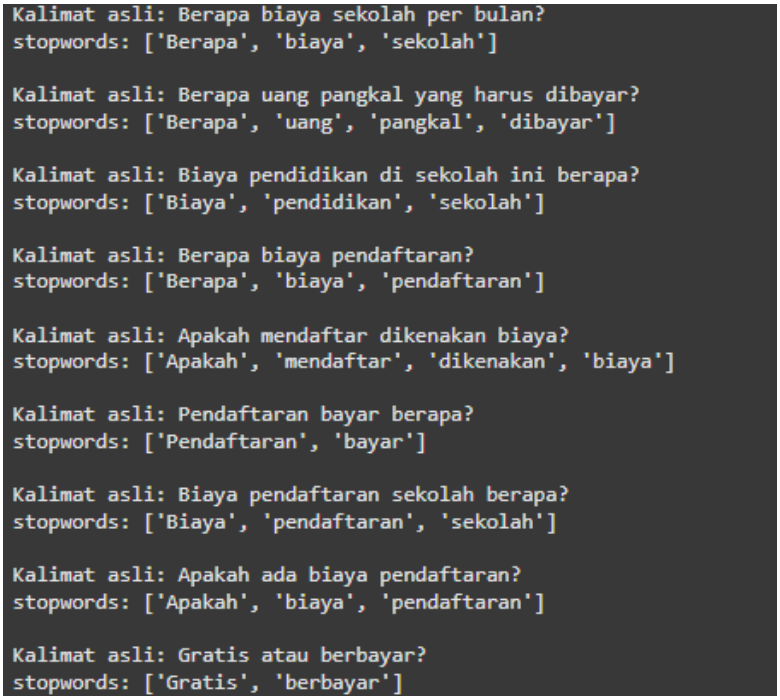
```
from nltk.corpus import stopwords

stop_words = set(stopwords.words('indonesian'))
ignoreLetters = ['?', '!', '.', ',']

def remove_stopwords(wordList, stop_words, ignoreLetters):
    return [word for word in wordList if word not in ignoreLetters and word not in stop_words]
```

Gambar 4. 9 Kode Program *Stopword Removal*

Pada gambar 4.5 mendefinisikan fungsi remove_stopwords yang bertujuan untuk membersihkan suatu list kata dari *stop words* bahasa Indonesia dan tanda baca tertentu (?', '!', '.', ',') dengan memanfaatkan modul stopwords dari nltk.corpus, sehingga menghasilkan list kata yang lebih penting untuk analisis teks. Pada gambar 4.3 menunjukan hasil dari *stopword removal.*



Gambar 4. 10 Hasil *Stopword Removal*

Hasil output pada gambar menunjukkan proses filtering yang menyeleksi kata-kata penting dari kalimat asli dengan menghapus kata-kata umum yang dianggap kurang relevan. Kalimat asli seperti "Berapa biaya sekolah per bulan?" menghasilkan kata-kata penting ['Berapa', 'biaya', 'sekolah'], di mana kata-kata seperti "per" dan "bulan" dihapus karena dianggap tidak signifikan, sementara kata kunci utama tetap dipertahankan.

d. Stemming

Stemming adalah proses mengubah kata berimbuhan ke bentuk dasarnya tanpa harus menjadi akar kata (root word). Sebagai contoh, kata seperti "mendengarkan," "dengarkan," dan "didengarkan" akan diubah menjadi "dengar." (Nugroho, 2019).

```python
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory

factory = StemmerFactory()
stemmer = factory.create_stemmer()

def stem_words(wordList, stemmer):
    return [stemmer.stem(word) for word in wordList]
```

Gambar 4. 11 Kode Program *Stemming*

Penelitian ini menggunakan *library* Sastrawi untuk melakukan stemming kata dalam bahasa Indonesia. Proses *stemming* dilakukan dengan menginstansiasi kelas StemmerFactory dan membuat objek *stemmer*. Fungsi *stem_words* yang didefinisikan kemudian digunakan untuk melakukan *stemming* pada setiap kata dalam list kata. Hasil dari proses *stemming* seperti gambar 4.4 .

```
Kalimat asli: Pendaftaran di tutup jam berapa ?
stemming: ['daftar', 'tutup', 'jam']

Kalimat asli: penutupan pendaftaran di jam berapa ?
stemming: ['tutup', 'daftar', 'jam']

Kalimat asli: Jam buka pendaftaran
stemming: ['jam', 'buka', 'daftar']

Kalimat asli: Jam tutup pendaftaran
stemming: ['jam', 'tutup', 'daftar']

Kalimat asli: Apa syarat pendaftaran sekolah?
stemming: ['apa', 'syarat', 'daftar', 'sekolah']
```

Gambar 4. 12 Hasil *Stemming*

3. Modeling

a. Bag of Word

Merubah hasil *Text Processing* menjadi bentuk index. Matriks yang dihasilkan dari proses ini digunakan sebagai input untuk model

pembelajaran mesin atau analisis teks lebih lanjut. Hasil output dari *Bag of Word* seperti pada tabel 4.2.

Tabel 4. 2 Hasil *Bag of Word*

Bag of Word
Pattern: ['formulir', 'daftar', 'mana'] Bag of Words: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

Output tersebut menunjukkan representasi *Bag of Words* untuk pola "Formulir pendaftaran ada dimana ?", yang telah diproses menjadi ['formulir', 'daftar', 'mana']. BoW adalah vektor biner yang menunjukkan keberadaan (1) atau ketidakhadiran (0) kata dalam kosakata. Pada pola ini, posisi kata 'formulir', 'daftar', dan 'mana' diisi dengan '1', sementara lainnya '0'. Representasi ini mempermudah model memahami data teks.

b. *Split Data*

Tahapan pemisahan data menggunakan pembagian 80% untuk data pelatihan dan 20% untuk data uji.

```
X_train, X_test, y_train, y_test = train_test_split(
    train_x, train_y, test_size=0.2, random_state=42, stratify=train_y
)

ros = RandomOverSampler(random_state=42)
X_resampled, y_resampled = ros.fit_resample(X_train, y_train)
```

Gambar 4. 13 *Source Code Split Data*

Gambar 4.13 Persiapan data sebelum digunakan untuk melatih model *machine learning*. Pertama, data dibagi menjadi data latih dan data uji menggunakan *train_test_split* untuk evaluasi performa model pada data yang belum pernah dilihat. Kemudian, *RandomOverSampler* digunakan untuk mengatasi ketidakseimbangan kelas dengan menduplikasi sampel dari kelas minoritas pada data latih, sehingga model dapat mempelajari pola dari kelas minoritas dengan lebih baik dan tidak bias terhadap kelas mayoritas.

c. Implementasi Algoritma ANN

Model *chatbot* ini dirancang menggunakan pendekatan *Artificial Neural Network* (ANN) dengan arsitektur *Multi-Layer Perceptron* (MLP). Struktur ANN ini terdiri dari dua *hidden layer Dense*, masing-masing dengan 128 dan 64 neuron, menggunakan fungsi aktivasi ReLU. Untuk mengurangi risiko *overfitting*, diterapkan dua *layer Dropout* dengan tingkat *dropout* sebesar 0.4. Pada *layer output*, digunakan *Dense layer* dengan fungsi aktivasi *softmax* untuk klasifikasi multi-kelas, di mana jumlah neuron pada *output layer* disesuaikan dengan jumlah kelas dalam data pelatihan. Model *Artificial Neural Network* dapat dilihat pada gambar 4.9.

```
model=Sequential()
model.add(Dense(128,activation='relu',input_shape=(len(X_resampled[0]),)))
model.add(Dropout(0.5))
model.add(Dense(64,activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(len(y_resampled[0]),activation='softmax'))
```

Gambar 4. 14 Kode Program Model ANN

Gambar 4.9 merupakan Kode program dari perancangan model *Artificial Neural Network.* Penjelasan dari kode program model diatas seperti berikut :

1) Inisialisasi model menggunakan Sequential API dari Keras, yang memungkinkan pemprosesan layer secara berurutan.
2) Layer pertama Lapisan Dense dengan 128 neuron ini menghubungkan seluruh fitur input ke setiap neuronnya, menggunakan fungsi aktivasi ReLU untuk memungkinkan jaringan mempelajari pola non-linear kompleks, dengan bentuk input yang sesuai dengan panjang vektor data pelatihan.
3) *Dropout layer* dengan *rate* 0.5 untuk mematikan 50% neuron selama proses pelatihan untuk menghindari *overfitting*.
4) *Hidden layer* ke 2 Lapisan Dense dengan 64 neuron berfungsi menyaring dan mengekstrak fitur-fitur penting dari data yang telah diolah sebelumnya, lalu menggunakan fungsi aktivasi ReLU untuk membantu model memahami pola-pola kompleks dalam data.
5) *Dropout layer* dengan rate 0.5 merupakan *dropout* pada layer ke 2 digunakan setelah *hidden layer 2.*

6) *Output layer Layer* dengan jumlah neuron mewakili satu kelas dalam klasifikasi. Fungsi aktivasi softmax menghasilkan probabilitas untuk setiap kelas, sehingga model dapat memprediksi kelas yang paling mungkin untuk input yang diberikan.

d. *Training model*

Setelah merancang model tahapan berikutnya adalah melakukan pelatihan pada model. Pelatihan model ini menggunakan *optimizer* SGD (*Stochastic Gradient Descent*). Penerapan *optimizer* SGD seperti pada gambar 4.10.

```
sgd=SGD(learning_rate=0.01, decay=1e-6, momentum=0.9, nesterov=True)
model.compile(loss='categorical_crossentropy', optimizer=sgd, metrics=['accuracy'])
history = model.fit(np.array(X_resampled),np.array(y_resampled),epochs=300, batch_size=10,verbose=1, validation_data=(X_test, y_test))
```

Gambar 4. 15 Kode Program *Training* model

Pada gambar 4.10 mendefinisikan optimizer SGD dengan learning rate 0.01 Menetapkan kecepatan pembaruan bobot dalam proses pelatihan, decay 1e-6 Mengurangi nilai learning rate secara bertahap untuk meningkatkan stabilitas pelatihan, momentum 0.9 Menambahkan momentum untuk mempercepat konvergensi dan mengurangi osilasi pada arah gradien, dan Nesterov diaktifkan, kemudian mengkompilasi model dengan fungsi kerugian *'categorical_crossentropy'*, *optimizer* SGD, dan metrik 'accuracy'. selanjutnya model dilatih menggunakan data train_x dan train_y

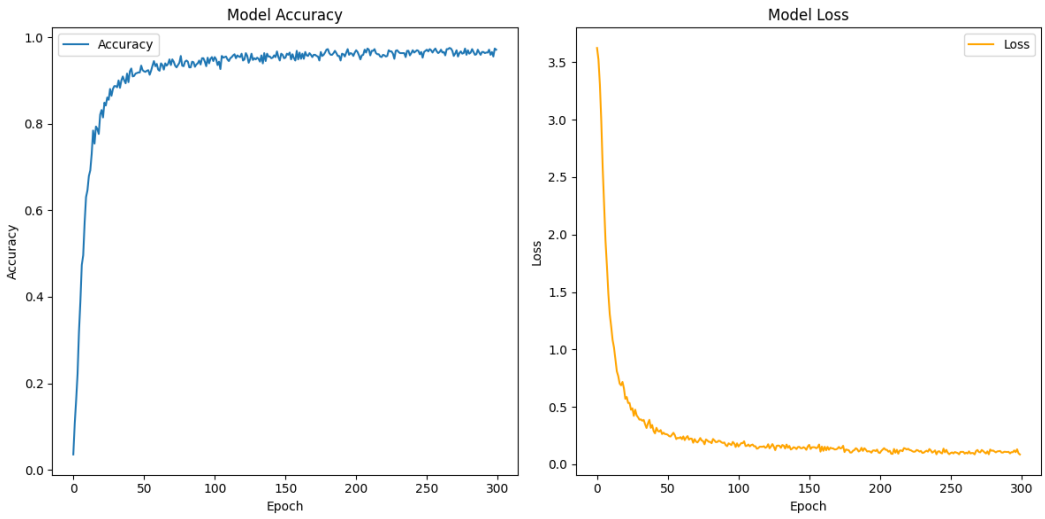
selama 300 epoch dengan batch size 10, dan menyimpan riwayat pelatihan dalam variabel history.

Pada penelitian dilakukan 4 kali pelatihan dengan ujicoba pada jumlah neuron *hidden layer* (32,16), (64, 32), dan (128, 64) Hasil pada uji coba pelatihan dilihat pada tabel 4.3.

Tabel 4. 3 Hasil Uji Coba Pada Jumlah Neuron

Jumlah neuron hidden layer	*Accuracy*	*Loss*
Neuron 32 dan 16	0.5156	1.4587
Neuron 64 dan 32	0.8217	0.5581
Neuron 128 dan 64	0.9702	0.0888

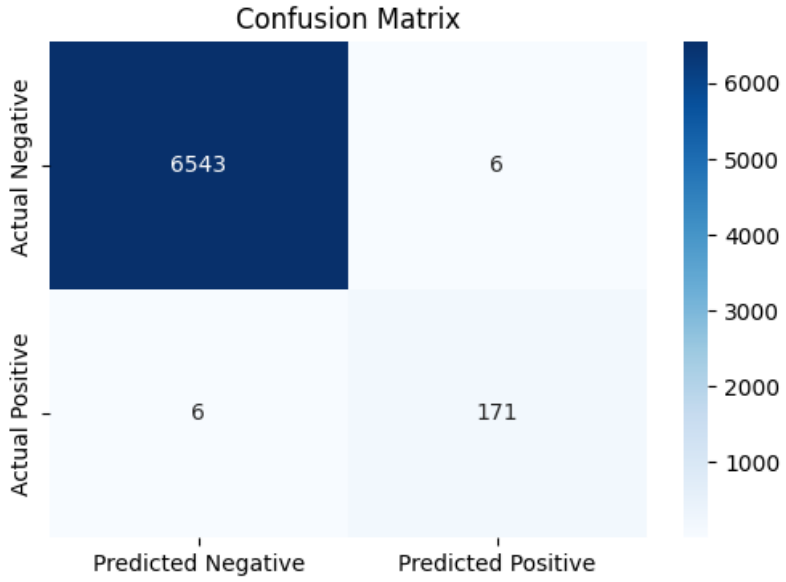
Hasil eksperimen menunjukkan bahwa peningkatan jumlah neuron pada lapisan tersembunyi berkorelasi positif dengan peningkatan akurasi model dan penurunan nilai *loss*. Konfigurasi dengan 32 dan 16 neuron pada lapisan tersembunyi belum mampu menangkap pola yang kompleks dalam data. Namun, dengan meningkatkan jumlah neuron menjadi 64 dan 32, kinerja model mengalami peningkatan signifikan. Konfigurasi dengan 128 dan 64 neuron menghasilkan performa terbaik, mendekati tingkat akurasi sempurna dan nilai *loss* yang sangat rendah.



Gambar 4. 16 Grafik Akurasi dan *Loss*

4. Evaluasi

Selanjut melakukan evaluasi model menggunakan metode *confusion matriks* untuk mencari kinerja model melalui hasil akurasi, presisi, recall, dan F1-score. Untuk melakukan evaluasi data dibagi menjadi 80 – 20, 80% untuk data training dan 20% untuk data uji.



Gambar 4. 17 Hasil *Confusion Matriks*

Pada gambar 4.13 menunjukan nilai *confusion matriks* dengan TP = 171, TN = 6543, FN = 6, FP = 6. selanjutnya akan dilakukan perhitungan untuk mencari nilai akurasi, *recall*, presisi, dan *f1-score*.

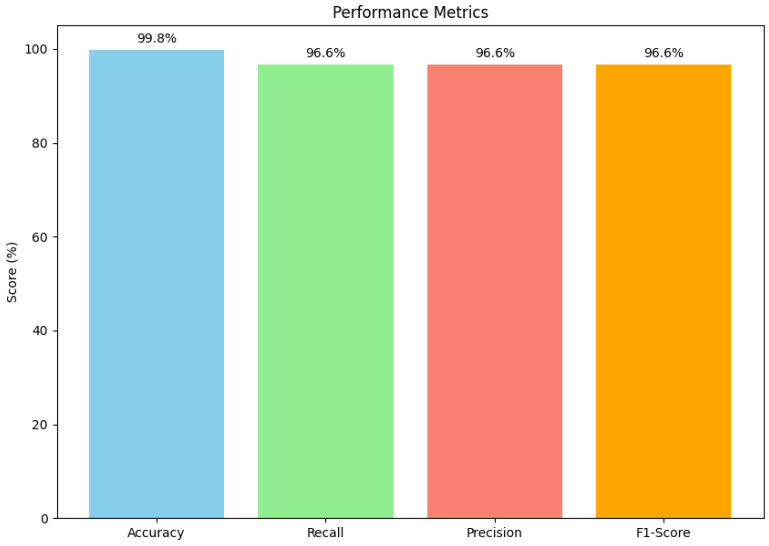
$$Akurasi = \frac{171 + 6543}{171 + 6543 + 6 + 6} = 0.998$$

$$Recal = \frac{171}{171 + 6} = 0.966$$

$$Persisi = \frac{171}{171 + 6} = 0.966$$

$$F1\ score = \frac{2 * (0.966 + 0.966)}{0.966 + 0.966} = 0.966$$

Dari hasil perhitungan ditas mendapatkan hasil akurasi, *recall*, F1-score, dan presisi pada *chatbot* menggunakan model *Artificial nural network* adalah 99,8% untuk akurasi, 96,6% untuk nilai *Recal*, 96,6% untuk nilai presisi, dan 96,6% untuk nilai *F1-score.* Visualisasi dari ke 4 matriks diatas dapat dilihat pada gambar 4.14.



Gambar 4. 18 Diagram Hasil Nilai *Performance Matriks*

Hasil akurasi, presisi, *F1-score* dan *recall* dapat dianggap menunjukkan kualitas klasifikasi yang baik atau tidak dengan mengacu pada pedoman evaluasi hasil klasifikasi pada tabel 4.3 (Gorunescu, 2011).

Tabel 4. 4 Pedoman parameter hasil klasifikasi

Rentang	**Hasil Klasifikasi**
90 – 100%	*Excellent classification*
80-90%	*Good classification*
70-80%	*Fair classification*
60-70%	*Poor classification*
50-60%	*Failure*

Hasil pelatihan model menunjukkan bahwa klasifikasi dataset menggunakan model Neural Network mendekati tingkat akurasi yang sempurna. Hal ini mengindikasikan bahwa model tersebut memiliki performa yang sangat baik dalam membedakan data dan mengelompokkannya ke dalam kategori atau kelas yang sesuai dengan benar.

5. Deployment

Pada tahap *deployment* ini, tujuan utamanya adalah untuk membangun antarmuka tampilan *website* PPDB dan *chatbot*. Pengembangan antarmuka ini menggunakan *Bootstrap* sebagai *frontend* dan *framework* Flask untuk membangun API *respon* dari untuk menciptakan antarmuka situs web yang menarik dan *respon* if.

a. Tahap pertama buat file projek dengan memasukan file seperti *chatbot*_model.h5, word.pkl, classes.pkl, dan intents.json yang merupakan hasil pelatihan model sebelumnya.

Gambar 4. 19 Struktur File

b. Pada app.py mengkode proses *chatbot* respon dan mempersiapkan *library* yang digunakan dan membuat *function* respon dan API nya menggunakan *framework* flask

```
from flask import Flask, request, jsonify
import random
import json
import pickle
import numpy as np
from keras.models import load_model
from flask_cors import CORS
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
import re
```

Gambar 4. 20 *Library* yang digunakan

Library yang digunakan mencakup Flask untuk membangun API *backend*, Flask-CORS untuk mendukung *Cross-Origin Resource Sharing*, Keras untuk memuat model *machine learning* yang telah dilatih (dengan fungsi load_model), NumPy untuk manipulasi array seperti pembentukan *bag-of-words*, Pickle untuk menyimpan dan memuat data serialisasi seperti *words*.pkl dan *classes*.pkl, serta Sastrawi untuk stemming kata-kata dalam bahasa Indonesia. Selain

itu, *library* random digunakan untuk memilih *respon* acak, json untuk membaca dan memproses file JSON, dan re untuk manipulasi teks dengan ekspresi reguler.

```python
# Chatbot logic functions
def clean_up_sentence(sentence):
    sentence_words = tokenize(sentence.strip())
    return [word for word in sentence_words if word.strip()]

def bag_of_words(sentence, words):
    sentence_words = tokenize(sentence.strip())
    print(f"Tokenisasi: {sentence_words}")
    sentence_words = [stemmer.stem(word.lower()) for word in sentence_words
    bag = [1 if word in sentence_words else 0 for word in words]
    # print(f"Bag of Words: {bag}")
    return np.array(bag)

def get_response(tag, intents_json):
    for intent in intents_json['intents']:
        if intent['tag'] == tag:
            return random.choice(intent['responses'])
    return "Maaf, saya tidak mengerti pertanyaan Anda."
```

Gambar 4. 21 *Source Code Logic Respon*

Kode program di atas adalah logika *chatbot* untuk memproses input pengguna. Fungsi *clean_up_sentence* membersihkan kalimat dengan tokenisasi menggunakan *tokenize* dan menghilangkan kata kosong. Fungsi *bag_of_words* mengubah kalimat menjadi representasi numerik berbasis *Bag of Words* dengan melakukan *stemming* dan mencocokkan kata dalam daftar kata (*words*). Fungsi *get_respon e* mengambil respon dari file intents_json berdasarkan tag yang diprediksi, dan jika tag tidak ditemukan, mengembalikan pesan default. Semua fungsi ini bekerja sama untuk memproses input teks dan menghasilkan respon yang relevan dari *chatbot*.

```python
def correct_typo(sentence, known_words, threshold=70):
    words = sentence.split()
    corrected_words = []
    for word in words:
        result = process.extractOne(word, known_words)
        if result:  # Pastikan hasil tidak kosong
            match, score, _ = result  # Ambil nilai kecocokan (match) dan skornya
            if score >= threshold:  # Jika skor kecocokan >= threshold
                corrected_words.append(match)
            else:
                corrected_words.append(word)
        else:
            corrected_words.append(word)  # Jika tidak ada hasil, gunakan kata asli
    return ' '.join(corrected_words)
```

Gambar 4. 22 *Source Code correct_typo*

Gambar 4.22 merupakan Fungsi *correct_typo* digunakan untuk memperbaiki *typo* dalam sebuah kalimat dengan membandingkan setiap kata dalam kalimat terhadap daftar kata yang dikenal (known_words). Fungsi ini memanfaatkan *process.extractOne* dari *library fuzzywuzzy* untuk mencari kata yang paling mirip dalam daftar kata yang dikenal, berdasarkan skor kecocokan. Jika skor kecocokan mencapai atau melebihi ambang batas *(threshold, default 70*), kata tersebut akan diganti dengan kata yang cocok jika tidak, kata aslinya dipertahankan. Kata-kata yang telah diperbaiki atau dipertahankan kemudian digabungkan kembali menjadi sebuah kalimat.

```python
def chatbot_response(text):
    # Koreksi typo pada input pengguna
    corrected_text = correct_typo(text, words)  # 'words' adalah daftar kata dari
    print(f"Input setelah koreksi: {corrected_text}")

    # Ubah input yang telah dikoreksi menjadi bag of words
    bow = bag_of_words(corrected_text, words)

    # Jika BoW kosong (tidak ada kata yang cocok), langsung respons default
    if not np.any(bow):  # Cek apakah semua elemen BoW adalah nol
        return "Maaf, saya tidak mengerti pertanyaan Anda. Silahkan hubungi admin

    prediction = model.predict(np.array([bow]))[0]  # Prediksi model
    max_prob = np.max(prediction)  # Probabilitas tertinggi
    predicted_class = classes[np.argmax(prediction)]  # Kelas yang diprediksi

    print(f"Kelas yang diprediksi: {predicted_class} (Akurasi: {max_prob:.6f})")

    # Logika respons berdasarkan threshold
    if max_prob > 0.5:  # Threshold tetap diatur di dalam fungsi
        response = get_response(predicted_class, intents)
        return response
    else:
        return "Maaf, saya tidak mengerti pertanyaan Anda."
```

Gambar 4. 23 *Source Code Function chatbot_respone*

Gambar 4.23 merupakan Fungsi *chatbot_response* bertugas memberikan respon *chatbot* berdasarkan input pengguna. Fungsi ini pertama-tama memperbaiki *typo* pada input menggunakan fungsi *correct_typo* dengan membandingkannya terhadap daftar kata yang dikenal (*words*). Setelah koreksi, input dikonversi menjadi representasi *bag of words* (BoW). Jika BoW kosong, *chatbot* langsung memberikan respon default. Jika tidak, input diprediksi oleh model, menghasilkan probabilitas prediksi tertinggi dan kelas yang diprediksi. Jika probabilitas prediksi melebihi ambang batas (0.5), *chatbot* memberikan respons yang sesuai berdasarkan data intents; jika tidak,

respon default tetap diberikan. Fungsi ini memastikan bahwa input yang mengandung typo tetap dapat diproses dengan baik oleh *chatbot*.

```python
@app.route('/get_response', methods=['POST'])
def chat_api():
    user_input = request.json.get("message")
    if not user_input:
        return jsonify({"response": "Silakan masukkan pesan valid."}), 400

    response = chatbot_response(user_input)
    return jsonify({"response": response})
```

Gambar 4. 24 *Source Code API Chatbot*

Fungsi *chat_api* adalah *endpoint* Flask untuk menerima input teks pengguna melalui POST *request*. Jika input kosong, fungsi mengembalikan respon *error*. Jika Berhasil, fungsi memproses input menggunakan *chatbot_respone* dan mengembalikan respon *chatbot* dalam format JSON.

c. Membuat tampilan menggunakan *bootstrap.*

```html
<!-- Chatbot Icon -->
<div class="chat-icon" id="chatbot-icon">
    💬
</div>

<!-- Chatbot Container -->
<div class="chat-container card" id="chat-container" style="display: none;">
    <div class="card-header bg-success text-white d-flex justify-content-between">
        <h5 class="mb-0">Chatbot</h5>
        <button id="close-chatbot" class="btn btn-sm btn-light">×</button>
    </div>
    <div class="card-body overflow-auto" id="chat-box" style="height: 350px;">
        <!-- Chat messages will appear here -->
    </div>

    <!-- Common Questions Row -->
 <div class="common-questions px-3 py-2 justify-content-md-center">
    <button class="btn btn-outline-secondary btn-sm d-inline-block question-btn">Cara daftar sekolah</button>
    <button class="btn btn-outline-secondary btn-sm d-inline-block question-btn">Syarat Pendaftaran</button>
    <button class="btn btn-outline-secondary btn-sm d-inline-block question-btn">Jadwal PPDB</button>
    <button class="btn btn-outline-secondary btn-sm d-inline-block question-btn">formulir pendaftaran</button>
</div>

    <div class="chat-footer d-flex align-items-center" id="chat-footer">
        <input type="text" id="user-input" placeholder="Ketik pesan Anda..." class="form-control" />
        <button id="send-btn" class="btn btn-success">Kirim</button>
    </div>
</div>
```

Gambar 4. 25 *Source Code Container*

Gambar 4.20 merupakan struktur HTML untuk antarmuka *chatbot*, terdiri dari ikon *chatbot* dengan ID *chatbot*-icon yang dapat diklik untuk membuka jendela *chatbot*. Jendela *chatbot* memiliki bagian header dengan tombol untuk menutup, sebuah area utama (*chat box*) untuk menampilkan pesan yang dapat digulir dengan tinggi 350px, dan tombol-tombol pertanyaan umum seperti "Cara daftar sekolah" dan "Syarat Pendaftaran" untuk membantu pengguna dengan cepat mengakses informasi. Bagian *footer* memungkinkan pengguna mengetikkan pesan di kolom input dan mengirimkannya melalui tombol “Kirim”

d. *Setting* API *chatbot* respon pada *javascript*

```
fetch("https://habdulghani-model-chatbot.hf.space/get_response", {
    method: "POST",
    headers: {
        "Content-Type": "application/json"
    },
    body: JSON.stringify({ message: userMessage })
})
.then(response => response.json())
.then(data => {
    // Remove waiting message
    chatBox.removeChild(waitingDiv);

    const botMessageDiv = document.createElement("div");
    botMessageDiv.className = "bot-message";

    // Replace \n with <br> to ensure new lines are rendered correctly
    let formattedResponse = data.response.replace(/\n/g, '<br>');

    // Check if there is a link and make it clickable
    formattedResponse = formattedResponse.replace(
        /(https?:\/\/[^\s]+)/g,
        '<a href="$1" target="_blank">$1</a>'
    );

    botMessageDiv.innerHTML = formattedResponse;

    chatBox.appendChild(botMessageDiv);

    // Scroll to the bottom
    chatBox.scrollTop = chatBox.scrollHeight;
})
```

Gambar 4. 26 *Source code Setting* API

Kode program diatas adalah untuk implementasi JavaScript untuk mengirim permintaan ke API *chatbot* menggunakan metode *fetch* dengan metode POST. Data dikirim dalam format JSON berisi pesan pengguna (*userMessage*). Setelah menerima respon dari API dalam format JSON, pesan bot akan diproses untuk menggantikan karakter *newline* (\n) dengan tag
 agar tampilan baris baru terlihat di HTML. Selain itu, jika respon berisi tautan (URL), tautan tersebut akan diubah menjadi tautan yang dapat diklik menggunakan elemen <a> dengan atribut target="*_blank*" untuk membukanya di tab baru. Pesan bot yang sudah diformat kemudian ditambahkan ke elemen *chat box* di halaman.

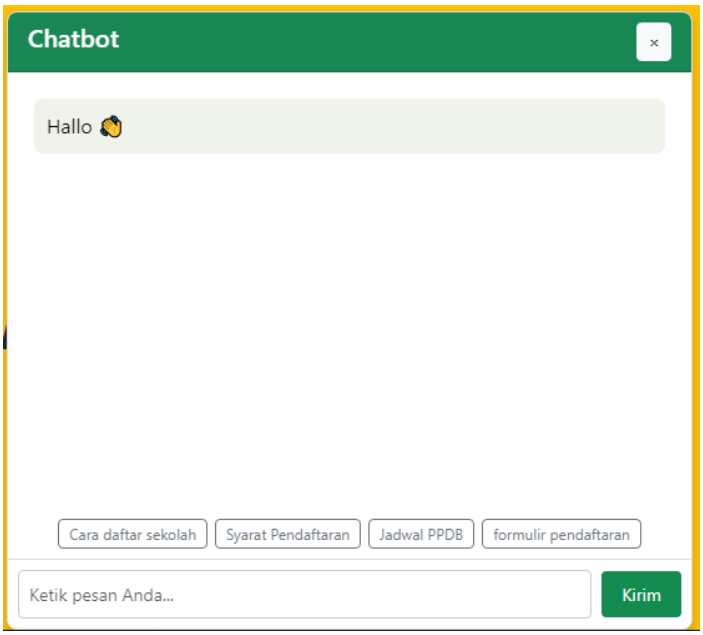
e. Hasil tampilan halaman depan *website*

Gambar 4. 27 Tampilan Awal *Website*

Halaman awal website ini dirancang menyerupai tampilan asli dari *website* PPDB resmi SMK Kesatrian Purwokerto, dengan tambahan fitur tombol chat di pojok kanan bawah. Tombol chat ini

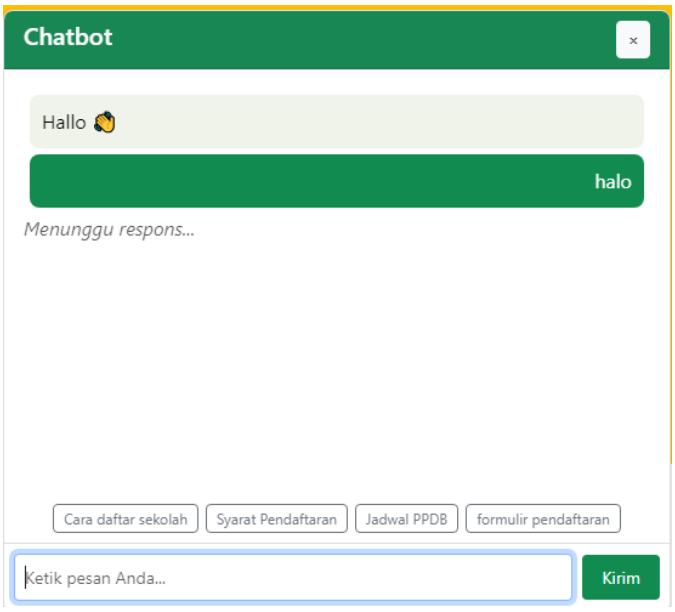
berfungsi untuk mengakses fitur *chatbot* yang bertujuan memfasilitasi pengujian dan interaksi dengan *chatbot* secara langsung.

f. Tampilan awal *chatbot*



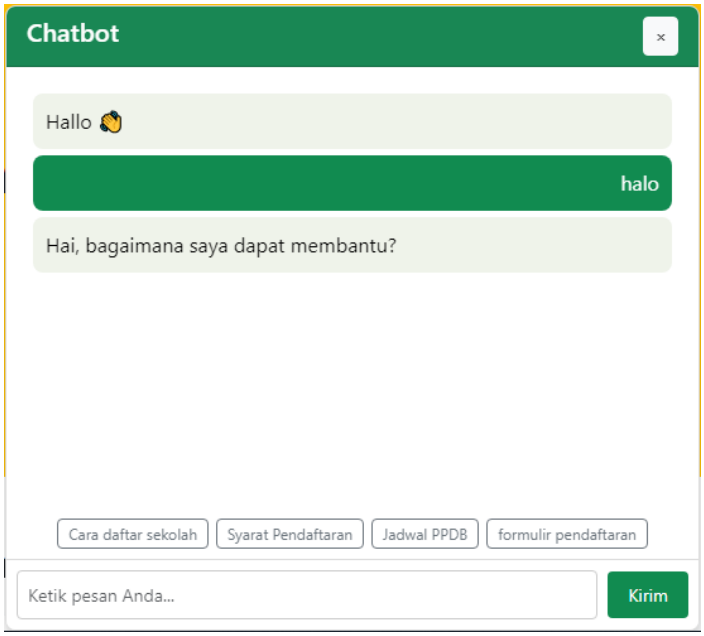
Gambar 4. 28 Tampilan Awal *Chatbot*

g. Tampilan *Chatbot* saat menunggu respon



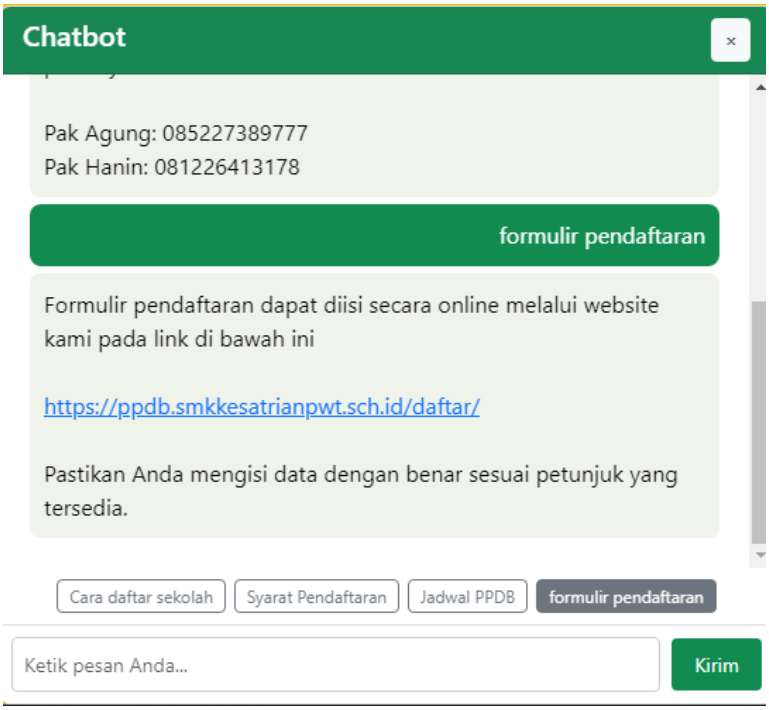
Gambar 4. 29 Tampilan *Chatbot* saat menunggu respon

h. Tampilan saat memberi respon



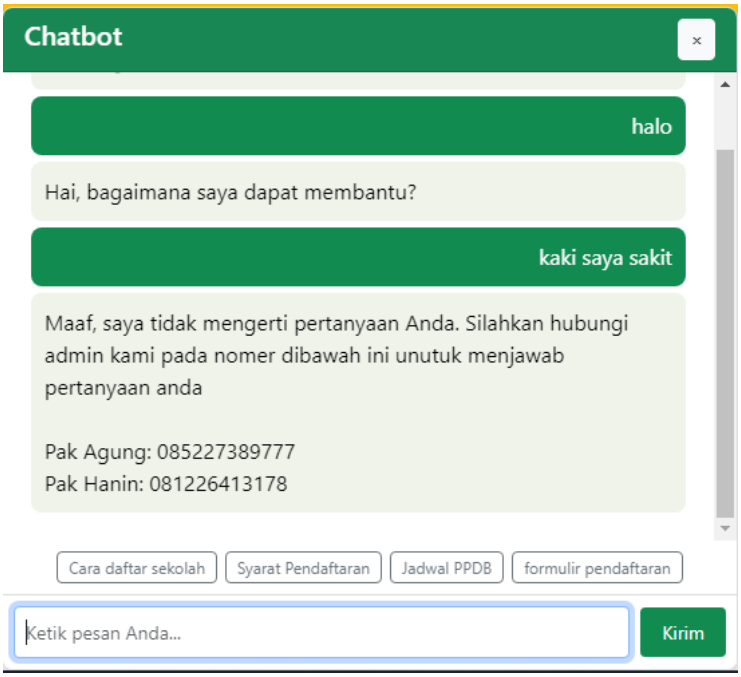
Gambar 4. 30 Tampilan *Chatbot* saat memberi respon

i. Tampilan respon dari *button* pertanyaan



Gambar 4. 31 Tampilan respon dari *button* pertanyaan

j. Tampilan inputan salah



Gambar 4. 32 Tampilan inputan salah

D. Pengujian

Tahapan pengujian *Blackbox Testing* bertujuan untuk memastikan bahwa aplikasi *chatbot* berfungsi sesuai dengan kebutuhan dan spesifikasi yang telah dirancang. Pengujian dilakukan dengan metode *alpha testing*, yang berfokus pada pengujian keluaran berdasarkan masukan tertentu tanpa memeriksa kode internal aplikasi, untuk memastikan respon yang dihasilkan sesuai dengan skenario pengujian. Selain itu, dilakukan juga *beta testing* secara langsung oleh pengguna akhir untuk mengevaluasi sejauh mana *chatbot* dapat membantu pengguna dalam mencari informasi PPDB secara efektif dan memberikan pengalaman yang memuaskan.

1. *Alpha Testing*

Skenario pengujian ini dengan menggunakan salah satu pattern pada setiap tag yang ada pada dataset. Jumlah tag pada daset sejumlah 35 tag yang berisi konteks pertanyaan terkait PPDB Pengujian ini menggunakan 37 pertanyaan inputan sesuai tag dan 2 inputan *error* hasil pengujian *balckbox* dapat dilihat pada tabel 4.

Tabel 4. 5 Pengujian *Blackbox*

No	Input pertanyan	Respon tag	Hasil respon yang diharapkan	Keterangan
1.	Apa kabar	*greeting*	*greeting*	Berhasil
2.	Terimakasih	*thanks*	*thanks*	Berhasil
3.	Assalamualaikum	salam	salam	Berhasil
4.	Apa yang bisa kamu lakukan?	*chatbot_can_do*	*chatbot_can_do*	Berhasil
5.	baiklah	*agree*	*agree*	Berhasil
6.	Kapan jadwal pendaftaran dibuka?	jadwal_pendaftaran	jadwal_pendaftaran	
7.	Jam berapa pendaftaran dibuka ?	jambuka_tutup	jambuka_tutup	Berhasil
8.	Apa syarat pendaftaran sekolah?	syarat_pendaftaran	syarat_pendaftaran	Berhasil
9.	cara mengisi formulir pendaftaran ?	formulir_pendaftaran	formulir_pendaftaran	Berhasil
10.	Biaya spp	spp_dan_uang_masuk	spp_dan_uang_masuk	Berhasil
11.	Biaya pendaftaran	biaya_pendaftaran	biaya_pendaftaran	Berhasil
12.	Jurusan yang dibuka	jurusan_yang_dibuka	jurusan_yang_dibuka	Berhasil
13.	Di mana lokasi sekolah?	lokasi_sekolah	lokasi_sekolah	Berhasil
14.	Fasilitas sekolah ?	fasilitas_sekolah	fasilitas_sekolah	Berhasil
15.	Akreditasi sekolah apa?	akreditasi_sekolah	akreditasi_sekolah	Berhasil
16.	Nilai rata-rata	nilai_pendaftaran	nilai_pendaftaran	Berhasil

17.	Apa itu jurusan DKV ?	jurusan_desain_komunikasi_visual	jurusan_desain_komunikasi_visual	Berhasil
18.	apa itu tkr ?	jurusan_teknik_kendaraan_ringan	jurusan_teknik_kendaraan_ringan	Berhasil
19.	Apa itu jurusan TSM?	jurusan_teknik_sepeda_motor	jurusan_teknik_sepeda_motor	Berhasil
20.	Apa itu TAV	jurusan_teknik_audio_video	jurusan_teknik_audio_video	Berhasil
21.	Apa iru TKJ	jenjang_karir_tkj	jurusan_teknik_komputer_dan_jaringan	Tidak berhasil
22.	Bagaimana cara mendaftar?	alur_pendaftaran	alur_pendaftaran	Berhasil
23.	cara daftar ulang ?	daftar_ulang	daftar_ulang	Berhasil
24.	Apa saja berkas yang di butuhkan untuk daftar ulang ?	berkas_daftar_ulang	berkas_daftar_ulang	Berhasil
25.	Kapan pengumuman hasil seleksi?	pengumuman_diterima	pengumuman_diterima	berhasil
26.	ukuran pas foto	ukuran_pas_foto	ukuran_pas_foto	Berhasil
27.	visi misi	visi_misi_sekolah	visi_misi_sekolah	Berhasil
28.	berapa jumlah siswa sekarang	jumlah_siswa_aktif	jumlah_siswa_aktif	Berhasil
29.	Informasi beasiswa	beasiswa	beasiswa	Berhasil
30.	Prospek kerja DKV?	jenjang_karir_dkv	jenjang_karir_dkv	Berhasil
31.	Prospek kerja TAV?	jenjang_karir_tav	jenjang_karir_tav	Berhasil
32.	Prospek kerja TKJ	jenjang_karir_tkj	jenjang_karir_tkj	Berhasil
33.	Prospek kerja TSM?	jenjang_karir_tsm	jenjang_karir_tsm	Berhasil
34.	Berapa biaya per jurusan?	biaya_perjurusan	biaya_perjurusan	Berhasil
35.	Apa itu PPDB?	apa_itu_ppdb	apa_itu_ppdb	Berhasil
36.	Prospek kerja TKR?	jenjang_karir_tkr	jenjang_karir_tkr	Berhasil
37.	Prestasi sekolah	prestasi_sekolah	prestasi_sekolah	Berhasil
38.	Kaki saya sakit	error	error	Berhasil
39.	akwjadja	error	error	Berhasil

Dari hasil pengujian *blackbox* dapat dilihat *chatbot* memberikan 39 respon sesuai prediksi dan 1 gagal. Selanjutnya menghitung akurasi dari hasil pengujian dengan menghitung jumlah prediksi benar dibagi jumlah keseluruhan pengujian (Rina, 2023).

$$Akurasi = \frac{Jumlah\ prediksi\ benar}{jumlah\ seluruh\ pengujian}$$

$$Akurasi = \frac{38}{39} = 0.9743$$

Hasil akurasi dari pengujian *blackbox* mendapatkan hasil 97%. Ini menandakan bahwa *chatbot* dapat memberikan respon yang sesuai dengan inputan pertanyaan dari pengguna.

2. *Beta Testing*

Metode pengujian beta akan dilakukan dengan membagikan kuesioner kepada 30 responden siswa/siswa kelas 10 SMK Kesatrain Purwokerto untuk mendapatkan umpan balik langsung mengenai pengalaman mereka menggunakan *chatbot*. Menurut Sugiyono (2019) Ukuran sampel yang layak dalam penelitian adalah antara 30 sampai dengan 500 responden, yang berarti sampel yang digunakan dalam penelitian ini sudah memenuhi kriteria representasi yang memadai.

Skala penilaian dari pertanyaan pada kuesioner menggunakan skala *likert.* Menurut (Sugiyono, 2019) Skala *Likert* adalah alat yang digunakan untuk mengukur sikap, pandangan, dan persepsi individu atau kelompok terhadap suatu fenomena sosial.

Tabel 4. 6 Tabel Skala Penilaian

Titik Respon	Bobot
Sangat Setuju (SS)	5
Setuju (S)	4
Cukup (C)	3
Tidak Setuju (TS)	2
Sangat Tidak Setuju (STS)	1

Skala penlilaian pada kuesioner seperti pada tabel 4.6 dimana respon sangat setuju (SS) mendapatkan bobot 5, Setuju (S) bobot 4, Cukup (C), Tidak Setuju (TS) bobot 4, dan sangat tidak setuju (STS) bobot 1.

Tabel 4. 7 Table Hasil Kuesioner

No	Pertanyaan	SS	S	C	TS	STS
1	Menurut Anda Setelah menggunakan *chatbot* ini, apakah dapat membantu memberikan Informasi PPDB?	15	15	0	0	0
2	Setelah menggunakan *chatbot* ini, apakah *chatbot* ini mempermudah dalam mendapatkan informasi?	14	15	1	0	0
3	Menurut Anda Informasi/jawaban yang diberikan *chatbot* sudah sesuai dengan yang ditanyakan?	10	15	5	0	0
4	Menurut Anda Apakah *chatbot* ini efisien dalam mencari informasi PPDB?	13	14	3	0	0
5	Menurut Anda apakah mudah dalam melakukan interaksi terhadap *chatbot* ?	10	13	6	0	0
6	Menurut Anda Apakah jawaban dari *chatbot* mudah dipahami?	14	13	3	0	0
7	Menurut Anda Apakah perlu pembaruan untuk menyempurnakan Aplikasi *chatbot* ini?	16	10	3	1	0

Tabel 4.7 merupakan hasil dari kuesioner yang di isi oleh 30 responden setelah mencoba menggunakan *chatbot*. Kuesioner berisikan pertanyaan terkait Efektivitas, Kemudahan, Akurasi, Kemudahan interaksi, dan kejelasan respon.

Untuk menghitung *indeks* setiap pertanyaan dapat dihitung dengan rumus :

$$Y = \frac{P}{Q} * 100\%$$

Y= Nilai *indeks*
P = Total skor
Q = Skor Tertinggi (jumlah responden * 5)

1) Menurut Anda Setelah menggunakan *chatbot* ini, apakah dapat membantu memberikan Informasi PPDB?

Tabel 4. 8 *indeks* Pertanyaan 1

Keterangan	Nilai	Jawaban	Skor	Persentase
Sangat Setuju	5	15	75	90%
Setuju	4	15	60	
Cukup	3	0	0	
Tidak Setuju	2	0	0	
Sangat Tidak Setuju	1	0	0	
Jumlah		30	135	

Tabel 4.8 meunjukan bahwa hasil perhitungan pertanyaan pertama mendapatkan 90% responden responden merasa *chatbot* efektif dalam membantu mereka mendapatkan informasi terkait PPDB.

2) Setelah menggunakan *chatbot* ini, apakah *chatbot* ini mempermudah dalam mendapatkan informasi?

Tabel 4. 9 *indeks* Pertanyaan 2

Keterangan	Nilai	Jawaban	Skor	Persentase
Sangat Setuju	5	14	70	89%
Setuju	4	15	60	
Cukup	3	1	3	
Tidak Setuju	2	0	0	
Sangat Tidak Setuju	1	0	0	
Jumlah		30	133	

Tabel 4.9 menunjukan hasil perhitungan pertanyaan kedua mendapatkan 89%, Hal ini menunjukkan bahwa *chatbot* telah dirancang dengan baik untuk memberikan pengalaman pengguna yang praktis dan efisien.

3) Menurut Anda Informasi/jawaban yang diberikan *chatbot* sudah sesuai dengan yang ditanyakan?

Tabel 4. 10 *indeks* Pertanyaan 3

Keterangan	Nilai	Jawaban	Skor	Persentase
Sangat Setuju	5	10	50	83%
Setuju	4	15	60	
Cukup	3	5	15	
Tidak Setuju	2	0	0	
Sangat Tidak Setuju	1	0	0	
Jumlah		30	125	

Tabel 4.10 menunjukan hasil perhitungan pertanyaan ketiga mendapatkan 83 %, menunjukkan bahwa sebagian besar pengguna merasa bahwa informasi atau jawaban yang diberikan oleh *chatbot* sudah sesuai dengan yang ditanyakan.

4) Menurut Anda Apakah *chatbot* ini efisien dalam mencari informasi PPDB?

Tabel 4. 11 *indeks* Pertanyaan 4

Keterangan	Nilai	Jawaban	Skor	Persentase
Sangat Setuju	5	13	65	87%
Setuju	4	14	56	
Cukup	3	3	9	
Tidak Setuju	2	0	0	
Sangat Tidak Setuju	1	0	0	
Jumlah		30	130	

Tabel 4.11 menunjukan hasil dari perhitungan pertanyaan keempat mendapatkan hasil 87%, Menunjukkan bahwa *chatbot* dianggap cukup efisien dalam mencari informasi PPDB oleh sebagian besar pengguna.

5) Menurut Anda apakah mudah dalam melakukan interaksi terhadap *chatbot* ?

Tabel 4. 12 *indeks* Pertanyaan 5

Keterangan	Nilai	Jawaban	Skor	Persentase
Sangat Setuju	5	10	50	80%
Setuju	4	13	52	
Cukup	3	6	18	
Tidak Setuju	2	0	0	
Sangat Tidak Setuju	1	0	0	
Jumlah		30	120	

Tabel 4.12 menunjukan bahwa pertanyaan kelima mendapatkan persentase 80% menunjukkan bahwa mayoritas pengguna merasa bahwa interaksi dengan *chatbot* cukup mudah.

6) Menurut Anda Apakah jawaban dari *chatbot* mudah dipahami ?

Tabel 4. 13 *indeks* Pertanyaan 6

Keterangan	Nilai	Jawaban	Skor	Persentase
Sangat Setuju	5	14	70	87%
Setuju	4	13	52	
Cukup	3	3	9	
Tidak Setuju	2	0	0	
Sangat Tidak Setuju	1	0	0	
Jumlah		30	131	

Tabel 4.13 menunjukan bahawa hasil perhitungan indeks pertanyaan keenam mendapatkan 87% menunjukkan bahwa sebagian besar pengguna merasa bahwa jawaban yang diberikan oleh *chatbot* mudah dipahami.

7) Menurut Anda Apakah perlu pembaruan untuk menyempurnakan Aplikasi *chatbot* ini?

Tabel 4. 14 *indeks* Pertanyaan 7

Keterangan	Nilai	Jawaban	Skor	Persentase
Sangat Setuju	5	16	80	87%
Setuju	4	10	40	
Cukup	3	3	9	
Tidak Setuju	2	1	2	
Sangat Tidak Setuju	1	0	0	
Jumlah		30	131	

Tabel 4.14 menunjukan dari hasil perhitungan indeks pertanyaan ketujuh mendapatkan 87% menunjukkan bahwa sebagian besar pengguna merasa bahwa *chatbot* sudah cukup baik, namun ada beberapa yang mungkin merasa perlu adanya pembaruan untuk meningkatkan kinerjanya.

Dari perhitungan *indeks* sebelumnya didapatkan nilai masing masing indeks dari hasil pertanyaan kuesioner. Nilai indeks yang sudah didapatkan selanjutnya di hitung rata-ratanya yang dimana akan dicari nilai interval persentase untuk mengetahui kepuasan pengguna terhadap *chatbot.*

$$Rata - rata = \frac{90 + 88{,}6 + 83{,}3 + 86{,}6 + 80 + 87{,}3 + 87{,}3}{7} = 86{,}19\%$$

Tabel 4. 15 Tabel Interpretasi Interval

Interval	Keterangan
0% - 19,99%	Sangat Buruk
20% - 39,99%	Kurang Baik
40% - 59,99%	Cukup
60% - 79,99%	Baik
80% - 100%	Sangat Baik

Berdasarkan hasil perhitungan rata-rata indeks dan mengacu pada tabel 4.15 mengenai interpretasi interval. Mendapatkan rata-rata mencapai 86,19% dapat disimpulkan bahwa *chatbot* PPDB secara keseluruhan memberi kinerja yang sangat baik.

BAB V
PENUTUP

A. Kesimpulan

Berdasarkan penelitian yang telah dilakukan mengenai pengembangan *chatbot* untuk layanan informasi Penerimaan Peserta Didik Baru (PPDB) di SMK Kesatrian Purwokerto, dapat disimpulkan hal-hal berikut:

1. Pengimplementasian model ANN pada *chatbot* berhasil memberikan informasi terkait PPDB secara cepat dan akurat, dengan tingkat akurasi model mencapai 97% selama pelatihan dan 97% dalam pengujian *blackbox alpha testing.*
2. Hasil *beta testing* dengan rata-rata kepuasan sebesar 86,19% menunjukkan *chatbot* mudah digunakan, baik dalam hal interaksi maupun pemahaman terhadap jawaban yang diberikan.
3. Pengolahan teks (*case folding, tokenizing, stopword removal, stemming*) secara signifikan meningkatkan kualitas data untuk pelatihan *chatbot*.
4. Evaluasi menunjukkan bahwa model memiliki akurasi 99% berdasarkan *confusion matrix*, dengan presisi, *recall*, dan *F1-score* masing-masing sebesar 96,6%.

B. Saran

Untuk pengembangan lebih lanjut, beberapa saran yang dapat diberikan adalah sebagai berikut :

1. Menambah jumlah dan variasi dataset untuk mencakup lebih banyak pola pertanyaan sehingga respons *chatbot* menjadi lebih kaya dan akurat.
2. Menambahkan fitur dashboard admin agar admin dapat menambah pertanyaan dan mengatur *chatbot* dengan lebih mudah.
3. Melakukan *hyperparameter tuning* lebih mendalam untuk memaksimalkan performa model.

Dengan menerapkan saran-saran tersebut, diharapkan *chatbot* dapat menjadi solusi yang lebih efektif dan memberikan kontribusi yang signifikan dalam mendukung layanan PPDB di SMK Kesatrian Purwokerto.

DAFTAR PUSTAKA

Arbizal, Y., Nurina Sari, B., & Garno. (2024). IMPLEMENTASI ALGORITMA ARTIFICIAL NEURAL NETWORK PADA CHATBOT WEBSITE PRODI INFORMATIKA UNSIKA. *Jurnal informasi dan Komputer*, *12*(1), 2024.

Bhashkar, K. (2019). *Conversational AI Chatbot using Deep Learning: How Bi-directional LSTM, Machine Reading Comprehension, Transfer Learning, Sequence to Sequence Model with multi-headed attention mechanism, Generative Adversarial Network, Self Learning based Sentiment Analysis and Deep Reinforcement Learning can help in Dialog Management for Conversational AI chatbot*. https://bhashkarkunal.medium.com/conversational-ai-chatbot-using-deep-learning-how-bi-directional-lstm-machine-reading-38dc5cf5a5a3.

Enterprise, J. (2019). *Python untuk Programmer Pemula*. Elex media komputindo.

Fadhallah. (2021). *WAWANCARA*. UNJ PRESS.

Fahmi Yusron Fiddin, Komarudin, A., & Melina, M. (2024). Chatbot Informasi Penerimaan Mahasiswa Baru Menggunakan Metode FastText dan LSTM. *Journal of Applied Computer Science and Technology*, *5*(1), 33–39. https://doi.org/10.52158/jacost.v5i1.648

Faurina, R., Revanza, D., & Sopran, A. (2023). Pengembangan Chatbot Menggunakan Deep Feed-Forward Neural Network sebagai Pusat Layanan Informasi Akademik. *Jurnal Eksplora Informatika*, *11*(2), 120–129. https://doi.org/10.30864/eksplora.v11i2.833

Gorunescu, F. (2011). *Data Mining Concepts, Models and Techniques*. Springer Berlin Heidelberg.

Harahap, R. N., & Muslim, K. (2020). PENINGKATAN AKURASI PADA PREDIKSI KEPRIBADIAN MBTI PENGGUNA TWITTER MENGGUNAKAN AUGMENTASI DATA. *Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK)*. https://doi.org/10.25126/jtiik.202073622

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

Iskandar, D., & Sriharyani, L. (2021). SOFT COMPUTING PENILAIAN KONDISI PERKERASAN JALAN BERBASIS ARTIFICIAL NEURAL

NETWORKS. *TAPAK (Teknologi Aplikasi Konstruksi) : Jurnal Program Studi Teknik Sipil*, *10*(2), 148. https://doi.org/10.24127/tp.v10i2.1584

Ivan, G., Hadi Asnal, Muhammad Nur Cahyadi, & Zaki Mubarok G. (2022). Perancangan Chatbot untuk Layanan Informasi Sekolah (Studi Kasus SMK Dwi Sejahtera Pekanbaru). *SATIN - Sains dan Teknologi Informasi*, *8*(2), 198–207. https://doi.org/10.33372/stn.v8i2.880

Krstinic, D., Braović, M., Šerić, L., & Božić-Štulić, D. (2020). *Multi-label Classifier Performance Evaluation with Confusion Matrix*. 1–14. https://doi.org/10.5121/csit.2020.100801

Kurniyawan, D. (2022). *Pengenalan Machine Learning dengan Python*. Elex Media Komputindo. https://books.google.co.id/books?id=ZutsEAAAQBAJ

Larasati, S., & Susetyo, Y. (2024). Development of a Web-Based Trading Term Application Using Flask Framework at PT. XYZ. *International Journal Software Engineering and Computer Science (IJSECS)*, *4*, 367–376. https://doi.org/10.35870/ijsecs.v4i1.2339

Ling, Q. (2023). Machine learning algorithms review. *Applied and Computational Engineering*, *4*(1), 91–98. https://doi.org/10.54254/2755-2721/4/20230355

Lubis, A., & Sumartono, I. (2023). RESOLUSI : Rekayasa Teknik Informatika dan Informasi Implementasi Layanan Akademik Berbasis Chatbot untuk Meningkatkan Interaksi Mahasiswa. *Media Online*, *3*(5). https://doi.org/https://doi.org/10.30865/resolusi.v3i5.767

Mustakim, F., Fauziah, & Hayati, N. (2021). Algoritma Artificial Neural Network pada Text-based Chatbot Frequently Asked Question (FAQ) Web Kuliah Universitas. *Jurnal Teknologi Informasi dan Komunikasi)*, *5*(4), 2021. https://doi.org/10.35870/jti

Nugraha, K. A., & Sebastian, D. (2021). Chatbot Layanan Akademik Menggunakan K-Nearest Neighbor. *Jurnal Sains dan Informatika*, *7*(1), 11–19. https://doi.org/10.34128/jsi.v7i1.285

Nugroho, K. S. (2019). *Dasar Text Preprocessing dengan Python*. https://ksnugroho.medium.com/dasar-text-preprocessing-dengan-python-a4fa52608ffe.

Nurhopipah, A., & Hasanah, U. (2020). Dataset Splitting Techniques Comparison For Face Classification on CCTV Images. *IJCCS (Indonesian Journal of*

Computing and Cybernetics Systems), *14*(4), 341. https://doi.org/10.22146/ijccs.58092

Nurul Puteri, A., Tamrin, F., Rahman Nasir, K., Widya Anggraeni, D., & Arafah, M. (2022). Aplikasi Chatbot untuk Layanan Informasi Penerimaan Mahasiswa Baru. *Seminar Nasional Teknik Elektro dan Informatika (SNTEI)*.

Paluang, P., Thavorntam, W., & Phairuang, W. (2024). Application of Multilayer Perceptron Artificial Neural Network (MLP-ANN) Algorithm for PM2.5 Mass Concentration Estimation during Open Biomass Burning Episodes in Thailand. *International Journal of Geoinformatics*, *20*(7), 28–42. https://doi.org/10.52939/ijg.v20i7.3401

Purnajaya, A. R., Lieputra, V., Tayanto, V., & Salim, J. G. (2022). Implementasi Text Mining untuk Mengetahui Opini Masyarakat Tentang Climate Change. *Journal of Information System and Technology*, *03*(03), 320–328.

Purwono, P., Dewi, P., Wibisono, S. K., & Dewa, B. P. (2022). Model Prediksi Otomatis Jenis Penyakit Hipertensi dengan Pemanfaatan Algoritma Machine Learning Artificial Neural Network. *Insect (Informatics and Security): Jurnal Teknik Informatika*, *7*(2), 82–90. https://doi.org/10.33506/insect.v7i2.1828

Ramadhani, S., Azzahra, D., & Z, T. (2022). Comparison of K-Means and K-Medoids Algorithms in Text Mining based on Davies Bouldin Index Testing for Classification of Student's Thesis. *Digital Zone: Jurnal Teknologi Informasi dan Komunikasi*, *13*(1), 24–33. https://doi.org/10.31849/digitalzone.v13i1.9292

Rina. (2023). *Memahami Confusion Matrix: Accuracy, Precision, Recall, Specificity, dan F1-Score untuk Evaluasi Model Klasifikasi*. https://esairina.medium.com/memahami-confusion-matrix-accuracy-precision-recall-specificity-dan-f1-score-610d4f0db7cf.

Rohmah, S., Wahyudi, W., & Pamungkas, F. (2021). Pengelolaan Penerimaan Peserta Didik Baru (PPDB) Berdasarkan Sistem Zonasi di SMP Negeri 1 Mlonggo Jepara. *Jawda: Journal of Islamic Education Management*, *1*(1), 25–34. https://doi.org/10.21580/jawda.v1i1.2020.6704

Rosad, A. M. (2019). IMPLEMENTASI PENDIDIKAN KARAKTER MELALUI MANAGEMEN SEKOLAH. *Tarbawi: Jurnal Keilmuan Manajemen Pendidikan*, *5*(02), 173. https://doi.org/10.32678/tarbawi.v5i02.2074

Sai, A. M. A., Balamurali, O., Karthikeya, M., & Anand, S. (2023). A Web-Based Chatbot for Indian Cities: A Comparison of CNN, ANN, and LSTM Models. *2023 14th International Conference on Computing Communication and Networking Technologies, ICCCNT 2023*. https://doi.org/10.1109/ICCCNT56998.2023.10307912

Sanjaya, M. O., Bukhori, S., & Furqon, M. `Ariful. (2023). Virtual Assistant for Thesis Technical Guide Using Artificial Neural Network. *Indonesian Journal of Artificial Intelligence and Data Mining*, *6*(2), 188. https://doi.org/10.24014/ijaidm.v6i2.23473

Sasongko, B. B., Malik, F., Ardiansyah, F., Rahmawati, A. F., Dharma Adhinata, F., & Rakhmadani, D. P. (t.t.). Pengujian Blackbox Menggunakan Teknik Equivalence Partitions pada Aplikasi Petgram Mobile. Dalam *Jurnal ICTEE* (Vol. 2, Nomor 1).

Sidik, M., Gunawan, B., Anggraini, D., & Korespondensi, P. (2021). PEMBUATAN APLIKASI CHATBOT KOLEKTOR DENGAN METODE EXTREME PROGRAMMING DAN STRATEGI FORWARD CHAINING. *Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK)*, *8*(2), 293–302. https://doi.org/10.25126/jtiik.202184298

Sugiyono. (2018). *Metode penelitian kuantitatif / Prof. Dr. Sugiyono* (Cet. 1). Alfabeta.

Sugiyono. (2019). *Metode Penelitian Kuantitatif Kualitatif dan R&D*. M. Dr. Ir. Sutopo. S. Pd. ALFABETA, cv.

Sujacka Retno, Rozzi Kesuma Dinata, & Novia Hasdyna. (2023). Evaluasi model data chatbot dalam natural language processing menggunakan k-nearest neighbor. *Jurnal CoSciTech (Computer Science and Information Technology)*, *4*(1), 146–153. https://doi.org/10.37859/coscitech.v4i1.4690

Suprapto, & Malik, A. . A. (2019). IMPLEMENTASI KEBIJAKAN DISKRESI PADA PELAYANAN KESEHATAN BADAN PENYELENGGARA JAMINAN KESEHATAN (BPJS). *Jurnal Ilmiah Kesehatan Sandi Husada*. https://akper-sandikarsa.e-journal.id

Syukri, S., & Samsuddin, S. (2019). Pengujian Algoritma Artificial Neural Network (ANN) Untuk Prediksi Kecepatan Angin. *Jurnal Nasional Komputasi dan Teknologi Informasi (JNKTI)*, *2*(1), 43. https://doi.org/10.32672/jnkti.v2i1.1056

Telaumbanua, F. D., Hulu, P., Nadeak, T. Z., Lumbantong, R. R., & Dharma, A. (2020). Penggunaan Machine Learning Di Bidang Kesehatan. *Jurnal Teknologi dan Ilmu Komputer Prima (JUTIKOMP)*, *2*(2). https://doi.org/10.34012/jutikomp.v2i2.657

Yudahana, A., Riadi, I., & Elvina, A. (2023). PERANCANGAN SISTEM INFORMASI PENDAFTARAN PESERTA DIDIK BARU (PPDB) BERBASIS WEB MENGGUNAKAN METODE RAPID APLLICATION DEVELOPMENT (RAD). *Rabit : Jurnal Teknologi dan Sistem Informasi Univrab*, *8*(1), 47–58. https://doi.org/10.36341/rabit.v8i1.2977

Yulia Puspaningrum, E., Satria Yudha, D. K., Wiji Utami, H., Vita Via, Y., Prakarsa Mandyartha, E., & Maulana, H. (2024). SIMAPA System Testing Using Alpha and Beta Tests. *NST Proceeding International Conference Partner*, 212–218. https://doi.org/http://dx.doi.org/10.11594/nstp.2024.4133

Zuraiyah, T. A., Utami, D. K., & Herlambang, D. (2019). IMPLEMENTASI CHATBOT PADA PENDAFTARAN MAHASISWA BARU MENGGUNAKAN RECURRENT NEURAL NETWORK. *Jurnal Ilmiah Teknologi dan Rekayasa*, *24*(2), 91–101. https://doi.org/10.35760/tr.2019.v24i2.2388

LAMPIRAN

Lampiran 1. Kartu Bimbingan

Lampiran 2. Surat Izin Penelitian

Lampiran 3. Dokumentasi Wawancara

Lampiran 4. Dokumentasi Pengujian

Lampiran 5. Brosur Sebagai Sumber dataset

Lampiran 6. Daftar Petanyaan Wawancara

Lampiran 7. *Source Code* Pembuatan Model Chatbot

```
from flask import Flask, request, jsonify
import random
import json
import pickle
import numpy as np
from keras.models import load_model
from flask_cors import CORS
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
import re
from rapidfuzz import process

factory = StemmerFactory()
stemmer = factory.create_stemmer()

# Load model dan preprocessing assets
model = load_model('chatbot_model.h5')
intents = json.loads(open('intents2.json').read())
words = pickle.load(open('words.pkl', 'rb'))
classes = pickle.load(open('classes.pkl', 'rb'))
ignoreLetters = ['?', '!', '.', ',']

app = Flask(__name__)
CORS(app)

def correct_typo(sentence, known_words, threshold=70):
    words = sentence.split()
```

```
    corrected_words = []
    for word in words:
        result = process.extractOne(word, known_words)
        if result:  # Pastikan hasil tidak kosong
            match, score, _ = result  # Ambil nilai kecocokan (match) dan 
skornya
            if score >= threshold:  # Jika skor kecocokan >= threshold
                corrected_words.append(match)
            else:
                corrected_words.append(word)
        else:
            corrected_words.append(word)  # Jika tidak ada hasil, gunakan kata 
asli
    return ' '.join(corrected_words)

# Fungsi tokenisasi menggunakan regex
def tokenize(sentence):
    # Tokenisasi dasar: ambil kata-kata alfanumerik
    return re.findall(r'\b\w+\b', sentence.lower())

# Chatbot logic functions
def clean_up_sentence(sentence):
    sentence_words = tokenize(sentence.strip())
    return [word for word in sentence_words if word.strip()]

def bag_of_words(sentence, words):
    sentence_words = tokenize(sentence.strip())
    print(f"Tokenisasi: {sentence_words}")

    sentence_words = [stemmer.stem(word.lower()) for word in sentence_words if 
word not in ignoreLetters]
    # print(f"Setelah stemming dan filtering: {sentence_words}")

    # for word in sentence_words:
    #     if word in words:
    #         print(f"Kata '{word}' ada pada words.")
    #     else:
    #         print(f"Kata '{word}' tidak ada pada words.")

    bag = [1 if word in sentence_words else 0 for word in words]
    # print(f"Bag of Words: {bag}")

    return np.array(bag)

def get_response(tag, intents_json):
    for intent in intents_json['intents']:
        if intent['tag'] == tag:
            return random.choice(intent['responses'])
    return "Maaf, saya tidak mengerti pertanyaan Anda."

def chatbot_response(text):
    # Koreksi typo pada input pengguna
    corrected_text = correct_typo(text, words)  # 'words' adalah daftar kata 
dari dataset
    print(f"Input setelah koreksi: {corrected_text}")

    # Ubah input yang telah dikoreksi menjadi bag of words
    bow = bag_of_words(corrected_text, words)

    # Jika BoW kosong (tidak ada kata yang cocok), langsung respons default
```

```
    if not np.any(bow):  # Cek apakah semua elemen BoW adalah nol
        return "Maaf, saya tidak mengerti pertanyaan Anda. Silahkan hubungi
admin kita terkait pertanyaan anda di bawah ini\n\nPak Agung: 085227389777\nPak
Hanin: 081226413178"

    prediction = model.predict(np.array([bow]))[0]  # Prediksi model
    max_prob = np.max(prediction)  # Probabilitas tertinggi
    predicted_class = classes[np.argmax(prediction)]  # Kelas yang diprediksi

    print(f"Kelas yang diprediksi: {predicted_class} (Akurasi: {max_prob:.6f})")

    # Logika respons berdasarkan threshold
    if max_prob > 0.5:  # Threshold tetap diatur di dalam fungsi
        response = get_response(predicted_class, intents)
        return response
    else:
        return "Maaf, saya tidak mengerti pertanyaan Anda."
# Route for chatbot response
@app.route('/get_response', methods=['POST'])
def chat_api():
    user_input = request.json.get("message")
    if not user_input:
        return jsonify({"response": "Silakan masukkan pesan valid."}), 400

    response = chatbot_response(user_input)
    return jsonify({"response": response})
```

Lampiran 8. Sourcode app.py

```
from flask import Flask, request, jsonify
import random
import json
import pickle
import numpy as np
from keras.models import load_model
from flask_cors import CORS
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
import re
from rapidfuzz import process

factory = StemmerFactory()
stemmer = factory.create_stemmer()

# Load model dan preprocessing assets
model = load_model('chatbot_model.h5')
intents = json.loads(open('intents2.json').read())
words = pickle.load(open('words.pkl', 'rb'))
classes = pickle.load(open('classes.pkl', 'rb'))
ignoreLetters = ['?', '!', '.', ',']

app = Flask(__name__)
CORS(app)

def correct_typo(sentence, known_words, threshold=70):
```

```
    words = sentence.split()
    corrected_words = []
    for word in words:
        result = process.extractOne(word, known_words)
        if result:  # Pastikan hasil tidak kosong
            match, score, _ = result  # Ambil nilai kecocokan (match) dan
skornya
            if score >= threshold:  # Jika skor kecocokan >= threshold
                corrected_words.append(match)
            else:
                corrected_words.append(word)
        else:
            corrected_words.append(word)  # Jika tidak ada hasil, gunakan kata
asli
    return ' '.join(corrected_words)

# Fungsi tokenisasi menggunakan regex
def tokenize(sentence):
    # Tokenisasi dasar: ambil kata-kata alfanumerik
    return re.findall(r'\b\w+\b', sentence.lower())

# Chatbot logic functions
def clean_up_sentence(sentence):
    sentence_words = tokenize(sentence.strip())
    return [word for word in sentence_words if word.strip()]

def bag_of_words(sentence, words):
    sentence_words = tokenize(sentence.strip())
    print(f"Tokenisasi: {sentence_words}")

    sentence_words = [stemmer.stem(word.lower()) for word in sentence_words if
word not in ignoreLetters]
    bag = [1 if word in sentence_words else 0 for word in words]
    return np.array(bag)

def get_response(tag, intents_json):
    for intent in intents_json['intents']:
        if intent['tag'] == tag:
            return random.choice(intent['responses'])
    return "Maaf, saya tidak mengerti pertanyaan Anda."

def chatbot_response(text):
    # Koreksi typo pada input pengguna
    corrected_text = correct_typo(text, words)  # 'words' adalah daftar kata
dari dataset
    print(f"Input setelah koreksi: {corrected_text}")

    # Ubah input yang telah dikoreksi menjadi bag of words
    bow = bag_of_words(corrected_text, words)

    # Jika BoW kosong (tidak ada kata yang cocok), langsung respons default
    if not np.any(bow):  # Cek apakah semua elemen BoW adalah nol
        return "Maaf, saya tidak mengerti pertanyaan Anda. Silahkan hubungi
admin kita terkait pertanyaan anda di bawah ini\n\nPak Agung: 085227389777\nPak
Hanin: 081226413178"

    prediction = model.predict(np.array([bow]))[0]  # Prediksi model
    max_prob = np.max(prediction)  # Probabilitas tertinggi
    predicted_class = classes[np.argmax(prediction)]  # Kelas yang diprediksi
```

```
    print(f"Kelas yang diprediksi: {predicted_class} (Akurasi: {max_prob:.6f})")

    # Logika respons berdasarkan threshold
    if max_prob > 0.5:  # Threshold tetap diatur di dalam fungsi
        response = get_response(predicted_class, intents)
        return response
    else:
        return "Maaf, saya tidak mengerti pertanyaan Anda."

# Route for chatbot response
@app.route('/get_response', methods=['POST'])
def chat_api():
    user_input = request.json.get("message")
    if not user_input:
        return jsonify({"response": "Silakan masukkan pesan valid."}), 400

    response = chatbot_response(user_input)
    return jsonify({"response": response})
```

Lampiran 9. Source Code Tampilan chatbot

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Chatbot</title>
   <link rel="icon" type="image/x-icon" href="Image/kesatrian (2).ico">
</head>
   <!-- Bootstrap CSS -->
   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">
   <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-
icons.min.css">
   <link rel="stylesheet" href="style.css">
</head>
<body>
   <header>
      <nav class="navbar navbar-expand-lg bg-dark py-3 fixed-top">
         <div class="container">
            <a class="fs-4 fw-bold navbar-brand text-white" href="#">PPDB ONLINE</a>
            <button class="navbar-toggler bg-success" type="button" data-bs-toggle="collapse" data-bs-
target="#navbarMenu" aria-controls="navbarMenu" aria-expanded="false" aria-label="Toggle navigation">
               <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse justify-content-center" id="navbarMenu">
```

```
            <ul class="navbar-nav mb-2 mb-lg-0 fw-bold">
              <li class="nav-item"><a href="#alur" class="nav-link text-white active">ALUR
PENDAFTARAN</a></li>
              <li class="nav-item"><a href="#formulir" class="nav-link text-white">FORMULIR
PENERIMAAN</a></li>
              <li class="nav-item"><a href="#kompetensi" class="nav-link text-white">KOMPETENSI
KEAHLIAN</a></li>
              <li class="nav-item"><a href="#daftarulang" class="nav-link text-white">DAFTAR
ULANG</a></li>
            </ul>
          </div>
        </div>
      </nav>
    </header>

    <main>
      <!-- Hero Section -->
      <section class="bg-warning text-center py-5" style="height: 100vh; display: flex; align-items: center;
justify-content: center;">
        <div class="container" style="width: auto;">
          <img src="Image/kesatrian.png" class="mb-3" style="max-width: 200px; height: auto;" alt="Logo
Sekolah">
          <h1 class="display-5 fw-bold">PENERIMAAN SISWA BARU</h1>
          <p class="lead">SMK KESATRIAN PURWOKERTO TA 2024/2025</p>
        </div>
      </section>

      <!-- Alur Pendaftaran -->
      <section id="alur" class="py-5 bg-light">
        <div class="container">
          <h2 class="text-center fw-bold mb-4">Alur Pendaftaran</h2>
          <div class="row g-4 justify-content-md-center">
            <!-- Card 1 -->
            <div class="col-md-4 text-center">
              <div class="card shadow-sm border-0">
                <div class="card-body">
                  <img src="Image/formulir3d.png" alt="Step 1" class="mb-3" style="width: 150px;">
                  <h5 class="card-title">1. ISI FORM PENDAFTARAN</h5>
                  <p class="card-text">Isi formulir pendaftaran dengan melengkapi persyaratan yang
dibutuhkan.</p>
                </div>
              </div>
            </div>
            <!-- Card 2 -->
            <div class="col-md-4 text-center">
              <div class="card shadow-sm border-0">
                <div class="card-body">
                  <img src="Image/download3d.png" alt="Step 2" class="mb-3" style="width: 150px;">
                  <h5 class="card-title">2. DOWNLOAD BUKTI PENDAFTARAN</h5>
                  <p class="card-text">Download berkas bukti pendaftaran sebagai referensi.</p>
                </div>
              </div>
            </div>
            <!-- Card 3 -->
            <div class="col-md-4 text-center">
              <div class="card shadow-sm border-0">
                <div class="card-body">
                  <img src="Image/diterima3d.png" alt="Step 3" class="mb-3" style="width: 150px;">
                  <h5 class="card-title">3. PENGUMUMAN DITERIMA</h5>
                  <p class="card-text">Masuk ke MENU Pengumuman untuk cek konfirmasi daftar
ulang.</p>
                </div>
```

```
                </div>
            </div>

            <div class="col-md-4 text-center">
                <div class="card shadow-sm border-0">
                    <div class="card-body">
                        <img src="Image/daftarulang3d.png" alt="Step 3" class="mb-3" style="width: 150px;">
                        <h5 class="card-title">4. DAFTAR ULANG</h5>
                        <p class="card-text">Untuk daftar ulang akan dibantu oleh petugas untuk proses daftar
ulangnya.</p>
                    </div>
                </div>
            </div>

            <div class="col-md-4 text-center">
                <div class="card shadow-sm border-0">
                    <div class="card-body">
                        <img src="Image/upload.png" alt="Step 3" class="mb-3" style="width: 150px;">
                        <h5 class="card-title">5.  UPLOAD BUKTI DAFTAR ULANG </h5>
                        <p class="card-text">Upload bukti daftar ulang pada menu Daftar Ulang.</p>
                    </div>
                </div>
            </div>
        </div>
    </div>
  </section>
</main>


<!-- Chatbot Container -->
<div class="chat-container card" id="chat-container">
  <div class="card-header bg-success text-white d-flex justify-content-between">
    <h5 class="mb-0">Chatbot</h5>
    <button id="close-chatbot" class="btn btn-sm btn-light">×</button>
  </div>
  <div class="card-body overflow-auto" id="chat-box" style="height: 350px;">
    <!-- Chat messages will appear here -->
  </div>

  <!-- Common Questions Row -->
  <div class="common-questions px-3 py-2 justify-content-md-center">
    <button class="btn btn-outline-secondary btn-sm d-inline-block question-btn">Cara daftar
sekolah</button>
    <button class="btn btn-outline-secondary btn-sm d-inline-block question-btn">Syarat
Pendaftaran</button>
    <button class="btn btn-outline-secondary btn-sm d-inline-block question-btn">Jadwal PPDB</button>
    <button class="btn btn-outline-secondary btn-sm d-inline-block question-btn">formulir
pendaftaran</button>
  </div>

  <div class="chat-footer d-flex align-items-center" id="chat-footer">
    <input type="text" id="user-input" placeholder="Ketik pesan Anda..." class="form-control" />
    <button id="send-btn" class="btn btn-success">Kirim</button>
  </div>
</div>

<!-- Chatbot Icon -->
<div class="chat-icon" id="chatbot-icon" style="display: none;">
  💬
</div>
```

```
    <!-- Bootstrap JS Bundle -->
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>
    <script>
        const chatIcon = document.getElementById("chatbot-icon");
const chatContainer = document.getElementById("chat-container");
const closeBtn = document.getElementById("close-chatbot");
const sendBtn = document.getElementById("send-btn");
const chatBox = document.getElementById("chat-box");
const userInput = document.getElementById("user-input");

// Hide chatbot and show icon when close button is clicked
closeBtn.addEventListener("click", () => {
    chatContainer.style.display = "none";
    chatIcon.style.display = "flex";
});
const initialMessageDiv = document.createElement("div");
        initialMessageDiv.className = "bot-message";
        initialMessageDiv.innerHTML = `
            Hallo 👋
        `;
        chatBox.appendChild(initialMessageDiv);
        chatBox.scrollTop = chatBox.scrollHeight;


// Show chatbot and hide icon when chat icon is clicked
chatIcon.addEventListener("click", () => {
    chatContainer.style.display = "flex";
    chatIcon.style.display = "none";

    // Cek apakah ini pertama kali chatbot dibuka
    if (!chatBox.querySelector(".bot-message")) {
        const initialMessageDiv = document.createElement("div");
        initialMessageDiv.className = "bot-message";
        initialMessageDiv.innerHTML = `
            Hallo 👋
        `;
        chatBox.appendChild(initialMessageDiv);
        chatBox.scrollTop = chatBox.scrollHeight;
    }
});

    document.querySelectorAll('.question-btn').forEach(button => {
    button.addEventListener('click', function() {
        const question = this.textContent;
        const userInput = document.getElementById('user-input');
        userInput.value = question; // Isi input dengan teks tombol
        document.getElementById('send-btn').click(); // Kirim otomatis
    });
});

        // Function to send a message
function sendMessage() {
    const userMessage = userInput.value.trim();
    if (!userMessage) return;

    // Display user message
    const userMessageDiv = document.createElement("div");
    userMessageDiv.className = "user-message";
    userMessageDiv.textContent = userMessage;
    chatBox.appendChild(userMessageDiv);
```

```
    // Clear input
    userInput.value = "";

    // Show waiting message
    const waitingDiv = document.createElement("div");
    waitingDiv.className = "waiting-message text-muted";
    waitingDiv.textContent = "Menunggu respons...";
    chatBox.appendChild(waitingDiv);
    chatBox.scrollTop = chatBox.scrollHeight;

    // Fetch chatbot response
    fetch("https://habdulghani-model-chatbot.hf.space/get_response", {
        method: "POST",
        headers: {
            "Content-Type": "application/json"
        },
        body: JSON.stringify({ message: userMessage })
    })
    .then(response => response.json())
    .then(data => {
        // Remove waiting message
        chatBox.removeChild(waitingDiv);

        const botMessageDiv = document.createElement("div");
        botMessageDiv.className = "bot-message";

        // Replace \n with <br> to ensure new lines are rendered correctly
        let formattedResponse = data.response.replace(/\n/g, '<br>');

        // Check if there is a link and make it clickable
        formattedResponse = formattedResponse.replace(
            /(https?:\/\/[^\s]+)/g,
            '<a href="$1" target="_blank">$1</a>'
        );

        botMessageDiv.innerHTML = formattedResponse;

        chatBox.appendChild(botMessageDiv);

        // Scroll to the bottom
        chatBox.scrollTop = chatBox.scrollHeight;
    })
    .catch(() => {
        // Remove waiting message and show error if fetch fails
        chatBox.removeChild(waitingDiv);
        const errorDiv = document.createElement("div");
        errorDiv.className = "bot-message text-danger";
        errorDiv.textContent = "Maaf, terjadi kesalahan. Coba lagi nanti.";
        chatBox.appendChild(errorDiv);
    });
}

// Add event listener for "Send" button
sendBtn.addEventListener("click", sendMessage);

// Add event listener for pressing "Enter" key
userInput.addEventListener("keydown", (event) => {
    if (event.key === "Enter") {
        sendMessage();
    }
});
    </script>
```

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