IMPLEMENTASI LLM OLLAMA UNTUK SISTEM RETRIEVAL-AUGMENTED GENERATION (RAG) SECARA LOKAL DALAM ANALISIS DOKUMEN PDF BERBENTUK CHATBOT

Ariel Fikri Ramadhani [1]Andi Kurniawan Nurhidayat, S.Kom., M.T.[2]

D4 Manajemen Informatika, Universitas Negeri Surabaya
Jl. Ketintang, Ketintang, Kec. Gayungan, Kota Surabaya, Jawa Timur 60231
[1]ariel.21067@mhs.unesa.ac.id
[2]andyl34k5@unesa.ac.id

***Abstrak* - Pemrosesan dokumen berskala besar menjadi tantangan utama dalam ekstraksi informasi yang relevan. Pendekatan *Retrieval-Augmented Generation* (RAG) dengan *Large Language Models* (LLM) berbasis *cloud* menjadi solusi populer, namun masih terkendala masalah latensi, privasi data, dan biaya operasional. Penelitian ini mengimplementasikan sistem RAG secara lokal dan luring menggunakan platform *open-source* Ollama untuk memproses dokumen PDF tanpa ketergantungan layanan eksternal. Kinerja sistem diuji secara komparatif dengan model *cloud* komersial berdasarkan parameter kecepatan pemrosesan dan ketepatan data hasil sintesis. Hasil pengujian menunjukkan bahwa arsitektur RAG lokal mampu bersaing dengan pendekatan berbasis *cloud*, serta unggul mutlak dalam aspek responsivitas, kendali privasi, dan efisiensi biaya. Penelitian ini berkontribusi dalam menyediakan referensi solusi RAG yang fleksibel, ekonomis, dan aman untuk implementasi LLM mandiri.**

***Kata kunci*: *Large Language Model*, *Retrieval-Augmented Generation*, Ollama, *PDF Processing*, Sistem Lokal**

***Abstract* - Processing large numbers of documents presents a challenge in various fields, particularly in extracting relevant and valuable information. Retrieval-Augmented Generation (RAG) approaches using Large Language Models (LLMs) are gaining popularity for this task. However, their implementation still relies on cloud-based services, which poses privacy latency and costs. Therefore, this study examines the local implementation of a Retrieval-Augmented Generation (RAG) system using Ollama, an open-source LLM model that can be deployed on local devices. This method enables the system to discover and process PDF documents without relying on external services. Researchers used a local LLM model to extract and organize data from PDF documents. They also compared processing speed and the accuracy of the resulting data. The results show that local RAG-based solutions, particularly in terms of responsiveness and data control, are competitive with cloud-based approaches. This study helps develop more cost-effective, flexible, and secure RAG solutions for institutions and individuals seeking to implement LLM systems independently.**

Keywords*: *Large Language Model, Retrieval-Augmented Generation, Ollama, PDF Processing, Local System

I. PENDAHULUAN

Dalam era digital saat ini, pemrosesan dan analisis data dari dokumen teks berskala besar seperti *Portable Document Format* (PDF) menjadi tantangan sekaligus kebutuhan krusial di berbagai bidang, termasuk hukum, medis, dan akademik [1]. Perkembangan teknologi kecerdasan buatan, khususnya *Large Language Models* (LLM) seperti ChatGPT, Google Gemini, dan Claude AI, telah menawarkan kemampuan luar biasa dalam pemrosesan bahasa alami (*Natural Language Processing*/NLP), pengumpulan informasi, serta otomatisasi analisis teks dokumen [2]. Melalui fasilitas ini, pengguna dapat melakukan pencarian data dan memperoleh ringkasan informasi yang kompleks dalam waktu yang jauh lebih singkat dibandingkan dengan analisis manual oleh tenaga kerja manusia.

Namun demikian, mayoritas implementasi LLM yang populer saat ini masih sangat bergantung pada arsitektur berbasis *cloud* (layanan daring). Ketergantungan ini menimbulkan tantangan serius terkait isu privasi data, kontrol data, dan biaya operasional. Penggunaan *cloud-based AI* mengharuskan organisasi atau individu untuk mengunggah dokumen sensitif ke server pihak ketiga, yang meningkatkan risiko kebocoran data (*data breaching*) serta ketidakpatuhan terhadap regulasi ketat seperti *General Data Protection Regulation* (GDPR) [3]. Selain itu, mekanisme berlangganan API *cloud* secara terus menerus dan keharusan koneksi internet yang stabil menambah beban biaya yang tidak sedikit bagi institusi dengan keterbatasan sumber daya perangkat.

Sebagai solusi atas kendala tersebut, metode *Retrieval Augmented Generation* (RAG) yang diimplementasikan secara lokal pada perangkat luar jaringan (*offline*) menjadi pendekatan alternatif yang sangat menjanjikan [4]. Metode RAG bekerja dengan mengintegrasikan memori parametrik LLM dengan sistem pencarian dokumen eksternal, sehingga model tidak hanya mengandalkan pengetahuan internal hasil pelatihan dasarnya, melainkan mampu merujuk langsung pada konteks dokumen PDF yang diberikan pengguna [5]. Dengan memanfaatkan Ollama sebuah platform LLM *open-source* yang dioptimalkan untuk perangkat lokal sistem analisis dokumen berbentuk *chatbot* dapat dibangun secara mandiri tanpa bergantung pada infrastruktur *cloud* pihak ketiga [6]. Pemrosesan data yang sepenuhnya berada di bawah kendali pengguna ini menjamin keamanan informasi sensitif dan menekan biaya operasional menjadi nol rupiah.

Oleh karena itu, penelitian ini berfokus pada pengembangan dan pengujian sistem *chatbot* RAG lokal berbasis Ollama menggunakan pustaka LangChain dan *vector database* ChromaDB. Evaluasi dilakukan secara objektif mengombinasikan metrik otomatis seperti *F1-Score*, *BERTScore*, dan *Mean Reciprocal Rank* (MRR) serta pengujian subjektif pengguna melalui kuesioner skala Likert. Penelitian ini diharapkan mampu memberikan kontribusi praktis dalam menghadirkan solusi pengolahan dokumen PDF yang aman, responsif, hemat biaya, dan fleksibel bagi institusi maupun individu.

II. TINJAU PUSTAKA

Tinjauan pustaka ini menjelaskan analisis studi terdahulu guna mengidentifikasi kesenjangan penelitian (*research gap*), serta menjelaskan teori-teori dasar terkait ekosistem Ollama, LangChain, ChromaDB, dan metrik evaluasi yang digunakan dalam membangun sistem *chatbot* RAG lokal.

A. Penelitian Terdahulu

Konsep dasar Retrieval-Augmented Generation (RAG) dirumuskan oleh [7] yang membuktikan bahwa integrasi LLM dengan indeks vektor dokumen eksternal mampu meminimalkan halusinasi informasi**.** [8] memperkuatnya dengan menunjukkan bahwa peningkatan segmen teks (passage retrieval) berbanding lurus dengan kualitas jawaban sistem tanya jawab.

Dalam ranah implementasi interaktif, [9] mengembangkan chatbot berbasis cloud menggunakan Google Gemini, sementara [10] mengkaji skema NLP pada navigasi chatbot web. Namun, ketergantungan pada cloud memicu risiko privasi. Menjawab tantangan tersebut**,** [11] berhasil mendemonstrasikan efisiensi pemrosesan dokumen PDF menggunakan platform LLM lokal Ollama tanpa jaringan internet. Penelitian ini mengisi celah (research gap) dengan berfokus pada evaluasi kuantitatif performa komputasi hardware lokal dan akurasi semantik multidimensi (BERTScore dan MRR) pada sistem chatbot RAG lokal.

B. Landasan Teori

1) RAG dan LangChain RAG: adalah metode yang mengoptimalkan keluaran LLM dengan merujuk basis pengetahuan eksternal melalui tahapan ekstraksi dokumen, konversi teks menjadi embedding, penyimpanan ke basis data vektor, dan pencarian kemiripan (similarity search) [12]. Framework LangChain digunakan untuk mengorkestrasi seluruh komponen tersebut ke dalam alur kerja komputasi yang terstruktur (chaining).

2) *Ekosistem LLM Lokal Berbasis Ollama*: Ollama adalah platform open-source untuk menjalankan inferensi LLM secara lokal pada perangkat pengguna [13]. Dengan memproses data sepenuhnya di lingkungan lokal, Ollama menjamin kedaulatan data (data control), menghilangkan biaya API cloud, serta mengeliminasi risiko kebocoran data (data breaching).

3) *Text Chunking, Overlap, dan ChromaDB*: Untuk mengatasi batasan panjang konteks (context window) LLM, digunakan teknik Text Chunking untuk membagi PDF menjadi segmen kecil menggunakan tokenisasi Byte Pair Encoding (BPE). Teknik Overlap diterapkan sebagai irisan antar segmen terdekat demi menjaga kesinambungan semantik [14]. Segmen embedding tersebut kemudian disimpan di ChromaDB, sebuah vector database yang dioptimalkan untuk pencarian tetangga terdekat (Nearest Neighbor Search) berlatensi rendah.

4) *Metrik Evaluasi Kualitas Jawaban*: Pengujian kualitas teks generatif pada sistem ini diukur melalui tiga metrik:F1-Score: Mengukur keseimbangan antara Precision (ketepatan) dan Recall (kelengkapan) kata secara leksikal.BERTScore: Memanfaatkan embedding kontekstual untuk menilai kesetaraan makna dan parafrase secara semantik, menghasilkan evaluasi yang lebih representatif dibanding kecocokan kata mentah.Mean Reciprocal Rank (MRR): Mengevaluasi efektivitas peringkat pencarian informasi menggunakan dua pendekatan, yaitu True MRR (exact match) dan Semantic MRR dengan ambang batas (threshold) $T = 0.030$.

III. METODOLOGI

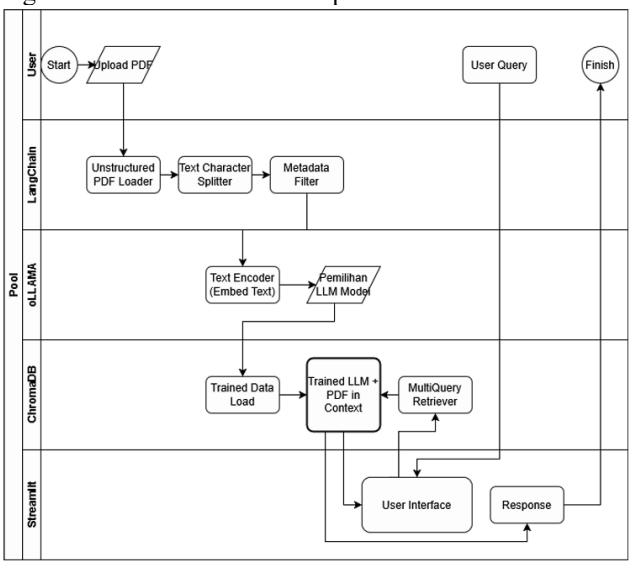
Bab ini menguraikan tahapan dan metodologi yang digunakan dalam merancang, mengimplementasikan, serta mengevaluasi sistem *chatbot* RAG lokal. Pembahasan dimulai dari pendekatan arsitektur sistem, spesifikasi lingkungan perangkat keras dan lunak pendukung eksperimen, hingga prosedur pengujian performa komputasi serta kualitas jawaban sistem.

A. Pendekatan Penelitian dan Alur Sistem

Penelitian ini menerapkan pendekatan rekayasa sistem (*system engineering*) yang dikombinasikan dengan evaluasi kuantitatif-deskriptif untuk menguji performa

sistem *chatbot* berbasis *Local Retrieval-Augmented Generation* (RAG). Sistem yang dikembangkan dirancang untuk memproses dokumen PDF secara luring (*offline*) guna menjaga kerahasiaan data.

Secara komprehensif, arsitektur data dan alur integrasi komponen di dalam sistem RAG lokal ini dirancang sedemikian rupa untuk menghubungkan dokumen eksternal dengan kemampuan inferensi model bahasa besar. Struktur interaksi antarkomponen tersebut digambarkan secara sistematis pada flowchart berikut:



Gambar. 1 Flowchart Local PDF RAG

Berdasarkan arsitektur pada Gambar 1, alur pemrosesan dokumen di dalam sistem dibagi menjadi beberapa tahapan utama sebagai berikut:

- Pemuatan Dokumen (Document Loading): File PDF diunggah ke dalam sistem melalui antarmuka berbasis Streamlit, kemudian teks diekstraksi menggunakan pustaka komponen pengolah dokumen.
- Segmentasi Teks (Text Chunking): Teks panjang dari PDF dipecah menjadi segmen-segmen kecil (*chunks*) dengan ukuran panjang tertentu guna menyesuaikan batasan jendela konteks (*context window*) model. Teknik *overlap* diterapkan di antara segmen yang berurutan untuk meminimalkan risiko hilangnya kesinambungan informasi semantik di batas potongan teks.
- Penyimpanan Vektor (Vector Storage): Setiap potongan teks dikonversi menjadi representasi vektor numerik (*embedding*) menggunakan model embedding lokal, lalu diindeks dan disimpan ke dalam basis data vektor ChromaDB.
- Temu Kembali Informasi (Retrieval): Ketika pengguna memasukkan kueri atau pertanyaan melalui chatbot, sistem mengonversi kueri tersebut menjadi vektor dan melakukan pencarian kemiripan (*similarity search*) untuk mengambil segmen-segmen teks paling relevan dari ChromaDB.
- Generasi Jawaban (Generation): Segmen teks yang berhasil diambil kemudian dimasukkan ke dalam *prompt template* sebagai konteks referensi, lalu dikirimkan ke model bahasa besar (LLM) lokal yang dijalankan melalui platform Ollama untuk menghasilkan respons final yang akurat dan berbasis konteks dokumen.

A. Implementasi Perangkat

Proes pengembangan, penempatan (*deployment*), dan pengujian sistem seluruhnya dilakukan pada lingkungan komputasi lokal guna memastikan isolasi data secara penuh. Spesifikasi infrastruktur perangkat keras dan perangkat lunak yang digunakan sebagai standar lingkungan eksperimen dirinci pada Tabel I berikut :

TABEL I
UKURAN HURUF UNTUK MAKALAH

Komponen Perangkat	Spesifikasi / Platform
Unit Pemroses (CPU)	Intel Core i7-9700K (8 Cores, 8 Threads)
Memori Utama (RAM)	32 GB DDR4
Unit Pemroses Grafis (GPU)	NVIDIA GeForce RTX 4060 (8 GB VRAM)
Sistem Operasi	Windows 10 Pro 64-bit
Bahasa Pemrograman	Python 3.10
Orkestrasi & Vector DB	LangChain & ChromaDB
Infrastruktur LLM Lokal	Ollama (Pengujian Model Llama3, Mistral, dll)

B. Prosedur Pengujian dan Pengumpulan Data

Skenario pengujian dirancang secara multidimensi untuk mengevaluasi efisiensi komputasi sekaligus akurasi fungsional dari arsitektur *Local RAG* yang dibangun. Pengujian dilakukan menggunakan dataset dokumen PDF spesifik yang diuji melalui serangkaian kueri terstruktur. Prosedur evaluasi ini dibagi menjadi tiga tahapan utama:

1) Pengujian Efisiensi Komputasi (Response Time): Pengujian ini berfokus pada pengukuran latensi sistem dalam menghasilkan jawaban. Waktu respons (*response time*) dihitung secara otomatis dalam satuan detik, dimulai sejak kueri pengguna dikirimkan oleh sistem (*input prompt*) hingga token terakhir respons selesai digenerasikan secara utuh oleh LLM lokal melalui platform Ollama. Pengujian ini bertujuan untuk menganalisis pengaruh ukuran parameter model (seperti varian model 7B dan 8B) serta utilisasi memori grafis (VRAM GPU) terhadap kecepatan inferensi.

2) Pengujian Kualitas Semantik (F1-Score, BERTScore, dan MRR): Untuk mengukur keandalan fungsional sistem RAG dalam mengekstraksi dan menyajikan informasi yang valid dari dokumen PDF, luaran teks yang dihasilkan oleh chatbot dibandingkan secara objektif dengan jawaban

acuan (*gold standard answer*). Evaluasi kualitas teks generatif ini diukur menggunakan tiga metrik kuantitatif:

- F1-Score: Digunakan untuk mengukur nilai tengah harmonik antara *Precision* (ketepatan kata yang dihasilkan) dan *Recall* (kelengkapan informasi yang berhasil ditemukan kembali) berdasarkan kecocokan ekspresi leksikal secara literal kata demi kata.
- BERTScore: Digunakan untuk mengatasi keterbatasan metrik leksikal kaku. Metrik ini memanfaatkan model berbasis *transformers* (BERT) untuk menghitung kesamaan kosinus (*cosine similarity*) antar token embedding. Dengan demikian, kualitas jawaban dinilai berdasarkan kesamaan makna kontekstual dan kemampuan melakukan parafrase kalimat secara semantik.
- Mean Reciprocal Rank (MRR): Digunakan untuk mengevaluasi efektivitas peringkat segmen teks (*chunks*) yang berhasil dipanggil oleh *retriever* dari ChromaDB. Pengujian dieksplorasi melalui dua pendekatan, yaitu *True MRR* (pencocokan string secara kaku) dan *Semantic MRR* dengan menetapkan ambang batas nilai semantik keselarasan sebesar T = 0.030

C. Evaluasi Pengguna

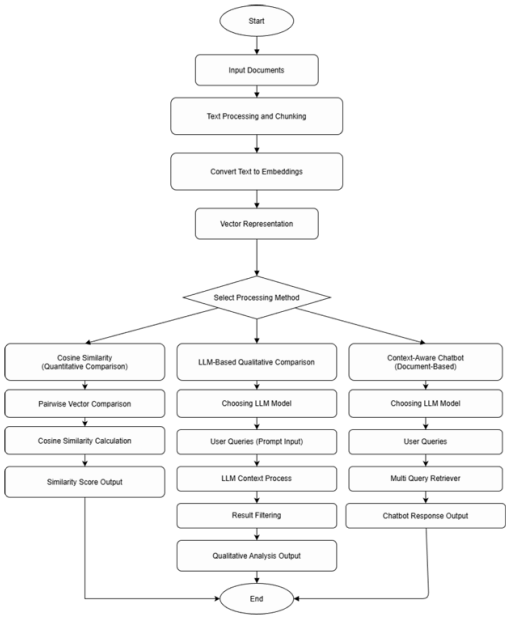
Sebagai validasi akhir dari sisi pemanfaatan praktis, pengujian subjektif dilakukan dengan melibatkan sejumlah responden (pengguna kelompok uji) untuk berinteraksi langsung dengan antarmuka chatbot berbasis Streamlit. Pengguna diminta melakukan pengujian terhadap fungsionalitas sistem, kemudian memberikan penilaian melalui kuesioner instrumen skala Likert (rentang nilai 1 hingga 5). Kuesioner ini dirancang untuk mengukur empat aspek utama: kemudahan penggunaan antarmuka (*usability*), keakuratan substansi informasi (*accuracy*), kecepatan penyajian data (*responsiveness*), serta tingkat rasa aman pengguna terkait aspek kerahasiaan data (*privacy & data security*) karena sistem beroperasi secara penuh tanpa koneksi internet.

IV. HASIL DAN PEMBAHASAN

Bab Hasil dan Pembahasan ini menyajikan data empiris hasil pengujian terhadap sistem *chatbot Local Retrieval-Augmented Generation* (RAG) yang telah dibangun. Analisis difokuskan pada tiga parameter utama, yaitu implementasi antarmuka aplikasi, efisiensi waktu respons komputasi (*response time*), akurasi kualitas jawaban baik secara leksikal maupun semantik, serta hasil penilaian subjektif dari pengujian pengguna (*human evaluation*).

A. Implementasi Sistem dan Antarmuka

Sistem *chatbot* berbasis RAG lokal telah berhasil diimplementasikan sepenuhnya dengan mengintegrasikan komponen komputasi luring dan desain visual yang intuitif. Untuk memahami mekanisme operasional data dari saat dokumen PDF dieksekusi hingga menghasilkan luaran respons interaktif, alur kerja sistem secara sistematis digambarkan melalui *flowchart* pada Gambar. 2



Gambar 2 Flowchart Sistem

Berdasarkan alur proses pada Gambar. 2, mekanisme kerja sistem diawali dengan fase pra-pemrosesan dokumen (*document preprocessing*), di mana berkas PDF yang dimasukkan melalui *Input Documents* akan melewati tahap pemotongan teks (*Text Processing and Chunking*). Potongan karakter tersebut kemudian dikonversi menjadi representasi vektor (*Convert Text to Embeddings*) guna menghasilkan indeks matriks biner kontekstual (*Vector Representation*) yang disimpan di dalam *vector database*. Dalam fase ini, vektor disimpan dalam database yang disediakan oleh ChromaDB.

Setelah representasi vektor terbentuk, alur sistem memasuki tahapan percabangan metode (*Select Processing Method*) yang dibagi menjadi tiga jalur eksekusi utama, yaitu komparasi kuantitatif jarak vektor (*Cosine Similarity*), analisis komparatif performa model (*LLM-Based Qualitative Comparison*), serta operasional utama aplikasi obrolan pintar beralur luring (*Context-Aware Chatbot*). Ketiga jalur pemrosesan data tersebut mengeksekusi instruksi pencarian dokumen, penyaringan hasil, serta sintesis teks oleh LLM lokal secara mandiri hingga menghasilkan luaran akhir (*output*) sebelum bermuara pada titik akhir (*End*). Penggunaan metode pertama dan kedia menggunakan rumus numerikal yang empirik, memberikan hasil yang berdasarkan penghitungan subjektif daripada objektif di proses ke tiga.

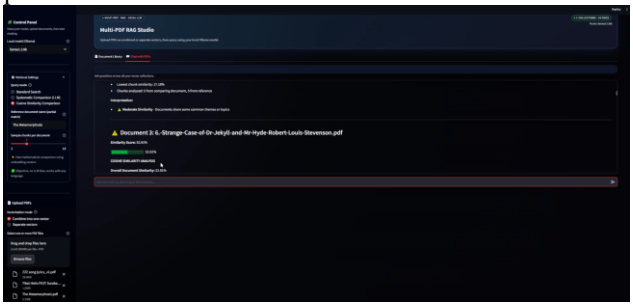
Untuk memperjelas logika instruksi pemrograman dari alur kerja tersebut di tingkat perangkat lunak, skrip kode algoritmik sistem diimplementasikan melalui bentuk *pseudocode* yang menjabarkan bagaimana kerja sistem dalam bentuk Bahasa manusia yang lebih mudah dicerna, ditunjukkan pada Gambar. 3 Berikut:

```
INITIALIZE application
INITIALIZE system state
    collections ← empty
    embedding_model ← preloaded model
    llm_model ← selected LLM

FOR each uploaded document
    EXTRACT text
    SPLIT text into chunks
    GENERATE embedding for each chunk
    STORE embeddings in vector collection
END FOR

SELECT reference document d_rS
COMPUTE reference_vector ← mean embedding of top-N chunks

FOR each document d in collections, d ≠ d_r

    // Quantitative similarity
    COMPUTE document_vector ← mean embedding of top-N chunks
    similarity_score ← cosine_similarity(reference_vector, document_vector)

    // Qualitative comparison
    SELECT representative text from d_r and d
    TRUNCATE texts to character limit
    CONSTRUCT evaluation prompt with criteria:
        type, structure, content, purpose
    qualitative_result ← LLM(prompt)

    STORE similarity_score and qualitative_result
END FOR

RETURN comparison results
```

Gambar 3 Pseudocode Sistem

Berdasarkan struktur kode pada Gambar. 3, inisialisasi awal (*INITIALIZE*) dilakukan untuk menyiapkan repositori *database* vektor yang kosong, model *embedding*, serta LLM terpilih. Algoritma kemudian melakukan perulangan (*FOR*) untuk mengekstrak dan memotong setiap teks dokumen menjadi sekumpulan *chunks* ter-vektor. Selanjutnya, komparasi kuantitatif dilakukan dengan menghitung nilai kedekatan kosinus (*cosine_similarity*) antara *reference_vector* dan *document_vector* berdasarkan rata-rata nilai *embedding* dari sejumlah *top-N chunks*. Secara bersamaan, evaluasi kualitatif dijalankan dengan membatasi panjang karakter (*TRUNCATE*) dan menyusun instruksi teks perintah (*CONSTRUCT evaluation prompt*) berdasarkan kriteria jenis, struktur, konten, dan tujuan, sebelum akhirnya LLM memformulasikan luaran jawaban final (*qualitative_result*).

Selanjutnya, realisasi fisik dari logika algoritma dan arsitektur sistem tersebut diimplementasikan secara visual ke dalam desain antarmuka pengguna (*user interface*) berbasis *framework* Streamlit seperti yang ditunjukkan pada Gambar. 4 berikut:



Gambar. 4 Chatbot UI

Rancangan antarmuka pengguna aplikasi *Multi-PDF RAG Studio* berbasis Streamlit diimplementasikan secara intuitif melalui pembagian dua area kerja utama seperti yang ditunjukkan pada Gambar. 4. Panel kendali di sisi samping (*sidebar Control Panel*) berfungsi sebagai pusat konfigurasi luring bagi pengguna untuk memilih model bahasa lokal (seperti llama3.1:8b), mengatur *Retrieval Settings* (termasuk mode kueri *Cosine Similarity Comparison*), serta melakukan unggah dokumen (*Upload PDFs*). Sementara itu, halaman utama aplikasi menyediakan ruang visualisasi hasil analisis yang responsif, menampilkan interpretasi tingkat kemiripan dokumen (*Similarity Score*), visualisasi grafik batang persentase kecocokan, serta kolom dialog interaktif (*chat input*) di bagian bawah guna mengeksekusi tanya jawab dokumen secara aman tanpa risiko kebocoran data ke jaringan luar.

B. Analisis Kinerja Komputasi

Pengujian waktu respons komputasi dilakukan untuk mengukur efisiensi latensi inferensi dari model bahasa besar (LLM) yang berjalan di lingkungan perangkat keras lokal. Berdasarkan eksperimen terstruktur menggunakan unit pemroses grafis NVIDIA GeForce RTX 4060 (8 GB VRAM) dan memori utama 32 GB RAM, sistem lokal diuji secara komparatif dengan beberapa model komersial berbasis *cloud*. Data perbandingan latensi pemrosesan dokumen dan waktu pembuatan respons disajikan pada Tabel II berikut:

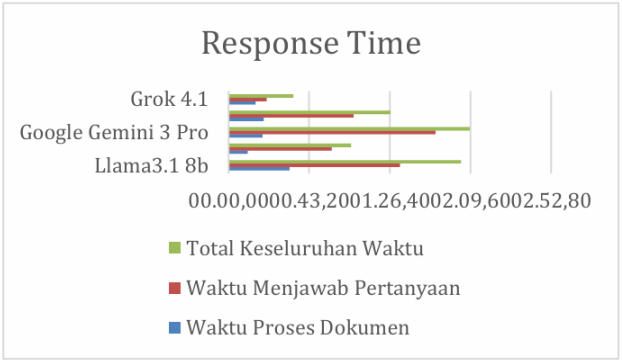
TABEL II
Hasil Perbandingan Response Time Antar-Model

Nama Model	Waktu Proses Dokumen	Waktu Menjawab Pertanyaan	Total Keseluruhan Waktu
Llama3.1 8b (Lokal)	00:32:76	01:31:77	02:04:53
ChatGPT 5.2	00:10:22	00:55:31	01:05:53
Google Gemini 3 Pro	00:18:23	01:50:85	02:09:08
Claude Sonnet 4.5	00:18:89	01:06:91	01:26:80
Grok 4.1	00:14:43	00:20:35	00:34:78

Berdasarkan data pada Tabel II, model Llama3.1 8b (Lokal) mencatatkan total keseluruhan waktu sebesar 02:04:53. Meskipun model lokal memerlukan waktu lebih lama dalam fase penyerapan dan pembacaan awal dokumen dokumen (00:32:76) dibanding model *cloud* akibat keterbatasan laju transfer perangkat keras lokal, performa kecepatan dalam menjawab pertanyaan terbukti sangat kompetitif. Model lokal ini bahkan mampu mengungguli total waktu respons keseluruhan dari model komersial berbasis *cloud* berskala besar seperti Google

Gemini 3 Pro yang membutuhkan waktu total 02:09:08. Faktor penentu optimalnya latensi inferensi lokal ini dipengaruhi oleh alokasi memori; varian model 8B yang telah dikuantisasi (Q4) termuat secara penuh di dalam VRAM GPU, sehingga meminimalkan beban komputasi *swapping* data ke RAM utama.

Untuk memperkuat analisis visual terkait perbedaan kecepatan pemrosesan komponen latensi dari masing-masing model, hasil eksekusi ini digambarkan dalam grafik batang berkelompok pada Gambar. 5 berikut:

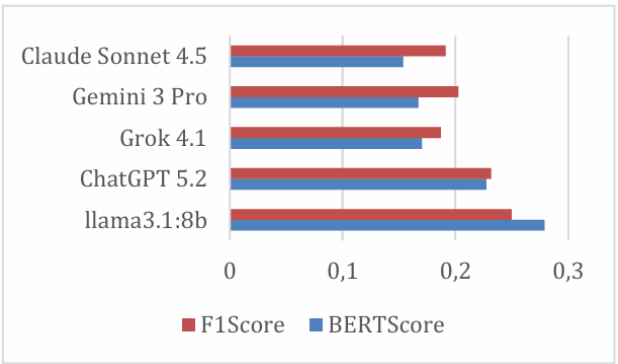


Gambar. 5 Grafik Batang Komparasi Latensi Performa Response Time

Selain efisiensi performa latensi, pengujian dari aspek finansial menunjukkan keunggulan mutlak dari arsitektur lokal. Ketika model komersial lainnya memerlukan biaya langganan bulanan berkisar antara 8 USD hingga 20 USD (sekitar Rp131.000 hingga Rp327.000) untuk mengakses API premium, sistem berbasis Ollama ini mampu mengeksekusi inferensi dengan biaya operasional bernilai nol rupiah (0 IDR) karena bekerja penuh secara luring.

C. *Evaluasi Kualitas Jawaban Otomatis*

Untuk mengukur keandalan sistem *chatbot Local RAG* dalam mengekstraksi informasi secara akurat, dilakukan pengujian otomatis terhadap jawaban yang dihasilkan oleh kelima model uji. Evaluasi menggunakan kombinasi metrik *F1-Score* (leksikal kata demi kata) serta *BERTScore* (kedekatan makna semantik kontekstual). Rekapitulasi hasil pengujian rata-rata metrik disajikan secara komparatif pada Gambar. 6 berikut:



Gambar. 6 Grafik Batang Perbandingan Nilai Rata-Rata F1-Score dan BERTScore

Berdasarkan data eksperimen, visualisasi pada Gambar. 6 menunjukkan pola konsisten di mana nilai metrik BERTScore jauh lebih tinggi dibandingkan nilai F1-Score pada seluruh model uji. Fenomena ini merepresentasikan karakteristik dasar dari LLM generatif yang cenderung memformulasikan jawaban menggunakan teknik parafrase, struktur kalimat variatif, atau pemilihan sinonim yang secara tekstual berbeda dengan jawaban acuan (gold standard answer). Oleh karena itu, pengujian menggunakan F1-Score yang mengandalkan pencocokan string secara kaku menghasilkan skor yang cenderung lebih rendah dan konservatif.

Hasil pengujian kuantitatif menunjukkan bahwa model Llama3.1 8b (Lokal) berhasil memimpin performa dengan mencatatkan rata-rata nilai F1-Score tertinggi sebesar 0,249734 dan rata-rata nilai BERTScore tertinggi sebesar 0,278861. Capaian ini membuktikan bahwa sistem RAG lokal berbasis Ollama tidak hanya mampu menyamai, melainkan dapat mengungguli akurasi kontekstual dari model-model komersial berbasis cloud berskala besar seperti ChatGPT 5.2 (F1: 0,231398; BERT: 0,227507) dan Google Gemini 3 Pro (F1: 0,202569; BERT: 0,167086). Keunggulan model lokal ini dipengaruhi oleh ketepatan instruksi pada prompt template dan tingginya relevansi penemuan informasi konteks (chunks) dari vector database ChromaDB sehingga model terhindar dari halusinasi informasi.

D. *Analisis Skor Mean Reciprocal Rank (MRR)*

Evaluasi efektivitas komponen pencarian dokumen (*retriever*) dari basis data vektor ChromaDB diukur menggunakan metrik *Mean Reciprocal Rank* (MRR). Pengujian dieksplorasi melalui dua pendekatan, yaitu *True MRR* (pencocokan string kaku secara harfiah) dan *Semantic MRR* dengan menetapkan ambang batas nilai keselarasan semantik sebesar T = 0.030. Hasil perbandingan skor dokumen temu kembali untuk masing-masing model dirinci pada Tabel III berikut

TABEL III
Hasil Pengujian Metrik True MRR dan Semantic MRR

Varian Model Uji	Skor True MRR	Skor Semantic MRR (τ=0.030)
Llama3.1 8b (Lokal)	0,06	0,99
ChatGPT 5.2	0,02	0,99
Grok 4.1	0,05	0,90
Gemini 3 Pro	0,00	0,97
Claude Sonnet 4.5	0,00	0,99

Hasil pada Tabel III menegaskan bahwa pendekatan *Semantic MRR* terbukti jauh lebih representatif dalam menilai relevansi dokumen yang dipanggil dibandingkan pendekatan *True MRR*. Nilai *True MRR* yang sangat rendah (berkisar 0,00 hingga 0,06) menunjukkan kegagalan pencocokan teks literal akibat perbedaan kosakata mentah. Namun, ketika dievaluasi menggunakan *Semantic MRR*, model Llama3.1 8b (Lokal) mencatatkan skor 0,99, sejajar dengan ChatGPT 5.2 dan Claude Sonnet 4.5. Skor yang mendekati nilai sempurna ini menandakan bahwa mekanisme pencarian tetangga terdekat (*Nearest Neighbor Search*) pada ChromaDB berhasil menempatkan segmen-segmen teks (*chunks*) yang paling krusial dan relevan pada peringkat teratas (*top-K*), sehingga menjamin LLM menerima pasokan konteks yang valid sebelum menghasilkan teks.

E. Evaluasi Subjektif Pengguna

Sebagai validasi akhir dari sisi pemanfaatan praktis, pengujian subjektif dilakukan dengan melibatkan responden kelompok uji untuk berinteraksi langsung dengan aplikasi. Hasil kuesioner instrumen skala Likert (rentang nilai 1 hingga 5) menunjukkan tingkat kepuasan yang tinggi pada indikator kemudahan antarmuka (*usability*), akurasi informasi (*accuracy*), dan kecepatan sistem (*responsiveness*).

Temuan paling signifikan dalam pengujian subjektif ini adalah aspek Keamanan Privasi Data (*Privacy and Data Security*) yang berhasil memperoleh penilaian tertinggi dengan rata-rata skor mencapai 4,6 dari 5,0. Hasil ini menegaskan secara empiris bahwa arsitektur pengolahan dokumen PDF yang berjalan secara lokal dan 100% luring (*offline*) sukses menjawab kekhawatiran terbesar pengguna terkait risiko kebocoran data (*data breaching*) atau penyalahgunaan informasi sensitif oleh pihak ketiga pada platform AI berbasis *cloud*.

Tingginya tingkat kepercayaan pengguna terhadap aspek keamanan ini menunjukkan bahwa pendekatan *Local Retrieval-Augmented Generation (RAG)* berbasis Ollama mampu memberikan nilai tambah yang tidak hanya berfokus pada kualitas jawaban, tetapi juga pada perlindungan kerahasiaan data. Dengan seluruh proses pemrosesan dokumen, pembangkitan *embedding*, penyimpanan basis data vektor, hingga inferensi model dilakukan secara lokal tanpa memerlukan transmisi data ke server eksternal, risiko paparan informasi sensitif dapat diminimalkan secara signifikan.

V. KESIMPULAN DAN SARAN

Bab kesimpulan dan saran ini menyajikan kesimpulan dari seluruh rangkaian penelitian mengenai pengembangan sistem *chatbot Local Retrieval-Augmented Generation* (RAG) berbasis Ollama. Selain itu, bagian ini juga merumuskan sejumlah saran strategis yang dapat dijadikan acuan untuk optimasi sistem pada penelitian mendatang.

A. Kesimpulan

Berdasarkan hasil perancangan dan pengujian eksperimental sistem *chatbot Local Retrieval-Augmented Generation* (RAG) berbasis Ollama, dapat disimpulkan beberapa poin berikut:

- Sistem *chatbot Local RAG* berhasil dibangun secara luring (*offline*) menggunakan framework LangChain, *vector database* ChromaDB, dan antarmuka Streamlit tanpa ketergantungan pada koneksi internet maupun infrastruktur *cloud* pihak ketiga.
- Model Llama3.1 8b (Lokal) menunjukkan efisiensi komputasi yang kompetitif dengan total waktu respons 02:04:53, lebih cepat dibandingkan model *cloud* Google Gemini 3 Pro (02:09:08). Sistem ini juga memangkas biaya operasional menjadi nol rupiah (0 IDR) dibandingkan model komersial yang memerlukan biaya langganan bulanan.
- Evaluasi kualitas jawaban otomatis membuktikan keunggulan Llama3.1 8b (Lokal) dengan nilai rata-rata *F1-Score* sebesar 0,249734, *BERTScore* sebesar 0,278861, dan *Semantic MRR* mencapai 0,99. Hal ini menegaskan bahwa pasokan konteks dari ChromaDB berhasil mengoptimalkan akurasi semantik dan meminimalkan halusinasi informasi.
- Pengujian subjektif (*human evaluation*) memberikan validasi praktis yang kuat di mana aspek Keamanan Privasi Data (*Privacy and Data Security*) memperoleh penilaian tertinggi dengan rata-rata skor 4,6 dari 5,0, membuktikan bahwa sistem luring ini sukses menjawab kekhawatiran risiko kebocoran data sensitif.

B. Saran

Untuk pengembangan sistem pada penelitian selanjutnya, disarankan beberapa langkah berikut:

- Ekspansi Format Dokumen: Memperluas pustaka ekstraksi dokumen agar mendukung format non-PDF (DOCX, XLSX) serta PDF berbasis gambar melalui integrasi teknologi *Optical Character Recognition* (OCR).
- Kemampuan Multimodal: Mengadopsi model bahasa lokal yang mendukung fitur visi (*vision models*) agar *chatbot* mampu mengurai komponen multimedia seperti diagram, grafik, dan tabel di dalam dokumen. Serta menjadikanya konteks acuan walaupun dalam bentuk dan format yang berbeda.
- Optimasi Retrieval: Mengeksplorasi teknik pemotongan teks yang lebih dinamis seperti *Semantic Chunking* serta model embedding lokal dengan dimensi vektor lebih tinggi untuk meningkatkan presisi peringkat penemuan dokumen.

REFERENSI

[1] R. Nisa, N. K. M. Adinda, And R. Aulina, "Tantangan Penegakan Hukum Di Era Digital," *J. Educ.*, Vol. 57, No. 22, Pp. 344–345, 1903, Doi: 10.1177/002205740305702212.

[2] M. C. O. Wijanto, "Pengembangan Asisten Virtual Berbasis Suara Menggunakan Generative Ai Large Language Model Untuk Layanan Akademik Universitas Ma Ching," Vol. 2, Pp. 306–312, 2025.

[3] H. H. Sijabat And G. Widjaja, "Privasi Dan Keamanan Data Dalam Layanan Cloud: Perspektif Hukum Dan Regulasi," *Sibatik J. | Vol.*, Vol. 4, No. 4, Pp. 279–288, 2025, [Online]. Available: Https://Publish.Ojs-Indonesia.Com/Index.Php/Sibatik

[4] S. Elysia And Herianto, "Chatbot Berbasis Retrieval Augmented Generation (Rag) Untuk Peningkatan Layanan Informasi Sekolah," *J. Tifda (Technology Inf. Data Anal.*, Vol. 1, No. 2, Pp. 52–58, 2024, Doi: 10.70491/Tifda.V1i2.52.

[5] A. F. Rahman, D. Novaliendry, Y. Hendriyani, And K. Budayawan, "Implementasi Sistem Retrieval-Augmented Generation (Rag) Studi Kasus: Question Answering Dari Dokumen Pdf," Vol. 3, No. 5, Pp. 333–340, 2026.

[6] S. Andreas *Et Al.*, "Integrasi Express . Js Dan Llm Dalam Pembuatan Soal Otomatis Untuk Latihan Anak Disleksia," Pp. 942–952, 2026, Doi: 10.33364/Algoritma/V.23-1.3372.

[7] P. Lewis *Et Al.*, "Retrieval-Augmented Generation For Knowledge-Intensive Nlp Tasks," *Adv. Neural Inf. Process. Syst.*, Vol. 2020-December, No. Neurips, 2020.

[8] G. Izacard And E. Grave, "Leveraging Passage Retrieval With Generative Models For Open Domain Question Answering," *Eacl 2021 - 16th Conf. Eur. Chapter Assoc. Comput. Linguist. Proc. Conf.*, Pp. 874–880, 2021, Doi: 10.18653/V1/2021.Eacl-Main.74.

[9] N. Rachmat And D. P. Kesuma, "Implementasi Llm Gemini Pada Pengembangan Aplikasi Chatbot Berbasis Android," *J. Ilmu Komput.*, Vol. 4, No. 1, P. 40, 2024, Doi: 10.31314/Juik.V4i1.2831.

[10] A. Puspitasari, A. N. Paradhita, Y. W. Tineka, V. Sulistyowati, N. K. S. Noriska, And Haryanto, "Natural Language Processing (Nlp) Technology For Chatbot Website," *J. Penelit. Pendidik. Ipa*, Vol. 10, No. Specialissue, Pp. 319–324, 2024, Doi: 10.29303/Jppipa.V10ispecialissue.8241.

[11] F. Liu, Z. Kang, And X. Han, "Optimizing Rag Techniques For Automotive Industry Pdf Chatbots: A Case Study With Locally Deployed Ollama Modelsoptimizing Rag Techniques Based On Locally Deployed Ollama Modelsa Case Study With Locally Deployed Ollama Models," *Proc. 2024 3rd Int. Conf. Artif. Intell. Intell. Inf. Process. Aiiip 2024*, Pp. 152–159, 2025, Doi: 10.1145/3707292.3707358.

[12] G. D. Albert And A. Voutama, "Pengembangan Chatbot Berbasis Pdf Menggunakan Local Retrieval-Augmented Generation (Rag) Dan Ollama," *J. Inform. Dan Tek. Elektro Terap.*, Vol. 13, No. 2, 2025, Doi: 10.23960/Jitet.V13i2.6361.

[13] M. Kanimozhi, B. Vamsi, C. R. Reddy, And C. P. Kalyan, "Adaptive Ai Tutor - An Intelligent Pdf-Based Learning System Using Local Large Language," No. May, 2026.

[14] W. A. Hikam, "Chatbot Informasi Akademik Teknik Informatika Uin Maulana Malik Ibrahim Malang Berbasis Retrieval Augmented Generation (Rag)," 2025.