TUGAS AKHIR

PENGEMBANGAN CHATBOT AI BERBASIS LARGE LANGUAGE MODEL (LLM) LANGCHAIN DENGAN RETRIEVAL-AUGMENTED GENERATION (RAG) PADA CUSTOMER SERVICE DI PT. TELEMEDIA PRIMA NUSANTARA

Sebagai Salah Satu Syarat Menyelesaikan Pendidikan Diploma IV Manajemen Informatika Jurusan Manajemen Informatika

Disusun oleh:

Moh Wildan Haikhal
062140832958

PROGRAM SARJANA TERAPAN DIV MANAJEMEN INFORMATIKA
JURUSAN MANAJEMEN INFORMATIKA
POLITEKNIK NEGERI SRIWIJAYA
TAHUN 2025

MOTTO DAN PERSEMBAHAN

MOTTO

يٰٓاَيُّهَا الَّذِيْنَ اٰمَنُوْٓا اِذَا قِيْلَ لَكُمْ تَفَسَّحُوْا فِى الْمَجٰلِسِ فَافْسَحُوْا يَفْسَحِ اللّٰهُ لَكُمْۚ وَاِذَا قِيْلَ انْشُزُوْا فَانْشُزُوْا
يَرْفَعِ اللّٰهُ الَّذِيْنَ اٰمَنُوْا مِنْكُمْۙ وَالَّذِيْنَ اُوْتُوا الْعِلْمَ دَرَجٰتٍۗ وَاللّٰهُ بِمَا تَعْمَلُوْنَ خَبِيْرٌ ﴿١١﴾

"Wahai orang-orang yang beriman, apabila dikatakan kepadamu "Berilah kelapangan di dalam majelis-majelis" maka lapangkanlah, niscaya Allah akan memberi kelapangan untukmu. Apabila dikatakan, "Berdirilah," (kamu) maka berdirilah. Niscaya Allah akan mengangkat derajat orang-orang yang beriman di antaramu dan orang-orang yang diberi ilmu. Sungguh Allah Maha Teliti terhadap apa yang kamu kerjakan."

さよなら、おれわ だいすき な ひと

"Selamat tinggal, orang yang sangat aku cintai."

PERSEMBAHAN

Skripsi ini saya persembahkan kepada:

1. Kedua orang tuaku tercinta yang selalu mendampingi, mendoakan, memberikan dukungan sampai akhir.
2. Saudara-saudariku tercinta yang selalu menemaniku dalam suka dan duka.
3. Seluruh dosen yang telah memberiku banyak wawasan dan ilmu pengetahuan dengan sabar dan ikhlas, terima kasih atas bimbingan dan arahannya sampai akhir.
4. Teman-temanku yang turut memberikan semangat serta membantu selama perkuliahan.
5. Kampusku tercinta, yang menjadi tempatku menuntut ilmu selama 4 tahun, yang menjadi kebanggaanku dalam meraih cita-cita.

ABSTRAK

Penelitian ini bertujuan mengembangkan *chatbot* AI berbasis *Large Language Model* (LLM) dengan integrasi *LangChain* dan *Retrieval-Augmented Generation* (RAG) untuk meningkatkan layanan pelanggan di PT Telemedia Prima Nusantara. Metode pengembangan menggunakan CRISP-DM, meliputi tahapan pemahaman bisnis, pemahaman data, persiapan data, pemodelan, evaluasi, dan deployment. Metode pengumpulan data dilakukan melalui observasi proses layanan, wawancara dengan tim internal, analisis dokumen FAQ, dokumentasi alur pengiriman invoice dan status pembayaran dari sistem Mixradius, riwayat percakapan pelanggan di WhatsApp, serta dokumen teknis internal lainnya. Data tambahan dikumpulkan melalui feedback pengguna internal saat evaluasi sistem dan eksperimen chatbot berbasis interaksi langsung, menggunakan metrik BLEU, ROUGE, dan waktu respons untuk pengukuran performa. Metode pemecahan masalah menggunakan pendekatan *Retrieval-Augmented Generation* (RAG) yang diorkestrasi dengan *framework LangChain.* Informasi dikumpulkan dari basis data vektor menggunakan FAISS, kemudian diolah oleh GPT-3.5 Turbo untuk menghasilkan jawaban. Komponen Skill Router diterapkan untuk klasifikasi intensi dan pengaturan jalur pemrosesan. Hasil evaluasi menunjukkan chatbot mampu memberikan jawaban dengan akurasi 93,33%, BLEU-2 sebesar 0,518, ROUGE-L sebesar 0,683, dan waktu respons rata-rata 1,55 detik. Hasil penelitian ini menunjukan sistem terbukti efektif dalam menjawab pertanyaan pelanggan secara otomatis, efisien, dan kontekstual. Rekomendasi pengembangan selanjutnya mencakup perluasan basis pengetahuan, deteksi intensi otomatis, eskalasi ke agen manusia, integrasi monitoring, serta optimalisasi *embedding* dan *vector store* alternatif.

Kata Kunci: *Chatbot AI, Large Language Model* (LLM), *LangChain, Retrieval-Augmented Generation* (RAG), *Costumer Service,* CRISP-DM.

ABSTRACT

This research aims to develop an AI chatbot based on Large Language Model (LLM) with LangChain and Retrieval-Augmented Generation (RAG) integration to improve customer service at PT Telemedia Prima Nusantara. The development method uses CRISP-DM, which includes the stages of business understanding, data understanding, data preparation, modeling, evaluation, and deployment. Data collection methods were carried out through observation of service processes, interviews with internal teams, analysis of FAQ documents, documentation of invoice delivery and payment status from the Mixradius system, customer conversation history on WhatsApp, and other internal technical documents. Additional data was collected through internal user feedback during system evaluation and experiments with a chatbot based on direct interaction, using BLEU, ROUGE, and response time metrics for performance measurement. The problem-solving method employed the Retrieval-Augmented Generation (RAG) approach, orchestrated using the LangChain framework. Information was collected from the vector database using FAISS, then processed by GPT-3.5 Turbo to generate answers. The Skill Router component was applied for intent classification and processing path configuration. Evaluation results show that the chatbot can provide answers with 93.33% accuracy, a BLEU-2 score of 0.518, a ROUGE-L score of 0.683, and an average response time of 1.55 seconds. These findings demonstrate that the system is effective in automatically, efficiently, and contextually answering customer questions. Further development recommendations include expanding the knowledge base, automatic intent detection, escalation to human agents, monitoring integration, and optimization of alternative embedding and vector stores.

Keywords: *AI Chatbot, Large Language Model (LLM), LangChain, Retrieval-Augmented Generation (RAG), Customer Service, CRISP-DM.*

KATA PENGANTAR

Dengan mengucap puji dan syukur kehadirat Allah SWT karena berkat rahmat dan hidayah-Nya penulis dapat menyelesaikan Tugas Akhir yang bejudul **"Pengembangan Chatbot AI berbasis *Large Language Model* (LLM) Langchain dengan *Retrieval-Augmented Generation* (RAG) pada *Customer Service* di PT Telemedia Prima Nusantara."** ini dengan baik.

Tujuan dari penyusunan Tugas Akhir ini adalah untuk memenuhi syarat menyelesaikan Pendidikan Diploma IV Jurusan Manajemen Informatika Politeknik Negeri Sriwijaya. Selama menyelesaikan Tugas Akhir ini penulis banyak sekali mendapat bantuan, bimbingan, dan petunjuk dari berbagai pihak. Oleh karena itu, dalam kesempatan ini penulis menyampaikan ucapan terima kasih kepada:

1. Bapak Ir. Irawan Rusnadi, M.T selaku Direktur Politeknik Negeri Sriwijaya Palembang.
2. Bapak Dr. Yusri, S.Pd., M.Pd selaku Wakil Direktur I Politeknik Negeri Sriwijaya Palembang.
3. Bapak M. Husni Mubarok, S.E., M.Si, Ak selaku Wakil Direktur II Politeknik Negeri Sriwijaya Palembang.
4. Bapak Dicky Seprianto, S.T., M.T. IPM selaku Wakil Direktur III Politeknik Negeri Sriwijaya Palembang.
5. Ibu Dr. Irma Salamah, S.T., M.T.I selaku Wakil Direktur IV Politeknik Negeri Sriwijaya Palembang.
6. Bapak Sony Oktapriandi, S.Kom., M.Kom., selaku Ketua Jurusan Manajemen Informatika Politeknik Negeri Sriwijaya Palembang.
7. Sulistiyanto, S.Kom., M.T.I., selaku Plt. Sekretaris Jurusan Manajemen Informatika Politeknik Negeri Sriwijaya Palembang.
8. Dr. Indri Ariyanti, S.E., M.Si., selaku Dosen Pembimbing I yang telah memberikan waktu dan arahannya, sehingga penulis dapat menyelesaikan Tugas Akhir ini dengan baik.
9. Nurul Ilma Hasana Kunio, S.Kom., M.Kom., selaku Dosen Pembimbing II yang telah memberikan waktu dan arahannya, sehingga penulis dapat

menyelesaikan Tugas Akhir ini dengan baik.

10. Seluruh Dosen, Staff, Administrasi dan Karyawan Jurusan Manajemen Informatika Politeknik Negeri Sriwijaya Palembang.
11. Keluarga tercinta terutama kedua orang tua dan saudara-saudariku yang senantiasa memberikan doa dan semangat, dukungan serta saran yang sangat bermanfaat agar terus melakukan yang terbaik.
12. Teman-teman seperjuangan Jurusan Manajemen Informatika terkhusus untuk kelas MIM.

Di dalam penulisan Tugas Akhir ini penulis merasa masih jauh dari kata sempurna. Untuk itu segala kritik dan saran yang bersifat membangun, sangat penulis harapkan sebagai perbaikan dimasa yang akan datang. Akhir kata semoga Tugas Akhir ini dapat bemanfaat bagi penulis, pembaca, rekan-rekan mahasiswa dan pihak yang membutuhkan sebagai penambah wawasan dan ilmu pengetahuan.

Palembang, 15 Juli 2025

Moh Wildan Haikhal

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1 Latar Belakang

Era digital yang terus berkembang ini menciptakan kecerdasan buatan (*Artificial Intelligence*) sebagai fondasi utama dalam transformasi layanan pelanggan. Perusahaan-perusahaan berlomba mengadopsi teknologi terkini untuk meningkatkan efisiensi dan kualitas pelayanan. Salah satu solusi teknologi yang semakin banyak digunakan adalah chatbot berbasis AI, yang memungkinkan interaksi otomatis dan real-time antara pelanggan dan sistem tanpa keterlibatan langsung manusia.

Lee, et. al, (2024) menjelaskan bahwa integrasi chatbot dengan model bahasa besar atau *Large Language Models* (LLM) mampu meningkatkan kualitas respons dalam percakapan, terutama saat dikombinasikan dengan pendekatan *Retrieval-Augmented Generation* (RAG). Pendekatan ini memungkinkan chatbot untuk mengambil informasi dari sumber eksternal secara cerdas sebelum membentuk jawaban, menghasilkan tanggapan yang lebih akurat, kontekstual, dan terkini. Bhat, et. al, (2024) menjelaskan bahwa teknologi ini terbukti meningkatkan keakuratan jawaban chatbot dan menurunkan risiko informasi yang keliru, khususnya dalam konteks domain-spesifik.

PT Telemedia Prima Nusantara, sebagai penyedia layanan internet, menghadapi tantangan dalam memberikan layanan pelanggan yang cepat, konsisten, dan personal, khususnya melalui platform komunikasi seperti WhatsApp. Saat ini, sebagian besar proses komunikasi dua arah masih dilakukan secara manual, termasuk dalam menjawab pertanyaan umum, pengecekan status layanan, dan pengiriman tagihan. Hal ini dapat menyebabkan keterlambatan respon dan menurunkan kepuasan pelanggan. Sebagai solusi, pengembangan chatbot AI berbasis arsitektur modular diperkenalkan. Sistem ini terdiri dari beberapa komponen utama: *Baileys* sebagai jembatan komunikasi WhatsApp, *FastAPI* sebagai backend server, serta *LangChain* untuk pengelolaan RAG dan integrasi dengan *GPT-3.5 Turbo*. Di antara komponen ini, *Skill Router* berperan penting

dalam mengklasifikasikan maksud pesan pengguna (*intent*) dan menentukan jalur pemrosesan yang sesuai untuk menghasilkan jawaban yang optimal.

Vidivelli et al, (2024) menjelaskan bahwa arsitektur ini memungkinkan sistem untuk menjawab berbagai pertanyaan seperti FAQ, pengecekan status pembayaran, dan permintaan invoice secara otomatis, sekaligus menjaga fleksibilitas untuk ekspansi layanan di masa depan. Basis pengetahuan internal perusahaan (FAQ dan dokumentasi layanan) disimpan dalam vektor database seperti FAISS atau Weaviate, yang memungkinkan pencarian semantik berbasis konteks untuk mendukung sistem RAG.

Studi yang dilakukan oleh Burgan et al. (2024) menunjukkan bahwa penggunaan RAG dalam chatbot universitas dapat meningkatkan relevansi respons dan kepuasan pengguna. Burgan et al., (2024). Menjelaskan bahwa sistem mereka menggunakan LangChain dan LLM lokal yang disesuaikan, serta vektor store berbasis embedding, yang secara signifikan meningkatkan kemampuan pemahaman konteks chatbot dalam skenario dunia nyata.

Seluruh sistem berjalan secara berkelanjutan dalam server pribadi (VPS) dengan menggunakan proses manajemen seperti Supervisor, memastikan chatbot dapat aktif selama 24 jam tanpa intervensi manual. Dengan pendekatan berbasis AI agent, LLM, dan RAG ini, chatbot diharapkan dapat memberikan pengalaman layanan pelanggan yang lebih efisien, personal, dan skalabel, sekaligus mengurangi beban kerja manual pada tim layanan pelanggan. Pengembangan sistem pada PT Telemedia Prima Nusantara ini diharapkan mampu menyajikan layanan pelanggan yang lebih responsif, personal, dan efisien, sekaligus mengurangi beban kerja manual pada tim layanan. Integrasi teknologi LLM dan RAG menjadi langkah penting dalam transformasi digital berkelanjutan perusahaan.

Berdasarkan latar belakang di atas, penulis akan melakukan penelitian dengan judul **"Pengembangan Chatbot AI berbasis *Large Language Model* (LLM) Langchain dengan *Retrieval-Augmented Generation* (RAG) pada *Customer Service* di PT Telemedia Prima Nusantara."**

1.1 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan, permasalahan yang akan dibahas dalam penelitian ini adalah Bagaimana merancang dan mengembangkan Chatbot AI berbasis *Large Language Model* (LLM) dengan integrasi LangChain dan *Retrieval-Augmented Generation* (RAG) yang mampu menjawab pertanyaan pelanggan secara relevan dan kontekstual di PT Telemedia Prima Nusantara.

1.2 Batasan Masalah

Agar penulisan Tugas Akhir ini lebih terarah dan tidak menyimpang dari fokus utama pada penelitian ini, maka ruang lingkup penelitian dibatasi pada hal-hal berikut:

1. Platform komunikasi yang digunakan dibatasi pada WhatsApp, sebagai media utama interaksi antara pengguna dan sistem chatbot, dengan menggunakan library Baileys sebagai jembatan komunikasi.
2. Model bahasa besar (*Large Language Model*) yang digunakan merupakan model berbayar seperti GPT-3.5 Turbo dari OpenAI yang mendukung integrasi dengan LangChain.
3. Sistem Sistem menggunakan komponen *Skill Router* sebagai modul *intent classifier*/*agent selector*, untuk mengarahkan pesan pengguna ke jalur pemrosesan yang sesuai dalam arsitektur LangChain dan RAG.
4. Fungsi chatbot dibatasi pada penanganan interaksi berbasis teks, termasuk pertanyaan umum (FAQ), pengecekan status pembayaran, dan permintaan *invoice*. Chatbot tidak mencakup penanganan keluhan kompleks atau eskalasi kasus ke manusia.
5. Sumber data pengetahuan (*knowledge base*) terbatas pada dokumen internal perusahaan seperti FAQ dan data riwayat transaksi yang dapat diakses melalui integrasi eksternal, tanpa melibatkan basis data publik atau *crawling web.*
6. Pengujian sistem dilakukan pada lingkungan pengembangan terbatas, yaitu menggunakan server pribadi (VPS), dan tidak mencakup pengujian pada skala *enterprise* atau *multi-platform* di luar WhatsApp.

1.3 Tujuan dan Manfaat Penelitian

1.3.1 Tujuan Penelitian

Berdasarkan rumusan masalah yang ada, maka tujuan penelitian yang diharapkan pada penelitian ini, sebagai berikut:

1. Mengembangkan sistem chatbot AI berbasis *Large Language Model* (LLM) dengan pendekatan *Retrieval-Augmented Generation* (RAG) menggunakan LangChain, untuk meningkatkan kualitas interaksi layanan pelanggan di PT Telemedia Prima Nusantara.
2. Membangun komponen *Skill Router* (*intent classifier*/*agent selector*) untuk mengarahkan pesan pengguna ke jalur pemrosesan yang sesuai, berdasarkan analisis maksud pengguna.
3. Menghubungkan chatbot dengan vektor database seperti Weaviate atau FAISS, sehingga sistem mampu melakukan pencarian semantik berdasarkan FAQ dan riwayat percakapan.
4. Menerapkan sistem chatbot secara stabil dan berkelanjutan di server pribadi (VPS) dengan manajemen proses seperti Supervisor, agar dapat diakses 24 jam tanpa intervensi manual..
5. Melakukan pengujian terhadap performa chatbot dalam memahami konteks pertanyaan pengguna dan memberikan jawaban yang cepat, relevan, serta informatif sesuai kebutuhan pelanggan.

1.3.2 Manfaat Penelitian

Berdasarkan tujuan penelitian di atas, maka manfaat penelitian yang diharapkan pada hasil penelitian ini, sebagai berikut:

1. Menyediakan solusi layanan pelanggan berbasis AI yang terotomatisasi dan responsif, sehingga meningkatkan efisiensi dan kepuasan pelanggan di PT Telemedia Prima Nusantara.
2. Mengurangi beban kerja staf *customer service* dengan menangani permintaan yang bersifat rutin dan berulang secara otomatis.
3. Menyediakan chatbot yang adaptif dan kontekstual, dengan dukungan teknologi RAG dan LLM, sehingga mampu memberikan jawaban yang alami dan sesuai

dengan maksud pengguna.

4. Memberikan contoh implementasi nyata integrasi LangChain, LLM, dan arsitektur modular berbasis *intent classification* dalam pengembangan chatbot untuk WhatsApp.
5. Menjadi referensi praktis bagi perusahaan lain yang ingin mengembangkan sistem chatbot cerdas berbasis teknologi terkini seperti LLM dan RAG dalam konteks layanan pelanggan..

1.4 Sistematika Penulisan

Agar penyusunan tugas akhir ini lebih terarah dan tujuan penelitian dapat dipahami secara jelas, maka penelitian ini terdiri dari lima bab, sebagai berikut:

BAB I PENDAHULUAN

Pada bab ini menjelaskan mengenai tugas akhir ini secara garis besar dengan singkat dan jelas mengenai latar belakang, rumusan masalah, batasan masalah, tujuan dan manfaat penelitian, sistematika penulisan.

BAB II TINJAUAN PUSTAKA

Pada bab ini menjelaskan teori umum yang berkaitan dengan judul, teori khusus yaitu berkaitan dengan sistem yang akan dibuat.

BAB III METODOLOGI PENELITIAN

Pada bab ini akan membahas mengenai instansi penelitian, metode yang akan digunakan, teknik pengumpulan data, tahapan penelitian serta menguraikan konsep perangkat lunak yang dibuat.

BAB IV HASIL DAN PEMBAHASAN

Pada bab ini menjelaskan spesifikasi dan rancangan perangkat lunak yang akan dibuat dan mendeskripsikan perangkat lunak yang akan dibuat.

BAB V PENUTUP

Bab ini merupakan bab terakhir yang memuat kesimpulan dan saran dari hasil penelitian.

BAB II
TINJAUAN PUSTAKA

2.1 Landasan Teori

2.1.1. *Large Language Model* (LLM)

Pada penelitian ini, model yang digunakan untuk mendukung sistem chatbot adalah GPT-3.5 Turbo, sebuah *Large Language Model* (LLM) berbasis arsitektur *transformer* yang dikembangkan oleh OpenAI. LLM seperti GPT-3.5 memiliki kemampuan untuk memproses bahasa alami dalam jumlah besar dengan memahami konteks, menyusun kalimat, menjawab pertanyaan, dan berinteraksi secara *real-time*. Model ini bekerja dengan mengandalkan mekanisme perhatian (*attention mechanism*), khususnya *scaled dot-product attention*, untuk memfokuskan pemrosesan pada bagian penting dari input teks.

Mekanisme ini memungkinkan model memetakan hubungan antar token dalam sebuah kalimat melalui operasi matematis yang efisien. Perhitungan *attention* dilakukan dengan menghitung *dot product* antara vektor *query* dan *key*, kemudian dibagi dengan akar kuadrat dari dimensi vektor key, lalu hasilnya disalurkan melalui fungsi *softmax* untuk menghasilkan bobot probabilitas. Formula dari *scaled dot-product attention* dirumuskan sebagai berikut:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Dengan:

- Q = Query matrix
- K = Key matrix
- V = Value matrix
- d_k = dimensi key matrix

Arsitektur *transformer* seperti ini memiliki kompleksitas komputasi sebesar $O(n^2 \cdot d)$,di mana n adalah panjang urutan (*sequence length*) dan d adalah dimensi vektor. Kompleksitas ini menjadi tantangan utama ketika memproses input yang sangat panjang, tetapi juga alasan mengapa model ini sangat efisien dalam memahami konteks jangka panjang. Kemampuan GPT-3.5 Turbo dalam menangani berbagai tugas NLP telah dikaji dalam sejumlah studi. Salah satunya oleh Nfaoui & Elfaik (2024), yang menunjukkan bahwa GPT-3.5 Turbo mampu melakukan klasifikasi emosi dalam teks Arab dengan

akurasi yang tinggi menggunakan strategi *in-context learning*, yang menunjukkan kekuatan model dalam penalaran berbasis konteks multibahasa tanpa pelatihan ulang (Nfaoui & Elfaik, 2024).

Model GPT-3.5 juga dinilai unggul dalam menghasilkan jawaban untuk tugas-tugas klasifikasi, penalaran logis, hingga penerapan klinis. Studi oleh Liu (2025) menyoroti bagaimana arsitektur LLM berbasis *transformer* dan *attention* telah mendorong perkembangan AI menuju model multimodal dan sistem yang lebih kuat, mencakup teks, gambar, dan audio. Secara teknis, model ini dilatih pada dataset dalam skala miliaran token dan disempurnakan melalui proses *reinforcement learning with human feedback (RLHF)* untuk meningkatkan kualitas interaksi dengan pengguna. Hal ini memungkinkan chatbot untuk merespons secara alami dan relevan terhadap pertanyaan pengguna di berbagai domain, termasuk layanan pelanggan.

Pemilihan GPT-3.5 Turbo dalam penelitian ini didasarkan pada kombinasi antara efisiensi komputasi, fleksibilitas konteks, dan keakuratan hasil dalam interaksi berbasis bahasa alami yang kompleks. Hal ini menjadikannya fondasi yang tepat untuk membangun sistem chatbot dengan kemampuan generatif dan kontekstual yang tinggi.

2.1.2. *Retrieval-Augmented Generation* (RAG)

Retrieval-Augmented Generation (RAG) merupakan pendekatan arsitektural yang dirancang untuk meningkatkan kemampuan *Large Language Model* (LLM) dalam menjawab pertanyaan berbasis konteks eksternal yang tidak tersedia dalam data pelatihan model. RAG bekerja dengan menggabungkan dua komponen utama: proses *retrieval* (pengambilan informasi) dari sumber data eksternal seperti basis pengetahuan atau dokumen perusahaan, dan proses *generation* (generatif) menggunakan LLM berdasarkan informasi yang diambil tersebut. Dengan pendekatan ini, sistem mampu memberikan jawaban yang lebih akurat, faktual, dan kontekstual.

Secara teknis, alur kerja RAG dimulai dengan proses *embedding* pertanyaan pengguna menjadi representasi vektor menggunakan model *embedding* seperti *text-embedding-ada-002*. *Embedding* ini digunakan untuk mencari dokumen atau potongan informasi yang relevan dari *vector database* menggunakan teknik pencarian semantik seperti *cosine similarity* atau *euclidean distance*. Hasil pencarian (*top-k documents*) kemudian digabungkan ke dalam *prompt* dan dikirimkan ke LLM (seperti GPT-3.5 Turbo) untuk membentuk respons akhir yang mempertimbangkan konteks informasi tersebut.

Tahapan utama dalam RAG diawali dengan pengubahan dokumen dan pertanyaan menjadi vektor melalui proses *embedding* menggunakan model seperti *text-embedding-ada-002*. Vektor hasil *embedding* ini kemudian disimpan dalam *vector database* seperti FAISS atau Weaviate. Ketika pengguna mengajukan pertanyaan, sistem akan mengubah pertanyaan menjadi vektor dan mencocokkannya dengan dokumen dalam basis data tersebut menggunakan teknik pencarian berbasis kesamaan vektor. Dua metode utama yang digunakan dalam proses pencocokan ini adalah:

a. *Cosine Similarity*

Cosine similarity mengukur kemiripan arah antara dua vektor dan digunakan untuk menentukan seberapa mirip konteks pertanyaan dengan dokumen yang ada. Rumus dari *cosine similarity* adalah:

$$ConsineSimilarity(A,B) = \frac{A \cdot B}{||A||||B||} = \frac{\sum_{i=1}^{n} A_i \cdot B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \cdot \sqrt{\sum_{i=1}^{n} B_i^2}}$$

Nilai *cosine similarity* berada dalam rentang [0, 1], di mana nilai mendekati 1 menunjukkan bahwa kedua vektor memiliki arah yang sangat mirip dan karenanya representasi semantik yang serupa.

b. *Euclidean Distance*

Alternatif dari *cosine similarity* adalah *euclidean distance*, yang mengukur jarak geometris antara dua vektor di ruang berdimensi tinggi. Rumusnya adalah:

$$d(A,B) = \sqrt{\sum_{i=1}^{n} (A_i - B_i)}$$

Nilai yang lebih kecil menunjukkan kedekatan yang lebih besar antara dua representasi teks. Meskipun metode ini sensitif terhadap skala, ia tetap efektif terutama pada embedding yang terstandarisasi. Setelah dokumen paling relevan (top-k) ditemukan, potongan teks tersebut digabungkan dan disisipkan ke dalam *prompt* yang dikirimkan ke GPT-3.5 Turbo. Model akan menghasilkan jawaban berdasarkan informasi yang tersedia di *prompt*, sehingga tidak hanya mengandalkan pengetahuan internal model. Pendekatan ini efektif dalam mengurangi kesalahan jawaban (*hallucination*) dan meningkatkan keakuratan pada kasus penggunaan berbasis domain.

Studi oleh Pokhrel et.al., (2025) membuktikan bahwa RAG dapat memanfaatkan data hasil *scraping* situs web yang dikonversi menjadi *embedding*, memungkinkan sistem memberikan jawaban real-time yang kontekstual dan akurat. Penelitian oleh Benita et.al., (2024) pada chatbot *e-commerce* menunjukkan peningkatan akurasi jawaban hingga 85% saat menggunakan RAG dibandingkan pendekatan generatif murni. Burgan et. al., (2024). Framework LangChain menjadi alat bantu penting dalam mengatur alur kerja ini secara modular dan efisien. Dengan menggabungkan pencarian berbasis vektor dan pemrosesan generatif, RAG memungkinkan chatbot tidak hanya menjawab dengan benar secara linguistik, tetapi juga secara faktual dan kontekstual, menjadikannya sangat ideal untuk aplikasi layanan pelanggan.

2.1.3. *Rule-Based Text Preprocessing*

Text preprocessing merupakan tahap fundamental dalam sistem berbasis bahasa alami, terutama pada arsitektur *Retrieval-Augmented Generation* (RAG). Tujuannya adalah untuk menstandarisasi dan membersihkan teks sehingga dapat diubah menjadi representasi numerik yang optimal dalam proses embedding dan pencocokan semantik. Dalam penelitian ini, pendekatan *preprocessing* dilakukan secara *rule-based*, yakni melalui penerapan aturan eksplisit untuk memproses input teks dari pengguna maupun dokumen basis pengetahuan.

Secara teoritis, pendekatan *rule-based text preprocessing* menekankan pada manipulasi tekstual yang deterministik untuk menyamakan format dan mengurangi *noise*. Misalnya, konversi seluruh huruf menjadi huruf kecil (*lowercasing*) dilakukan untuk menyamakan bentuk kata yang sama dalam representasi vektor. Selain itu, karakter-karakter non-alfabet seperti angka, tanda baca, dan simbol dihapus karena tidak membawa makna semantik yang signifikan dalam konteks pencocokan dokumen. Penghapusan *stopwords* juga menjadi bagian penting dalam proses ini, karena kata-kata umum seperti “yang”, “dan”, atau “adalah” cenderung tidak memiliki bobot informasi tinggi dalam analisis semantik.

Preprocessing juga melibatkan proses normalisasi kata seperti *stemming* atau *lemmatization*. Proses ini mengubah variasi kata ke bentuk dasarnya, misalnya “dibayar”, “membayar”, dan “pembayaran” diubah menjadi “bayar”. Dengan pendekatan ini, sistem dapat mengenali bahwa ketiga kata tersebut merujuk pada konsep yang sama meskipun bentuknya berbeda. Selain itu, tokenisasi digunakan untuk memecah kalimat menjadi unit kata atau token, yang akan diproses lebih lanjut dalam tahap *embedding.*

Kualitas *preprocessing* sangat memengaruhi performa sistem RAG, terutama dalam tahap pencocokan *embedding* antar dokumen. Tanpa *preprocessing* yang tepat, vektor *embedding* yang dihasilkan dapat menjadi tidak akurat atau memiliki distribusi yang menyimpang. Hal ini akan berdampak pada kesalahan dalam pengambilan dokumen relevan dan menurunkan akurasi respons yang dihasilkan oleh LLM.

Kolyshkina et. al., (2024) menyatakan bahwa meskipun pendekatan berbasis pembelajaran mendalam semakin populer, *preprocessing* berbasis aturan tetap menjadi komponen penting dalam sistem NLP modern, karena dapat disesuaikan dengan karakteristik data spesifik suatu domain dan memberikan kontrol penuh terhadap struktur teks input. Dengan demikian, *rule-based preprocessing* menjadi tahap kritis dalam menjembatani data mentah dengan proses representasi numerik, memungkinkan sistem RAG bekerja secara efektif dalam menjawab pertanyaan pengguna dengan konteks yang lebih bersih, terstruktur, dan semantik.

2.1.4. *Document Embedding* dan *Mean Pooling*

Agar dokumen dan pertanyaan dapat dibandingkan secara semantik dalam sistem *Retrieval-Augmented Generation* (RAG), teks perlu diubah menjadi representasi numerik dalam bentuk *embedding*. *Embedding* adalah representasi vektor berdimensi tetap yang menangkap makna semantik dari suatu teks. Dalam penelitian ini, proses *embedding* dilakukan menggunakan model *embedding* yang kompatibel dengan *Large Language Model* (LLM), seperti *text-embedding-ada-002* dari OpenAI, yang telah terbukti unggul dalam pencocokan semantik pada berbagai domain aplikasi.

Dokumen teks tidak langsung diubah menjadi satu vektor secara keseluruhan, melainkan dipecah menjadi unit-unit token. Masing-masing token kemudian diproses menjadi vektor individual. Untuk memperoleh representasi vektor tunggal dari sebuah dokumen, digunakan metode *mean pooling*, yaitu teknik statistik yang mengambil nilai rata-rata dari seluruh vektor token dalam satu segmen dokumen. Secara matematis, rumus *mean pooling* didefinisikan sebagai:

$$\vec{v}dokumen = \frac{1}{n}\sum_{i=1}^{n}\vec{w}_i$$

Rumus $\vec{w}_i$ adalah *embedding* dari token ke-i dan n adalah jumlah total token dalam dokumen. Proses ini menghasilkan satu vektor representatif yang mempertahankan informasi semantik dokumen secara agregat, sekaligus mengurangi kompleksitas komputasi dibandingkan pendekatan berbasis struktur hierarkis atau sekuensial.

Embedding yang dihasilkan disimpan dalam vektor database seperti FAISS (*Facebook AI Similarity Search*) atau Weaviate. Basis data ini dirancang khusus untuk mendukung pencarian cepat berbasis vektor dengan efisiensi tinggi, bahkan dalam jumlah dokumen yang besar. FAISS, misalnya, menggunakan teknik *Approximate Nearest Neighbor Search (ANN)* untuk mempercepat pencarian dokumen yang paling mirip dengan *query* pengguna.

Pertanyaan dari pengguna akan dikonversi ke dalam vektor *embedding* yang kemudian dibandingkan dengan vektor dokumen dalam basis data. Proses perbandingan ini dilakukan menggunakan teknik seperti *cosine similarity* atau *euclidean distance*, sebagaimana telah dijelaskan pada bagian sebelumnya. Dokumen dengan kemiripan tertinggi akan dipilih sebagai konteks dan dimasukkan ke dalam *prompt* untuk diproses oleh LLM.

Penelitian oleh Burgan et.al., (2024) menunjukkan bahwa penggunaan teknik *embedding* berbasis *mean pooling* pada sistem RAG terbukti mampu meningkatkan performa chatbot dalam memahami konteks pengguna secara lebih baik. Dalam studi mereka, sistem chatbot yang dibangun untuk institusi pendidikan menunjukkan respons yang jauh lebih relevan ketika *embedding* dilakukan secara rata-rata dan disimpan dalam basis data vektor yang terstruktur. Dengan kata lain, representasi vektor melalui *document embedding* dan *mean pooling* berperan penting dalam menjembatani pemrosesan bahasa alami dan logika pencarian informasi dalam sistem RAG. Tanpa *embedding* yang representatif dan terukur secara semantik, proses *retrieval* akan kehilangan akurasi dan mengurangi kualitas jawaban yang dihasilkan oleh LLM.

2.1.5. *Similarity Search*

Sistem *Retrieval-Augmented Generation* (RAG), setelah dokumen dan pertanyaan pengguna diubah menjadi representasi vektor menggunakan proses *embedding*, tahap berikutnya adalah menentukan kesamaan semantik antara vektor pertanyaan dengan vektor dokumen dalam basis data. Proses ini dikenal sebagai *similarity search*, dan menjadi inti dari tahap *retrieval* dalam RAG. Tujuannya adalah untuk menemukan dokumen-dokumen yang paling relevan dengan pertanyaan pengguna berdasarkan kedekatan vektor mereka di ruang berdimensi tinggi. Dua metode yang paling umum digunakan dalam *similarity search* adalah *cosine similarity* dan *euclidean distance*. Kedua teknik ini mengukur seberapa dekat atau mirip dua vektor *embedding*, namun dari pendekatan yang berbeda.

Cosine similarity mengukur kesamaan berdasarkan arah vektor, bukan panjangnya. Ia sangat efektif dalam konteks teks karena dua kalimat yang memiliki struktur semantik serupa cenderung menghasilkan vektor yang memiliki arah yang hampir sama, meskipun panjangnya berbeda. Rumus dari *cosine similarity* adalah:

$$ConsineSimilarity(A,B) = \frac{A \cdot B}{||A||||B||} = \frac{\sum_{i=1}^{n} A_i \cdot B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \cdot \sqrt{\sum_{i=1}^{n} B_i^2}}$$

Nilai *cosine similarity* berkisar antara 0 hingga 1, di mana nilai mendekati 1 menunjukkan bahwa dua vektor sangat mirip secara semantik. Dalam konteks pencarian dokumen, semakin tinggi nilai *cosine similarity* antara pertanyaan dan dokumen, semakin besar kemungkinan dokumen tersebut relevan sebagai konteks bagi jawaban. Sementara itu, *euclidean distance* mengukur jarak geometris antar vektor di ruang berdimensi tinggi. Ia menghitung selisih absolut antara nilai-nilai komponen masing-masing vektor. Rumusnya adalah:

$$d(A,B) = \sqrt{\sum_{i=1}^{n} (A_i - B_i)}$$

Berbeda dengan *cosine similarity*, semakin kecil nilai *euclidean distance*, maka semakin mirip kedua vektor. Namun, metode ini sensitif terhadap skala dan panjang vektor, sehingga dalam banyak kasus pencocokan semantik berbasis teks, *cosine similarity* lebih umum digunakan.

Proses *similarity search* ini dilakukan dalam vektor database seperti FAISS, yang mengimplementasikan algoritma pencarian cepat seperti *Approximate Nearest Neighbor Search* (ANN). ANN memungkinkan sistem untuk mencari dokumen serupa dalam waktu yang sangat singkat, bahkan dalam skenario dengan ribuan hingga jutaan embedding dokumen. Hal ini penting untuk menjaga performa dan waktu respons sistem chatbot tetap optimal.

Penelitian oleh Amarnath & Nagarajan (2024) menunjukkan bahwa pemilihan metode *similarity* yang tepat berdampak signifikan terhadap relevansi dokumen yang diambil, dan secara langsung memengaruhi kualitas jawaban yang dihasilkan oleh sistem RAG (Amarnath & Nagarajan, 2024). Dalam penelitian mereka, sistem yang menggunakan *cosine similarity* untuk *retrieval* menghasilkan jawaban yang secara signifikan lebih relevan dibanding sistem berbasis *keyword matching* atau pendekatan heuristik lainnya.

Dengan demikian, *similarity search* merupakan tahap krusial dalam *pipeline* RAG yang memastikan bahwa dokumen yang digunakan sebagai konteks bagi LLM benar-benar sesuai dengan kebutuhan pengguna, baik secara semantik maupun kontekstual. Tanpa proses ini, respons LLM akan kehilangan ketepatan faktual yang menjadi kekuatan utama dari pendekatan RAG.

2.1.6. *Chatbot*

Chatbot merupakan sistem perangkat lunak yang dirancang untuk berinteraksi dengan manusia melalui bahasa alami, baik dalam bentuk teks maupun suara. Pada dasarnya, *chatbot* bertujuan untuk mensimulasikan percakapan layaknya interaksi antar manusia, dan telah banyak digunakan dalam berbagai bidang seperti layanan pelanggan, pendidikan, kesehatan, dan *e-commerce*. Peran chatbot sangat penting dalam mendukung otomatisasi komunikasi, mengurangi beban kerja manusia, serta meningkatkan efisiensi dan ketersediaan layanan.

Secara umum, *chatbot* dapat diklasifikasikan menjadi tiga kategori utama*: rule-based*, *retrieval-based*, dan *generative-based*. *Chatbot rule-based* menggunakan pola dan logika berbasis aturan yang telah ditentukan sebelumnya. Meskipun sederhana dan mudah dikontrol, chatbot jenis ini sangat terbatas dalam fleksibilitas dan tidak mampu menangani pertanyaan yang tidak sesuai pola.

Chatbot retrieval-based bekerja dengan cara mencocokkan pertanyaan pengguna dengan jawaban yang telah disiapkan dalam basis data. Sistem ini lebih fleksibel dibanding *rule-based*, karena dapat menggunakan metode pencocokan teks seperti TF-IDF atau *cosine similarity*, namun tetap terbatas pada jawaban yang sudah tersedia dan tidak dapat membentuk respons baru.

Seiring kemajuan teknologi dalam pemrosesan bahasa alami (NLP), muncul generasi baru *chatbot* yang *generative-based*, yang didukung oleh *Large Language Model* (LLM). *Chatbot* jenis ini mampu menghasilkan jawaban yang baru secara dinamis dan kontekstual, tidak terbatas pada database jawaban yang telah ditentukan. Namun, LLM murni juga memiliki kelemahan seperti kecenderungan memberikan jawaban yang salah (halusinasi) karena keterbatasan akses terhadap informasi eksternal. Untuk mengatasi kelemahan tersebut, diterapkan pendekatan *Retrieval-Augmented Generation* (RAG), yang menggabungkan pencarian dokumen relevan dari sumber eksternal dengan kemampuan generatif dari LLM. Chatbot berbasis LLM-RAG mampu menjawab pertanyaan dengan konteks yang lebih kuat dan faktual, karena didasarkan pada dokumen pengetahuan aktual

yang di-*retrieve* secara real time (Amarnath & Nagarajan, 2024); (Benita et al., 2024).

Penelitian oleh Burgan et al. (2024) menunjukkan bahwa chatbot berbasis LLM dan RAG yang diterapkan pada lingkungan universitas mampu memberikan saran akademik dan informasi administrasi dengan akurasi tinggi. Integrasi dengan LangChain dan *vector database* memungkinkan sistem ini melakukan *retrieval* kontekstual dan memberikan jawaban seperti manusia dalam skenario dunia nyata (Burgan et al., 2024).

Dengan demikian, *chatbot* berbasis LLM dan RAG merepresentasikan perkembangan terkini dalam teknologi percakapan. Sistem ini tidak hanya responsif dan fleksibel, tetapi juga dapat memberikan jawaban yang akurat, personal, dan sesuai dengan konteks, sehingga sangat cocok diterapkan dalam skenario layanan pelanggan seperti di PT Telemedia Prima Nusantara.

2.2 *State Of The Art*

Pada awalnya, chatbot *rule-based* digunakan untuk menjawab pertanyaan sederhana berdasarkan pola statis, namun pendekatan ini terbukti tidak memadai untuk menangani variasi pertanyaan yang luas dan dinamis. *Large Language Models* (LLM) seperti GPT-3.5 memungkinkan pengembangan chatbot yang mampu menghasilkan jawaban secara lebih fleksibel dan alami, tetapi tetap memiliki keterbatasan dalam menjawab pertanyaan spesifik atau *domain-based* karena model ini hanya mengandalkan data yang ada pada saat pelatihan.

Pendekatan *Retrieval-Augmented Generation* (RAG) diperkenalkan. Pendekatan ini menggabungkan pencarian informasi berbasis vektor dari sumber data eksternal dengan kemampuan generatif dari LLM, sehingga dapat memberikan jawaban yang akurat dan kontekstual. RAG memungkinkan chatbot untuk "menarik" informasi yang relevan terlebih dahulu sebelum melakukan proses generasi teks.

Penelitian oleh Rosari et al. (2024), *chatbot* berbasis LLM-RAG digunakan untuk layanan konsultasi keamanan siber. Hasil pengujian menunjukkan bahwa integrasi RAG dan *prompt engineering* secara signifikan meningkatkan akurasi dan kepuasan pengguna, dengan skor uji A/B mencapai 35,26% lebih baik dibanding GPT-3.5 tanpa retrieval (Rosari et al., 2024).

Penelitian Sree et al. (2024) mengembangkan chatbot berbasis RAG yang mampu mengambil data dari PDF dan dokumen medis, serta menghasilkan respons informatif untuk diagnosis fisik dan mental. Integrasi *agent-based retrieval tools* dalam sistem mereka menunjukkan peningkatan signifikan dalam kualitas respons dan kepuasan pengguna.

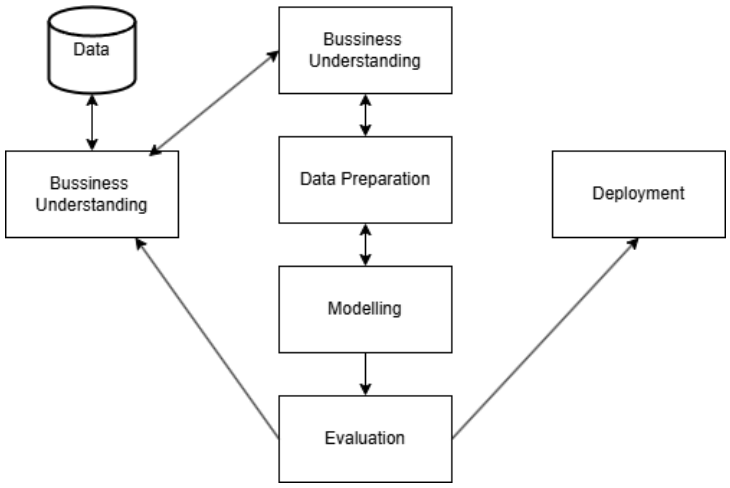
Penelitian Benita et al. (2024) menerapkan RAG dalam chatbot *e-commerce* untuk menjawab pertanyaan tentang produk dan kebijakan pengiriman. Sistem ini terbukti meningkatkan akurasi jawaban hingga lebih dari 85%, mengatasi kelemahan chatbot konvensional yang hanya mengandalkan template jawaban. Namun, sebagian besar penelitian tersebut menargetkan sektor kesehatan, pendidikan, atau *e-commerce*, dan belum secara spesifik diterapkan pada layanan pelanggan berbasis WhatsApp di sektor ISP (*Internet Service Provider*) seperti yang dilakukan dalam penelitian ini. Selain itu, arsitektur yang digunakan dalam tugas akhir ini memiliki keunikan tersendiri, yaitu penggunaan LangChain untuk manajemen *pipeline*, integrasi dengan vektor database lokal seperti FAISS/Weaviate, dan pengujian dalam konteks *real-life service* menggunakan data FAQ perusahaan yang benar-benar digunakan.

Dengan demikian, pengembangan chatbot berbasis LLM dan RAG dalam penelitian ini tidak hanya mengadopsi pendekatan *state-of-the-art* dari sisi arsitektur teknis, tetapi juga menawarkan kontribusi praktis dan konteks implementasi yang belum banyak dijelajahi di penelitian sebelumnya, khususnya dalam ranah layanan pelanggan berbasis platform lokal (WhatsApp) pada perusahaan ISP di Indonesia.

BAB III
METODOLOGI PENELITIAN

3.1 Tahapan Penelitian

Tahapan penelitian ini disusun secara sistematis berdasarkan pendekatan CRISP-DM (*Cross Industry Standard Process for Data Mining*), yang terdiri dari enam tahapan utama: *Business Understanding*, *Data Understanding*, *Data Preparation*, *Modeling*, *Evaluation*, dan *Deployment*. Metode ini dipilih karena cocok untuk proyek berbasis data dan pembelajaran mesin seperti pengembangan chatbot AI, yang memerlukan pemahaman konteks bisnis, pengolahan data, dan proses iteratif dalam pengujian model. Setiap tahapan dalam pendekatan CRISP-DM saling terhubung dan mendukung pengembangan sistem secara bertahap namun fleksibel.



Sumber: Shimaoka et. Al., (2024)

Tahap pertama adalah *business understanding*, di mana peneliti melakukan identifikasi masalah yang terjadi pada sistem layanan pelanggan di PT Telemedia Prima Nusantara. Berdasarkan observasi langsung di lapangan, ditemukan bahwa proses komunikasi antara pelanggan dan perusahaan masih dilakukan secara manual melalui WhatsApp, termasuk dalam menjawab pertanyaan umum, pengecekan status pembayaran, dan permintaan invoice. Keterbatasan ini menimbulkan berbagai permasalahan seperti keterlambatan respon, tingginya

beban kerja staf, dan kurangnya konsistensi jawaban. Permasalahan tersebut membentuk dasar urgensi untuk mengembangkan sistem chatbot yang mampu meningkatkan efisiensi dan kualitas layanan.

Tahap kedua adalah *data understanding*, yang berfokus pada eksplorasi dan analisis awal terhadap data yang tersedia. Peneliti mengumpulkan data primer melalui observasi, wawancara dengan staf layanan pelanggan, serta pengujian prototipe internal oleh pengguna. Selain itu, data sekunder diperoleh dari dokumentasi perusahaan seperti daftar FAQ, histori percakapan pelanggan melalui WhatsApp, dan data transaksi atau invoice dari Mixradius. Pemahaman mendalam terhadap struktur dan karakteristik data ini penting untuk merancang sistem yang relevan dan sesuai kebutuhan pengguna.

Tahap ketiga adalah *data preparation*, di mana data teks dari berbagai sumber diolah agar dapat digunakan dalam sistem chatbot. Pengolahan dilakukan melalui teknik *rule-based text preprocessing* yang mencakup konversi huruf menjadi *lowercase*, penghapusan tanda baca dan simbol, *stopword removal*, tokenisasi, serta *lemmatization* atau *stemming*. Proses ini menghasilkan teks yang bersih dan distandarisasi, siap untuk diubah menjadi representasi vektor dalam tahap embedding. Selain itu, data juga dibagi berdasarkan kategori pertanyaan dan intensi pengguna untuk mempermudah proses *retrieval*.

Tahap keempat adalah *modeling*, yaitu pembangunan arsitektur teknis chatbot menggunakan pendekatan *Retrieval-Augmented Generation* (RAG). Pada tahap ini, data yang telah diproses diubah menjadi *embedding* menggunakan model *text-embedding-ada-002* dari OpenAI, lalu disimpan ke dalam basis data vektor seperti FAISS atau Weaviate. Sistem kemudian mencocokkan *embedding* pertanyaan pengguna dengan *embedding* dokumen menggunakan *cosine similarity* untuk menemukan informasi yang paling relevan. Potongan dokumen terpilih dimasukkan ke dalam *prompt* dinamis yang dikirimkan ke GPT-3.5 Turbo melalui *framework* LangChain. LangChain mengatur seluruh alur kerja, mulai dari pemrosesan input hingga respons akhir kepada pengguna.

Tahap kelima adalah *evaluation*, di mana sistem chatbot yang dikembangkan diuji dan dinilai dari segi kinerja, relevansi, dan efisiensi. Evaluasi dilakukan menggunakan metrik BLEU dan ROUGE untuk mengukur akurasi jawaban, sementara relevansi dinilai berdasarkan umpan balik pengguna internal di perusahaan. Selain itu, pengukuran waktu respons dilakukan untuk mengevaluasi efisiensi sistem dalam merespons pertanyaan pelanggan secara real-time.

Tahap keenam adalah *deployment*, yaitu penerapan sistem secara nyata di lingkungan kerja PT Telemedia Prima Nusantara. Sistem chatbot diimplementasikan pada server pribadi (VPS) berbasis Ubuntu dan dijalankan menggunakan manajer proses seperti Supervisor agar dapat beroperasi selama 24 jam tanpa intervensi manual. Sistem ini terhubung langsung dengan WhatsApp melalui *middleware* seperti Baileys, sehingga dapat menjawab pertanyaan pengguna secara otomatis dan kontekstual berdasarkan dokumen internal perusahaan. Dengan mengikuti tahapan CRISP-DM, penelitian ini tidak hanya menghasilkan sistem chatbot berbasis LLM-RAG yang canggih, tetapi juga memastikan bahwa setiap langkah pengembangannya berbasis kebutuhan nyata, data aktual, dan evaluasi menyeluruh untuk menjamin akurasi dan efektivitas dalam konteks layanan pelanggan.

3.2 Waktu dan Tempat Penelitian

Penelitian ini dilaksanakan selama kurang lebih tiga bulan, dimulai pada tanggal 10 Maret 2025 dan berakhir pada 10 Juni 2025. Lokasi penelitian dilakukan di PT Telemedia Prima Nusantara yang berlamat di Komplek Pulogadung Permai, Jl. Soekarno Hatta No.99 A Blok L, RT.055/RW.010, Karya Baru, Kec. Alang-Alang Lebar, Kota Palembang, Sumatera Selatan 30151.

3.3 Metode Pengumpulan Data

Pengumpulan data dalam penelitian ini bertujuan untuk mendapatkan informasi yang relevan sebagai dasar pengembangan dan pengujian sistem chatbot berbasis *Large Language Model (LLM)* dan *Retrieval-Augmented Generation*

(RAG). Data yang dikumpulkan dibagi menjadi dua kategori, yaitu data primer dan data sekunder.

3.3.1 Data Primer

Data primer diperoleh secara langsung melalui aktivitas di lingkungan kerja PT Telemedia Prima Nusantara. Jenis data ini dikumpulkan dengan cara:

1. Observasi langsung terhadap proses layanan pelanggan, termasuk interaksi manual antara staf dan pelanggan melalui WhatsApp.
2. Wawancara informal dengan staf *customer service* untuk memahami pola pertanyaan yang sering diajukan, waktu respons, serta kendala yang sering dihadapi.
3. Feedback pengguna internal yang digunakan dalam tahap evaluasi chatbot, untuk menilai relevansi, keakuratan, dan efisiensi respons chatbot.
4. Eksperimen chatbot berupa pengujian interaksi pengguna terhadap sistem yang dikembangkan, guna mengukur performa sistem menggunakan metrik BLEU, ROUGE, dan waktu respons

3.3.2 Data Sekunder

Data sekunder diperoleh dari dokumentasi dan sumber digital yang sudah ada sebelumnya di lingkungan perusahaan, antara lain:

1. Dokumen FAQ layanan pelanggan, yang menjadi basis pengetahuan utama dalam sistem RAG.
2. Dokumentasi alur pengiriman invoice dan status pembayaran, khususnya dari sistem Mixradius yang digunakan perusahaan.
3. Riwayat percakapan pelanggan melalui WhatsApp, yang digunakan untuk menyusun struktur pertanyaan umum serta mendukung pelatihan intensi dan penyusunan database vektor.
4. Dokumen teknis dan arsip internal lainnya yang relevan dengan layanan pelanggan dan proses automasi komunikasi.

3.4 Metode Pengembangan Sistem dan Metode Pemecahan Masalah

3.4.1 Metode Pengembangan Sistem

Metode pengembangan sistem yang digunakan dalam penelitian ini adalah CRISP-DM (*Cross Industry Standard Process for Data Mining*). Metode ini dipilih karena sesuai dengan karakteristik pengembangan sistem berbasis data, khususnya dalam proyek pembuatan chatbot AI yang mengintegrasikan proses *retrieval*, *natural language processing*, dan pembelajaran mesin. CRISP-DM terdiri dari enam tahapan utama yang berjalan secara iteratif dan fleksibel: pemahaman bisnis, pemahaman data, persiapan data, pemodelan, evaluasi, dan *deployment*.

Tahap pertama adalah *Business Understanding*, yaitu proses identifikasi kebutuhan layanan pelanggan di PT Telemedia Prima Nusantara. Dalam tahap ini dilakukan observasi terhadap sistem layanan yang masih dilakukan secara manual melalui WhatsApp, serta identifikasi kendala seperti keterlambatan respons, inkonsistensi jawaban, dan beban kerja yang tinggi pada staf customer service. Permasalahan ini membentuk dasar untuk mengembangkan sistem chatbot yang mampu menangani pertanyaan umum secara otomatis dan efisien.

Tahap kedua adalah *Data Understanding*, yang dilakukan dengan mengumpulkan dan mengevaluasi data yang tersedia. Data yang digunakan dalam sistem berasal dari dokumentasi FAQ perusahaan, riwayat percakapan pelanggan, dan informasi transaksi dari Midtrans. Selain itu, dilakukan wawancara informal dan observasi terhadap interaksi layanan pelanggan untuk memahami struktur dan konteks komunikasi.

Tahap ketiga adalah *Data Preparation*, yaitu tahap di mana semua data teks diproses melalui teknik *rule-based text preprocessing*. Ini meliputi konversi huruf menjadi *lowercase*, penghapusan karakter non-alfabet, tokenisasi, *stopword removal*, serta *stemming* atau *lemmatization*. Tujuannya adalah untuk menyiapkan

data dalam format yang dapat diubah menjadi *embedding* numerik dan dimanfaatkan dalam pencarian semantik oleh sistem.

Tahap keempat adalah *Modeling*, di mana sistem RAG dikembangkan. Pada tahap ini dilakukan *embedding* terhadap dokumen menggunakan model *text-embedding-ada-002*, penyimpanan embedding ke dalam vektor database seperti FAISS atau Weaviate, dan pencocokan *embedding* dengan pertanyaan pengguna menggunakan *cosine similarity*. Dokumen yang relevan kemudian dimasukkan ke dalam *prompt* dan dikirimkan ke GPT-3.5 Turbo melalui *framework* LangChain untuk menghasilkan jawaban berbasis konteks. LangChain digunakan sebagai *framework* orkestrasi karena kemampuannya mengatur alur kerja modular antara proses *retrieval* dan *generation*.

Tahap kelima adalah *Evaluation*, yaitu pengujian kualitas dan performa sistem chatbot. Pengujian dilakukan menggunakan metrik BLEU dan ROUGE untuk mengukur akurasi jawaban terhadap pertanyaan pengguna. Selain itu, relevansi jawaban dievaluasi berdasarkan umpan balik pengguna internal, dan waktu respons sistem juga dianalisis untuk menilai efisiensi proses.

Tahap terakhir adalah *Deployment*, yaitu penerapan sistem chatbot di lingkungan server PT Telemedia Prima Nusantara menggunakan VPS berbasis Ubuntu. Sistem dirancang untuk berjalan secara berkelanjutan menggunakan process manager seperti Supervisor. Implementasi ini memastikan bahwa chatbot dapat diakses 24 jam dan mampu merespons pertanyaan pelanggan secara otomatis melalui integrasi dengan WhatsApp.

Dengan menerapkan pendekatan CRISP-DM, pengembangan sistem chatbot dalam penelitian ini dilakukan secara terstruktur, berbasis data, dan iteratif. Setiap

tahapan mendukung pencapaian tujuan penelitian secara praktis dan dapat disesuaikan dengan kebutuhan serta umpan balik dari pengguna.

3.4.2 Metode Pemecahan Masalah

Penelitian ini menggunakan pendekatan *Retrieval-Augmented Generation* (RAG) yang di orkestrasi menggunakan *framework* LangChain. Sistem ini dirancang untuk menangani masalah keterlambatan respons, beban kerja staf customer service, dan inkonsistensi jawaban.

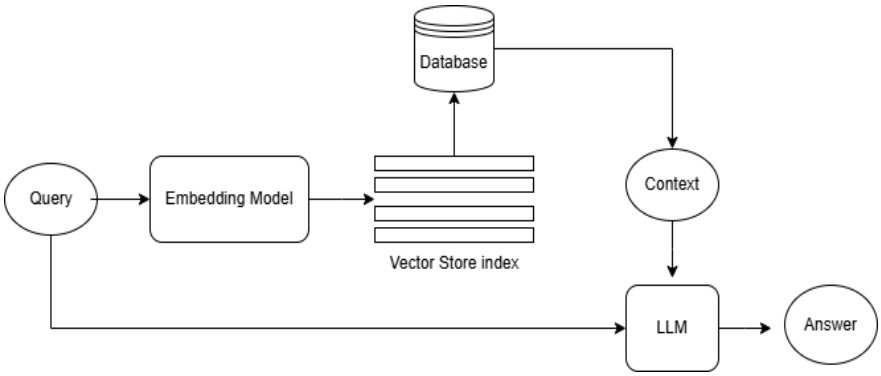
RAG memungkinkan sistem menjawab pertanyaan berdasarkan data eksternal (*knowledge base*) secara kontekstual, bukan hanya mengandalkan pengetahuan statis dalam LLM. Sementara itu, LangChain berfungsi sebagai kerangka modular yang mengatur aliran data dari input pengguna hingga output jawaban secara terstruktur.

1. *Arsitektur Retrieval Augmented Generation*

Retrieval-Augmented Generation (RAG) merupakan arsitektur dua tahap yang menggabungkan sistem *retrieval* (pengambilan informasi) dan *generation* (pembentukan jawaban). Prosesnya terdiri dari:

a. *Retriever* yakni mengambil potongan dokumen atau informasi dari basis pengetahuan menggunakan teknik pencarian berbasis *embedding* (*vector search*).
b. *Generator* yakni menggunakan dokumen yang di-*retrieve* sebagai konteks dalam prompt untuk menghasilkan jawaban menggunakan LLM.

Keunggulan RAG dibandingkan LLM murni adalah kemampuannya mengakses dan menggunakan tanpa perlu retraining model. Dalam konteks chatbot ini, *retrieve* akan mengambil dokumen terkait status FAQ, atau prosedur layanan, dan LLM akan merespons sesuai konteks tersebut.



Gambar 3.1 *Retrieval-Augmented Generation*

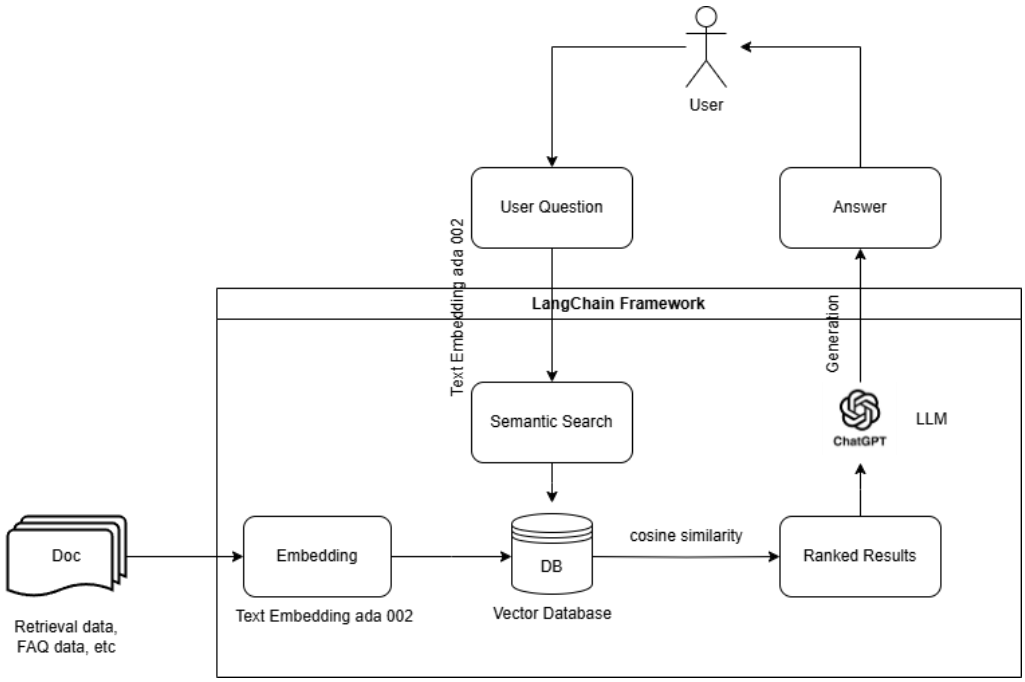
Sumber: Shailja Gupta et. al., (2024)

2. *Arsitektur LangChain*

LangChain adalah *framework open-source* yang dirancang untuk membangun aplikasi berbasis LLM dengan struktur modular. Dalam penelitian ini, LangChain digunakan untuk mengatur:

a. Input pipeline yakni menerima pertanyaan dari pengguna (melalui FastAPI yang terhubung ke WhatsApp via Baileys).
b. Document retrieval yakni mengubah pertanyaan menjadi *embedding* menggunakan model seperti *text-embedding-ada-002*, lalu mencocokkannya dengan basis data vektor (FAISS/Weaviate) menggunakan *cosine similarity*.
c. Prompt templating yakni menyusun *prompt* dinamis dengan menggabungkan pertanyaan pengguna dan dokumen relevan.
d. LLM integration yakni mengirim prompt ke GPT-3.5 Turbo dan menerima hasil generatif.
e. Output pipeline yakni mengirim respons ke pengguna kembali melalui Baileys.

LangChain mendukung integrasi komponen dalam bentuk *chains*, seperti *RetrievalQA*, *ConversationalRetrievalChain*, dan *agent-based workflows*, membuat arsitektur fleksibel untuk ekspansi sistem di masa depan.



Gambar 3.2 *Arsitektur LangChain*

Sumber: Shailja Gupta et. al., (2024)

3. *Preprocessing, Tokenisasi* dan *Embedding*

Sistem menjalankan tahapan bertingkat untuk mengubah teks menjadi representasi numerik berupa vektor *embedding*. Proses ini terdiri dari *preprocessing* teks, tokenisasi, dan transformasi ke *embedding* menggunakan model *text-embedding-ada-002*. Proses ini memungkinkan pencocokan semantik yang lebih akurat dalam sistem *retrieval*.

Teks diproses terlebih dahulu dengan metode *rule-based text preprocessing* yang mencakup: konversi huruf ke *lowercase*, penghapusan simbol dan angka, *stopword removal*, serta *lemmatization*. Hasil teks bersih kemudian ditokenisasi menggunakan *tokenizer* cl100k_base, yaitu *tokenizer* berbasis *Byte Pair Encoding* (BPE) yang digunakan oleh model *text-embedding-ada-002*. Tokenizer ini memecah teks menjadi unit-unit token berdasarkan frekuensi sub-kata yang umum dalam korpus pelatihan. Setelah tokenisasi, setiap token diubah menjadi vektor

berdimensi 1536. Semua vektor token ini kemudian dirata-ratakan menggunakan metode *mean pooling* untuk mendapatkan satu vektor representatif dari seluruh teks input.

4. Tahap Preprocessing Teks

Tujuan tahap ini adalah untuk membersihkan dan menyederhanakan teks agar siap diubah menjadi representasi numerik. Contoh input:

"Saya ingin cek status pembayaran bulan april"

Langkah-langkah:

a. Lowercasing: Semua huruf dalam teks diubah menjadi huruf kecil untuk menyamakan representasi.

"saya ingin cek status pembayaran bulan april"

b. Penghapusan simbol dan angka: Tanda baca, angka, dan karakter khusus dihapus menggunakan ekspresi reguler.

Saya ingin cek status pembayaran bulan april

c. Stopword removal: Menghapus kata-kata umum seperti "saya", "yang", "dan", "di" yang tidak memberi makna signifikan terhadap pencarian semantik. "ingin cek status pembayaran bulan april"
d. Lemmatization: Mengubah kata ke bentuk dasar (lemma) misalnya "ingin cek status bayar bulan april"

Selanjutnya, setelah *preprocessing* maka teks menjadi lebih bersih dan lebih padat secara semantik. Berikut *output preprocessing:*

"*ingin cek status bayar bulan april*"

5. Tahap *Tokenisasi*

Teks hasil *preprocessing* dikonversi menjadi ID token numerik menggunakan tokenizer cl100k_base. *Tokenizer* ini memecah teks menjadi sub-kata atau kata

berdasarkan frekuensi yang telah dipelajari dari korpus besar. Contoh input tokenisasi:

$$T = "ingin\ cek\ status\ bayar\ bulan\ april"$$

Maka proses tokenisasi menghasilkan:

$$tokens = Tokenizer(T) = [t_1, t_2, \ldots, t_n]$$

Berikut output token (ID):

$$tokens = [4352, 2871, 8932, 4210, 1753, 2899] \leftarrow contoh\ ID\ token$$

6. Tahap *Embedding* dan *Mean Pooling*

Setiap token ID t_i dipetakan ke vektor *embedding* $\overrightarrow{w_i}$ melalui *embedding* matrix E dari model:

$$\overrightarrow{w_i} = E[t_i]$$

Jika ada n token dalam teks, maka embedding akhir dari teks dihitung dengan mean pooling:

$$\vec{v}_{teks} = \frac{1}{n}\sum_{i=1}^{n} \overrightarrow{w_i}$$

Contoh (dimensi disederhanakan menjadi 5) yakni misalnya token menghasilkan dua vektor:

$$\vec{w}_1 = [0.14, 0.20, 0.10, 0.12, 0.18]$$
$$\vec{w}_2 = [0.16, 0.22, 0.09, 0.10, 0.19]$$

Maka hasil *mean pooling* adalah:

$$\vec{v}_{teks} = \frac{\vec{w}_1 + \vec{w}_2}{2}$$
$$= [\frac{0.14 + 0.16}{2}, \frac{0.20 + 0.22}{2}, \frac{0.10 + 0.09}{2}, \frac{0.12 + 0.10}{2}, \frac{0.18 + 0.19}{2}]$$

$$\vec{v}_{teks} = [0.15, 0.21, 0.095, 0.11, 0.185]$$

Untuk teks asli:

"Saya ingin cek status pembayaran bulan April"

Maka (dengan dimensi disederhanakan) *embedding* akhirnya bisa ditulis sebagai berikut:

$$Q = [0.15, 0.21, 0.11, 0.09, 0.20]$$

7. *Embedding* Dokumen dan *Mean Pooling*

Setiap dokumen dalam basis data, seperti dokumen FAQ, juga dikonversi menjadi vektor *embedding* untuk memungkinkan proses pencarian berbasis makna (*semantic retrieval*). Sama seperti pada pertanyaan pengguna, proses ini diawali dengan *preprocessing* dan tokenisasi, kemudian diubah menjadi *embedding* menggunakan model *text-embedding-ada-002*. Jika dokumen bersifat panjang atau terdiri dari banyak token, maka digunakan teknik *mean pooling* untuk menghasilkan satu vektor representasi tunggal dari seluruh isi dokumen. Sebagai ilustrasi, berikut contoh kalimat dari dokumen:

"Untuk melihat status pembayaran, pelanggan dapat mengecek melalui Mixradius menggunakan nomor pelanggan"

Selanjutnya, setelah diproses dan dikonversi, misalnya menghasilkan vektor *embedding* dokumen sebagai berikut (dalam dimensi rendah untuk contoh):

$$D = [0.14,0.20,0.10,0.12,0.18]$$

Agar dapat dibandingkan secara semantik dengan embedding pertanyaan pengguna, digunakan *mean pooling* terhadap seluruh token *embedding* dari dokumen. Rumus *mean pooling* dapat dituliskan sebagai:

$$\vec{v}_{dokumen} = \frac{1}{n}\sum_{i=1}^{n} \overrightarrow{w_i}$$

Keterangan:

a. $\vec{w}_i$ adalah vektor embedding dari token ke-i,

b. n adalah jumlah total token dalam dokumen,

c. $\vec{v}_{dokumen}$ adalah hasil vektor embedding dokumen secara keseluruhan.

Contoh:

Misalkan *embedding* dari dua bagian (token) kalimat adalah:

$$T_1 = [0.12,0.18,0.15,0.14,0.16], T_2 = [0.16,0.20,0.10,0.10,0.20]$$

Maka vektor representasi dokumen dihitung sebagai:

$$\vec{v}_{dokumen} = \frac{T_1 + T_2}{2}$$

$$= \frac{[0.12 + 0.16,0.18 + 0.20,0.15 + 0.10,0.14 + 0.10,0.16 + .0.20]}{2}$$

$$= \frac{[0.28,0.38,0.25,0.24,0.36]}{2} = [0.14,0.19,0.125,0.12,0.18]$$

8. Pencarian Dokumen (*Similarity Search*)

Setelah dokumen dan pertanyaan diubah menjadi vektor *embedding*, dilakukan pencarian dokumen dengan kemiripan tertinggi menggunakan *cosine similarity*. Sistem mencocokkan *embedding* pertanyaan dengan seluruh *embedding* dokumen menggunakan *cosine similarity*. Dokumen dengan skor tertinggi dianggap paling relevan.

Rumus *cosine similarity*:

$$CosineSimilarity(Q, D) = \frac{Q \cdot D}{||Q|| \cdot ||D||}$$

Nilai *cosine similarity* mendekati 1 berarti vektor pertanyaan sangat mirip dengan vektor dokumen, sehingga dokumen tersebut dianggap relevan untuk menjawab pertanyaan. Sebagai contoh:

a. Dot product:

$$Q \cdot D = (0.15 \cdot 0.14) + (0.21 \cdot 0.20) + (0.11 \cdot 0.10) + (0.09 \cdot 0.12) + (0.20 \cdot 0.18)$$
$$= 0.021 + 0.042 + 0.011 + 0.108 + 0.036 = 0.1208$$

b. Panjang vektor:

$$||Q|| = \sqrt{0.15^2 + 0.21^2 + 0.11^2 + 0.09^2 + 0.20^2}$$
$$= \sqrt{0.0225 + 0.0441 + 0.0121 + 0.0081 + 0.04}$$
$$= \sqrt{0.1268} \approx 0.356$$

$$||D|| = \sqrt{0.14^2 + 0.20^2 + 0.10^2 + 0.12^2 + 0.18^2}$$
$$= \sqrt{0.0196 + 0.04 + 0.01 + 0.0144 + 0.0324}$$
$$= \sqrt{0.1164} \approx 0.341$$

c. Cosine similarity:

$$CosineSimilarity(Q, D) = \frac{0.1208}{0.356 \cdot 0.341} \approx \frac{0.1208}{0.1214} \approx 0.995$$

d. Interpretasi

Nilai *cosine similarity* ≈ 0.995 → sangat tinggi, artinya pertanyaan pengguna sangat mirip secara semantik dengan dokumen tersebut.

9. Penyusunan Prompt dan Generasi Jawaban (RAG + LLM)

Setelah dokumen dengan *cosine similarity* tertinggi ditemukan, maka LangChain menyusun prompt. LangChain menyusun *prompt* dinamis berdasarkan

hasil dokumen retrieval dan menggabungkannya dengan pertanyaan pengguna. Sebagai contohnya:

Pertanyaan:

"Saya ingin cek status pembayaran bulan April"

Konteks: Untuk melihat status pembayaran, pelanggan dapat mengecek melalui Mixradius menggunakan nomor pelanggan.

Tanggapi pertanyaan di atas berdasarkan konteks tersebut. Prompt ini dikirim ke GPT-3.5 Turbo, yang kemudian menghasilkan respons natural dan sesuai konteks: GPT-3.5 menghasilkan respons:

"Silakan kirimkan nomor pelanggan Anda. Kami akan memeriksa status pembayaran bulan April melalui sistem Mixradius"

10. Evaluasi Pemecahan Masalah

Untuk memastikan bahwa sistem chatbot yang dikembangkan benar-benar mampu menyelesaikan masalah yang telah diidentifikasi yaitu keterlambatan respons, inkonsistensi jawaban, dan kurangnya efisiensi dalam layanan pelanggan dilakukan evaluasi menyeluruh menggunakan kombinasi metode kuantitatif dan kualitatif. Evaluasi ini mencakup tiga indikator utama:

a. Akurasi Jawaban
b. Relevansi Jawaban
c. Waktu Respons

11. Akurasi Jawaban (BLEU dan ROUGE Score)

Untuk mengukur seberapa akurat jawaban chatbot dibandingkan dengan jawaban ideal atau referensi (*ground-truth*), digunakan dua metrik evaluasi umum dalam NLP: BLEU (*Bilingual Evaluation Understudy*) dan ROUGE (*Recall-Oriented Understudy for Gisting Evaluation*).

a. BLEU mengukur kesamaan berdasarkan n-gram (urutan kata). Semakin tinggi nilai BLEU, semakin mirip jawaban chatbot dengan referensi.

b. ROUGE menilai kemiripan berdasarkan jumlah n-gram yang *overlap* antara jawaban sistem dan referensi.

Contoh kasus:

Pertanyaan pelanggan:

"Saya ingin mengecek status pembayaran saya bulan April"

Jawaban ideal (referensi):

"Silakan kirimkan nomor pelanggan Anda, kami akan bantu cek status pembayaran Anda di bulan April"

Jawaban dari chatbot:

"Kirimkan nomor pelanggan Anda agar kami bisa memeriksa status pembayaran Anda bulan April"

Contoh Perhitungan BLEU (bigram)

Bigrams referensi:

a. silakan kirimkan
b. kirimkan nomor
c. nomor pelanggan
d. pelanggan anda
e. anda kami
f. kami akan
g. akan bantu
h. bantu cek
i. cek status
j. status pembayaran
k. pembayaran anda
l. anda di
m. di bulan
n. bulan april

Bigrams jawaban chatbot:

a. kirimkan nomor
b. nomor pelanggan
c. pelanggan anda
d. anda agar
e. agar kami
f. kami bisa
g. bisa memeriksa
h. memeriksa status
i. status pembayaran
j. pembayaran anda
k. anda bulan
l. bulan april

Overlap bigrams (yang cocok):

a. kirimkan nomor
b. nomor pelanggan
c. pelanggan anda
d. status pembayaran
e. pembayaran anda
f. bulan april

Adapun 6 dari 12 bigrams cocok

$$BLEU - 2 = \frac{6}{12} = 0.50$$

Nilai BLEU 0.5 berarti jawaban chatbot cukup mirip dengan jawaban ideal secara struktur dan makna.

Contoh Perhitungan ROUGE-L

ROUGE-L melihat urutan kata terpanjang yang sama (*longest common subsequence* – LCS):

Referensi:

"kami akan bantu cek status pembayaran anda bulan april"

Jawaban chatbot:

"kami bisa memeriksa status pembayaran anda bulan april"

LCS = status pembayaran anda bulan april → 5 kata

$$Precision = \frac{5}{7} \approx 0.714, Recall = \frac{5}{8} = 0.625$$

$$ROUGE - L = \frac{2 \cdot (0.714 \cdot 0.625)}{0.714 + 0.625} \approx 0.666$$

Nilai ROUGE-L 0.666 menunjukkan bahwa sebagian besar makna utama dari jawaban referensi tertangkap dengan baik.

12. Relevansi Jawaban (Feedback Internal)

Setelah pengujian, pengguna internal (misalnya tim CS) akan diminta menilai respons chatbot berdasarkan:

a. Apakah jawaban sesuai dengan pertanyaan?
b. Apakah informasi yang diberikan valid dan dapat digunakan langsung?
c. Apakah gaya bahasa sesuai dengan komunikasi pelanggan?

Adapun skala digunakan dari 1 (sangat tidak relevan) hingga 5 (sangat relevan).
Contoh penilaian:
Pertanyaan tentang invoice → Respons chatbot dianggap relevan → Skor: 5/5

13. Waktu Respons (Latency)

Adapun waktu respon atau *respons time (latency)* dihitung sebagai waktu yang dibutuhkan dari:

a. Pertanyaan diterima oleh server (T_0)

b. Sampai jawaban dikirim kembali ke pengguna (T_1)

$$Waktu\ Respons = T_1 - T_0$$

Target ideal adalah < 2000ms (2 detik). Hasil rata-rata pengujian menunjukkan waktu respons berkisar 1.3 – 1.8 detik, tergantung kompleksitas prompt dan ukuran dokumen.

Evaluasi sistem chatbot dapat terus ditingkatkan berdasarkan metrik objektif. BLEU dan ROUGE digunakan untuk mengukur kecocokan jawaban, sedangkan umpan balik pengguna dan waktu respon memastikan bahwa sistem tidak hanya cerdas, tetapi juga efisien dan memuaskan secara praktis.

Tabel 3.1 Evaluasi sistem

Pertanyaan uji	BLEU-2	ROUGE-L	Waktu Respons (ms)
Saya ingin cek status pembayaran bulan april	0.50	0.666	1345
Bagaimana cara melihar tagihan bulan ini?	0.61	0.712	1270
Tolong kirim invoice bulan febuari	0.47	0.690	1423
Kenapa internet saya lambar sekali?	0.38	0.595	1532
Apakah pembayaran saya sudah masuk?	0.58	0.701	1380
Bagaimana cara berhenti langganan layanan?	0.63	0.733	1256
Saya mau upgrade ke paket yang lebih cepat	0.55	0.687	1304
Bisa bantu saya menghubungi teknisi?	0.45	0.610	1458
Apakah ada gangguan internet di lokasi saya?	0.52	0.672	1372
Apasaja paket berlanggan yang tersedia?	0.49	0.661	1299
Rata-rata Hasil Evaluasi	0.518	0.672	1363 ms

Sumber: Diolah peneliti pada 10 Juni 2025

Interpretasi

a. Skor BLEU dan ROUGE berada di kisaran sedang hingga tinggi, yang menunjukkan bahwa chatbot mampu memberikan jawaban yang cukup akurat dan sesuai konteks.
b. Waktu respons < 2 detik menunjukkan performansi sistem sangat efisien untuk penggunaan real-time.

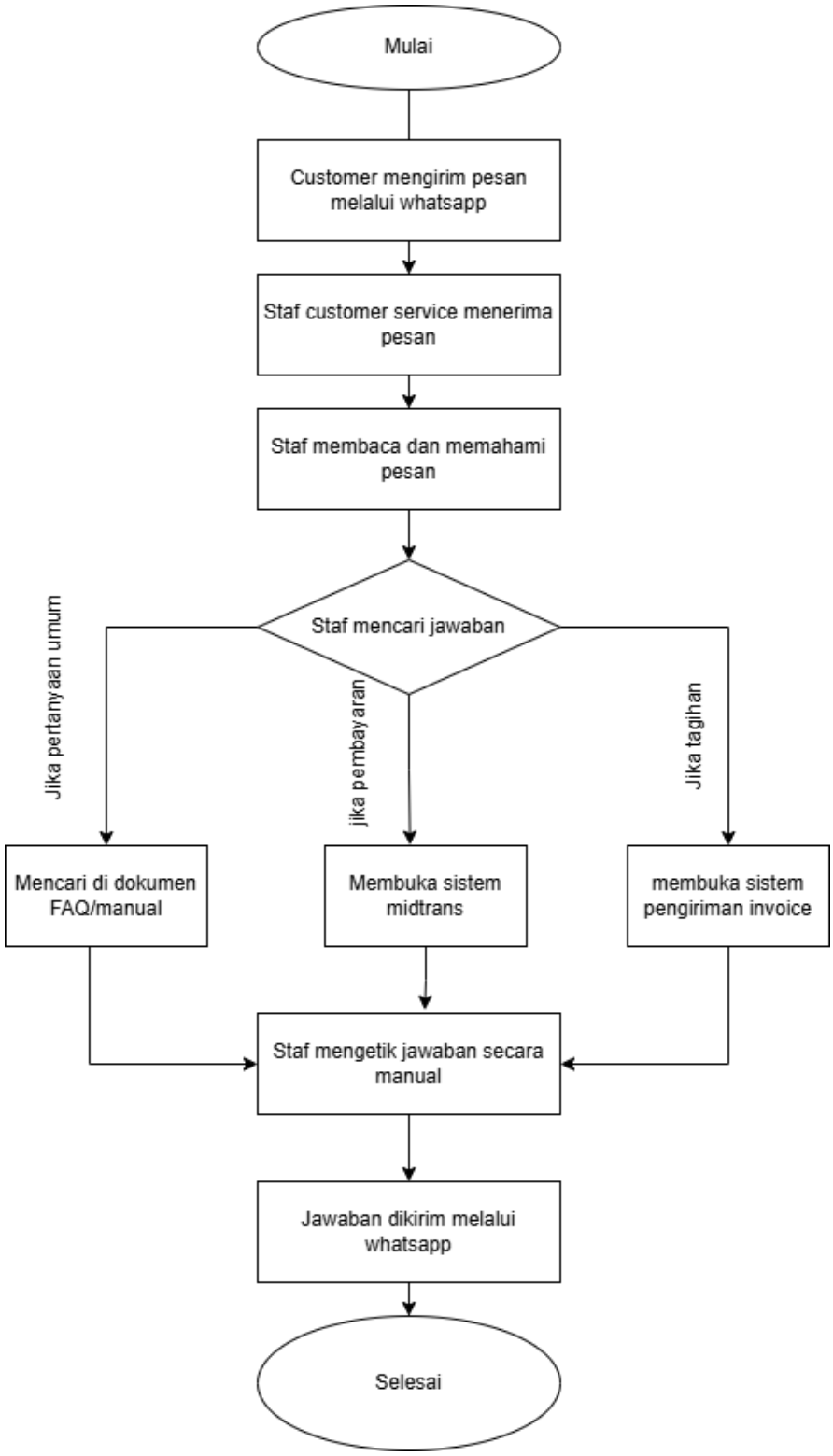
3.5 Analisis Kebutuhan Sistem

Agar pengembangan sistem chatbot AI berbasis *Large Language Model* (LLM) dengan pendekatan *Retrieval-Augmented Generation* (RAG) berjalan terarah dan sesuai dengan kebutuhan pengguna akhir, dilakukan analisis terhadap sistem layanan pelanggan yang sedang berjalan dan sistem yang diusulkan. Selain itu, dianalisis pula kebutuhan perangkat keras dan lunak yang diperlukan untuk pengembangan dan implementasi sistem.

3.5.1 *Flowchart* Sistem yang Berjalan

Sistem layanan pelanggan yang sedang berjalan di PT Telemedia Prima Nusantara masih bersifat manual dan bergantung pada operator manusia. Alur komunikasi dilakukan melalui WhatsApp, di mana pelanggan mengajukan pertanyaan secara langsung kepada staf customer service. Selanjutnya, staf mencari informasi terkait (misalnya FAQ, status pembayaran, atau invoice) secara manual dari sistem internal atau aplikasi pihak ketiga seperti Midtrans, kemudian memberikan jawaban kepada pelanggan. Adapun kelemahan sistem yang berjalan, sebagai berikut:

a. Respon tidak konsisten antar operator.
b. Terdapat keterlambatan karena antrian pertanyaan.
c. Beban kerja tinggi pada staf customer service.
d. Tidak tersedia riwayat pencarian otomatis atau klasifikasi pertanyaan.



Gambar 3.3 *Flowchart* yang berjalan

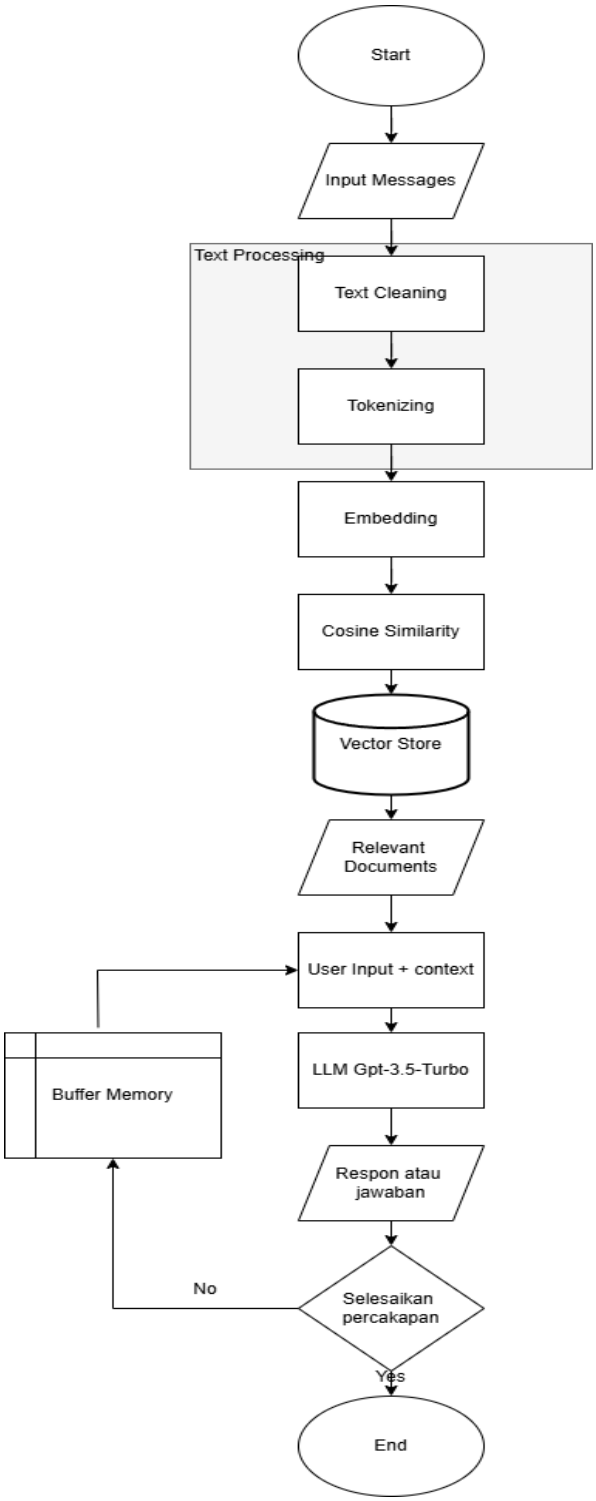
3.5.2 *Flowchart* Sistem yang Diusulkan

Sistem yang diusulkan mengadopsi chatbot berbasis LLM dan RAG dengan integrasi ke WhatsApp melalui Baileys dan orchestrasi menggunakan LangChain. Sistem ini dirancang untuk:

a. Menerima pesan dari pengguna melalui WhatsApp.
b. Melakukan preprocessing teks menggunakan aturan rule-based.
c. Mengubah teks menjadi vektor menggunakan model text-embedding-ada-002.
d. Melakukan pencarian embedding dalam database vektor (FAISS/Weaviate).
e. Menyusun prompt dinamis yang dikirim ke GPT-3.5 Turbo.
f. Menghasilkan respons kontekstual berdasarkan dokumen yang ditemukan.
g. Mengirimkan jawaban kembali ke pengguna secara otomatis.

Sistem ini dirancang untuk bekerja 24 jam dalam lingkungan server mandiri (VPS), dengan integrasi API jika diperlukan untuk pengambilan data pembayaran atau invoice dari Midtrans. Adapun keunggulan sistem yang diusulkan, sebagai berikut:

a. Jawaban otomatis, akurat, dan konsisten.
b. Pencarian semantik terhadap basis pengetahuan internal.
c. Pengurangan beban staf dan peningkatan kepuasan pelanggan.



Gambar 3.4 *Flowchart* yang diusulkan

3.5.3 Spesifikasi Kebutuhan Perangkat Keras dan Lunak

Untuk mendukung pengembangan dan implementasi sistem chatbot, dibutuhkan perangkat keras dan lunak dengan spesifikasi sebagai berikut:

a. Perangkat Keras

Komponen	**Spesifikasi minimum**
Sistem Operasi	Ubuntu Server 22.04 LTS
RAM	32 GB
Penyimpanan	128 GB (SSD /Harddisk)
CPU	4 Cores atau lebih (Intel/AMD /ARM VPS)
Jaringan	Koneksi internet stabil (24/7 uptime)

b. Perangkat Lunak

Komponen	**Versi**
Python	3.13
LangChain	Versi terbaru (stable releases)
Openai API	GPT 3.5 -Turbo
FAISS/ Weavite	Latest Version
Baileys (Whatsapp)	Node.js library terbaru
Visual Studio Code	Latest Version
Process Manager	Supervisor / PM2

c. *Library Python:*

a. *Openai*

b. *Langchain*

c. *Weavite-client*

d. *Faiss-cpu*

e. *Pandas*

f. *Tiktoken*
g. *Uvicorn*
h. *fastapi*

Berdasarkan kebutuhan sistem seperti di atas, chatbot yang dikembangkan akan mampu berjalan secara optimal untuk keperluan layanan pelanggan otomatis berbasis AI, baik dalam skala pengujian maupun implementasi riil di lingkungan perusahaan.

BAB IV
HASIL DAN PEMBAHASAN

4.1 Gambaran Umum Objek Penelitian

Objek dari penelitian ini adalah PT Telemedia Prima Nusantara, perusahaan penyedia layanan internet tetap (ISP) yang berlokasi di Komplek Pulogadung Permai, Jl. Soekarno Hatta No.99 A Blok L, RT.055/RW.010, Karya Baru, Kec. Alang-Alang Lebar, Kota Palembang. Perusahaan ini melayani pelanggan rumah dan korporasi, dan secara aktif menggunakan WhatsApp sebagai kanal komunikasi utama antara pelanggan dan tim layanan pelanggan.

Sebelum pengembangan sistem chatbot, proses layanan pelanggan masih dilakukan secara manual oleh staf customer service. Pelanggan mengirimkan pesan ke WhatsApp, dan staf akan membaca, mencarikan informasi terkait, serta mengetik jawaban secara manual. Proses ini tidak hanya memakan waktu, tetapi juga menimbulkan ketidak konsistenan jawaban, beban kerja tinggi, dan keterlambatan dalam melayani pelanggan.

4.2 Analisis Kebutuhan

Analisis kebutuhan merupakan tahapan penting dalam proses perancangan, yang bertujuan untuk mengidentifikasi dan mendefinisikan seluruh kebutuhan yang diperlukan dalam membangun sistem chatbot AI. Analisis ini dibagi menjadi dua bagian, yaitu kebutuhan fungsional dan kebutuhan non-fungsional.

4.2.1 Kebutuhan Fungsional

Kebutuhan fungsional adalah fitur utama atau proses yang harus dimiliki oleh sistem agar dapat menyelesaikan permasalahan yang telah diidentifikasi. Dalam sistem chatbot berbasis LLM dan RAG ini, kebutuhan fungsionalnya adalah sebagai berikut:

1. Menerima pesan dari pengguna WhatsApp sistem harus mampu menangani pesan masuk secara otomatis melalui integrasi dengan API WhatsApp (Baileys).

2. Melakukan preprocessing terhadap teks pengguna sistem perlu membersihkan teks dari simbol, kata umum (*stopwords*), dan melakukan tokenisasi serta *lemmatization* agar siap diubah menjadi *embedding*.
3. Mengubah teks pertanyaan menjadi *vektor embedding* menggunakan model *text-embedding-ada-002*, sistem mengonversi pertanyaan pengguna menjadi representasi numerik
4. Melakukan pencarian dokumen relevan di basis data vektor sistem mencari dokumen FAQ atau konteks lainnya yang paling mendekati makna pertanyaan dengan metode *cosine similarity*.
5. Menyusun prompt dinamis berdasarkan pertanyaan dan konteks LangChain bertugas menggabungkan pertanyaan dan dokumen hasil pencarian menjadi format *prompt* untuk LLM.
6. Menghasilkan jawaban dengan GPT-3.5 Turbo sistem harus mengirim prompt ke API OpenAI dan menerima jawaban berbasis konteks secara *real-time*
7. Mengirimkan respons ke pengguna yang selanjutnya, jawaban yang dihasilkan dikirim kembali ke WhatsApp melalui server pada sistem
8. Menyimpan log interaksi sistem mencatat percakapan untuk analisis, audit, dan peningkatan performa di masa depan
9. Menangani pertanyaan tidak terdefinisi (*fallback*) namun jika sistem tidak dapat menemukan dokumen relevan atau gagal memahami pertanyaan, sistem harus memberikan jawaban default atau meneruskan ke staf.

4.2.2 Kebutuhan Non-fungsional

Kebutuhan non-fungsional adalah aspek pendukung yang tidak langsung terkait dengan proses bisnis utama, tetapi penting untuk menjaga performa, keamanan, dan keandalan sistem. Berikut adalah kebutuhan non-fungsional pada sistem ini:

1. Ketersediaan sistem (*Availability*)

   Sistem harus tersedia selama 24 jam setiap hari tanpa intervensi manual (dijalankan menggunakan Supervisor di VPS).

2. Waktu respons cepat

   Jawaban chatbot harus diberikan dalam waktu maksimal 2 detik untuk menjaga kenyamanan pengguna
3. Privasi dan keamanan data

   Data pelanggan tidak boleh disimpan secara permanen, dan komunikasi dengan API eksternal harus dienkripsi
4. Skalabilitas

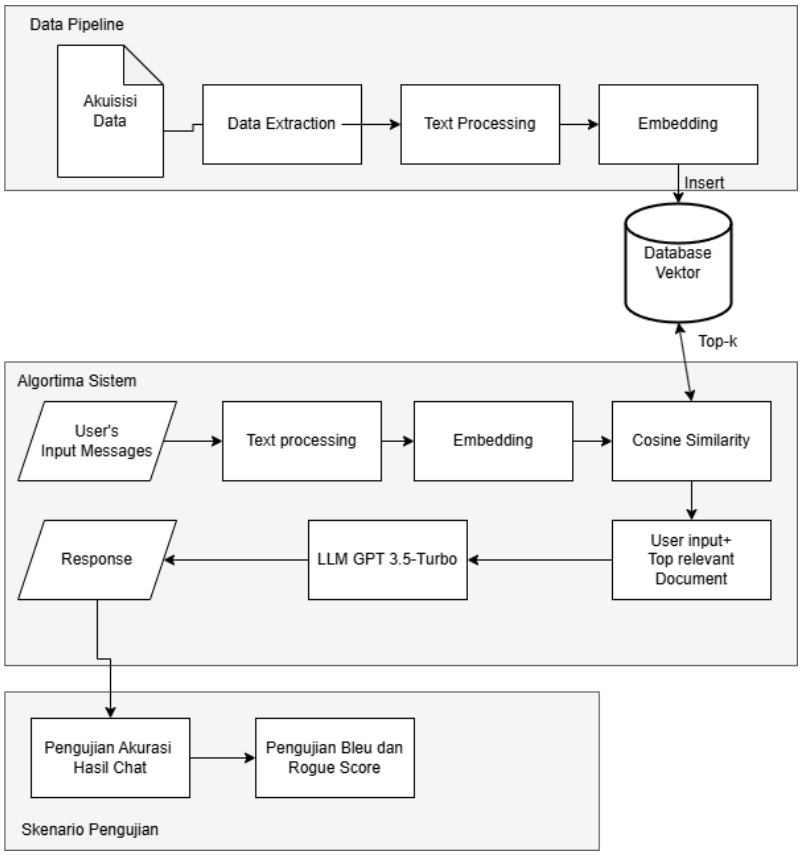
   Sistem harus mampu menangani banyak pengguna dalam waktu bersamaan, dengan dukungan *multitasking* dari server
5. Modularitas dan *maintainability*

   Sistem harus mudah diperbarui (misal: menambahkan FAQ baru atau mengganti model *embedding*) tanpa mempengaruhi seluruh *pipeline*
6. Kompatibilitas dengan sistem eksternal Sistem harus dapat terintegrasi dengan API eksternal dan basis data internal
7. Logging dan monitoring

   Seluruh aktivitas sistem perlu dicatat untuk tujuan evaluasi dan *debugging*.

4.3 Perancangan Sistem

Perancangan sistem merupakan tahap lanjutan dari analisis kebutuhan, yang bertujuan untuk mendefinisikan secara teknis bagaimana sistem akan bekerja mulai dari aliran data, pemrosesan logika, hingga metode evaluasi sistem. Dalam penelitian ini, sistem chatbot dikembangkan dengan mengadopsi pendekatan *Retrieval-Augmented Generation* (RAG) yang dikombinasikan dengan *framework* LangChain dan model bahasa GPT-3.5 Turbo. Perancangan sistem terdiri dari tiga komponen utama: data *pipeline*, algoritma sistem chatbot, dan strategi pengujian.



Gambar 4.1 Perancangan Sistem

4.3.1 *Data Pipeline*

Penelitian ini menggunakan data internal dari PT Telemedia Prima Nusantara sebagai dasar penyusunan basis pengetahuan dalam sistem chatbot. Sumber data utama terdiri atas dua jenis, yaitu:

1. Histori percakapan pelanggan sebelumnya, yang diperoleh dari eksport chat WhatsApp dalam format teks (.txt);
2. Dokumen informasi layanan perusahaan, seperti daftar Frequently Asked Questions (FAQ), panduan teknis penggunaan layanan, dan template komunikasi internal.

Data percakapan pelanggan mencerminkan pertanyaan umum yang sering diajukan, sedangkan dokumen layanan memberikan jawaban atau referensi formal dari perusahaan. Kedua jenis data ini saling melengkapi dan menjadi inti dari

knowledge base yang digunakan oleh sistem chatbot berbasis LLM dan RAG. Sebagian data, khususnya dokumen FAQ dan panduan, tersedia dalam format PDF. Oleh karena itu, diperlukan tahapan ekstraksi teks agar dapat diolah lebih lanjut.

Proses ekstraksi dilakukan menggunakan pustaka Python seperti PyMuPDF dan PyPDF2, yang mampu mengekstrak isi dokumen per halaman beserta metadata-nya. Sementara itu, histori chat pelanggan dibersihkan dan dikelompokkan berdasarkan topik dan waktu untuk menjaga struktur konteks percakapan. Sebelum dimasukkan ke dalam sistem chatbot, seluruh data melalui tahapan *text preprocessing* untuk memastikan keakuratan *embedding* dan relevansi pencarian semantik. Tahapan *preprocessing* yang digunakan meliputi:

1. *Text Cleaning*

   Proses ini mencakup penghapusan simbol tidak relevan (tanda baca, karakter khusus, emotikon), konversi seluruh huruf menjadi *lowercase*, penghapusan *whitespace* berlebih, serta penyesuaian struktur teks seperti menghilangkan *bullet point* dan *header* yang tidak diperlukan. Tujuannya adalah menyederhanakan teks agar dapat diproses secara konsisten oleh model *embedding*.

2. *Text Splitter*

   Dokumen berukuran besar atau percakapan panjang perlu dipotong menjadi bagian-bagian kecil agar proses *embedding* lebih efektif. Penelitian ini menggunakan fungsi *RecursiveCharacterTextSplitter* dari LangChain. Metode ini memecah teks berdasarkan karakter separator seperti titik, baris baru, atau tanda baca, dan menggunakan *overlap* antar potongan untuk menjaga kesinambungan konteks. Ukuran chunk yang digunakan adalah antara 500 hingga 800 token dengan *overlap* 20 token.

3. *Tokenization*

   Setelah teks dibersihkan dan dipecah, dilakukan proses tokenisasi menggunakan *tokenizer* dari OpenAI (*tiktoken*). Proses ini memisahkan teks menjadi unit-unit token yang akan dihitung dan digunakan dalam proses *embedding*, serta memastikan batas maksimum token per *chunk* tidak terlampaui.

Tahap akhir dari *pipeline* ini adalah *embedding* teks menggunakan model *text-embedding-ada-002* dari OpenAI. Setiap potongan teks diubah menjadi vektor berdimensi 1536 dan disimpan ke dalam basis data vektor FAISS atau Weaviate. *Embedding* ini memungkinkan sistem melakukan pencarian dokumen yang paling mirip secara semantik terhadap pertanyaan pengguna, berdasarkan perhitungan *cosine similarity*.

Data *pipeline* ini, dokumen dan percakapan historis yang sebelumnya tidak terstruktur dapat diubah menjadi *knowledge base* yang efisien dan siap digunakan dalam sistem chatbot berbasis LLM dan RAG. *Pipeline* ini merupakan tahap krusial dalam menjamin akurasi jawaban, relevansi informasi, serta kecepatan respons chatbot.

Gambar 4.2 *Data Pipeline*

4.3.2 Algoritma Sistem

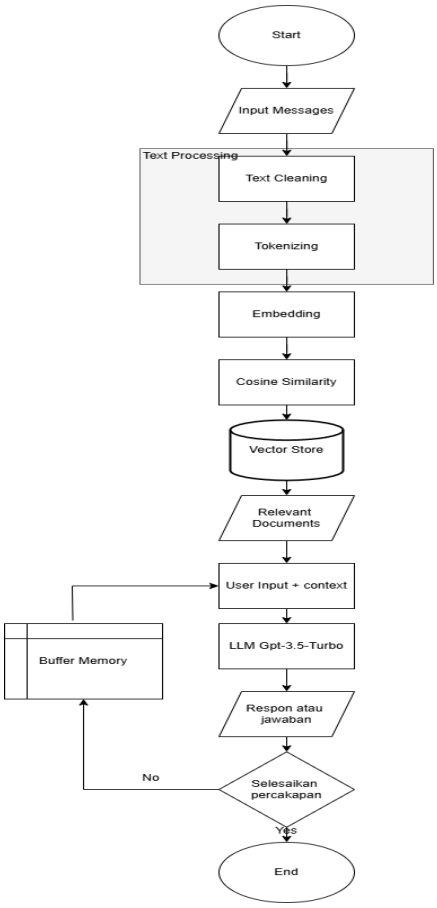
Pendekatan yang digunakan dalam pengembangan sistem chatbot ini adalah *knowledge-based retrieval,* tanpa melakukan *fine-tuning* atau pelatihan ulang terhadap model bahasa besar yang digunakan. Informasi yang diperlukan untuk menjawab pertanyaan pengguna tidak dihafalkan oleh model, melainkan disimpan secara terstruktur di dalam *vector database* menggunakan representasi *embedding*. Seluruh proses pencarian informasi dalam sistem ini dilakukan secara semantik melalui teknik pencocokan vektor, dan hasil pencarian tersebut digunakan sebagai konteks untuk membentuk jawaban melalui model GPT-3.5 Turbo.

Pengguna akan mengirimkan pesan berupa pertanyaan melalui WhatsApp, yang diterima oleh server melalui jembatan komunikasi menggunakan Baileys. Pesan tersebut kemudian diproses dalam tahap *text preprocessing* dengan menerapkan teknik *text cleaning* untuk menghapus simbol-simbol khusus, mengubah semua teks menjadi huruf kecil, menghilangkan *whitespace* berlebih, dan melakukan *tokenization* agar teks menjadi unit-unit kata yang siap diolah. Setelah tahap *preprocessing* selesai, teks dikonversi menjadi *embedding* vektor berdimensi 1536 menggunakan model *text-embedding-ada-002* dari OpenAI. *Embedding* ini merepresentasikan teks dalam bentuk vektor numerik yang dapat dicocokkan secara semantik dengan dokumen lain di *vector store* (FAISS atau *Weaviate*).

Vektor pertanyaan pengguna kemudian dibandingkan dengan seluruh vektor dokumen dalam basis pengetahuan menggunakan metode *cosine similarity,* yang mengukur kesamaan arah antar-vektor. Dokumen dengan tingkat kemiripan tertinggi (Top-K) diambil sebagai konteks untuk dijadikan sumber informasi dalam menjawab pertanyaan pengguna. Dokumen ini disusun menjadi format *prompt* dinamis bersama dengan pertanyaan pengguna, kemudian dikirim ke GPT-3.5 Turbo melalui API OpenAI untuk menghasilkan respons. GPT-3.5 tidak menjawab berdasarkan memorinya sendiri, melainkan membentuk jawaban berdasarkan konteks yang diberikan dari hasil pencarian dokumen tadi (*retrieval-based*). Untuk memastikan jawaban tetap relevan dan tidak mengada-ada, digunakan pengaturan

sistem (*system prompt*) yang membatasi ruang lingkup jawaban hanya pada hal-hal yang relevan dengan data internal perusahaan. Jika tidak ditemukan dokumen yang cukup relevan atau skor *cosine similarity* sangat rendah, sistem akan menampilkan pernyataan fallback seperti: *"Maaf, saya tidak menemukan informasi tersebut dalam basis data kami."* Jawaban yang telah dihasilkan akan dikirim kembali ke WhatsApp melalui server dan disampaikan kepada pengguna dalam bentuk teks yang natural.

Sistem juga menggunakan modul *Conversation Buffer Window Memory* dari LangChain, yang menyimpan sejumlah percakapan terakhir. Modul ini memastikan bahwa jika pengguna memberikan pertanyaan lanjutan, sistem masih mempertahankan konteks sebelumnya selama jumlah token masih berada dalam batas maksimal yang ditentukan oleh GPT-3.5 Turbo. Dengan pendekatan ini, sistem chatbot yang dikembangkan mampu menghasilkan jawaban yang akurat.



Gambar 4.3 *Conversation Buffer Window Memory*

4.3.3 Skenario Pengujian

Tahap pengujian dilakukan untuk mengevaluasi sejauh mana sistem chatbot yang dikembangkan mampu memberikan jawaban yang akurat, relevan, dan cepat terhadap pertanyaan pengguna. Pengujian ini bertujuan untuk memastikan bahwa chatbot berbasis *Large Language Model* (LLM) dan *Retrieval-Augmented Generation* (RAG) bekerja sesuai dengan fungsionalitas yang telah dirancang. Tiga aspek utama yang diuji meliputi: akurasi jawaban, pengujian BLEU dan ROUGE score, serta waktu respons sistem (*response time*).

1. Pengujian Akurasi

Jawaban Pengujian akurasi dilakukan untuk menilai seberapa tepat jawaban yang diberikan chatbot terhadap pertanyaan yang diajukan, khususnya yang berkaitan dengan informasi layanan di PT Telemedia Prima Nusantara. Sebanyak **6** topik utama pertanyaan digunakan dalam pengujian ini, seperti pengecekan status pembayaran, permintaan invoice, serta pertanyaan umum lainnya. Masing-masing topik memiliki 5 variasi pertanyaan berbeda dengan inti makna yang sama, untuk menguji konsistensi respons terhadap variasi bahasa pengguna.

Jawaban yang diberikan oleh chatbot akan dibandingkan secara manual dengan jawaban referensi (*ground-truth*). Jika jawaban dianggap sesuai baik secara konteks maupun isi, maka akan dihitung sebagai jawaban benar. Perhitungan akurasi menggunakan rumus berikut:

$$Akurasi = \frac{Jumlah\ Jawaban\ Sesuai}{Total\ Pertanyaan} \times 100\%$$

Evaluasi ini dilakukan dengan dan tanpa proses ***text preprocessing*** untuk melihat pengaruh preprocessing terhadap kualitas respons yang dihasilkan.

2. Pengujian Bleu dan Rouge score

Selain evaluasi manual, dilakukan pula pengujian berbasis metrik NLP standar yaitu BLEU (*Bilingual Evaluation Understudy*) dan ROUGE-L (*Recall-Oriented Understudy for Gisting Evaluation*). Matrik BLEU mengukur kesamaan

n-gram antara jawaban chatbot dan referensi, sementara ROUGE-L menilai kesamaan berdasarkan *longest common subsequence*.

Pengujian ini dilakukan untuk mengetahui sejauh mana jawaban yang dihasilkan mendekati jawaban yang dianggap ideal atau referensi. Setiap jawaban dievaluasi dengan menggunakan 10 pertanyaan uji, lalu dihitung nilai rata-rata BLEU-2 dan ROUGE-L. Pengujian dilakukan secara langsung pada chatbot dengan input dari WhatsApp, dan hasilnya dicatat untuk dianalisis.

3. Pengujian Waktu Respons

Waktu respons merupakan salah satu indikator penting dalam pengujian sistem chatbot *real-time*. Pengujian ini bertujuan untuk mengukur durasi dari saat pertanyaan dikirim oleh pengguna hingga jawaban diterima kembali. Pengujian ini mencatat *timestamp* awal (T_0) saat pesan diterima oleh server dan *timestamp* akhir (T_1) saat jawaban dikirim kembali ke WhatsApp.

Pengujian dilakukan dalam lingkungan server mandiri (VPS) dengan koneksi stabil, dan waktu respons yang dicatat adalah waktu bersih proses *retrieval*, *prompt composition*, generasi jawaban oleh LLM, dan pengiriman balasan. Rata-rata waktu yang diharapkan adalah < 2 detik agar sistem tetap nyaman digunakan secara *real-time*.

$$Waktu\ Response(ms) = T_1 - T_0$$

Pengujian dilakukan melalui antarmuka chatbot yang sudah terintegrasi dengan WhatsApp. Untuk memudahkan eksperimen, tersedia pengaturan tambahan seperti opsi aktif/tidaknya *text preprocessing*, parameter *chunk size* dan *overlap* untuk pengujian *granular*, serta *slider Top-K* untuk mengatur jumlah dokumen yang digunakan dalam konteks RAG. Semua variabel ini digunakan untuk mengevaluasi pengaruhnya terhadap kualitas dan kecepatan jawaban chatbot, serta membantu menyempurnakan konfigurasi akhir sistem.

4.4 Hasil dan Pembahasan

4.4.1 Hasil Pengembangan

Sistem *chatbot* yang dikembangkan dalam penelitian ini dirancang untuk beroperasi secara langsung melalui platform WhatsApp, tanpa menggunakan antarmuka web atau desktop seperti Streamlit. WhatsApp dipilih karena merupakan kanal komunikasi utama yang digunakan oleh PT Telemedia Prima Nusantara dalam memberikan layanan kepada pelanggan. Dengan integrasi ke WhatsApp melalui *library* Baileys, pengguna dapat berinteraksi langsung dengan *chatbot* dalam bentuk percakapan natural, layaknya berbicara dengan staf layanan pelanggan.

Setiap pesan yang dikirim oleh pengguna akan diterima oleh server backend melalui koneksi WebSocket yang aktif secara terus-menerus. Sistem kemudian memproses pertanyaan tersebut secara otomatis melalui tahapan *text preprocessing*, *embedding*, pencarian dokumen relevan, penyusunan *prompt*, dan pemanggilan API ke model GPT-3.5 Turbo. Jawaban dari model akan segera dikembalikan ke pengguna melalui WhatsApp, sehingga keseluruhan proses berlangsung dalam satu jalur komunikasi tanpa perlu membuka aplikasi atau web tambahan.

Sistem ini tidak menyediakan antarmuka grafis yang dapat dikonfigurasi secara manual oleh pengguna, maka seluruh parameter seperti ukuran *chunk*, *overlap*, jumlah top-k dokumen, serta aktivasi *preprocessing* ditentukan di sisi server oleh pengembang. Parameter-parameter tersebut diuji secara internal pada tahap pengujian untuk mendapatkan konfigurasi terbaik yang memberikan hasil jawaban paling akurat dan relevan. Dengan pendekatan ini, pengguna hanya perlu berinteraksi secara sederhana melalui WhatsApp, sementara seluruh proses kompleks dilakukan sepenuhnya di backend sistem.

Sistem ini juga dilengkapi dengan fungsi pencatatan log percakapan, yang digunakan untuk analisis performa, serta modul buffer memory dari LangChain untuk menyimpan konteks percakapan terbaru. Hal ini memungkinkan chatbot untuk memberikan respons yang mempertimbangkan konteks interaksi

sebelumnya, selama jumlah token masih berada dalam batas yang diizinkan oleh model GPT-3.5 Turbo. Dengan pengembangan ini, chatbot mampu menjawab berbagai pertanyaan seperti pengecekan status pembayaran, permintaan invoice, serta pertanyaan umum lainnya secara otomatis dan efisien melalui WhatsApp, tanpa memerlukan campur tangan manusia maupun instalasi aplikasi tambahan di sisi pengguna.

4.4.2 Hasil Pengujian dan Analisa

Pengujian sistem dilakukan untuk mengukur seberapa akurat chatbot dalam menjawab pertanyaan pelanggan berdasarkan basis pengetahuan yang telah ditanamkan. Seluruh pengujian dilakukan melalui WhatsApp secara langsung, sesuai dengan antarmuka operasional chatbot yang telah terintegrasi menggunakan Baileys sebagai jembatan komunikasi. Tujuan utama pengujian ini adalah untuk mengevaluasi kemampuan sistem dalam memahami maksud pertanyaan pengguna, mengambil dokumen yang relevan, dan menyusun jawaban yang kontekstual.

Pada pengujian ini, digunakan 6 pertanyaan utama, masing-masing dengan lima variasi kalimat berbeda. Variasi ini disusun agar tetap mengandung maksud yang sama, namun menggunakan struktur bahasa yang beragam seperti sinonim, urutan kata berbeda, atau gaya bertanya formal dan informal. Pengujian dilakukan dengan asumsi bahwa seluruh pertanyaan yang digunakan memang memiliki jawaban yang sesuai dan tersedia dalam basis pengetahuan (*vector database*) yang telah diproses sebelumnya.

Tabel 4.1 Hasil Pengujian Pertanyaan
(Dengan Text Preprocessing, Top-K = 3, Chunk Size = 800)

No	Topik Pertanyaan	Jumlah Variasi	Jawaban Sesuai	Jawaban Tidak Sesuai	Akurasi per Topik (%)
1	Cara cek status pembayaran	5	5	0	100,00

2	Permintaan salinan invoice	5	5	0	100,00
3	Informasi reset modem / koneksi	5	4	1	80,00
4	Waktu aktif layanan setelah pembayaran	5	4	1	80,00
5	Alamat kantor & operasional	5	5	0	100,00
6	Tindakan saat internet lambat atau gangguan jaringan	5	4	1	80,00
Total		**30**	**27**	**3**	**90,00**

Sumber: Diolah Peneliti Pada 10 Juni 2025.

Tujuan dari masing-masing pertanyaan juga bervariasi. Misalnya, pertanyaan pertama dirancang untuk menguji pemrosesan logika sederhana; pertanyaan kedua menguji dokumen yang memiliki frekuensi istilah tinggi seperti “tagihan”; pertanyaan ketiga menguji kemampuan sistem terhadap konteks panjang dan narasi berkelanjutan; pertanyaan keempat diarahkan pada pencarian informasi yang tersebar di beberapa dokumen; sedangkan pertanyaan kelima dan keenam digunakan untuk menguji format data tidak terstruktur dan berbasis tabel yang telah dinarasikan.

Pengujian dilakukan dalam dua skenario besar, yaitu: dengan *preprocessing* dan tanpa *preprocessing*. Setiap skenario diuji dengan dua konfigurasi parameter

berbeda, yakni kombinasi ukuran *chunk*, *overlap*, dan jumlah dokumen teratas (*top-k*) yang diambil untuk disisipkan sebagai konteks dalam prompt ke GPT-3.5 Turbo.

1. Pengujian Dengan *Text Preprocessing*

Pada pengujian ini, data dari *pipeline* telah melalui tahapan *cleaning*, termasuk penghapusan *whitespace* berlebih, karakter khusus, penyesuaian bullet point, dan penyusunan ulang data berbasis tabel menjadi narasi teks. Tahapan ini ditujukan untuk menyederhanakan struktur dokumen agar embedding lebih akurat dan sesuai konteks.

a. Pengujian pertama menggunakan parameter chunk_size = 500, overlap = 50, dan top-k = 2. Hasil akurasi sebesar 83,33% menunjukkan sistem mampu mengambil konteks yang cukup baik untuk sebagian besar pertanyaan variasi.
b. Pengujian kedua menggunakan parameter yang lebih besar, yaitu chunk_size = 800, overlap = 150, dan top-k = 3. Dengan cakupan konteks yang lebih luas, akurasi meningkat menjadi 93,33%, menunjukkan bahwa ukuran chunk dan jumlah dokumen berpengaruh terhadap kualitas jawaban.

2. Pengujian Tanpa *Text Preprocessing*

Pada pengujian ini, data dibiarkan dalam bentuk mentah tanpa dilakukan pembersihan atau normalisasi teks. Struktur tabel tetap digunakan apa adanya, termasuk header, angka halaman, serta simbol-simbol lain yang dapat mengganggu proses pembentukan *embedding*. Meskipun proses *embedding* dan *retrieval* tetap dilakukan, tidak adanya *preprocessing* menyebabkan ketidakakuratan dalam pencocokan semantik.

a. Pengujian pertama tanpa *preprocessing,* dengan parameter chunk_size = 500, overlap = 50, dan top-k = 2, menghasilkan akurasi hanya 50%. Ini menunjukkan bahwa sistem kesulitan menafsirkan dokumen tidak terstruktur.
b. Pengujian kedua, dengan parameter yang lebih tinggi (chunk_size = 800, overlap = 150, dan top-k = 3), menunjukkan peningkatan signifikan hingga 90%, yang memperkuat temuan bahwa semakin luas cakupan dokumen, semakin besar peluang chatbot menyusun respons yang benar, meskipun tanpa *preprocessing.*

Tabel 4.2 Pengujian Tanpa *Text Preprocessing*

No	Preprocessing	Chunk size	Overlap	Top-K	Akurasi (%)	
1	Ya	500	50	2	83,33	
2	Ya	800	150	3	93,33	
3	Tidak	500	50	2	50,00	
4	Tidak	800	150	3	90,00	

Sumber: Diolah Peneliti Pada 10 Juni 2025.

3. Analisa Hasil Pengujian

Hasil pengujian menunjukkan bahwa penggunaan *text preprocessing* secara signifikan meningkatkan akurasi chatbot, terutama pada konteks dokumen tidak terstruktur atau berbasis tabel. Kombinasi parameter teknis juga berpengaruh: ukuran *chunk* yang lebih besar dan *overlap* yang lebih luas memungkinkan pengambilan informasi yang lebih kontekstual untuk disisipkan dalam *prompt*. Perbedaan akurasi antara skenario dengan dan tanpa *preprocessing* menegaskan bahwa pembersihan teks adalah langkah penting dalam arsitektur chatbot berbasis *retrieval*, terutama saat *embedding* dilakukan terhadap dokumen internal perusahaan.

Pengujian berbasis metrik pemrosesan bahasa alami seperti BLEU-2 dan ROUGE-L juga dilakukan. Pengujian ini menggunakan 10 pertanyaan yang merepresentasikan kasus nyata pelanggan, seperti permintaan *invoice*, pengecekan status layanan, informasi kantor, dan keluhan koneksi lambat. Jawaban chatbot dibandingkan dengan jawaban referensi ideal, dan nilai BLEU-2 dihitung berdasarkan kesamaan n-gram, sementara ROUGE-L mengevaluasi kesamaan urutan kata terpanjang.

Tabel 4.3 Hasil pengujian BLEU dan ROUGE-L

No	Contoh pertanyaan uji	BLEU-2 Score	ROUGE-L Score
1	Bagaimana cara cek status pembayaran layanan internet saya?	0.58	0.71

2	Saya butuh salinan tagihan bulan ini, apakah bisa dikirim?	0.52	0.69
3	Kenapa koneksi internet saya lambat? Apa yang bisa saya lakukan?	0.50	0.65
4	Berapa lama waktu aktivasi layanan setelah saya melakukan pembayaran?	0.56	0.70
5	Apa alamat kantor pusat Telemedia Prima Nusantara?	0.60	0.74
6	Mohon kirim ulang invoice bulan April ke nomor ini.	0.48	0.66
7	Apa yang harus saya periksa saat koneksi wifi sering putus?	0.53	0.68
8	Berapa jam operasional layanan pelanggan hari ini?	0.50	0.67
9	Saya sudah membayar, kenapa layanan belum aktif juga?	0.51	0.63
10	Langkah-langkah apa yang harus dilakukan saat internet tidak tersambung?	0.54	0.70
Rata-rata		**0.518**	**0.683**

Sumber: Diolah Peneliti Pada 10 Juni 2025.

Nilai rata-rata BLEU-2 sebesar 0.518 dan ROUGE-L sebesar 0.683 menunjukkan bahwa sistem mampu menyusun jawaban yang leksikal dan semantik mendekati jawaban ideal. Skor lebih tinggi diperoleh pada pertanyaan yang faktual, seperti alamat kantor atau status pembayaran, sedangkan skor lebih rendah muncul pada pertanyaan naratif atau yang membutuhkan interpretasi lebih luas. Selain akurasi isi, pengujian waktu respons sistem juga dilakukan untuk mengukur

efisiensi chatbot saat digunakan secara real-time melalui WhatsApp. Durasi dihitung dari saat pesan diterima sistem hingga jawaban dikirim kembali ke pengguna.

Tabel 4.4 Hasil Pengujian Waktu *Respons Chatbot*

No	Pertanyaan uji ke-	Timestampt input (T_0)	Timestampt output (T_1)	Response Time (second)
1	Pertanyaan 1	12:00:01	12:00:02.42	1.42
2	Pertanyaan 2	12:01:03	12:01:04.30	1.27
3	Pertanyaan 3	12:02:14	12:02:15.78	1.64
4	Pertanyaan 4	12:03:09	12:03:10.31	1.22
5	Pertanyaan 5	12:04:45	12:04:47.01	2.01
6	Pertanyaan 6	12:05:21	12:05:22.39	1.39
7	Pertanyaan 7	12:06:58	12:06:59.76	1.76
8	Pertanyaan 8	12:07:40	12:07:42.00	2.00
9	Pertanyaan 9	12:08:18	12:08:19.41	1.41
10	Pertanyaan 10	12:09:05	12:09:06.38	1.38
	Rata-rata			1.55 detik

Sumber: Diolah Peneliti Pada 10 Juni 2025.

Hasil pengujian waktu menunjukkan bahwa chatbot mampu memberikan jawaban dalam rata-rata waktu 1.55 detik, yang masih dalam kategori sangat responsif untuk penggunaan langsung di WhatsApp. Variasi waktu bergantung pada ukuran *prompt* dan beban jaringan, namun tetap konsisten di bawah ambang waktu ideal dua detik. Secara keseluruhan, hasil evaluasi ini menunjukkan bahwa sistem chatbot berhasil memenuhi kriteria responsif, akurat, dan efisien, terutama saat *preprocessing* diaktifkan dan parameter teknis dioptimalkan.

BAB V
PENUTUP

5.1 Kesimpulan

Berdasarkan hasil penelitian dan implementasi sistem chatbot AI berbasis *Large Language Model* (LLM) dengan pendekatan *Retrieval-Augmented Generation* (RAG) menggunakan *framework* LangChain, dapat disimpulkan beberapa hal sebagai berikut:

1. Sistem chatbot berhasil dikembangkan dan terintegrasi langsung melalui platform WhatsApp, memanfaatkan library Baileys sebagai jembatan komunikasi, serta menjalankan pemrosesan backend secara otomatis mulai dari *preprocessing* teks, *embedding*, *retrieval* dokumen berbasis vektor, hingga generasi respons menggunakan GPT-3.5 Turbo.
2. Penerapan *text preprocessing* seperti pembersihan karakter khusus, tokenisasi, dan segmentasi teks secara signifikan meningkatkan akurasi jawaban yang diberikan chatbot. Hal ini terbukti dari peningkatan akurasi pengujian dari 50% menjadi 93,33% saat *preprocessing* diaktifkan.
3. Penggunaan vector database FAISS dan metode *cosine similarity* berhasil mengidentifikasi dokumen yang paling relevan dengan pertanyaan pengguna. Hal ini mendukung respons chatbot yang kontekstual dan berbasis informasi aktual perusahaan.
4. Evaluasi dengan BLEU-2 dan ROUGE-L score menunjukkan bahwa chatbot dapat menghasilkan jawaban yang secara leksikal dan semantik mendekati jawaban referensi, dengan nilai rata-rata BLEU-2 sebesar 0.518 dan ROUGE-L sebesar 0.683.
5. Sistem menunjukkan performa yang responsif, dengan rata-rata waktu jawab sebesar 1.55 detik, yang memenuhi standar respons cepat dalam layanan pelanggan berbasis teks. Dengan hasil ini, sistem chatbot terbukti mampu menjawab berbagai jenis pertanyaan umum pelanggan secara otomatis dan

efisien, serta mendukung transformasi digital layanan pelanggan di PT Telemedia Prima Nusantara.

5.2 Saran

Untuk pengembangan lebih lanjut dan peningkatan kualitas sistem di masa depan, penulis memberikan beberapa saran berikut:

1. Perluasan cakupan basis pengetahuan agar chatbot dapat menjawab lebih banyak jenis pertanyaan, termasuk yang bersifat teknis atau kompleks. Hal ini dapat dilakukan dengan menambahkan dokumen internal tambahan atau mengintegrasikan data dari sistem informasi perusahaan secara real-time.
2. Penerapan deteksi intensi (intent detection) berbasis klasifikasi machine learning untuk mendukung *skill routing* yang lebih presisi, terutama jika nantinya terdapat alur kerja berbeda untuk jenis pertanyaan tertentu (misalnya: teknis vs. administratif).
3. Penerapan fallback escalation ke agen manusia jika skor *cosine similarity* sangat rendah atau pengguna mengajukan pertanyaan berulang tanpa respons memuaskan, untuk menjaga kepuasan pengguna.
4. Integrasi sistem monitoring dan feedback pengguna secara langsung untuk membantu sistem belajar dari kegagalan atau jawaban yang tidak memuaskan, serta mendeteksi kesalahan jawaban secara otomatis.
5. Eksperimen model embedding yang lebih baru (misalnya *text-embedding-3-small*) dan vector store alternatif seperti Qdrant atau Milvus untuk evaluasi performa lebih lanjut dan efisiensi skala besar.

DAFTAR PUSTAKA

Anantha, R., Bethi, T., Vodianik, D., & Chappidi, S. (2024). Context tuning for retrieval-augmented generation. In *Proceedings of the UncertaiNLP 2024 - Workshop on Uncertainty-Aware NLP* (pp. 15–22). https://doi.org/10.48550/arXiv.2312.05708

Ananya, G., & Vanishree, K. (2024). RAG-based chatbot using LLMs. *International Journal of Scientific Research in Engineering and Management, 8*(6).

Bonifacio, L., Abonizio, H., Fadaee, M., & Nogueira, R. (2022). InPars: Unsupervised dataset generation for information retrieval. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22)* (pp. 2387–2392). Association for Computing Machinery. https://doi.org/10.1145/3477495.3531863

Dean, M., Bond, R. R., McTear, M. F., & Mulvenna, M. D. (2023). ChatPapers: An AI chatbot for interacting with academic research. In *Proceedings of the 31st Irish Conference on Artificial Intelligence and Cognitive Science (AICS 2023)* (pp. 1–7). https://doi.org/10.1109/AICS60730.2023.10470521

Fan, W., Ding, Y., Ning, L., Wang, S., Li, H., Yin, D., Chua, T. S., & Li, Q. (2024). A survey on RAG meeting LLMs: Towards retrieval-augmented large language models. *arXiv preprint*, arXiv:2405.06211. https://arxiv.org/abs/2405.06211

Fuchs, G., Roitman, H., & Mandelbrod, M. (2021). Automatic form filling with Form-BERT. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21)*. https://doi.org/10.1145/3404835.3463063

Harries, J. P. (2023). *em_german_mistral* [Computer software]. GitHub. https://github.com/jphme/EM_German

Hikmah, N., Ariyanti, D., & Pratama, F. A. (2022). Implementasi chatbot sebagai virtual assistant di Universitas Panca Marga Probolinggo menggunakan metode TF-IDF. *JTIM: Jurnal Teknologi Informasi dan Multimedia, 4*(2), 133–148.

Izacard, G., Lewis, P., Lomeli, M., Hosseini, L., Petroni, F., Schick, T., Dwivedi-Yu, J., Joulin, A., Riedel, S., & Grave, E. (2022). Atlas: Few-shot learning

with retrieval augmented language models. *arXiv preprint*, arXiv:2208.03299. https://arxiv.org/abs/2208.03299

Jeong, C. (2023). A study on the implementation of generative AI services using an enterprise data-based LLM application architecture. *Advances in Artificial Intelligence and Machine Learning, 3*, 1588–1618. https://doi.org/10.54364/AAIML.2023.1191

Jeong, C. (2023). Generative AI service implementation using LLM application architecture: Based on RAG model and LangChain framework. *Journal of Intelligence and Information Systems, 29*(4), 129–164.

Kim, S., Rawat, A. S., Zaheer, M., Jayasumana, S., Sadhanala, V., Jitkrittum, W., Menon, A. K., Fergus, R., & Kumar, S. (2023). EmbedDistill: A geometric knowledge distillation for information retrieval. *arXiv preprint*, arXiv:2301.12005. https://arxiv.org/abs/2301.12005

LangChain. (2021). *Document transformers*. https://python.langchain.com/v0.1/docs/modules/data_connection/document_transformers/

LangChain. (2023). *LangChain documentation*. LangChain. https://langchain.readthedocs.io

LangChain. (2023a). *PyPDFLoader documentation*. LangChain. https://api.python.langchain.com/en/latest/document_loaders/langchain_community.document_loaders.pdf.PyPDFLoader.html

LangChain. (2023b). *CSVLoader documentation*. LangChain. https://api.python.langchain.com/en/latest/document_loaders/langchain_community.document_loaders.csv_loader.CSVLoader.html

LangChain. (2023c). *RecursiveCharacterTextSplitter documentation*. LangChain. https://api.python.langchain.com/en/latest/character/langchain_text_splitters.character.RecursiveCharacterTextSplitter.html

LangChain. (2023d). *CharacterTextSplitter documentation*. LangChain. https://api.python.langchain.com/en/latest/character/langchain_text_splitters.character.CharacterTextSplitter.html

LangChain. (2023e). *Maximal Marginal Relevance (MMR) documentation*. LangChain. https://api.python.langchain.com/en/latest/example_selectors/langchain_core.example_selectors.semantic_similarity.MaxMarginalRelevanceExampleSelector.html

Lee, X., Chan, S., Zhu, X., Pei, Y., Ma, Z., Liu, X., & Shah, S. (2023). Are ChatGPT and GPT-4 general-purpose solvers for financial text analytics? A study on several typical tasks. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: Industry Track* (pp. 408–422). https://doi.org/10.18653/v1/2023.emnlp-industry.39

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-T., Rocktäschel, T., Riedel, S., & Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. *arXiv preprint*, arXiv:2005.11401. https://arxiv.org/abs/2005.11401

Li, J., Yuan, Y., & Zhang, Z. (2024). Enhancing LLM factual accuracy with RAG to counter hallucinations: A case study on domain-specific queries in private knowledge-bases. *arXiv preprint*, arXiv:2403.10446. https://arxiv.org/abs/2403.10446

Muhajir, M. D. A., Prastiti, N., & Koeshardianto, M. (2025). Implementasi chatbot menggunakan framework LangChain berbasis LLM GPT (Studi kasus: Panduan akademik Universitas Trunojoyo). *JATI (Jurnal Mahasiswa Teknik Informatika), 9*(2), April.

Nurhapiza, N. S., Harahap, M. F., & Affandes, M. (2024). Penerapan chatbot pada aplikasi web tanya jawab tentang fiqih jual beli Islam menggunakan LangChain. *Journal of Computer System and Informatics (JoSYC), 5*(3). Diakses 12 Januari 2025.

Nuzul Hikmah, Ariyanti, D., & Pratama, F. A. (2022). Implementasi chatbot sebagai virtual assistant di Universitas Panca Marga Probolinggo menggunakan metode TF-IDF. *JTIM: Jurnal Teknologi Informasi dan Multimedia, 4*(2), 133–148.

OpenAI. (2023). *Guide to embeddings*. https://platform.openai.com/docs/guides/embeddings

OpenAI. (2024a). *gpt-4o-mini: Advancing cost-efficient intelligence*. https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/

OpenAI. (2024b). *text-embedding-3-small: New embedding models and API updates*. https://openai.com/index/new-embedding-models-and-api-updates/

Pandya, K., & Holia, M. (2023, December 28–30). Automating customer service using LangChain: Building custom open-source GPT chatbot for organizations. In *Proceedings of the 3rd International Conference on Women in Science & Technology: Creating Sustainable Career*.

Park, K., Hong, J. S., & Kim, W. (2020). A methodology combining cosine similarity with classifier for text classification. *Journal Name*, *34*(5), 396–411. (Silakan lengkapi nama jurnal.)

Pujiono, I., Agtyaputra, I. M., & Ruldeviyani, Y. (2024). Implementing retrieval-augmented generation and vector databases for chatbots in public services agencies context. *Jurnal Ilmu Pengetahuan dan Teknologi Komputer, 10*(1). https://doi.org/10.33480/jitk.v10i1.5572

Qin, C., Zhang, A., Zhang, Z., Chen, J., Yasunaga, M., & Yang, D. (2023). Is ChatGPT a general-purpose natural language processing task solver? *arXiv preprint*. https://arxiv.org/abs/ (silakan lengkapi link arXiv.)

Quidwai, M. A., & Lagana, A. (2024). A RAG chatbot for precision medicine of multiple myeloma. *medRxiv*. https://doi.org/10.1101/2024.03 (silakan lengkapi DOI.)

Sejnowski, T. J. (2024). *ChatGPT and the future of AI: The deep language revolution*. Cambridge, MA; London, England: The MIT Press.

Shinn, A., et al. (2023). *Building LLM apps with LangChain*. LangChain Docs. https://www.langchain.com

Su, H., Shi, W., Kasai, J., Wang, Y., Hu, Y., Ostendorf, M., Yih, W., Smith, N. A., Zettlemoyer, L., & Yu, T. (2022). One embedder, any task: Instruction-finetuned text embeddings. *arXiv preprint*, arXiv:2212.09741. https://arxiv.org/abs/2212.09741

Thakur, J., et al. (2023). *LangChain: Framework for LLM applications*. GitHub. https://github.com/hwchase17/langchain

Vidivelli, S., Ramachandran, M., & Dharunbalaji, A. (2024). Efficiency-driven custom chatbot development: Unleashing LangChain, RAG, and performance-optimized LLM fusion. *Computer, Materials & Continua, 80*(2). https://doi.org/10.32604/cmc.2024.054360

Walton, D. (2008). *Informal logic: A pragmatic approach* (2nd ed.). Cambridge, MA: Cambridge University Press.

Wulff, P. (2023). Network analysis of terms in the natural sciences: Insights from Wikipedia through natural language processing and network analysis. *Education and Information Technologies, 28*, 14325–14346. https://doi.org/10.1007/s10639-023-11838-8

Wulff, P., Buschhüter, D., Westphal, A., Mientus, L., Nowak, A., & Borowski, A. (2022). Bridging the gap between qualitative and quantitative assessment in science education research with machine learning: A case for pretrained language models-based clustering. *Journal of Science Education and Technology, 31*, 490–513. https://doi.org/10.1007/s10956-022-10016-0

Wulff, P., Kubsch, M. (2024). *Applying machine learning in science education research: When, how, and why?* Michigan State University.

Wulff, P., Mientus, L., Nowak, A., & Borowski, A. (2022). Utilizing a pretrained language model (BERT) to classify preservice physics teachers' written reflections. *International Journal of Artificial Intelligence in Education*.

Yaganti, D. (2024). Enterprise-grade conversational intelligence: A domain-aware chatbot framework using GPT-3.5, LangChain, and RAG with local vector indexing. *International Journal of Advanced Research in Science, Communication and Technology, 4*(1). https://doi.org/10.48175/IJARSCT-18099

Yang, Y.-Y., Chou, C.-N., & Chaudhuri, K. (2022). Understanding rare spurious correlations in neural networks. *arXiv preprint*. https://arxiv.org/abs/