PENGEMBANGAN *CHATBOT* BERBASIS *FRAMEWORK* RASA PADA *WEBSITE* BANK SAMPAH SRIWILIS

Candra Bella Vista[1], Mellyana Tundjung[2], Triana Fatmawati[3], Endah Septa Sintiya[4]

[1,2,3,4] Jurusan Teknologi Informasi, Politeknik Negeri Malang
[1]bellavista@polinema.ac.id, [2]2141720061@student.polinema.ac.id, [3]Triana@polinema.ac.id,
[4]endah.septa@polinema.ac.id

Abstrak

Perkembangan teknologi informasi dalam bidang *Natural Language Processing* (NLP) membuka peluang pemanfaatan *chatbot* sebagai solusi layanan informasi berbasis *website* yang interaktif dan responsive di tengah keterbatasan tenaga pelayanan dan waktu operasional. *Chatbot* memungkinkan pengguna memperoleh informasi secara real-time tanpa keterlibatan operator manusia secara langsung, sehingga dapat meningkatkan efisiensi dan ketersediaan layanan. Penelitian ini bertujuan mengembangkan *chatbot* berbasis *Framework Really Awesome Software Automation* (RASA) pada *Website* Bank Sampah Sriwilis serta menganalisis pengaruh konfigurasi pipeline *Natural Language Understanding* (NLU) terhadap performa klasifikasi *intent*. Metode pengembangan sistem menggunakan model *Waterfall* yang meliputi tahap analisis kebutuhan, perancangan sistem, implementasi, dan pengujian. Dataset disusun dalam bahasa Indonesia, terdiri dari 9 *intent* dengan total 250 kalimat. Eksperimen dilakukan terhadap tiga konfigurasi pipeline, yaitu DIETClassifier sebagai model *baseline*, DIETClassifier dengan penambahan fitur leksikal melalui RegexFeaturizer dan LexicalSyntacticFeaturizer, serta LogisticRegressionClassifier sebagai model pembanding. Evaluasi kinerja model dilakukan menggunakan metrik accuracy, precision, recall, dan F1-score. Hasil penelitian menunjukkan bahwa model berbasis DIETClassifier memberikan peningkatan performa akurasi sebesar 5% dibandingkan Logistic Regression. Konfigurasi model dengan penambahan *pipeline RegexFeaturizer* dan *LexicalSyntacticFeaturizer* menghasilkan nilai *accuracy* terbaik sebesar 93%, *precision* 93%, *recall* 91%, dan *F1-score* 91%. Dengan demikian, pemilihan konfigurasi pipeline yang tepat serta penerapan fitur tambahan berpengaruh signifikan terhadap peningkatan performa *chatbot* berbasis RASA pada layanan informasi Bank Sampah Sriwilis.

Kata kunci: *Chatbot*, *Framework* RASA, Klasifikasi *intent*

1. Pendahuluan

Perkembangan teknologi informasi telah mendorong banyak institusi untuk mengadopsi solusi digital dalam meningkatkan kualitas layanan mereka. Bank Sampah Sriwilis sebagai lembaga sosial yang bergerak di bidang pengelolaan sampah, menghadapi tantangan dalam menyampaikan informasi secara merata kepada nasabah, terutama terkait prosedur, jadwal penjemputan, saldo tabungan, dan edukasi lingkungan. Informasi yang seharusnya mudah diakses sering kali terhambat oleh keterbatasan tenaga pelayanan serta waktu operasional yang terbatas. Salah satu teknologi yang dapat menjadi solusi permasalahan ini adalah *chatbot*, yang memanfaatkan Natural Language Processing (NLP) untuk memfasilitasi interaksi manusia dan komputer secara alami (Muliyono & Sumijan, 2021). *Chatbot* mampu mengidentifikasi maksud (*intent*) pengguna dan mengekstraksi informasi penting (*entity*) dari kalimat yang diberikan (Puspita et al., 2025). *Chatbot* terbukti efektif dalam menyampaikan informasi secara cepat, efisien, dan dapat diakses kapan saja, sehingga menjadi solusi ideal untuk mengatasi keterbatasan layanan manual (Setyono & Wibisono, S., 2024).

Salah satu *framework* yang banyak digunakan dalam pengembangan *chatbot* berbasis NLP adalah RASA. RASA merupakan *framework* open-source yang menyediakan modul *Natural Language Understanding* (NLU) dan *Dialogue Management* yang fleksibel serta dapat dikustomisasi sesuai kebutuhan sistem (Bocklisch et al., 2017). Pada RASA, proses klasifikasi *intent* dan ekstraksi *entity* umumnya menggunakan arsitektur DIET (*Dual Intent and Entity Transformer*) yang mengintegrasikan pembelajaran berbasis *transformer* untuk menangani tugas multi-label *classification* dan *sequence tagging* secara simultan (Astuti et al., 2024). Penggunaan *framework* RASA memungkinkan pengembangan *chatbot* dengan performa yang baik meskipun menggunakan dataset berukuran terbatas. Selain itu, RASA menyediakan berbagai konfigurasi *pipeline* yang memungkinkan pengembang melakukan eksperimen terhadap kombinasi *featurizer* dan *classifier* untuk memperoleh performa optimal.

Penelitian terdahulu menyebutkan bahwa *framework* RASA dapat memahami maksud pengguna dan merespons dengan tingkat akurasi tinggi (Fauzia et al., 2021),(Apriliyanto & Arief,

2020). *Framework* RASA juga telah terbukti mampu meningkatkan efisiensi layanan di berbagai institusi, termasuk perguruan tinggi (Arthana et al., 2021) (Fauzia et al., 2021), pemerintahan (Rachman et al., 2023), dan sektor wisata (Rosida & Hadiono, 2024). Namun pada penerapannya, konfigurasi *pipeline* dan jumlah data latih berpengaruh signifikan terhadap performa klasifikasi *intent* (Supreetha & Sandhya. S, 2022).

Sistem *chatbot* juga perlu menangani kondisi ketika model tidak mampu mengenali maksud pengguna dengan tingkat kepercayaan yang memadai (Cannavaro, 2023). Dalam *framework* RASA, kondisi tersebut ditangani melalui mekanisme *fallback response*, yaitu respons otomatis yang diberikan ketika skor prediksi *intent* berada di bawah ambang batas tertentu (RASA Technologies, 2023). Pengaturan ambang batas dan strategi *fallback* menjadi aspek penting untuk menjaga kualitas interaksi serta mencegah respons yang keliru akibat kesalahan klasifikasi.

Berdasarkan uraian diatas sebagian besar penelitian masih berfokus pada implementasi sistem tanpa melakukan eksplorasi komparatif terhadap variasi konfigurasi pipeline dan pengaturan *fallback* yang tersedia dalam *framework* RASA. Sehingga, penelitian ini bertujuan untuk mengembangkan *chatbot* berbasis *Framework* RASA pada *website* Bank Sampah Sriwilis serta melakukan eksperimen terhadap beberapa konfigurasi *pipeline* NLU untuk menganalisis performa klasifikasi *intent*. Hasil penelitian diharapkan dapat memberikan kontribusi dalam pemilihan konfigurasi model yang sesuai untuk pengembangan *chatbot* layanan informasi berbasis bahasa Indonesia.

2. Metode

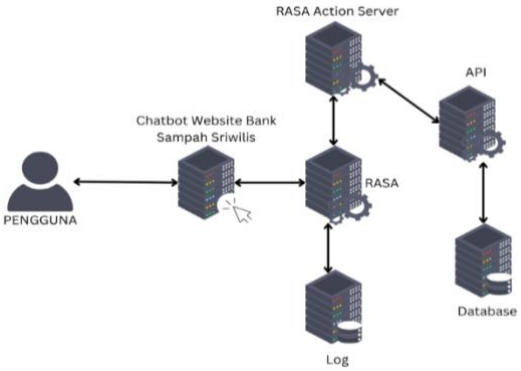
Penelitian ini dilakukan dengan pendekatan *Software Development Life* Cycle (SDLC) menggunakan model *Waterfall*. Model ini dipilih karena memiliki alur kerja yang sistematis dan terstruktur, sesuai untuk proyek dengan kebutuhan yang telah terdefinisi secara jelas di awal (Wahid, 2020). Tahapan *waterfall* dimulai dari analisis kebutuhan, perancangan sistem, implementasi, dan pengujian.

2.1 Analisis Kebutuhan

Pada tahap ini dilakukan identifikasi kebutuhan layanan informasi pada *Website* Bank Sampah Sriwilis. Mengidentifikasi kebutuhan sistem melalui wawancara dengan pengelola Bank Sampah Sriwilis dan observasi alur layanan manual, serta dokumentasi FAQ. Informasi yang dianalisis meliputi jam operasional, jenis sampah yang diterima, harga sampah, prosedur menabung, lokasi, dan pendaftaran. Hasil analisis kebutuhan digunakan sebagai dasar dalam menentukan daftar *intent* dan *entity* yang akan digunakan dalam sistem *chatbot*.

2.2 Perancangan Sistem

Tahapan perancangan sistem meliputi tahapan perancangan arsitektur sistem dan perancangan dataset. Gambar 1 menunjukkan arsitektur sistem.



Gambar 1. Arsitektur Sistem

Sistem *chatbot* berbasis RASA ini diintegrasikan ke *website* menggunakan API dan webhook, dengan arsitektur yang terdiri dari *frontend* (*website* Laravel), *backend* (*chatbot* RASA), serta *server hosting* untuk komunikasi *real-time* antara pengguna dan *chatbot*.

2.2.1 RASA *Framework*

Penelitian ini menggunakan RASA *Open Source* sebagai *framework* utama dalam pengembangan *chatbot*. RASA merupakan *framework* berbasis Python yang dirancang untuk membangun sistem percakapan (*conversational AI*) dengan pendekatan modular yang fleksibel dan dapat dikustomisasi (RASA Technologies, 2023). *Framework* ini banyak digunakan dalam pengembangan *chatbot* berbasis *Natural Language Processing* (NLP) karena menyediakan komponen lengkap untuk klasifikasi intent, ekstraksi entity, serta manajemen dialog.

Secara umum, arsitektur RASA terdiri atas dua komponen utama, yaitu RASA NLU dan RASA Core (Gujjar & Kumar, 2022). RASA NLU bertanggung jawab dalam memahami pesan pengguna dengan melakukan dua tugas utama, yaitu *Intent Classification* dan *Entity Extraction*. *Intent classification* bertugas mengidentifikasi maksud pengguna dari teks masukan. *Entity Extraction* mengekstraksi informasi spesifik dari teks yang relevan dengan intent tertentu. RASA *Core* berfungsi mengelola alur percakapan berdasarkan konteks dialog. Komponen ini menentukan respons yang akan diberikan sistem setelah intent dikenali. Pengelolaan dialog dilakukan menggunakan *stories* untuk mendefinisikan alur percakapan berbasis skenario, *rules* untuk menangani respons deterministik seperti sapaan atau *fallback*, dan policies untuk menentukan strategi pemilihan respons.

RASA menggunakan pendekatan modular berbasis pipeline yang terdiri atas beberapa komponen pemrosesan teks, seperti tokenizer,

featurizer, dan classifier. Pipeline bertanggung jawab mengubah teks mentah menjadi representasi numerik sebelum diproses oleh model klasifikasi. Pendekatan modular ini memungkinkan dilakukan eksperimen terhadap berbagai konfigurasi pipeline untuk menganalisis pengaruhnya terhadap performa klasifikasi intent (Talenggoran et al., 2025).

2.2.2 Perancangan Dataset

Data yang digunakan untuk membangun *chatbot* berupa sampel dialog dan *intent* yang dikumpulkan dari *Frequently Asked Questions* (FAQ). Data ini diperoleh melalui wawancara dengan petugas Bank Sampah dan survei yang telah divalidasi oleh mitra. Tabel 1 menunjukkan contoh data FAQ.

Tabel 1. Contoh Data FAQ

Pertanyaan	Jawaban
"Bagaimana prosedur menyetor sampah?"	"Untuk menyetor sampah, nasabah harus memilah sampah sesuai jenisnya terlebih dahulu. Kemudian, datang ke lokasi Bank Sampah pada jam operasional, dan petugas akan menimbang serta mencatat sampah yang disetor."
"Jenis sampah apa saja yang dapat disetorkan?"	"Jenis sampah yang dapat disetorkan meliputi sampah organik, sampah anorganik (plastik, kaca, logam), kertas, karton, dan sampah elektronik"
"Untuk penghitungan saldo dilakukan oleh siapa?"	"Untuk saldo itu dicek langsung sama petugas."

Data tersebut kemudian melalui tahap data *preprocessing* untuk memastikan kualitasnya (Ruidungan & Jacobus, 2021). Proses ini meliputi *Data Cleaning*, yaitu menghapus duplikasi, kesalahan ejaan, informasi tidak relevan. Selanjutnya *normalization*, yaitu menstandarkan format data, menyamakan huruf kapital, memperbaiki singkatan, tanda baca. *Intent Mapping* yaitu, mengelompokkan pertanyaan berdasarkan tujuan. Tahap selanjutnya *data Categorization* yaitu, memisahkan data menjadi *intent* dan *entities* agar lebih terstruktur. Dari proses tersebut dataset dikelompokkan menjadi 9 *intent* dengan total 250 kalimat. Tabel 2 menunjukkan distribusi dataset.

Tabel 2. Distribusi Dataset

Intent	Jumlah Data
Greeting	30
Jam_operasional	35
Lokasi	30
Jenis_sampah	35
Harga_sampah	30
Cara_menabung	25
Pendaftaran	20
Cek_saldo	25
Goodbye	20

Tabel 2 menunjukkan distribusi dataset berdasarkan intent yang digunakan dalam pelatihan *chatbot*. Dataset terdiri dari sembilan intent utama yang merepresentasikan kebutuhan informasi pengguna, dengan jumlah data per intent berkisar antara 20 hingga 35 data. Distribusi ini disusun relatif seimbang untuk memastikan setiap intent dapat dipelajari secara optimal oleh model, sehingga meningkatkan akurasi klasifikasi dan mengurangi bias terhadap intent tertentu.

2.3 Implementasi

Tahap implementasi merupakan fase pengembangan sistem berdasarkan hasil perancangan yang telah dilakukan pada tahap sebelumnya. Pada tahap ini setelah dilakukan tahap *preprocessing*, data ditransformasikan ke dalam format *YAML Ain't Markup Language* (YAML) sesuai standar *framework* RASA. Data pelatihan dalam format ini terdiri dari *Intents* atau tujuan pengguna, *Stories* atau alur percakapan, *Rules* atau alur percakapan pasti, dan *Domain* atau ruang lingkup kerja *chatbot* (Gujjar & Kumar, 2022). Selanjutnya dilakukan pelatihan model, pengaturan mekanisme *fallback*, serta integrasi *chatbot* ke dalam Website Bank Sampah Sriwilis.

2.3.1 Konfigurasi *Pipeline* NLU

Implementasi NLU dilakukan dengan mengonfigurasi pipeline pada file *config.yml* sesuai dengan desain eksperimen yang telah dirancang. Pipeline terdiri atas komponen tokenizer, featurizer, dan classifier (RASA Technologies, 2023). Tokenizer yang digunakan adalah *WhitespaceTokenizer* untuk memecah kalimat berdasarkan spasi. Selanjutnya, *CountVectorsFeaturizer* digunakan untuk mengubah teks menjadi representasi numerik berbasis frekuensi kemunculan kata (*bag-of-words*).

Pada model berbasis deep learning digunakan *DIETClassifier* yang mampu melakukan klasifikasi *intent* dan ekstraksi entity secara simultan menggunakan pendekatan *transformer* (Narendra & Setyaningsih, 2021). Parameter pelatihan seperti jumlah *epochs* disesuaikan untuk meningkatkan konvergensi model. Pada penelitian ini digunakan tiga konfigurasi pipeline untuk dianalisis performanya.

Model A menggunakan konfigurasi standar RASA dengan DIETClassifier sebagai komponen utama. DIET (*Dual Intent and Entity Transformer*) merupakan arsitektur berbasis *transformer* yang dirancang untuk melakukan klasifikasi *intent* dan ekstraksi entity secara simultan (Narendra & Setyaningsih, 2021).

Model B menambahkan komponen *featurizer* untuk memperkaya representasi teks sebelum diproses oleh DIETClassifier. RegexFeaturizer berfungsi untuk mengidentifikasi pola tertentu dalam teks menggunakan ekspresi reguler, misalnya angka

atau kata kunci khusus. LexicalSyntacticFeaturizer digunakan untuk mengekstraksi fitur berbasis struktur leksikal seperti huruf kapital, posisi kata, dan pola morfologi. Penambahan fitur ini bertujuan meningkatkan kemampuan model dalam membedakan *intent* yang memiliki kemiripan struktur kalimat.

Model C menggunakan pendekatan *LogisticRegressionClassifier* sebagai pembanding terhadap model berbasis *transformer*. LogisticRegressionClassifier menggunakan regresi logistik untuk melakukan klasifikasi *intent* berdasarkan fitur numerik dari CountVectorsFeaturizer.

2.3.2 Pelatihan Model

Pelatihan menghasilkan model dalam bentuk file terkompresi yang disimpan pada direktori */models/*. Setiap konfigurasi pipeline dilatih menggunakan dataset yang sama untuk memastikan konsistensi eksperimen. Proses pelatihan bertujuan untuk membangun model klasifikasi *intent* berdasarkan data latih yang telah disusun. Model yang dihasilkan kemudian digunakan pada tahap evaluasi.

2.3.3 Implementasi *Fallback Response*

Untuk menghindari respons yang tidak relevan akibat prediksi dengan tingkat kepercayaan rendah, sistem menerapkan mekanisme *fallback response*. *Fallback* akan aktif ketika nilai *confidence score intent* berada di bawah ambang batas tertentu.

Pengaturan *fallback* dilakukan melalui konfigurasi *RulePolicy* dengan menentukan parameter *core_fallback_threshold*. Ketika threshold tidak terpenuhi, sistem akan menjalankan *action_default_fallback* yang berisi respons klarifikasi kepada pengguna (RASA Technologies, 2023). Mekanisme ini bertujuan meningkatkan keandalan sistem dan menjaga kualitas interaksi pengguna.

2.4 Evaluasi

Pengujian dilakukan dalam dua bentuk, yaitu pengujian fungsional dan pengujian kinerja model. Pengujian fungsional menggunakan metode *blackbox testing.* Pengujian ini bertujuan memastikan setiap *intent* dan respons berjalan sesuai skenario yang dirancang. pengujian model *chatbot* RASA, evaluasi dilakukan menggunakan metrik uji performa yang sesuai dengan karakteristik model dan studi kasus yang ditangani. Metrik utama yang digunakan adalah *Confusion Matrix* beserta parameter turunannya, yaitu *Precision*, *Recall*, dan *F1-Score*, untuk mengukur akurasi respons *chatbot* dalam memahami dan memberikan jawaban yang relevan (Amelia & Sutabri, 2024).

Akurasi menunjukkan proporsi prediksi *intent* yang benar terhadap seluruh data uji. Nilai akurasi yang tinggi menunjukkan bahwa model mampu mengklasifikasikan sebagian besar *intent* dengan benar. Precision mengukur tingkat ketepatan prediksi pada masing-masing kelas *intent*. Nilai precision yang tinggi menunjukkan bahwa model jarang memberikan prediksi positif yang salah. Recall mengukur kemampuan model dalam mengenali seluruh data dari suatu kelas *intent*. Nilai recall yang tinggi menunjukkan bahwa model mampu menangkap sebagian besar contoh dari kelas tersebut. F1-score merupakan rata-rata harmonis antara precision dan recall, sehingga memberikan gambaran keseimbangan antara kedua metrik tersebut.

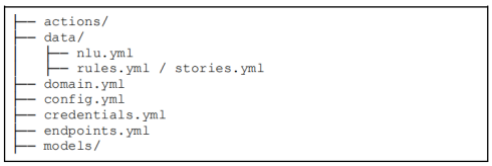
3. Hasil dan Pembahasan

3.1 Hasil Implementasi Sistem

Implementasi sistem dilakukan dengan mengintegrasikan *chatbot* berbasis RASA ke dalam aplikasi web Laravel. Proses ini mencakup instalasi RASA, pembuatan *webhook middleware*, komunikasi API menggunakan HTTP request, dan penyajian tampilan *chatbot* pada halaman *frontend* Laravel.

3.1.1 Instalasi dan *setup* RASA

Instalasi RASA dilakukan pada lingkungan virtual menggunakan perintah `pip install RASA` dan `RASA init`. Struktur folder yang dihasilkan seperti pada Gambar 2.



Gambar 2. Struktur Folder *Framework* RASA

Struktur ini menyediakan komponen yang diperlukan untuk mendefinisikan intent, respons, logika dialog, dan koneksi dengan *web frontend.*

3.1.2 Implementasi Proyek RASA

Pada implementasi ini, komponen penting yang dikonfigurasi meliputi file `nlu.yml`, `domain.yml`, dan `rules.yml`, yang berperan dalam memahami maksud pengguna, merespons dengan tepat, serta mengatur alur percakapan. File `nlu.yml` digunakan untuk melatih model mengenali *intent*). Gambar 3 adalah contoh file `nlu.yml`.

```
version: "3.1"

nlu:
- intent: informasi_layanan
examples: |

- Apa saja layanan yang ditawarkan oleh Bank
Sampah Sriwilis?
- Apa itu Bank Sampah Sriwilis?
- Apa keuntungan menjadi anggota Bank Sampah
Sriwilis?

- intent: tanya_jenis_sampah
examples: |

- Jenis sampah apa saja yang diterima di Bank
Sampah Sriwilis?
- Sampah jenis apa yang bisa saya setor ke Bank
Sampah Sriwilis?
- Sampah apa yang diterima di Bank Sampah
Sriwilis?
```

Gambar 3. Contoh File `nlu.yml`

File `domain.yml` digunakan untuk mendefinisikan *intent*, respons, dan aksi *chatbot*. Gambar 4 menunjukkan contoh file `domain.yml`.

```
version: "3.1"

intents:
- informasi_layanan
- tanya_jenis_sampah

responses:

utter_informasi_layanan:
- text: |

Bank Sampah Sriwilis menawarkan berbagai layanan:
1. Penukaran sampah dengan saldo**: Anda dapat
menukarkan sampah yang telah dikumpulkan dengan
saldo yang bisa digunakan untuk berbagai
keperluan.

utter_tanya_jenis_sampah:
- text: |

Bank Sampah Sriwilis menerima jenis sampah sebagai
berikut:
1. Plastik: Botol plastik, kantong plastik, dan
wadah plastik lainnya.
2. Kertas: Koran, majalah, karton, dan kertas
bekas lainnya.
3. Kaca: Botol kaca, wadah kaca lainnya.
4. Logam: Kaleng, aluminium foil, dan lainnya.
```

Gambar 4. Contoh File `nlu.yml`

Untuk mengatur alur percakapan berbasis aturan digunakan file `rules.yml`. File ini mengatur aturan logika sederhana berbasis pola input pengguna. Gambar 5 menunjukkan contoh file `rules.yml`.

```
version: "3.1"

- rule: Provide service information
steps:
- intent: informasi_layanan
- action: utter_informasi_layanan

- rule: Ask about types of waste
steps:
- intent: tanya_jenis_sampah
- action: utter_tanya_jenis_sampah
```

Gambar 5. Contoh File `nlu.yml`

3.1.3 Pelatihan Model dan Menjalankan *Server*

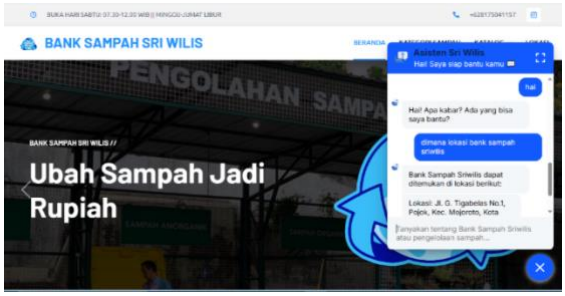
Pelatihan model dilakukan dengan perintah `RASA train`, dan *server* RASA dijalankan menggunakan `RASA run --enable-api` serta `RASA run actions`.

3.1.4 Konfigurasi Kebijakan (*Policies*)

RulePolicy adalah salah satu jenis kebijakan (*policy*) dalam RASA yang digunakan untuk mengatur alur percakapan berbasis aturan eksplisit yang telah ditentukan dalam file `rules.yml`. konfigurasi `core_fallback_action_name` dan `fallback_action_name` keduanya mengacu pada tindakan yang akan dijalankan ketika *chatbot* tidak dapat memutuskan langkah berikutnya atau tidak yakin terhadap *intent* pengguna. Dalam hal ini, keduanya diarahkan ke `action_default_fallback`, yaitu aksi bawaan yang biasanya memberikan respon seperti "Maaf, saya tidak mengerti pertanyaan Anda."

3.2 Hasil Implementasi Antarmuka Sistem

Pengguna dapat mengakses *chatbot* melalui widget pada halaman website dan berinteraksi secara real-time. Setiap pesan yang dikirimkan akan diproses oleh RASA Server untuk mengidentifikasi *intent* dan *entity* sebelum menghasilkan respons yang sesuai. Gambar 6 menunjukkan hasil implementasi fitur *chatbot* pada website bank sampah Sriwilis.



Gambar 6. Fitur *Chatbot* Bank Sampah Sriwilis

3.2 Hasil Pengujian Kinerja Model

Pengujian kinerja dilakukan terhadap tiga konfigurasi pipeline, yaitu Model A (DIET *baseline*), Model B (DIET + fitur tambahan), dan Model C (*Logistic Regression*). Konfigurasi tiap model ditunjukkan pada Tabel 3, 4, dan 5.

Tabel 3. Konfigurasi Model A

Komponen	Parameter	Nilai
Tokenizer	WhitespaceTokenizer	Default
Featurizer	CountVectorsFeaturizer	Lowercase= True
Classifier	DIETClassifier	epoch=100
Entity Mapping	EntitySynonymMapper	Default
Respons	ResponseSelector	epoch=100

Tabel 4. Konfigurasi Model B

Komponen	Parameter	Nilai
Tokenizer	WhitespaceTokenizer	Default
Featurizer	CountVectorsFeaturizer	Lowercase= True
	RegexFeaturizer	Default
	LexicalSyntaxticFeaturizer	Default
Classifier	DIETClassifier	epoch=100
Entity Mapping	EntitySynonymMapper	Default
Respons	ResponseSelector	epoch=100

Tabel 5. Konfigurasi Model C

Komponen	Parameter	Nilai
Tokenizer	WhitespaceTokenizer	Default
Featurizer	CountVectorsFeaturizer	Lowercase= True
Classifier	LogisticRegressionClassifier	max_iter = 200

Evaluasi terhadap tiap jenis model dilakukan menggunakan data uji sebesar 20% dari total dataset dengan metrik *accuracy, precision, recall,* dan F1-*score*. Tabel 6 menunjukkan perbandingan kinerja model.

Tabel 6. Perbandingan Kinerja Model

Model	Akurasi	Presisi	Recall	F1-Score
Model A	0.91	0.90	0.89	0.89
Model B	0.93	0.93	0.91	0.91
Model C	0.86	0.85	0.83	0.84

Berdasarkan hasil pada Tabel 6, model A yang menggunakan DIETClassifier sebagai *baseline* menunjukkan performa yang cukup baik dengan nilai accuracy sebesar 0.91. Model ini mampu mengklasifikasikan sebagian besar intent dengan benar dan menunjukkan kestabilan antara *precision* dan *recall*. Hal ini sejalan dengan penelitian (Narendra & Setyaningsih, 2021) yang menyatakan bahwa DIET efektif dalam menangani klasifikasi intent dan ekstraksi entitas secara simultan menggunakan arsitektur *transformer* ringan. Namun demikian, masih terdapat beberapa kesalahan klasifikasi pada intent yang memiliki kemiripan kosakata.

Model B menunjukkan performa terbaik dengan *accuracy* sebesar 0.93 dan F1-score sebesar 0.91. Peningkatan performa ini menunjukkan bahwa penambahan ekstraksi fitur dengan RegexFeaturizer dan LexicalSyntacticFeaturizer membantu model dalam membedakan pola kalimat yang memiliki struktur serupa. Penambahan fitur leksikal memungkinkan sistem mengenali variasi morfologi dan pola token tertentu yang relevan dengan intent spesifik. Dengan demikian, konfigurasi *pipeline* yang lebih kaya fitur terbukti meningkatkan kemampuan generalisasi model.

Model C yang menggunakan *Logistic Regression Classifier* menunjukkan performa terendah dengan akurasi sebesar 0.86. Meskipun masih mampu mengenali *intent* dasar seperti *greeting* dan *goodbye* dengan baik, model ini mengalami penurunan performa pada *intent* yang memiliki variasi struktur kalimat yang kompleks.

Sebagai contoh, *intent* jenis_sampah dan harga_sampah sering kali memiliki kemiripan struktur kalimat karena sama-sama menggunakan kata tanya seperti “apa” atau “berapa”. Pada Model C yang menggunakan pendekatan *bag-of-words*, kesalahan klasifikasi lebih sering terjadi karena model hanya mempertimbangkan frekuensi kemunculan kata tanpa memahami pola linguistik yang lebih kompleks. Sebaliknya, pada Model B, penambahan fitur leksikal membantu model mengenali perbedaan pola frasa, seperti keberadaan kata “harga” yang dominan pada intent harga_sampah dan frasa “apa saja” yang lebih umum pada intent jenis_sampah.

Peningkatan performa juga terlihat pada intent cek_saldo, yang memiliki variasi kalimat seperti “Berapa saldo saya?”, “Uang saya tinggal berapa?”, dan “Cek tabungan saya”. Variasi struktur tersebut menuntut model untuk memahami hubungan semantik antar kata. Model B menunjukkan recall yang lebih tinggi dibandingkan Model A dan C, yang menunjukkan bahwa kombinasi arsitektur DIETClassifier dan fitur tambahan memberikan kemampuan generalisasi yang lebih baik terhadap variasi bahasa alami.

4 Kesimpulan

Penelitian ini berhasil mengembangkan *chatbot* berbasis *Framework* RASA pada *Website* Bank Sampah Sriwilis dengan menerapkan metode pengembangan *Waterfall* dan melakukan eksperimen terhadap beberapa konfigurasi *pipeline Natural Language Understanding* (NLU). Sistem yang dibangun mampu mengenali intent pengguna serta memberikan respons secara *real-time*.

Berdasarkan hasil pengujian kinerja model, konfigurasi *pipeline* berpengaruh terhadap performa klasifikasi *intent*. Model berbasis DIETClassifier menunjukkan performa yang lebih baik dibandingkan LogisticRegressionClassifier, yang mengindikasikan bahwa pendekatan berbasis *transformer* lebih efektif dalam menangani variasi bahasa alami dibandingkan metode *machine learning* klasik berbasis frekuensi kata. Selain itu, penambahan komponen *RegexFeaturizer* dan *LexicalSyntacticFeaturizer* pada Model B memberikan peningkatan accuracy dan F1-score dibandingkan model *baseline*, sehingga menunjukkan bahwa strategi *feature engineering* masih relevan dalam meningkatkan kualitas klasifikasi.

Pengembangan selanjutnya dapat dilakukan dengan memperluas jumlah dan variasi dataset untuk meningkatkan kemampuan generalisasi model. Selain itu, integrasi *chatbot* dengan basis data aktual untuk fitur seperti tukar_saldo dapat meningkatkan fungsionalitas sistem. Penelitian lanjutan juga dapat mengeksplorasi penggunaan *pretrained language models* atau pendekatan berbasis *transformer* yang lebih besar untuk meningkatkan akurasi klasifikasi.

Daftar Pustaka:

Amelia, R., & Sutabri, T. (2024). Analisis Strategi Sukses Model Bisnis Startup E-Commerce di Era Digital Menggunakan Metode Value Proposition Design. *Uranus : Jurnal Ilmiah Teknik Elektro, Sains Dan Informatika*, *2*(4), 23–40. https://doi.org/10.61132/uranus.v2i4.467

Apriliyanto, E., & Arief, R. (2020). *Chatbot* untuk Pengendalian Hama Tanaman Padi dengan Metode Artificial Intelligence Markup Language dan Normalisasi Identification Of Diseases In Rice Plant Using *Chatbot* With Methode Artificial Intelligence Markup Language and Normalization. *Research : Journal of Computer*, *3*(2), 67–73.

Arthana, I., Dewi, L., Seputra, K., & Marti, N. (2021). Undiksha Virtual Assistant (Shavira) : Integration Frequency Asked Question with Rasa *Framework*. *Jurnal Sains Dan Teknologi*, *10*(2), 264–273.

Astuti, W., Wibawa, A. P., Haviluddin, H., & Darwis, H. (2024). DIET Classifier Model Analysis for Words Prediction in Academic *Chatbot*. *ILKOM Jurnal Ilmiah*, *16*(1), 59–67. https://doi.org/10.33096/ilkom.v16i1.1598.59-67

Bocklisch, T., Faulkner, J., Pawlowski, N., & Nichol, A. (2017). *Rasa: Open Source Language Understanding and Dialogue Management*. http://arxiv.org/abs/1712.05181

Cannavaro, N. (2023). Aplikasi *Chatbot* untuk Layanan Akademik Menggunakan Platform RASA Open Source dengan Fitur Two Stage *Fallback*. *Jurnal Ilmu Komputer Dan Informatika*, *3*(1), 53–64. https://doi.org/10.54082/jiki.73

Fauzia, L., Hadiprakoso, R. B., & Girinoto. (2021). Implementation of *Chatbot* on University Website Using RASA *Framework*. *2021 4th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*, 373–378. https://doi.org/10.1109/ISRITI54043.2021.9702821

Gujjar, J. P., & Kumar, V. N. (2022). Open Source *Chatbot* Development *Framework*-RASA. *Asian Journal of Advances in Research*, *5*(1), 451–453. http://arxiv.org/abs/1712.05181

Ketut Resika Arthana, I., Joni Erawati Dewi, L., Agus Seputra, K., & Wayan Marti, N. (2021). Undiksha Virtual Assistant (Shavira) : Integration Frequency Asked Question with Rasa *Framework*. *Jurnal Sains Dan Teknologi*, *10*(2).

Muliyono, M., & Sumijan, S. (2021). Identifikasi *Chatbot* dalam Meningkatkan Pelayanan Online Menggunakan Metode Natural Language Processing. *Jurnal Informatika Ekonomi Bisnis*, 142–147. https://doi.org/10.37034/infeb.v3i4.102

Narendra, L. W., & Setyaningsih, E. R. (2021). Designing a Transactional Smart Assistant in Indonesian using Rasa *Framework*. *2021 7th International Conference on Electrical, Electronics and Information Engineering (ICEEIE)*, 1–6. https://doi.org/10.1109/ICEEIE52663.2021.9616946

Puspita, D., Wahyuni, E. D., & Suryanto, T. L. M. (2025). Application of NLP and Rasa for Intent Classification in Durga Historical Texts. *Bit-Tech*, *8*(2), 1336–1346. https://doi.org/10.32877/bt.v8i2.2887

Rachman, A., Mardhiyah, I., & Jannah, M. (2023). KLIK: Kajian Ilmiah Informatika dan Komputer Implementasi *Chatbot* FAQ pada Aplikasi Monev Kinerja Direktorat Jenderal Anggaran Menggunakan *Framework* Rasa Open Source. *Media Online*, *4*(1), 62–72. https://doi.org/10.30865/klik.v4i1.1020

Rasa Technologies. (2023). *Rasa Open Source Documentation*. https://rasa.com/docs/

Rosida, A. A., & Hadiono, K. (2024). IMPLEMENTASI *CHATBOT* BERBASIS *FRAMEWORK* RASA UNTUK SISTEM REKOMENDASI WISATA DI SEMARANG. *JIPI (Jurnal Ilmiah Penelitian Dan Pembelajaran Informatika)*, *9*(3), 1374–1384. https://doi.org/10.29100/jipi.v9i3.5380

Ruidungan, D. G. S., & Jacobus, A. (2021). *Chatbot* Development for an Interactive Academic Information Services using the Rasa Open Source *Framework*. *Jurnal Teknik Elektro Dan Komputer*, *10*(1), 61–68.

Supreetha, H. V., & Sandhya. S. (2022). Implementation of an Educational *Chatbot* using Rasa *Framework*. *International Journal of Innovative Technology and Exploring Engineering*, *11*(9), 29–35. https://doi.org/10.35940/ijitee.G9189.0811922

Talenggoran, R., Krisnawati, L. D., & Virginia, G. (2025). Implementation of RASA *Framework* To Build FAQ Bot System As Bureau Information Service 3. *Jurnal Terapan Teknologi Informasi*, *9*(2), 81–92. https://doi.org/10.21460/jutei.2025.92.431

Wahid, A. (2020). Analisis Metode Waterfall Untuk Pengembangan Sistem Informasi. *Jurnal Ilmu-Ilmu Informatika Dan Manajemen STMIK*, 1–5.

Halaman ini sengaja dikosongkan