# **BAB IV HASIL DAN PEMBAHASAN**

## **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.

## **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 dibalas 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.

## **Pengembangan *Chatbot***

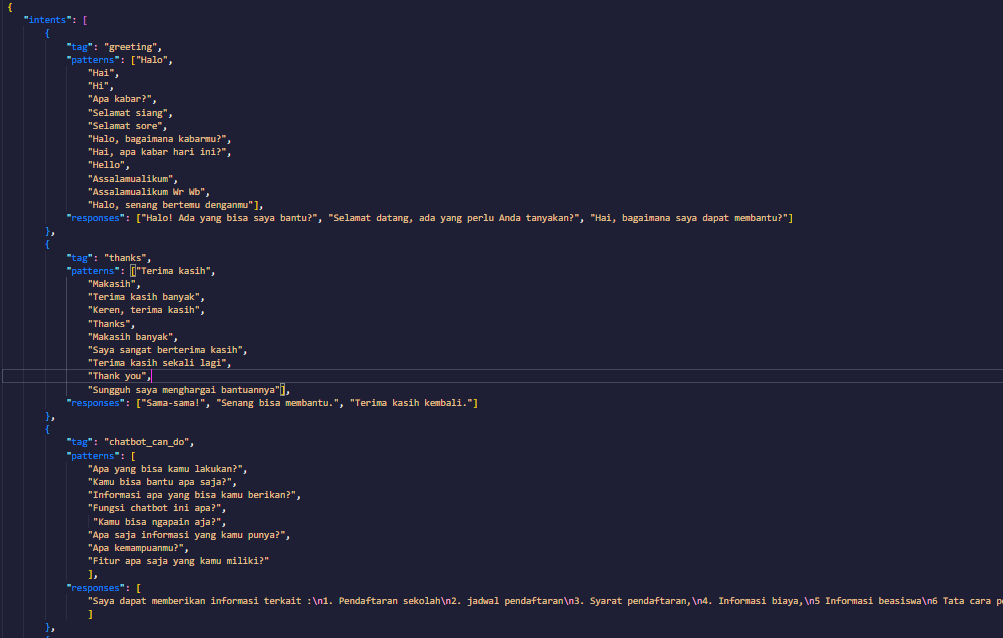
1. ***Data Collection***

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 :

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

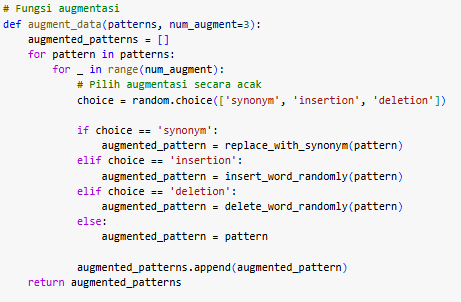
Pada penelitian ini, terdapat 295 dataset pola pertanyaan, 38 tag, dan 44 respon. Pada gambar 4.1 merupakan struktur dari dataset.



Gambar 4. 1 Struktur dataset JSO

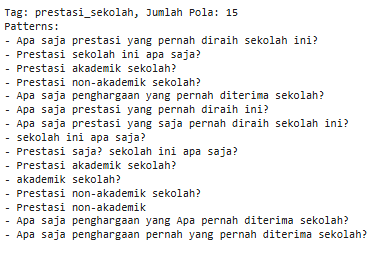
Pada gambar 4.1 merupakan struktur dari dataset yang digunakan terdapat 3 bagian yaitu *tag*, *patterns,* dan *respon es.* Sebagai contoh, tag "*greeting*" merupakan label yang mewakili maksud sapaan. Pada tag ini, terdapat beberapa pola pertanyaan pada bagian patterns, seperti"Hai", "Hi", "Apa kabar?", 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).



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.



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.



Gambar 4. 4 Hasil Sebelum Augmentasi Data



Gambar 4. 5 Hasil Setelah Augmentasi Data

1. ***Text processing***
2. *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.



Gambar 4. 6 Kode Program Case Folding

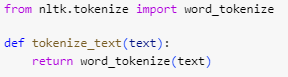
Pada gambar 4.2 merupakan *function* untuk *case folding* yang memiliki parameter (Text) yang dimana akan mengembalikan teks yang sudah dirubah menjadi huruf kecil menggunakan method .lower(). Untuk hasil *case folding* seperti pada tabel 4.1.

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? |

1. *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.



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.

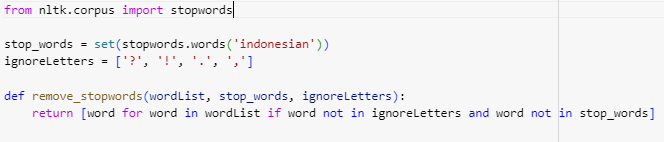


Gambar 4. 8 Hasil Tokenizing

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.

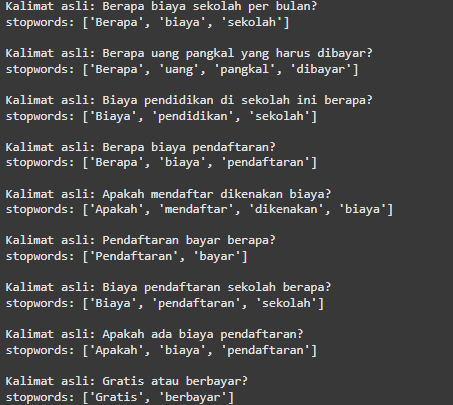
1. *Stopword Removal*

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.



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.*

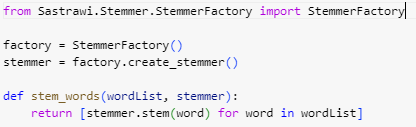


Gambar 4. 10 Hasil Stopword Removal

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.

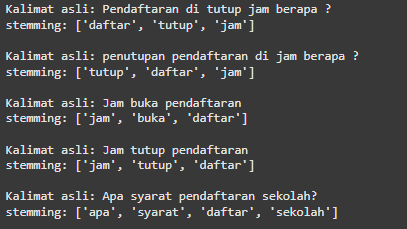
1. *Stemming*

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).



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 .



Gambar 4. 12 Hasil Stemming

1. ***Modeling***
2. *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.

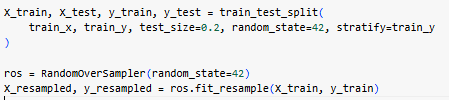
Tabel 4. 2 Hasil Bag of Word

|  |
| --- |
| 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.

1. *Split Data*

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

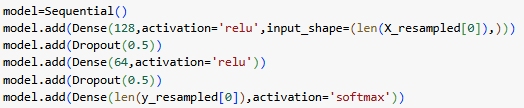


Gambar 4. 13 Source Code Split Data

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.

1. 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.



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.
7. *Training model*

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



Gambar 4. 15 Kode Program Training model

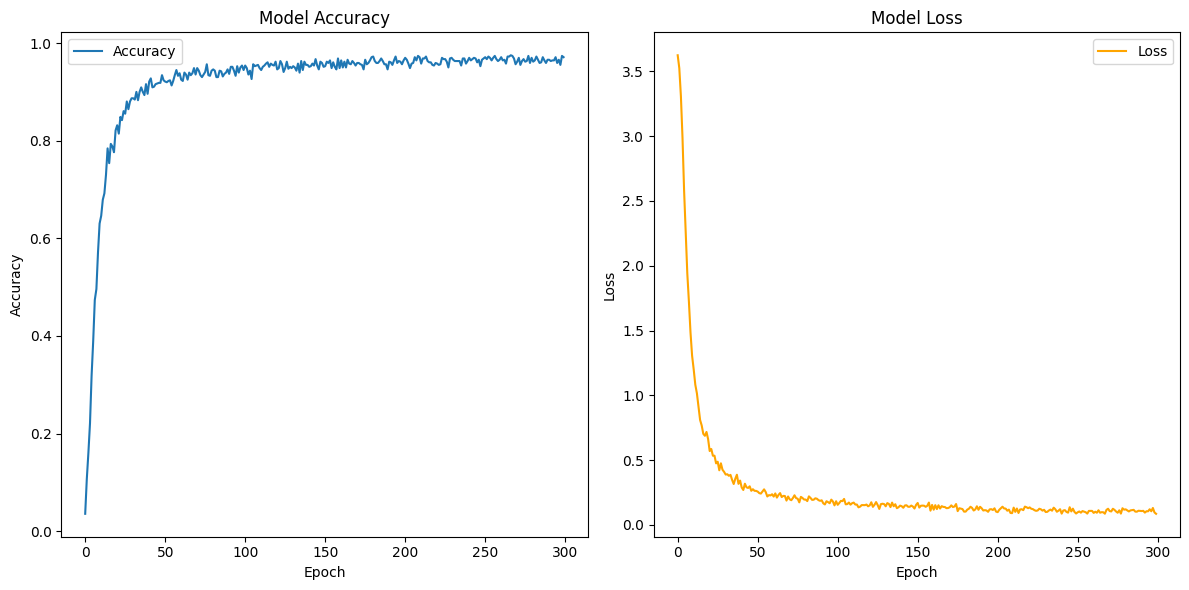
Pada gambar 4.10 mendefinisikan optimizer SGD dengan learning rate 0.01 Menetapkan kecepatan pembaruan bobot dalam proses pelatihan, decay 1e-6 Mengurangi nilai learning rate secara bertahap untuk meningkatkan stabilitas pelatihan, momentum 0.9 Menambahkan momentum untuk mempercepat konvergensi dan mengurangi 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 |

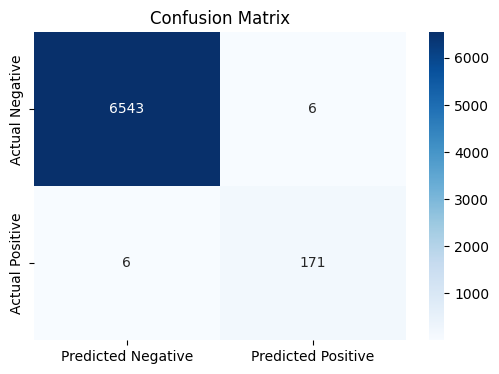
Hasil eksperimen menunjukkan bahwa peningkatan jumlah neuron pada lapisan tersembunyi berkorelasi positif dengan peningkatan akurasi model dan penurunan nilai *loss*. Konfigurasi dengan 32 dan 16 neuron pada lapisan tersembunyi belum mampu menangkap pola yang kompleks dalam data. Namun, dengan meningkatkan 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

1. **Evaluasi**

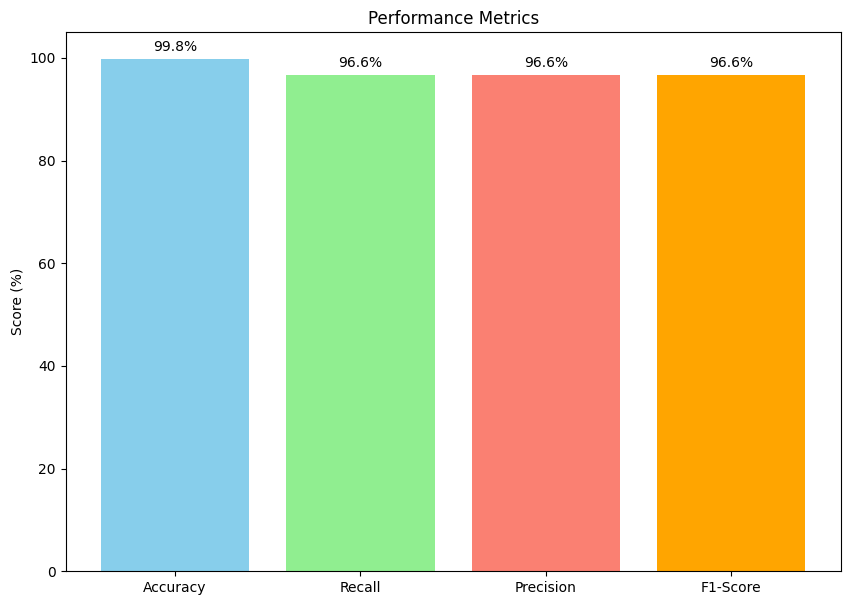
Selanjut melakukan evaluasi model menggunakan metode *confusion matriks* untuk mencari kinerja model melalui hasil akurasi, presisi, recall, dan F1-score. Untuk melakukan evaluasi data dibagi menjadi 80 – 20, 80% untuk data training dan 20% untuk data uji.



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*.

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.

1. ***Deployment***

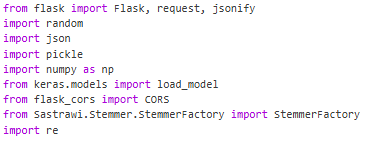
Pada tahap *deployment* ini, tujuan utamanya adalah untuk membangun antarmuka tampilan *website* PPDB dan *chatbot*. Pengembangan antarmuka ini menggunakan *Bootstrap* sebagai *frontend* dan *framework* Flask untuk membangun API *respon*  dari untuk menciptakan antarmuka situs web yang menarik dan *respon*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

1. 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.



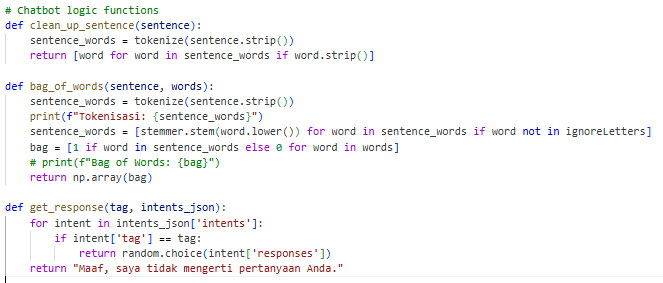
Gambar 4. 19 Struktur File

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



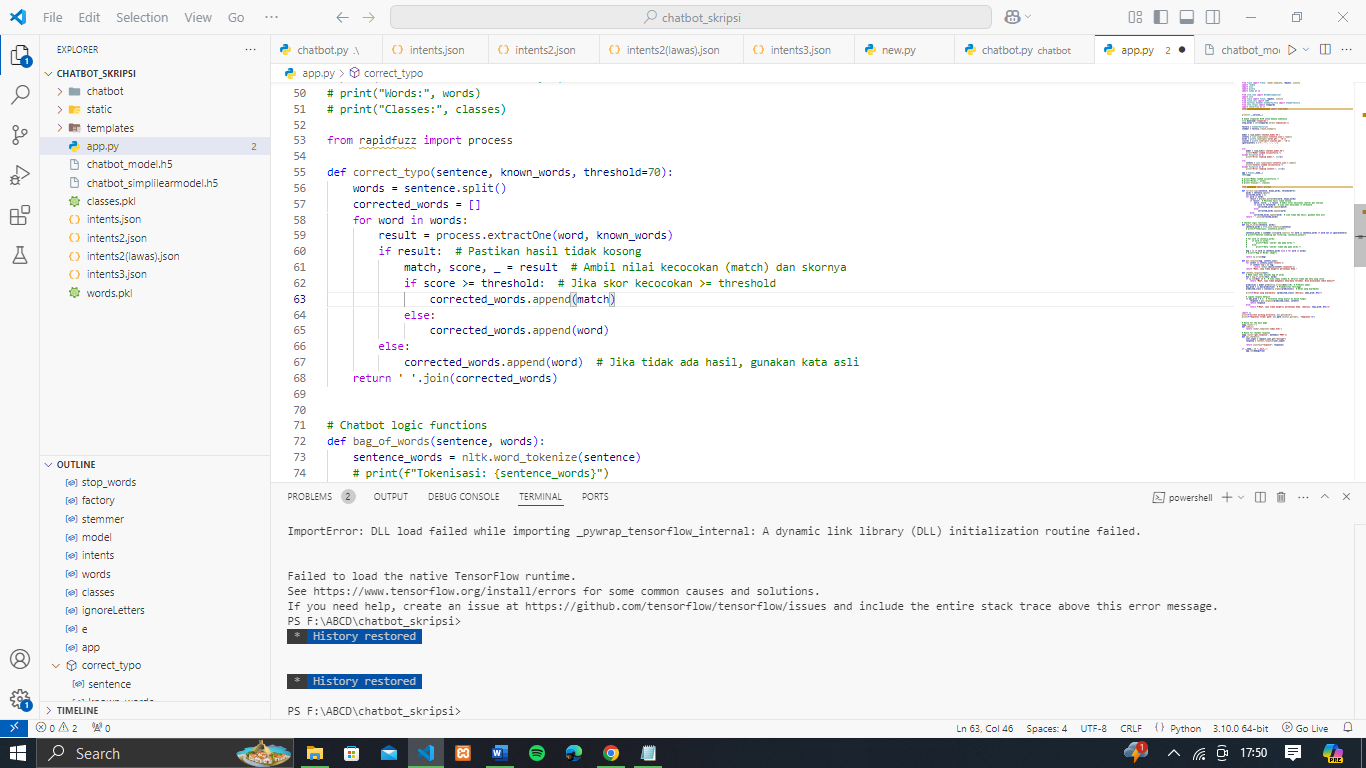
Gambar 4. 20 Library yang digunakan

*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.



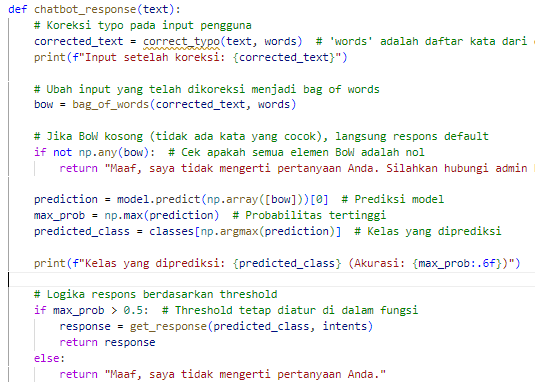
Gambar 4. 21 Source Code Logic Respon

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\_response* 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*.



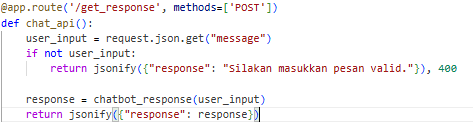
Gambar 4. 22 Source Code correct\_typo

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.



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*.



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.

1. Membuat tampilan menggunakan *bootstrap.*



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”.

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



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 <br> 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.

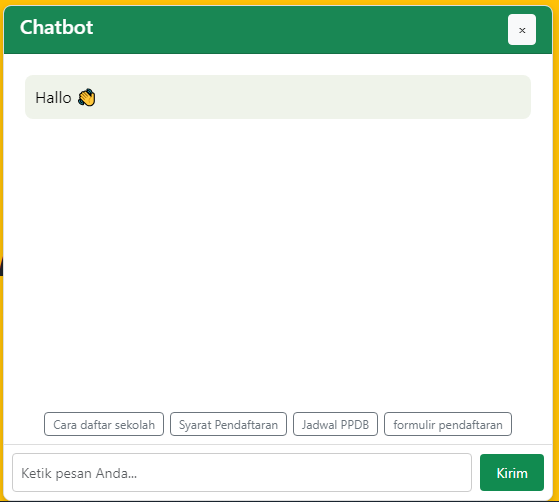
1. Hasil tampilan halaman depan *website*



Gambar 4. 27 Tampilan Awal Website

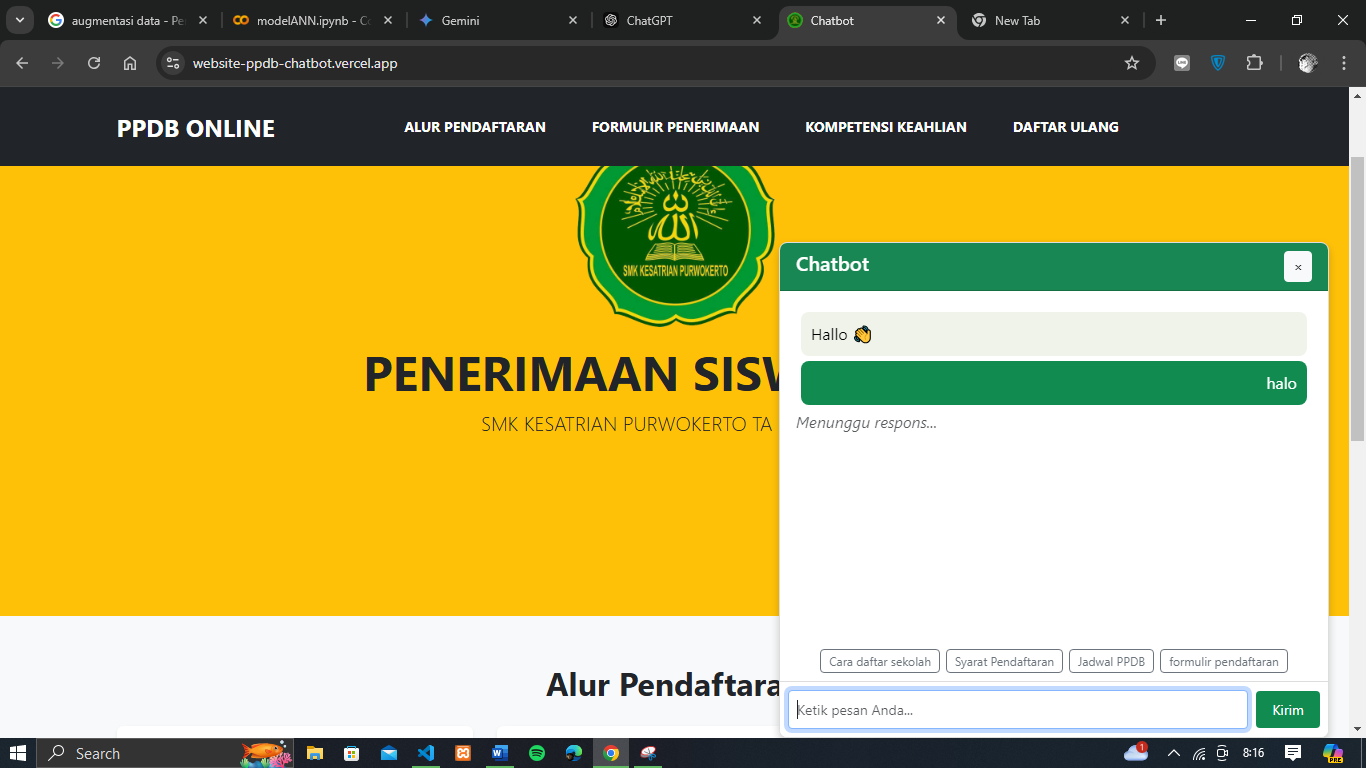
Halaman awal website ini dirancang menyerupai tampilan asli dari *website* PPDB resmi SMK Kesatrian Purwokerto, dengan tambahan fitur tombol chat di pojok kanan bawah. Tombol chat ini berfungsi untuk mengakses fitur *chatbot* yang bertujuan memfasilitasi pengujian dan interaksi dengan *chatbot* secara langsung.

1. Tampilan awal *chatbot*



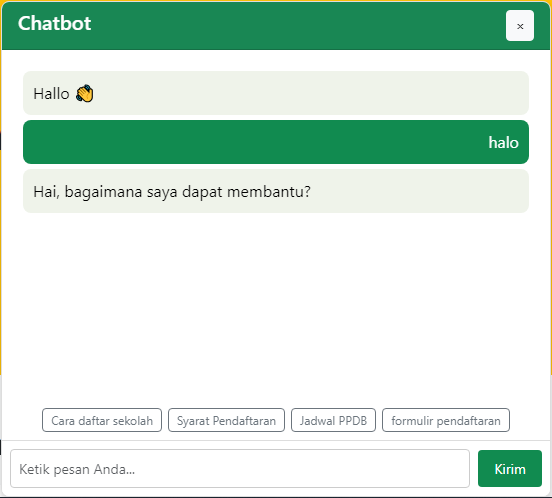
Gambar 4. 28 Tampilan Awal Chatbot

1. Tampilan *Chatbot* saat menunggu respon



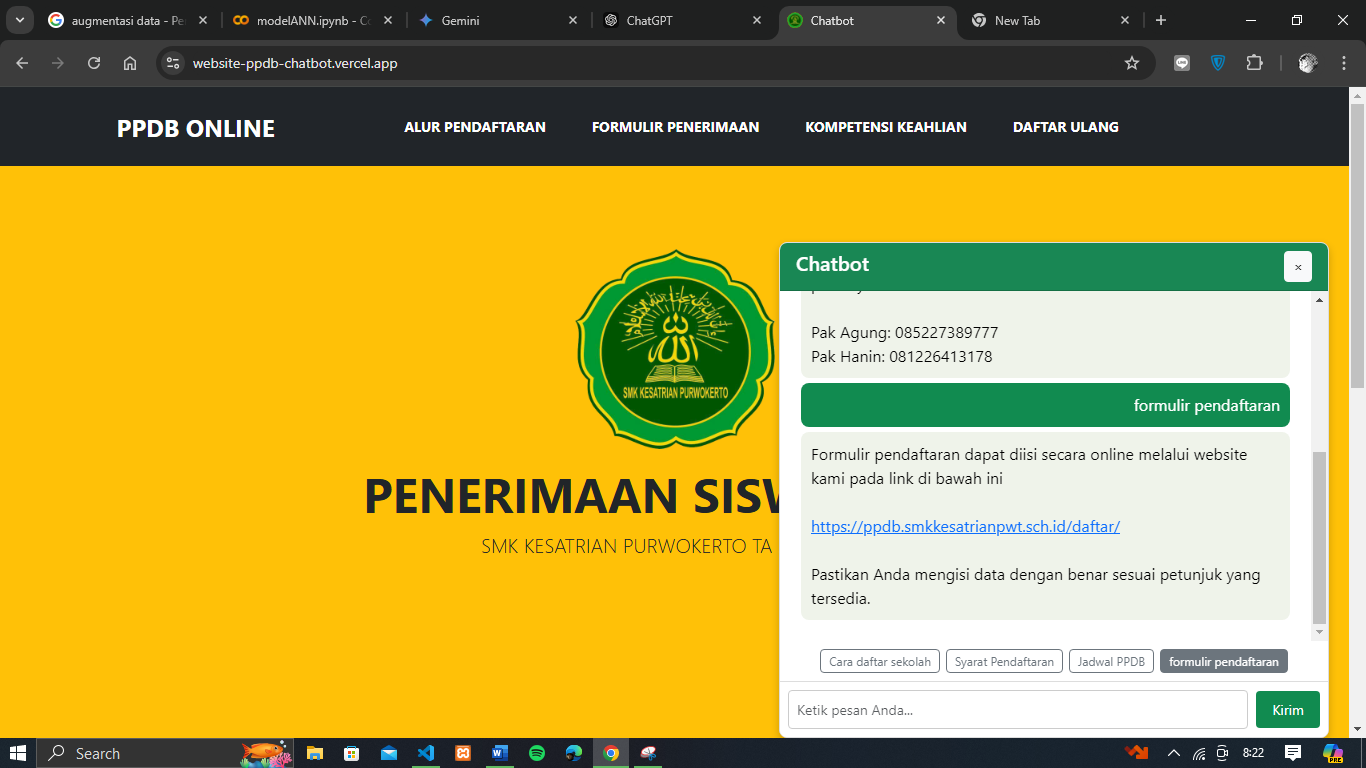
Gambar 4. 29 Tampilan Chatbot saat menunggu respon

1. Tampilan saat memberi respon



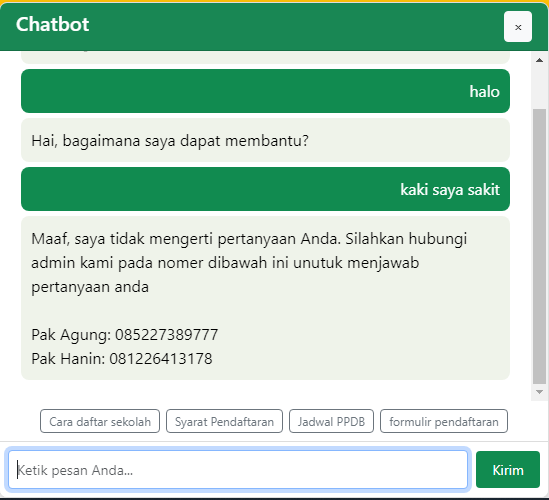
Gambar 4. 30 Tampilan Chatbot saat memberi respon

1. Tampilan respon dari *button* pertanyaan



Gambar 4. 31 Tampilan respon dari button pertanyaan

1. Tampilan inputan salah



Gambar 4. 32 Tampilan inputan salah

## **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 Tabel Uji Alpha Testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Input pertanyan** | **Respon tag** | **Hasil respon yang diharapkan** | **Keterangan** |
|  | Apa kabar | greeting | greeting | Berhasil |
|  | Terimakasih | thanks | thanks | Berhasil |
|  | Assalamualaikum | salam | salam | Berhasil |
|  | Apa yang bisa kamu lakukan? | chatbot\_can\_do | chatbot\_can\_do | Berhasil |
|  | baiklah | agree | agree | Berhasil |
|  | Kapan jadwal pendaftaran dibuka? | jadwal\_pendaftaran | jadwal\_pendaftaran |  |
|  | Jam berapa pendaftaran dibuka ? | jambuka\_tutup | jambuka\_tutup | Berhasil |
|  | Apa syarat pendaftaran sekolah? | syarat\_pendaftaran | syarat\_pendaftaran | Berhasil |
|  | cara mengisi formulir pendaftaran ? | formulir\_pendaftaran | formulir\_pendaftaran | Berhasil |
|  | Biaya spp | spp\_dan\_uang\_masuk | spp\_dan\_uang\_masuk | Berhasil |
|  | Biaya pendaftaran | biaya\_pendaftaran | biaya\_pendaftaran | Berhasil |
|  | Jurusan yang dibuka | jurusan\_yang\_dibuka | jurusan\_yang\_dibuka | Berhasil |
|  | Di mana lokasi sekolah? | lokasi\_sekolah | lokasi\_sekolah | Berhasil |
|  | Fasilitas sekolah ? | fasilitas\_sekolah | fasilitas\_sekolah | Berhasil |
|  | Akreditasi sekolah apa? | akreditasi\_sekolah | akreditasi\_sekolah | Berhasil |
|  | Nilai rata-rata | nilai\_pendaftaran | nilai\_pendaftaran | Berhasil |
|  | Apa itu jurusan DKV ? | jurusan\_desain\_komunikasi\_visual | jurusan\_desain\_komunikasi\_visual | Berhasil |
|  | apa itu tkr ? | jurusan\_teknik\_kendaraan\_ringan | jurusan\_teknik\_kendaraan\_ringan | Berhasil |
|  |  |  |  |  |
| Tabel 4. 6 Tabel Uji Alpha Testing (Lanjutan) | | | | |
|  | Apa itu jurusan TSM? | jurusan\_teknik\_sepeda\_motor | jurusan\_teknik\_sepeda\_motor | Berhasil |
|  | Apa itu TAV | jurusan\_teknik\_audio\_video | jurusan\_teknik\_audio\_video | Berhasil |
|  | Apa iru TKJ | jenjang\_karir\_tkj | jurusan\_teknik\_komputer\_dan\_jaringan | Tidak berhasil |
|  | Bagaimana cara mendaftar? | alur\_pendaftaran | alur\_pendaftaran | Berhasil |
|  | cara daftar ulang ? | daftar\_ulang | daftar\_ulang | Berhasil |
|  | Apa saja berkas yang di butuhkan untuk daftar ulang ? | berkas\_daftar\_ulang | berkas\_daftar\_ulang | Berhasil |
|  | Kapan pengumuman hasil seleksi? | pengumuman\_diterima | pengumuman\_diterima | berhasil |
|  | ukuran pas foto | ukuran\_pas\_foto | ukuran\_pas\_foto | Berhasil |
|  | visi misi | visi\_misi\_sekolah | visi\_misi\_sekolah | Berhasil |
|  | berapa jumlah siswa sekarang | jumlah\_siswa\_aktif | jumlah\_siswa\_aktif | Berhasil |
|  | Informasi beasiswa | beasiswa | beasiswa | Berhasil |
|  | Prospek kerja DKV? | jenjang\_karir\_dkv | jenjang\_karir\_dkv | Berhasil |
|  | Prospek kerja TAV? | jenjang\_karir\_tav | jenjang\_karir\_tav | Berhasil |
|  | Prospek kerja TKJ | jenjang\_karir\_tkj | jenjang\_karir\_tkj | Berhasil |
|  | Prospek kerja TSM? | jenjang\_karir\_tsm | jenjang\_karir\_tsm | Berhasil |
|  | Berapa biaya per jurusan? | biaya\_perjurusan | biaya\_perjurusan | Berhasil |
|  | Apa itu PPDB? | apa\_itu\_ppdb | apa\_itu\_ppdb | Berhasil |
|  | Prospek kerja TKR? | jenjang\_karir\_tkr | jenjang\_karir\_tkr | Berhasil |
|  | Prestasi sekolah | prestasi\_sekolah | prestasi\_sekolah | Berhasil |
|  | Kaki saya sakit | error | error | Berhasil |
|  | 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).

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

1. *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. 7 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. 8 Table Hasil Kuesioner

| **No** | **Pertanyaan** | **SS** | **S** | **C** | **TS** | **STS** |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | Menurut Anda setelah menggunakan *chatbot* ini, apakah dapat membantu memberikan Informasi terkait 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 |
|  |  |  |  |  |  |  |
| Tabel 4. 9 Table Hasil Kuesioner(Lanjutan) | | | | | | |
| 4 | Menurut Anda Apakah *chatbot* ini efisien dalam mencari informasi PPDB selama 24 jam ? | 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, Efiisiensi, Kemudahan interaksi, dan kejelasan respon.

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

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. 10 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.

1. Setelah menggunakan *chatbot* ini, apakah *chatbot* ini mempermudah dalam mendapatkan informasi?

Tabel 4. 11 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.

1. Menurut Anda Informasi/jawaban yang diberikan *chatbot*sudah sesuai dengan yang ditanyakan?

Tabel 4. 12 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.

1. Menurut anda apakah *chatbot* ini efisien dalam mencari informasi PPDB selama 24 jam ?

Tabel 4. 13 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.

1. Menurut anda apakah mudah dalam melakukan interaksi terhadap *chatbot* ?

Tabel 4. 14 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.

1. Menurut anda Apakah jawaban dari *chatbot* mudah dipahami ?

Tabel 4. 15 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.

1. Menurut anda Apakah perlu pembaruan untuk menyempurnakan Aplikasi *chatbot*ini?

Tabel 4. 16 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.*

Tabel 4. 17 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.

Hasil uji beta terhadap *chatbot* PPDB di SMK Kesatrian Purwokerto menunjukkan rata-rata kepuasan pengguna sebesar 86,19%. Sebagian besar pengguna menilai *chatbot* sangat membantu dalam memberikan informasi terkait PPDB selama 24 jam, mempermudah akses informasi, dan menyajikan jawaban yang mudah dipahami. Skor tertinggi terdapat pada pertanyaan mengenai efektivitas *chatbot* dalam memberikan informasi (90%), yang menunjukkan bahwa sistem ini mampu memenuhi kebutuhan pengguna secara signifikan. Dengan hasil tersebut, dapat disimpulkan bahwa *chatbot* sudah berjalan dengan baik dan memberikan pengalaman pengguna yang positif, meskipun pengembangan lebih lanjut tetap diperlukan untuk menyempurnakan fungsionalitasnya.