

# Implementasi LLM Pada *Chatbot* PMB Universitas Muhammadiyah Sorong Menggunakan Metode RAG Berbasis *Website*

Syamsudin Aliphadji Talaohu<sup>\*1</sup>, Rendra Soekarta<sup>2</sup>, Muhammad Surahmanto<sup>3</sup>

<sup>1,2,3</sup> Prodi Teknik Informatika, Universitas Muhammadiyah Sorong

E-mail: <sup>\*1</sup>[syamsudintalaohu@gmail.com](mailto:syamsudintalaohu@gmail.com), <sup>2</sup>[rendrasoekarta@um-sorong.ac.id](mailto:rendrasoekarta@um-sorong.ac.id), <sup>3</sup>[surahmanto@um-sorong.ac.id](mailto:surahmanto@um-sorong.ac.id)

## Abstrak

Dalam beberapa tahun terakhir, pemanfaatan kecerdasan buatan (AI) dalam sektor pendidikan mengalami pertumbuhan yang pesat. Salah satu teknologi AI yang mulai banyak diterapkan adalah *chatbot* berbasis Large Language Model (LLM), yang dirancang untuk menyajikan informasi secara interaktif dan responsif. Dalam konteks Penerimaan Mahasiswa Baru (PMB), Universitas Muhammadiyah Sorong masih menghadapi tantangan dalam menyediakan layanan informasi yang cepat dan akurat, karena sistem yang digunakan masih berbasis manual melalui media sosial. Hal ini menyebabkan proses tanya jawab menjadi lambat dan kurang efisien, terutama pada saat puncak pendaftaran. Penelitian ini bertujuan untuk mengimplementasikan *chatbot* berbasis LLM dengan metode Retrieval-Augmented Generation (RAG) untuk mendukung layanan PMB di Universitas Muhammadiyah Sorong. Metode RAG memungkinkan *chatbot* untuk memahami bahasa alami dan mengakses informasi eksternal secara real-time. Hasil penelitian menunjukkan bahwa *chatbot* dapat merespons pertanyaan calon mahasiswa dengan akurat dan memperoleh skor usability testing 88,5%, yang menandakan sistem ini mudah digunakan dan bermanfaat. Selain itu, hasil pengujian menggunakan BERTScore menunjukkan bahwa *chatbot* yang dikembangkan memiliki performa yang tinggi dengan tingkat akurasi sebesar 93%, dengan rata-rata precision 90%, recall 95%, dan f1-score 92%. Hasil ini menunjukkan bahwa *chatbot* mampu memberikan jawaban dengan tingkat keakuratan yang tinggi, sehingga dapat meningkatkan efisiensi dan kualitas layanan PMB di Universitas Muhammadiyah Sorong.

**Kata kunci**— *Chatbot*, Large Language Model (LLM), Retrieval-Augmented Generation (RAG), Penerimaan Mahasiswa Baru (PMB).

## 1. PENDAHULUAN

Kemajuan teknologi kecerdasan buatan (AI) telah memberikan dampak signifikan dalam berbagai aspek kehidupan, termasuk pendidikan. Salah satu penerapan yang semakin populer adalah *chatbot* berbasis Large Language Model (LLM), yang mampu memberikan informasi secara interaktif dan responsive [1]. Dalam konteks Penerimaan Mahasiswa Baru (PMB), kecepatan dan akurasi informasi menjadi faktor penting untuk menarik minat calon mahasiswa [2]. Dengan kemampuan beroperasi sepanjang waktu tanpa membutuhkan tenaga manusia, *chatbot* berbasis AI menawarkan solusi inovatif untuk meningkatkan komunikasi antara lembaga pendidikan dan calon mahasiswa [3].

Namun, teknologi ini belum sepenuhnya diterapkan oleh semua institusi, termasuk Universitas Muhammadiyah Sorong, yang masih menghadapi tantangan dalam menyediakan layanan PMB secara optimal. Saat ini, respons terhadap pertanyaan calon mahasiswa baru masih dilakukan secara manual melalui media sosial tanpa adanya dukungan sistem terintegrasi, sehingga memakan waktu lebih lama dan memerlukan tenaga manusia untuk menjawab banyak

pertanyaan secara bersamaan, terutama selama puncak pendaftaran. Akibatnya, calon mahasiswa dapat merasa kurang terlayani dengan baik, yang berpotensi menurunkan jumlah pendaftar. Oleh karena itu, diperlukan solusi teknologi modern yang mampu meningkatkan efisiensi dan kecepatan layanan PMB untuk memenuhi kebutuhan calon mahasiswa dengan lebih baik.

Untuk mengatasi masalah tersebut, teknologi *chatbot* berbasis kecerdasan buatan (AI) menggunakan metode *Retrieval-Augmented Generation* (RAG) dapat menjadi solusi yang tepat. Metode ini memungkinkan *chatbot* untuk memahami bahasa alami melalui teknologi *Large Language Model* (LLM) dan mengakses data eksternal secara *real-time* dengan pendekatan RAG serta lebih fleksibel karena mampu beroperasi sepanjang waktu tanpa memerlukan tenaga manusia [4][5]. Teknologi ini juga dapat menangani banyak pertanyaan dalam waktu bersamaan tanpa menurunkan kualitas respons dengan layanan yang lebih cepat dan efisien serta meningkatkan kepuasan penggunaannya [6]. Salah satu model LLM yang populer adalah *GPT-3.5 Turbo* dari OpenAI, model ini banyak dipilih karena kemampuannya dalam memahami konteks percakapan dan memberikan respons cepat, akurat dan memiliki kestabilan dengan performa yang sudah teruji serta memiliki kemudahan integrasi melalui API [7][8]. Dengan memanfaatkan teknologi ini dapat membantu layanan penerimaan mahasiswa baru di Universitas Muhammadiyah Sorong untuk menangani banyak pertanyaan dari calon mahasiswa baru dalam waktu bersamaan dengan layanan yang lebih cepat dan efisien.

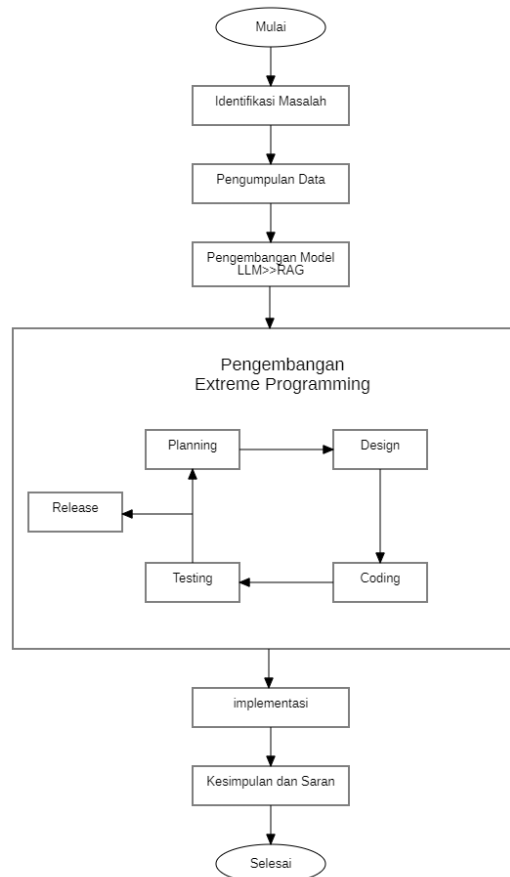
Penelitian terkait yang dilakukan oleh Lantana dkk (2023) dengan judul “Rancang Bangun *Chatbot* Berbasis *Rule-Based* Sebagai Pusat Informasi Calon Mahasiswa Baru di Universitas Nasional” Penelitian ini berfokus untuk mengembangkan *chatbot* berbasis *rule-based* dengan tujuan untuk menjadi pusat informasi calon mahasiswa baru di Universitas Nasional. Pendekatan yang digunakan adalah *rule-based* dengan logika *if/then*. Hasilnya menunjukkan bahwa *chatbot* yang dikembangkan dapat menjawab pertanyaan umum dari calon mahasiswa baru [9]. Namun, pendekatan yang digunakan dalam studi ini memiliki kelemahan signifikan dalam memahami variasi bahasa alami, karena *chatbot* hanya mampu merespon pertanyaan sesuai pola yang telah ditentukan, sehingga gagal menangani input di luar skenario [10]. Penelitian lain oleh Ishlahuddin dkk (2020) dengan judul “Sistem tanya jawab dengan menggunakan telegram *chatbot* di STMIK Muhammadiyah Paguyangan Brebes” juga menggunakan metode *rule-based*, Namun memiliki keterbatasan yaitu sistem menjadi kurang efisien karena respons melambat seiring bertambahnya aturan serta *chatbot* hanya melayani menu yang telah tersedia [11]. Sebagai perbandingan, penelitian selanjutnya menawarkan kontribusi baru dengan mengintegrasikan teknologi *Large Language Model* (LLM) dengan metode *Retrieval-Augmented Generation* (RAG) pada *chatbot* untuk mendukung layanan Penerimaan Mahasiswa Baru (PMB). Teknologi ini memungkinkan sistem memberikan respons fleksibel terhadap pertanyaan dalam bahasa alami [12], termasuk variasi pertanyaan yang belum pernah dicakup pada penelitian berbasis aturan (*rule-based*) sebelumnya.

## 2. METODE PENELITIAN

### 2.1 Tahapan Penelitian

Penelitian ini melalui sejumlah tahapan yang terstruktur, dimulai dari proses identifikasi masalah, dilanjutkan dengan pengumpulan data yang relevan, kemudian pengembangan *chatbot* serta model *Large Language Model* (LLM) dengan pendekatan *Retrieval-Augmented Generation* (RAG). Setelah itu, dilanjutkan dengan tahapan pengembangan sistem menggunakan metode *Extreme Programming*, implementasi sistem, hingga tahap akhir berupa penyusunan kesimpulan dan saran. Rangkaian langkah-langkah dalam penelitian ini dapat dilihat secara lebih jelas pada gambar 1.

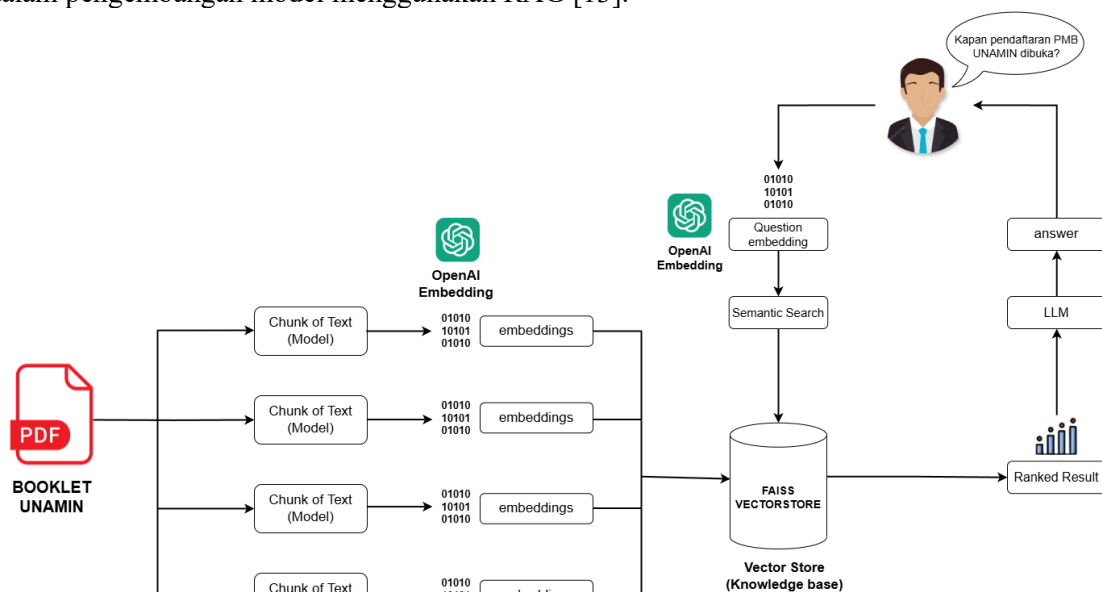
---



Gambar 1 Alur Penelitian

## 2. 2 Pengembangan Model LLM Menggunakan Metode RAG

Pada tahap ini, peneliti akan mengembangkan model LLM menggunakan metode RAG untuk meningkatkan kinerja chatbot PMB Universitas Muhammadiyah Sorong dalam menjawab pertanyaan terkait penerimaan mahasiswa baru. Berikut adalah langkah-langkah atau arsitektur dalam pengembangan model menggunakan RAG [13].



Gambar 2 Alur LLM Menggunakan RAG

Berdasarkan alur dari sistem pada gambar 2 tersebut dapat dijelaskan sebagai berikut :

1. *Pemanggilan Knowledge Base*  
Pada tahap pertama yaitu pemanggilan *knowledge base*, sistem akan menggunakan data dari sumber *Booklet UNAMIN* sebagai basis pengetahuan. Data dari pdf ini akan diproses untuk dijadikan sumber informasi yang dapat diakses oleh *chatbot*.
2. *Preprocessing Data*  
Pada fase ini dilakukan proses pra-pemrosesan data, yang mencakup pembagian data dari pdf menjadi *chunk of text*. *Chunking* adalah proses membagi teks menjadi unit-unit yang lebih kecil untuk mempermudah pemahaman dan penyajian informasi. Bagian-bagian teks ini dibagi sesuai dengan struktur sintaksis seperti frasa kata benda atau frasa kata kerja. Setiap potongan teks yang dihasilkan memiliki panjang maksimal 2048 karakter.
3. *Embedding*  
Setelah *chunking*, tahap berikutnya adalah *embedding*, yaitu mengonversi teks ke dalam format yang dapat dipahami oleh mesin, berupa vektor numerik (contoh: 0101). Setiap bagian data (*chunk of text*) akan diubah menjadi representasi vektor menggunakan *OpenAI Embeddings*. *OpenAI Embeddings* akan mengonversi setiap bagian data menjadi vektor numerik yang menggambarkan makna semantik dari teks yang bersangkutan. Dalam proses ini, kata-kata dengan makna serupa akan memiliki representasi nilai yang saling mendekati.
4. *Membangun Vector Store*  
Pada fase ini potongan-potongan teks yang telah diubah menjadi vektor disimpan dalam *vector store* yang berfungsi sebagai basis pengetahuan untuk *chatbot*. Dalam penelitian ini, peneliti menggunakan pustaka (*library*) *FAISS* untuk menyimpan vektor tersebut. Dengan ini, data dari sumber pdf yang diolah menjadi vektor siap untuk digunakan sebagai *knowledge base* bagi sistem *chatbot*.
5. *Pencarian Semantik*  
Ketika seorang pengguna mengajukan pertanyaan ke sistem, pertanyaan tersebut juga akan diubah menjadi representasi vektor melalui proses *embedding* menggunakan *OpenAI Embeddings*. Setelah itu, dilakukan pencarian kemiripan (*similarity search*) pada *vector store* untuk mencari dan menemukan beberapa jawaban yang relevan dengan pertanyaan tersebut.
6. *Penyaringan dan Generasi Jawaban*  
Setelah beberapa jawaban relevan ditemukan melalui pencarian semantik, jawaban-jawaban tersebut akan diurutkan berdasarkan tingkat relevansinya. *LLM* akan memanfaatkan konteks yang diperoleh dari pencarian *semantic*, jawaban yang paling relevan akan diproses oleh *LLM* dan akan menghasilkan jawaban yang paling tepat sesuai dengan pertanyaan pengguna

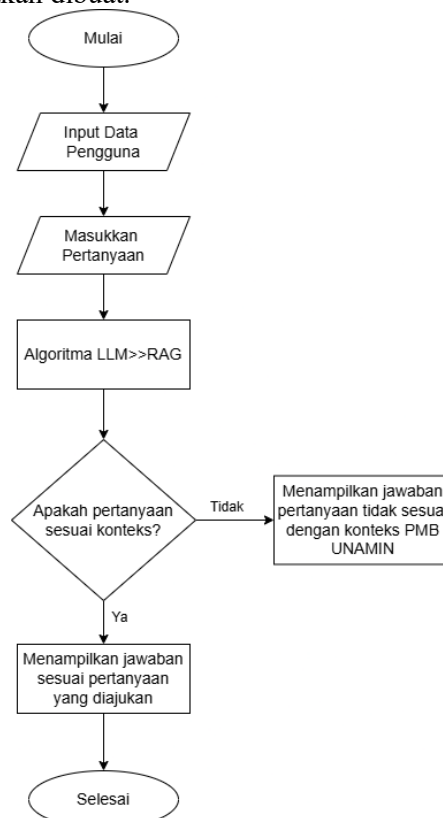
### 2.3 Pengembangan Sistem (Extreme Programming)

Pada tahap pengembangan sistem, peneliti akan merancang dan membangun sebuah sistem untuk mengembangkan perangkat lunak menggunakan Extreme Programming (XP) [14][15]. Berikut ini adalah tahapan-tahapan yang akan diterapkan:

1. *Planning*  
Pada fase ini, dimulai dengan merinci konteks bisnis sistem, mendefinisikan output, mengidentifikasi fitur sistem, menetapkan fungsi sistem yang dibuat, menentukan estimasi waktu dan biaya pengembangan sistem, serta menggambarkan alur pengembangan sistem.
-

## 2. Design

Pada fase perancangan desain ini, peneliti akan menyusun diagram alir seperti *Flowchart* yang berguna untuk memudahkan pemahaman alur logika dan memvisualisasikan hubungan antar langkah secara sistematis. Berikut ini adalah representasi visual dari *flowchart* sistem yang akan dibuat.



Gambar 3 *Flowchart* Sistem

Berdasarkan gambar 3 pada *flowchart* sistem di atas, proses dimulai dengan input data pengguna seperti nama, asal sekolah, minat bakat, dan lain-lain, selanjutnya pengguna memasukkan pertanyaan yang ingin diajukan terkait PMB UNAMIN, lalu akan diproses oleh algoritma LLM dengan metode RAG. Sistem kemudian memeriksa apakah pertanyaan tersebut sesuai konteks PMB UNAMIN atau tidak. Jika pertanyaan tidak sesuai konteks, sistem akan memberikan jawaban bahwa pertanyaan tidak sesuai konteks PMB UNAMIN. Namun, jika pertanyaan sesuai, sistem akan menampilkan jawaban sesuai dengan pertanyaan yang diajukan oleh pengguna.

## 3. Coding

Pada fase ini, peneliti mengembangkan sistem dengan menggunakan bahasa pemrograman *Python* dan mengimplementasikan metode RAG untuk *Large Language Model*. *Framework Flask* digunakan untuk membangun antarmuka pengguna (*interface*) sistem. Kode program ditulis menggunakan *Visual Studio Code*, dan pengembangan dilakukan dengan metode *Extreme Programming* (XP) untuk memastikan kualitas dan efisiensi pengembangan sistem.

## 4. Testing

Pada fase ini, pengujian pertama akan dilakukan menggunakan *BERTScore* untuk menilai akurasi jawaban yang dihasilkan oleh sistem, selanjutnya dilakukan pengujian awal sistem dengan menggunakan metode *blackbox testing*. Dalam pengujian tersebut akan terfokuskan pada pemeriksaan apakah fitur-fitur yang telah dibuat dalam sistem sesuai dengan perpindahan *activity*. Pada tahap kedua, pengujian dilakukan melalui *usability testing* dimana pengguna sistem langsung menguji sistem tersebut.

### 3. HASIL DAN PEMBAHASAN

#### 3.1 Hasil Pengumpulan Data

Pada penelitian ini, peneliti menggunakan sumber *booklet* UNAMIN yang kemudian isinya dipindahkan lagi agar hanya berupa teks saja sebagai sumber *knowledge base* atau basis pengetahuan. Data pdf ini terbagi menjadi 2 bagian yaitu sumber 1 dan sumber 2, seperti pada tabel 1.

Tabel 1 Data *Knowledge Base*

| No | Nama PDF | Keterangan            |
|----|----------|-----------------------|
| 1. | Sumber 1 | Informasi Pendaftaran |
| 2. | Sumber 2 | Informasi Kampus      |

Berdasarkan tabel 1 tersebut, data dari *booklet* yang dipindahkan kemudian dibagi menjadi 2 pdf, yaitu sumber 1 dengan isi informasi pendaftaran dan sumber 2 dengan isi informasi kampus. Hal ini dilakukan agar pemrosesan data pdf dapat lebih mudah dibaca oleh *chatbot*.

#### 3.2 Pengembangan Model LLM Menggunakan Metode RAG

##### 1. Pemanggilan *Knowledge Base*

Sumber *knowledge base* yang digunakan pada penelitian ini berasal dari *booklet* UNAMIN yang dibagi menjadi 2 pdf. Pembagian ini bertujuan untuk mempermudah sistem dalam mengelompokkan informasi sehingga *chatbot* dapat memberikan jawaban yang lebih akurat sesuai dengan kebutuhan pengguna. Proses pemanggilan *knowledge base* ini dapat dilihat pada gambar 4.

```
pdf_directory = "pd"

# **Fungsi untuk Memuat Semua PDF**
def load_pdfs():
    pdf_files = glob.glob(os.path.join(pdf_directory, "*.pdf"))
    all_docs = []

    for pdf_file in pdf_files:
        print(f"♦ Memproses file: {pdf_file}")
        loader = PyPDFLoader(pdf_file)
        data = loader.load()
```

Gambar 4 Pemanggilan *Knowledge Base*

Berdasarkan gambar 4 tersebut data dari sumber pdf ini dimuat dengan mencari dan membaca semua file pdf dalam direktori penyimpanan atau folder menggunakan fungsi *load pdf*, proses ini akan secara otomatis membaca setiap dokumen pdf yang ada menggunakan library dari *PyPDFLoader*.

##### 2. *Preprocessing* Data

Pada tahap ini merupakan tahap *preprocessing* data untuk memastikan bahwa informasi dalam sebuah dokumen pdf tadi dapat diolah dengan lebih efisien dalam sistem *chatbot* menggunakan *Retrieval-Augmented Generation (RAG)*. Data atau *chunk* ini akan diolah dengan menggunakan *splitter* teks. Berikut kode *preprocessing* data dapat dilihat pada gambar 5.

```
# Split dokumen agar lebih efisien dalam FAISS
text_splitter = RecursiveCharacterTextSplitter(chunk_size=2048, chunk_overlap=248)
docs = text_splitter.split_documents(data)

# Tambahkan ke daftar dokumen
all_docs.extend(docs)

return all_docs
```

Gambar 5 *Preprocessing* Data

Pada gambar 5 di atas data dari pdf dibagi menjadi bagian-bagian kecil (*chunk*) menggunakan metode *RecursiveCharacterTextSplitter*. Proses ini bertujuan untuk

membagi teks menjadi unit yang lebih kecil sehingga lebih mudah dikelola dan diproses oleh sistem pencarian berbasis vektor. Setiap chunk teks memiliki ukuran maksimal 2048 karakter dengan *overlap* sebesar 248 karakter untuk menjaga kesinambungan informasi di antara potongan teks.

### 3. *Embedding*

Pada tahap ini, dilakukan proses *embedding*, yaitu mengonversi teks yang telah diproses sebelumnya menjadi representasi vektor numerik agar dapat dipahami oleh mesin. Proses ini memungkinkan *chatbot* untuk menangkap makna semantik dari setiap potongan teks dalam knowledge base.

### 4. Membangun *Vector Store*

Pada tahap ini adalah tahap membangun *vector store* untuk menyimpan hasil *embedding* dalam format yang dapat diakses dengan cepat oleh *chatbot*. Pada penelitian ini peneliti menggunakan *FAISS* (*Facebook AI Similarity Search*) sebagai *vector store*. Berikut kode yang digunakan untuk membangun *vector store* dapat dilihat pada gambar 7.

```
FAISS_PATH = "faiss_index"
if os.path.exists(FAISS_PATH):
    print("♦ Memuat FAISS dari VectorStore...")
    vectorstore = FAISS.load_local(FAISS_PATH, embeddings)
else:
    print("♦ Membuat FAISS baru...")
    pdf_docs = load_pdfs() # Muat PDF sebelum membuat FAISS
```

Gambar 7 Membangun *Vectorstore*

Pada gambar 7, terlihat bahwa sistem terlebih dahulu memeriksa apakah *vector store* *FAISS* sudah ada di dalam direktori penyimpanan. Jika sudah ada, sistem akan langsung memuat kembali *FAISS* dari penyimpanan agar dapat digunakan tanpa perlu membangun ulang. Namun, jika belum ada, sistem akan membuat *FAISS* baru dengan memproses file pdf yang telah diekstraksi sebelumnya.

### 5. Pencarian Semantik

Pada fase ini dilakukan pencarian semantik untuk mencari jawaban yang relevan berdasarkan pertanyaan pengguna. sistem akan mengonversi pertanyaan yang diberikan oleh pengguna ke dalam bentuk vektor kemudian mencocokkannya dengan vektor-vektor yang tersimpan dalam *vector store*. Dalam hal ini peneliti menggunakan  $k=15$  atau 15 hasil teratas agar *chatbot* bisa memberikan jawaban yang paling sesuai dengan pertanyaan pengguna.

### 6. Penyaringan dan Generasi Jawaban

Pada tahap ini, sistem akan menyaring hasil yang ditemukan dan menggunakan *Large Language Model (LLM)* untuk menghasilkan jawaban yang paling tepat berdasarkan konteks yang ada. Berikut kode yang digunakan untuk penyaringan dan generasi jawaban dapat dilihat pada gambar 9.

```
updated_prompt_template = ChatPromptTemplate.from_messages(
    [
        ("system", f"{system_prompt}\n\nInformasi Pengguna: {user_context}"),
        ("human", f"{history_context}\n\nUser: {user_input}"),
    ]
)

# Mendapatkan jawaban dari model menggunakan RAG
question_answer_chain = create_stuff_documents_chain(llm, updated_prompt_template)
rag_chain = create_retrieval_chain(retriever, question_answer_chain)
response = rag_chain.invoke({"input": user_input})
ai_response = response["answer"]
```

Gambar 9 Penyaringan dan Generasi Jawaban

Pada gambar 9 terdapat *updated prompt template* yang berguna untuk menggabungkan informasi pengguna dan riwayat percakapan sebelumnya untuk memberikan konteks yang lebih lengkap dalam menjawab pertanyaan. Kemudian, sistem menggunakan *RAG chain* untuk menggabungkan hasil pencarian yang relevan dengan *question-answer chain* yang berfungsi untuk memproses input dan menghasilkan jawaban.

### 3.3 Pengujian BERTScore

Pengujian model *LLM* pada *chatbot* dilakukan dengan menggunakan *BERTScore* untuk menilai akurasi jawaban yang dihasilkan oleh model *GPT-3.5 Turbo*. *BERTScore* menilai kemiripan makna antara respon atau jawaban dari sistem dan jawaban acuan dengan mempertimbangkan representasi vektor kata yang dihasilkan. Sampel pertanyaan yang digunakan dalam pengujian ini dapat dilihat pada tabel 2.

Tabel 2 Sampel Pertanyaan

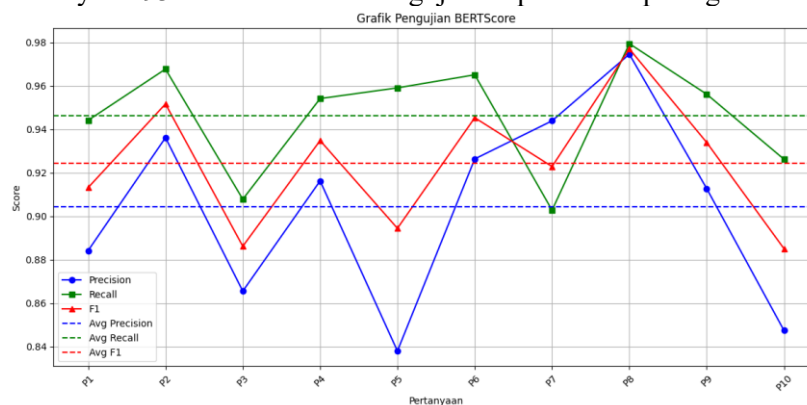
| No. | Pertanyaan                                                                   |
|-----|------------------------------------------------------------------------------|
| 1.  | Kapan waktu pendaftaran dibuka?                                              |
| 2.  | Apa saja persyaratan pendaftaran?                                            |
| 3.  | Bagaimana cara mendaftar online?                                             |
| 4.  | Bagaimana cara mendaftar offline?                                            |
| 5.  | Dimana lokasi tempat pendaftaran?                                            |
| 6.  | Apa saja program studi yang ada di Universitas Muhammadiyah Sorong (UNAMIN)? |
| 7.  | Apa akreditasi kampus UNAMIN di tanah papua?                                 |
| 8.  | Apa visi dan misi UNAMIN?                                                    |
| 9.  | Dimana lokasi kampus UNAMIN?                                                 |
| 10. | Apa saja fasilitas di kampus Universitas Muhammadiyah Sorong?                |

Berdasarkan tabel 2 tersebut pengujian dilakukan dengan menggunakan 10 sampel pertanyaan umum yang biasanya diajukan oleh calon mahasiswa baru mengenai topik PMB. Berikut merupakan hasil pengujian yang didapatkan yaitu *precision*, *recall* dan *f-1 score*.

Tabel 3 Hasil Pengujian BERTScore

| Pertanyaan | BERTScore (%) |          |            |
|------------|---------------|----------|------------|
|            | <i>P</i>      | <i>R</i> | <i>F-1</i> |
| 1          | 88%           | 94%      | 91%        |
| 2          | 94%           | 97%      | 95%        |
| 3          | 87%           | 91%      | 89%        |
| 4          | 92%           | 95%      | 93%        |
| 5          | 84%           | 96%      | 89%        |
| 6          | 93%           | 97%      | 95%        |
| 7          | 94%           | 90%      | 92%        |
| 8          | 97%           | 98%      | 98%        |
| 9          | 91%           | 96%      | 93%        |
| 10         | 85%           | 93%      | 89%        |
| Rata-rata  | 90%           | 95%      | 92%        |

Pada tabel 3 tersebut menunjukkan hasil pengujian dengan menggunakan *BERTScore* menghasilkan rata-rata *precision* 90%, *recall* 95%, dan *f-1 score* 92%, serta hasil akurasi yang didapatkan yaitu 93%. Grafik Hasil Pengujian dapat dilihat pada gambar 10 berikut.

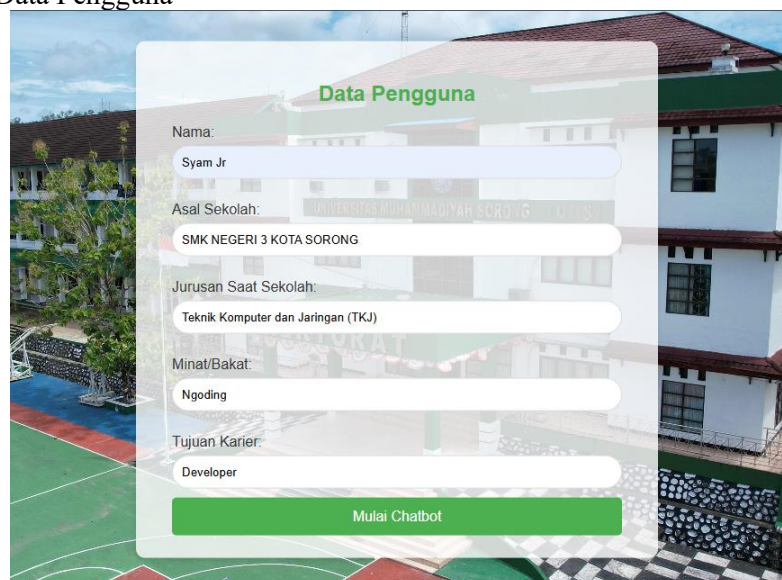


Gambar 10 Grafik Hasil Pengujian BERTScore

Pada gambar 10 tersebut merupakan grafik evaluasi model berdasarkan *Precision*, *Recall*, dan *F1-Score* pada sepuluh pertanyaan, dengan garis putus-putus yang merepresentasikan rata-rata masing-masing metrik.

### 3.4 Implementasi Website

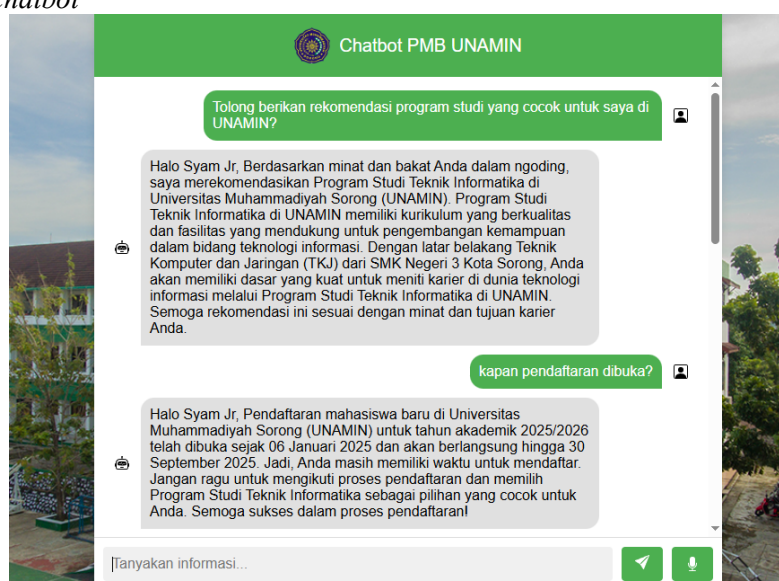
#### 1. Halaman Data Pengguna



Gambar 11 Implementasi Halaman Data Pengguna

Pada gambar 11 merupakan hasil implementasi halaman data pengguna dari website yang telah dibuat. Halaman ini terdiri dari beberapa kolom input. Pengguna akan mengisi data tersebut sebelum melanjutkan ke halaman *chatbot*. Setelah data diinput, pengguna dapat menekan tombol mulai *chatbot* untuk masuk pada halaman *chatbot*. Halaman ini bertujuan untuk mengumpulkan informasi dasar dari pengguna yang nantinya digunakan dalam memperkaya konteks dari memori *chatbot*, sehingga *chatbot* juga dapat memberikan rekomendasi program studi yang sesuai di Universitas Muhammadiyah Sorong (UNAMIN), terutama jika pengguna masih merasa bingung dalam memilih program studi yang tepat untuk mereka,

#### 2. Halaman Chatbot



Gambar 11 Implementasi Halaman Chatbot

Pada gambar 11 merupakan hasil implementasi halaman *chatbot* dari website yang telah dibuat, pada halaman ini sebelum pengguna mengajukan pertanyaan terdapat beberapa tombol navigasi di bagian atas yang berfungsi sebagai *quick suggestion* atau saran cepat, pengguna juga bisa mengajukan pertanyaan terkait rekomendasi program studi sesuai data yang diisikan pada halaman data pengguna sebelumnya. Di bagian bawah, terdapat kolom untuk mengetikkan pertanyaan atau menggunakan *microphone* untuk bertanya memakai suara. Setelah pertanyaan diinputkan lalu menekan tombol kirim atau dengan menggunakan suara, sistem akan memprosesnya dan memberikan hasil respons dalam bentuk teks.

#### 4. KESIMPULAN

Berdasarkan implementasi dan pengujian yang telah dilakukan terhadap *chatbot* berbasis website untuk mendukung layanan Penerimaan Mahasiswa Baru (PMB) di Universitas Muhammadiyah Sorong menggunakan teknologi *Large Language Model* (LLM) dengan metode *Retrieval-Augmented Generation* (RAG), maka diperoleh beberapa kesimpulan sebagai berikut:

1. Sistem telah berhasil diimplementasikan pada *chatbot* berbasis website dengan menggunakan teknologi *Large Language Model* (LLM) dan metode *Retrieval-Augmented Generation* (RAG), *chatbot* mampu memberikan respons yang relevan dan akurat terhadap pertanyaan calon mahasiswa terkait Penerimaan Mahasiswa Baru (PMB) di Universitas Muhammadiyah Sorong, serta memperoleh hasil evaluasi *Usability Testing* dengan skor sebesar 88,5%, yang mengindikasikan bahwa sistem ini mudah digunakan dan berguna untuk calon mahasiswa.
2. Hasil pengujian *chatbot* menggunakan *BERTScore* menunjukkan bahwa *chatbot* yang diimplementasikan dengan metode *RAG* memiliki performa yang tinggi dengan mendapatkan hasil tingkat akurasi sebesar 93%, dengan rata-rata *precision* sebesar 90%, *recall* sebesar 95%, dan *f1-score* sebesar 92%. Hasil ini menunjukkan bahwa sistem *chatbot* dapat memberikan jawaban dengan tingkat keakuratan yang tinggi dalam mendukung layanan PMB di Universitas Muhammadiyah Sorong.

#### 5. SARAN

Peneliti menyadari bahwa penelitian ini masih belum sempurna. Dengan demikian menurut peneliti perlu adanya pengembangan untuk mendapatkan hasil yang lebih baik. Adapun saran untuk penelitian selanjutnya yaitu :

1. Menambahkan fitur pengenalan gambar agar *chatbot* dapat menerima dan memproses input dalam bentuk gambar.
2. Meningkatkan kemampuan pengenalan suara agar *chatbot* dapat memahami berbagai logat dan aksen daerah.
3. Mengintegrasikan *chatbot* dengan sistem pendaftaran online sehingga calon mahasiswa dapat langsung melakukan proses pendaftaran melalui *chatbot*.
4. Meningkatkan pemahaman *chatbot* terhadap konteks percakapan yang lebih kompleks agar respons yang diberikan semakin akurat dan natural.

#### DAFTAR PUSTAKA

- [1] H. Khandakar, S. A. Fazal, K. F. Afnan, and K. K. Hasan, "Implications of artificial intelligence chatbot models in higher education," *IAES Int. J. Artif. Intell.*, vol. 13, no. 4, pp. 3808–3813, 2024, doi: 10.11591/ijai.v13.i4.pp3808-3813.
  - [2] F. Y. Fiddin, A. Komarudin, and M. Melina, "Chatbot Informasi Penerimaan Mahasiswa Baru Menggunakan Metode FastText dan LSTM," *J. Appl. Comput. Sci. Technol.*, vol. 5, no. 1, pp. 33–39, 2024, doi: 10.52158/jacost.v5i1.648.
-

- [3] S. Rifky, “Dampak Penggunaan Artificial Intelligence Bagi Pendidikan Tinggi,” *Indones. J. Multidiscip. Soc. Technol.*, vol. 2, no. 1, pp. 37–42, 2024, doi: 10.31004/ijmst.v2i1.287.
  - [4] J. Chen, H. Lin, X. Han, and L. Sun, “Benchmarking Large Language Models in Retrieval-Augmented Generation,” *Proc. AAAI Conf. Artif. Intell.*, vol. 38, no. 16, pp. 17754–17762, 2024, doi: 10.1609/aaai.v38i16.29728.
  - [5] M. A. K. Raiaan *et al.*, “A Review on Large Language Models: Architectures, Applications, Taxonomies, Open Issues and Challenges,” *IEEE Access*, vol. 12, no. January, pp. 26839–26874, 2024, doi: 10.1109/ACCESS.2024.3365742.
  - [6] I. Ichsanudin, R. Pratama, and B. Sisephaputra, “Pengembangan Sistem Helpdesk Menggunakan Chatbot Dengan Metode Retrieval-Augmented Generation ( RAG ),” *JINACS (Journal Informatics Comput. Sci.)*, vol. 06, no. 2, pp. 696–710, 2024.
  - [7] M. Gupta, C. Akiri, K. Aryal, E. Parker, and L. Praharaj, “From ChatGPT to ThreatGPT: Impact of Generative AI in Cybersecurity and Privacy,” *IEEE Access*, vol. 11, no. August, pp. 80218–80245, 2023, doi: 10.1109/ACCESS.2023.3300381.
  - [8] A. P. Mulia, P. R. Piri, and C. Tho, “Usability Analysis of Text Generation by ChatGPT OpenAI Using System Usability Scale Method,” *Procedia Comput. Sci.*, vol. 227, pp. 381–388, 2023, doi: 10.1016/j.procs.2023.10.537.
  - [9] D. A. Lantana, S. Ningsih, T. Waluyo, and Winarsih, “Rancang Bangun Chatbot Berbasis Rule-Based Sebagai Pusat Informasi Calon Mahasiswa Baru Di Universitas Nasional Penulis 1),” *JUNSIBI (Jurnal Sist. Inf. Bisnis)*, vol. 4, no. 1, pp. 34–42, 2023, doi: 10.55122/junsibi.v4i1.695.
  - [10] K. Srivastava and N. Shekokar, “Design of machine learning and rule based access control system with respect to adaptability and genuineness of the requester,” *EAI Endorsed Trans. Pervasive Heal. Technol.*, vol. 6, no. 24, pp. 1–12, 2020, doi: 10.4108/eai.24-9-2020.166359.
  - [11] F. Ishlakhuddin, A. Basir, and N. Nurlaela, “Rancang Bangun Sistem Tanya-jawab Berbasis Aturan STMIK Muhammadiyah Paguyangan Brebes dengan Menggunakan Telegram Chatbot,” *J. Inform. J. Pengemb. IT*, vol. 5, no. 3, pp. 100–105, 2020, doi: 10.30591/jpit.v5i3.2900.
  - [12] S. K. Freire, C. Wang, and E. Niforatos, “Conversational Assistants in Knowledge-Intensive Contexts: An Evaluation of LLM- versus Intent-based Systems,” vol. 1, no. 1, pp. 1–10, 2024, [Online]. Available: <http://arxiv.org/abs/2402.04955>
  - [13] Y. Gao *et al.*, “Retrieval-Augmented Generation for Large Language Models: A Survey,” pp. 1–21, 2023, [Online]. Available: <http://arxiv.org/abs/2312.10997>
  - [14] C. Binardo, “Pengembangan Sistem Pendaftaran Kejuaraan Karate Berbasis Web dengan Pendekatan Extreme Programing,” *J. Inform. Dan Rekayasa Perangkat Lunak*, vol. 2, no. 2, pp. 276–284, 2021, doi: 10.33365/jatika.v2i2.932.
  - [15] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, “Bertscore: Evaluating Text Generation With Bert,” *8th Int. Conf. Learn. Represent. ICLR 2020*, pp. 1–43, 2020.
-