BAB II

LANDASAN TEORI

2.1 Penelitian Terdahulu

Tabel 2.0.1 Penelitian Terdahulu

Penulis dan Tahun	Judul	Metodologi dan Hasil
Yasmin & Fudholi (2025) [10]	Pengembangan chatbot Informasi Hukum Layanan Publik Berbasis retrieval-augmented generation Menggunakan Langchain Dan openai di ombudsman DIY	Menciptakan chatbot untuk urusan legal berbasis Chroma vector database untuk Ombudsman DIY **Hasil:** Skor tinggi di Perceived Usefulness (12.0) dan Accuracy (18.6).
Yun et al. (2024) [11]	Improving citizen-government interactions with generative artificial intelligence: Novel human-computer interaction strategies for policy understanding through large language models	Menggunakan *framework* RAG dan LLM untuk memahami dokumen kebijakan dari China dan AS. **Hasil:** Meraih 90.67% akurasi untuk AS dan 85.58% untuk kebijakan China.
Pujiono et al. (2024) [12]	Implementing Retrieval-Augmented Generation and Vector Databases for Chatbots in Public Services Agencies Context	Membandingkan model-model OpenAI (GPT-4, GPT-3.5) untuk query kebijakan finansial menggunakan Pinecone. **Hasil:** GPT-4 lebih superior dengan rata-rata *cosine similarity score* 0.404
Son et al. (2025) [13]	Development and Evaluation of a Retrieval-Augmented Generation-Based Electronic Medical Record Chatbot System	Mengintegrasikan model *fine-tuned embedding* (BGE-M3) dan Solar Mini Chat API untuk menavigasi manual EMR. **Hasil:** BGE-M3 mencapai akurasi *retrieval* sebesar 97,6% dengan latensi kueri di bawah 10 ms.
Kizi & Suh (2025) [14]	Design and performance evaluation of LLM-Based RAG pipelines for chatbot services in international student admissions	Melakukan perbandingan berbagai alur kerja (*pipeline*) RAG (MMR, *Dense*, *Hybrid*) dan LLM (GPT-4o, LLaMA3, OpenChat). **Hasil**: model *open-source* yang teroptimasi (OpenChat) mampu menyamai

Penulis dan Tahun	Judul	Metodologi dan Hasil
		performa GPT-4o, namun secara signifikan meningkatkan latensi.
Husain et al. (2025) [15]	Development of an Academic Services Chatbot Based on Retrieval-Augmented Generation (RAG)	Mengimplementasikan alur kerja (*pipeline*) RAG multi-tahap (*hierarchical chunking*, *reranking*) menggunakan Gemini 2.5 Pro. **Hasil:** kueri faktual memperoleh skor *faithfulness* sebesar 0,91, namun pada kueri penalaran (*reasoning*), *context recall* menurun menjadi 0,59
Samuel et al. (2024) [16]	AgroLLM: Connecting Farmers and Agricultural Practices through Large Language Models	Chatbot RAG untuk PDF menggunakan LangChain dan GPT/LLaMA.
Kaif et al. (2025) [17]	Gemini MultiPDF Chatbot: Multiple Document RAG Chatbot using Gemini	Membangun *chatbot* menggunakan LangChain dan Gemini untuk menganalisis beberapa dokumen PDF secara simultan. **Hasil:** integrasi RAG secara signifikan mengurangi biaya API dan meningkatkan presisi respons.
Faisal et al. (2025) [18]	Implementation of Fine-Tuning LLM Model on New Student Registration Chatbot at Muhammadiyah University of Makassar	Melakukan penalaan halus (*fine-tuning*) pada Gemini 2.5 Flash melalui Vertex AI menggunakan 1.430 pasangan tanya-jawab spesifik domain selama 37 *epoch*. **Hasil:** pencapaian akurasi sebesar 97,6% dan tingkat kepuasan pengguna sebesar 87,5%.
Hanindya et al. (2025) [19]	Smart Chatbot Design to Support PDF Document Analysis at Politeknik Indonusa Surakarta	Mengikuti model pengembangan 4D (*Define, Design, Develop, Disseminate*) untuk mengembangkan prototipe GPT-4/LangChain terkait pedoman kampus. **Hasil:** tingkat akurasi sebesar 90% dan tingkat kepuasan pengguna yang tinggi (4,5/5).

Sejumlah penelitian telah secara luas mengeksplorasi penerapan *Retrieval-Augmented Generation* (RAG) dan *Large Language Models* (LLMs) untuk meningkatkan aksesibilitas informasi di berbagai sektor, dengan penekanan signifikan pada layanan akademik dan admisi. Faisal et al. melakukan *fine-tuning* pada model Gemini 2.5 Flash untuk *chatbot* pendaftaran mahasiswa baru, dan berhasil mencapai tingkat akurasi sebesar 97,6% serta tingkat kepuasan pengguna

mencapai 87,5% [9]. Sejalan dengan hal tersebut, Kizi dan Suh meneliti konfigurasi *pipeline* RAG untuk penerimaan mahasiswa internasional, dimana hasil penelitian menunjukkan bahwa meskipun model proprieter seperti GPT-4o unggul pada awalnya, model *open-source* yang teroptimasi seperti OpenChat mampu menyamai atau bahkan melampaui performa tersebut dengan skor RAGAS rata-rata 0,7377 berbanding 0,7249 milik GPT-4o, walaupun dengan latensi yang lebih tinggi [14]. Lebih lanjut, Husain et al. mengimplementasikan *multi-stage* RAG untuk bimbingan akademik universitas, yang melaporkan skor *faithfulness* sebesar 0,91 untuk kueri faktual, namun mengidentifikasi adanya hambatan performa pada tugas penalaran di mana *context recall* menurun hingga 0,59 [15]. Melengkapi upaya tersebut, Hanindya dkk. (2025) mengembangkan *chatbot* untuk analisis dokumen PDF menggunakan model pengembangan 4D, yang menghasilkan akurasi respon sebesar 90% dan skor *usability* 4,5 dari skala 5 [19].

Dalam ranah tata kelola publik, pemahaman kebijakan, dan transparansi hukum, para peneliti berfokus pada demokratisasi informasi birokrasi yang kompleks. Yun et al. mengombinasikan RAG dengan LLM untuk menginterpretasikan kebijakan regional di Amerika Serikat dan Tiongkok, di mana sistem tersebut mencapai akurasi 90,67% untuk kebijakan AS dan 85,58% untuk kebijakan Tiongkok [11]. Dalam upaya serupa untuk meningkatkan akses publik terhadap data hukum, Yasmin dan Fudholi merancang *chatbot* untuk Ombudsman DIY yang menerima skor ahli tinggi untuk aspek akurasi (18,6) dan kejelasan (8,4) pada skala relatif [10]. Menyoroti sektor keuangan dalam layanan publik, Pujiono et al. membandingkan berbagai model OpenAI untuk menjawab kueri berbasis regulasi, dan mengidentifikasi GPT-4 sebagai model unggulan dengan skor *cosine similarity* rata-rata 0,404 [12]. Secara kolektif, studi-studi ini mengindikasikan bahwa meskipun sistem RAG sangat efektif untuk pengambilan dokumen literal, diperlukan proses pembersihan data yang lebih kuat guna menangani konteks regulasi yang terspesialisasi.

Implementasi RAG juga menunjukkan potensi transformatif dalam domain yang sangat terspesialisasi seperti pertanian dan kesehatan. Samuel et al. mengembangkan AgroLLM, sebuah *chatbot* pertanian yang menjembatani

kesenjangan antara buku teks teoretis dan pengetahuan praktis bertani [16]. Temuan mereka menunjukkan bahwa ChatGPT-4o mini melampaui model alternatif lainnya dengan akurasi 93% ketika diperkuat oleh kerangka kerja RAG. Di sektor kesehatan, Son et al. berfokus pada optimasi alur kerja administratif rumah sakit dengan mengintegrasikan panduan *Electronic Medical Record* (EMR). Melalui *fine-tuning* pada model *embedding* BGE-M3, penelitian tersebut mencapai akurasi pengambilan data yang luar biasa sebesar 97,6% dengan rata-rata latensi kueri kurang dari 10 ms [13].

Terakhir, berbagai penelitian menekankan pada efisiensi teknis dan optimasi biaya dari kerangka kerja AI tersebut. Kaif et al. menyoroti keuntungan ekonomi dari integrasi RAG dengan model Gemini, yang menunjukkan bahwa pengambilan token yang relevan saja, alih-alih menggunakan seluruh konteks dokumen, dapat mengurangi biaya per panggilan untuk Gemini 1.5 Pro dari sekitar $0,5 menjadi $0,005 [17]. Hal ini selaras dengan temuan yang lebih luas bahwa RAG tidak hanya meningkatkan relevansi kontekstual dari respon yang dihasilkan dengan mendasarkannya pada data terverifikasi, tetapi juga membuat penerapan LLM berperforma tinggi menjadi lebih berkelanjutan bagi institusi dengan sumber daya terbatas.

Berdasarkan sintesis terhadap berbagai literatur yang sudah disebutkan sebelumnya mengenai sistem *Retrieval-Augmented Generation* (RAG), ditemukan bahwa meskipun teknologi ini telah berkembang pesat, masih terdapat kendala signifikan dalam melakukan penalaran multi-tahap (*multi-hop reasoning*) dan sintesis informasi kompleks, di mana performa sistem cenderung menurun drastis ketika *query* memerlukan penggabungan data dari berbagai bagian dokumen yang terpisah. Selain kendala kognitif tersebut, terdapat hambatan teknis yang nyata dalam pemrosesan dokumen dengan tata letak visual yang rumit, seperti tabel, grafik, dan struktur multi-kolom, yang sering kali tidak mampu diekstraksi secara akurat oleh alat pemindaian standar sehingga menurunkan kualitas informasi yang diperoleh. Celah penelitian lainnya berkaitan dengan adanya dilema antara akurasi dan latensi, di mana model *open-source* sering kali memerlukan waktu respon yang lebih lama untuk menyamai performa model proprieter, yang menjadi tantangan

bagi aplikasi *real-time*. Lebih jauh lagi, mayoritas arsitektur RAG yang ada saat ini masih bersifat linear dan statis, sehingga kurang memiliki kemampuan untuk melakukan koreksi mandiri (*self-correction*) atau mengintegrasikan data dinamis secara *real-time*. Terakhir, masih terdapat kekurangan dalam studi yang melibatkan validasi longitudinal oleh pengguna manusia dalam lingkungan profesional nyata untuk mengukur dampak fungsional sistem terhadap efisiensi beban kerja administratif.

Penelitian ini hadir untuk mengisi celah-celah tersebut melalui pengembangan sistem AI *chatbot* yang diimplementasikan pada dokumen Rencana Komunitas Resmi (*Official Community Plan*) Kota Surrey. Dari sisi teknis, penelitian ini mengatasi hambatan ekstraksi struktur dokumen kompleks dengan mengintegrasikan kerangka kerja LlamaIndex dan LlamaCloud, khususnya penggunaan LlamaParse untuk mencapai akurasi ekstraksi tinggi pada dokumen kebijakan yang kaya akan elemen visual. Secara metodologis, tesis ini memberikan kontribusi dengan menerapkan pendekatan *human-in-the-loop*, di mana sistem divalidasi langsung oleh praktisi pemerintahan guna menjawab kebutuhan akan validasi fungsional dalam konteks profesional. Melalui integrasi teknologi ini, penelitian tidak hanya memberikan solusi atas keterbatasan ekstraksi data, tetapi juga menjadi fondasi bagi transisi dari sistem RAG konvensional menuju arsitektur *Agentic RAG* yang lebih cerdas, adaptif, dan mampu melakukan penalaran lintas dokumen secara komprehensif bagi kepentingan publik.

2.2 Teori yang berkaitan

2.2.1 Artificial Intelligence

Terdapat berbagai perspektif dalam mendefinisikan Kecerdasan Buatan (*Artificial Intelligence*/AI) di dalam literatur ilmiah. Dari sudut pandang komunikasi dan interaksi, AI didefinisikan sebagai kapasitas nyata dari mesin non-manusia atau entitas artifisial untuk melaksanakan tugas, menyelesaikan masalah, berkomunikasi, berinteraksi, serta bertindak secara logis sebagaimana yang dilakukan oleh manusia secara biologis [20]. Definisi ini menekankan

pada kemampuan fungsional mesin dalam meniru perilaku dan logika manusia dalam lingkup dunia nyata.

Secara konseptual, inteligensi yang ada pada manusia maupun mesin dapat dipahami sebagai kemampuan maksimal untuk mencapai tujuan-tujuan baru secara sukses melalui proses persepsi-kognitif atau komputasi. Namun, terdapat argumen dalam kajian literatur bahwa sistem yang berkembang saat ini lebih banyak menunjukkan pencapaian artifisial (*artificial achievement*) melalui efisiensi pemrosesan, alih-alih merepresentasikan kecerdasan artifisial yang sepenuhnya utuh dalam pengertian kognitif yang mendalam [21]. Di sisi lain, meskipun istilah AI telah diadopsi secara luas di berbagai sektor termasuk pendidikan tinggi, beberapa tinjauan mencatat bahwa definisi AI sering kali masih bersifat samar dan tidak konsisten di berbagai literatur ilmiah, yang menunjukkan bahwa diskursus mengenai terminologi ini masih terus berkembang seiring dengan kemajuan teknologi tersebut [22].

2.2.2 Large Language Model (LLM)

Large Language Model (LLM) dideskripsikan sebagai klasifikasi model bahasa yang memanfaatkan jaringan saraf tiruan dengan kapasitas miliaran parameter. Model ini dilatih menggunakan dataset teks tak berlabel dalam skala masif melalui metode *self-supervised learning*. Secara teknis, LLM umumnya mengadopsi arsitektur transformer yang mengintegrasikan mekanisme self-attention guna menangkap konteks informasi jarak jauh secara akurat [23].

Selain itu, kajian literatur lain mendefinisikan LLM sebagai model kecerdasan buatan tingkat lanjut yang dirancang khusus untuk memahami dan menghasilkan bahasa manusia. Keunggulan model ini terletak pada jumlah parameter dan volume data pelatihan yang sangat besar, sehingga memungkinkan pencapaian performa yang optimal dalam berbagai tugas pemahaman teks, generasi bahasa, hingga fungsi penalaran kompleks [24].

2.2.3 Natural Language Processing (NLP)

Natural Language Processing (NLP) didefinisikan sebagai pendekatan komputasi untuk menganalisis teks dengan memanfaatkan data terstruktur maupun tidak terstruktur. Sebagai cabang dari kecerdasan buatan (*Artificial Intelligence*) dan ilmu komputer, bidang ini berfokus pada pengembangan metode yang memungkinkan komunikasi bahasa alami antara manusia dan perangkat komputer [25]. Secara teoritis, kerangka kerja NLP mencakup dua komponen utama, yaitu *Natural Language Understanding* (NLU) yang merujuk pada proses pemahaman bahasa oleh mesin terhadap input manusia (manusia ke mesin), serta *Natural Language Generation* (NLG) yang berkaitan dengan kemampuan mesin dalam memproduksi bahasa alami untuk berinteraksi dengan manusia (mesin ke manusia) [25].

2.2.4 Official Community Plan (OCP)

Official Community Plan (OCP) didefinisikan sebagai kerangka kerja panduan tata guna lahan dan kebijakan pada tingkat lokal yang diadopsi sebagai peraturan daerah (*bylaw*) [3]. Instrumen ini berfungsi sebagai dasar hukum yang mengarahkan pengambilan keputusan pemerintah kota dalam berbagai sektor strategis, termasuk tata guna lahan, transportasi, energi, perumahan, serta kebijakan terkait lainnya [3].

Di wilayah British Columbia, regulasi tingkat provinsi (Bill 27) mewajibkan setiap pemerintah kota untuk mengintegrasikan target pengurangan emisi Gas Rumah Kaca (*greenhouse gas*), kebijakan lingkungan, dan aksi nyata ke dalam dokumen OCP mereka. Hal ini memosisikan OCP sebagai instrumen sentral dalam upaya mitigasi dan adaptasi perubahan iklim di tingkat lokal. Kekuatan utama OCP terletak pada cakupannya yang luas dan integratif, yang meliputi berbagai dimensi kemasyarakatan seperti tata guna lahan, sistem transportasi, ekosistem sensitif, ruang terbuka hijau (taman), infrastruktur pekerjaan umum, hingga kebijakan sosial. Pendekatan terintegrasi ini dipandang sebagai elemen

krusial dalam menanggulangi isu-isu kontemporer yang kompleks, khususnya perubahan iklim [3].

2.3 Framework/Algoritma yang digunakan

2.3.1 Design Science Research Methodology (DSRM)

Design Science Research Methodology (DSRM) dapat didefinisikan sebagai sebuah sistem yang mencakup prinsip, praktik, dan prosedur yang diterapkan untuk melaksanakan penelitian *design science* dalam disiplin sistem informasi. Metodologi ini dikembangkan untuk memenuhi tiga tujuan utama: konsistensi dengan literatur penelitian terdahulu, penyediaan model proses nominal untuk melakukan penelitian *design science*, serta penyediaan model mental untuk mempresentasikan dan mengevaluasi hasil penelitian tersebut. DSRM berfungsi sebagai kerangka kerja yang diterima secara umum guna memastikan bahwa artefak yang diciptakan melalui proses penelitian memiliki rigoritas ilmiah dan kegunaan praktis yang tinggi [27].

Struktur alur metodologi DSRM terdiri dari enam tahapan utama sebagai berikut:

1) Identifikasi Masalah dan Motivasi (*Problem Identification and Motivation*): Tahap ini melibatkan pendefinisian masalah penelitian secara spesifik serta pemberian justifikasi atas nilai atau urgensi dari solusi yang diusulkan.
2) Definisi Tujuan untuk Solusi (*Define the Objectives for a Solution*): Tahap ini bertujuan untuk menyimpulkan tujuan-tujuan yang ingin dicapai oleh solusi berdasarkan definisi masalah dan pengetahuan mengenai apa yang mungkin serta layak secara teknis untuk dilakukan.
3) Desain dan Pengembangan (*Design and Development*): Tahap inti ini berfokus pada penciptaan artefak, yang dapat berupa konstruk, model, metode, atau instansiasi, serta penentuan fungsionalitas dan arsitektur artefak tersebut.

4) Demonstrasi (*Demonstration*): Tahap ini menunjukkan efektivitas penggunaan artefak dalam memecahkan satu atau lebih instansi masalah, baik melalui eksperimen, simulasi, studi kasus, maupun aktivitas pembuktian lainnya.
5) Evaluasi (*Evaluation*): Tahap ini melibatkan observasi dan pengukuran mengenai seberapa baik artefak tersebut mendukung solusi bagi masalah yang telah diidentifikasi dengan membandingkan tujuan solusi terhadap hasil observasi nyata.
6) Komunikasi (*Communication*): Tahap akhir ini difokuskan pada penyebaran informasi mengenai masalah, kepentingan artefak, kebaruan, dan efektivitas desain kepada peneliti lain maupun praktisi profesional melalui publikasi ilmiah atau saluran komunikasi lainnya [27].

2.3.2 Waterfall

Metode *Waterfall* merupakan salah satu pendekatan dalam *Software Development Life Cycle* (SDLC) yang bersifat linier dan sekuensial. Kerangka kerja ini membagi proses pengembangan perangkat lunak ke dalam beberapa tahapan fungsional yang baku dan terdefinisi secara jelas. Karakteristik utama dari metode ini menetapkan bahwa transisi ke tahapan berikutnya hanya dapat dilakukan setelah seluruh aktivitas pada tahapan sebelumnya telah diselesaikan secara menyeluruh dan mendapatkan validasi.

Secara terstruktur, tahapan di dalam metode *Waterfall* meliputi:

1) Analisis Kebutuhan (*Requirements Analysis*): Tahap awal yang bertujuan untuk mengidentifikasi, mengumpulkan, dan mendokumentasikan secara menyeluruh seluruh spesifikasi dan kebutuhan sistem yang akan dibangun.
2) Desain Sistem (*System Design*): Proses penerjemahan spesifikasi kebutuhan ke dalam cetak biru arsitektur perangkat lunak, termasuk

perancangan struktur data, arsitektur sistem, dan representasi antarmuka.

3) Implementasi (*Implementation*): Tahap konstruksi di mana rancangan desain yang telah dibuat mulai ditransformasikan ke dalam bentuk baris kode program atau pembuatan modul-modul perangkat lunak.
4) Pengujian (*Verification/Testing*): Proses evaluasi dan validasi terhadap integrasi modul sistem untuk memastikan bahwa fungsionalitas perangkat lunak telah memenuhi kriteria spesifikasi yang ditetapkan sebelumnya.
5) Penerapan (*Deployment*): Tahap instalasi dan pengoperasian sistem yang telah dinyatakan lolos uji ke dalam lingkungan produksi nyata agar dapat diakses oleh pengguna akhir.
6) Pemeliharaan (*Maintenance*): Fase pasca-penerapan yang mencakup perbaikan kesalahan (*bug fixing*), optimasi performa sistem, serta penyesuaian perangkat lunak terhadap perubahan lingkungan operasional.

Pendekatan *Waterfall* sangat ideal diimplementasikan pada proyek pengembangan yang memiliki spesifikasi kebutuhan yang jelas, matang, dan cenderung tidak mengalami perubahan di tengah jalan. Kendati menawarkan keunggulan dalam hal disiplin dokumentasi dan manajemen proyek yang terstruktur, metodologi ini memiliki tingkat fleksibilitas yang lebih rendah dalam mengakomodasi perubahan kebutuhan dibandingkan dengan pendekatan yang bersifat iteratif [32].

2.3.3 Agile

Dalam konteks *Software Development Life Cycle* (SDLC), *Agile* didefinisikan sebagai pendekatan pengembangan perangkat lunak yang adaptif dan berkualitas tinggi melalui pemanfaatan tim berskala kecil, penyempurnaan desain secara berkesinambungan, serta pengujian berkala yang didasarkan pada umpan balik dan respons cepat terhadap perubahan. Berbeda dengan metodologi tradisional yang bersifat linear dan kaku, *Agile* menekankan pada

siklus proyek yang iteratif dan evolusioner, gaya kepemimpinan yang kolaboratif, serta pengelolaan pengetahuan berbasis interaksi langsung (*tacit knowledge*).

Secara fundamental, operasionalisasi *Agile* dalam SDLC berakar pada empat nilai inti *Agile Manifesto*, yaitu:

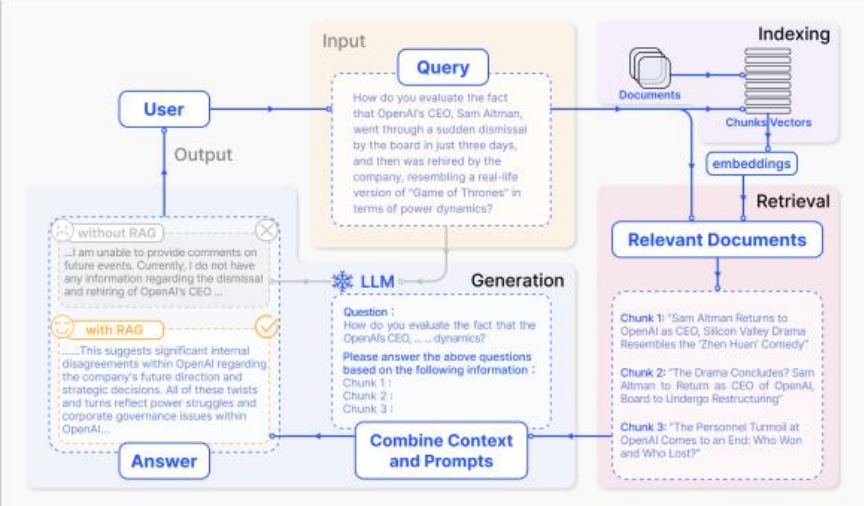
1) Fokus pada Individu dan Interaksi: Mengutamakan komunikasi antar-anggota tim di atas ketergantungan pada proses dan alat kerja (*tools*).
2) Orientasi pada Produk Berfungsi: Memprioritaskan pengiriman perangkat lunak yang dapat berjalan dengan baik dibandingkan penyusunan dokumentasi yang terlalu komprehensif.
3) Kolaborasi dengan Pelanggan: Menempatkan pengguna atau klien sebagai mitra aktif yang terlibat langsung sepanjang siklus pengembangan, bukan sekadar objek negosiasi kontrak.
4) Responsif terhadap Perubahan: Menekankan fleksibilitas untuk beradaptasi dengan dinamika kebutuhan proyek daripada mematuhi rencana awal secara kaku.

Lebih lanjut, agilitas dalam pengembangan perangkat lunak juga direpresentasikan sebagai kemampuan struktural untuk melakukan penyesuaian objektif skala kecil secara tangkas tanpa menimbulkan dampak disruptif yang signifikan terhadap anggaran maupun jadwal proyek. Melalui karakteristik tim lintas fungsi yang mandiri (*self-organizing*) dan adaptif, metode *Agile* mengintegrasikan mekanisme deteksi dan respons (*sense-and-respond*) guna mencapai keberhasilan implementasi sistem di lingkungan yang dinamis [33].

2.3.4 Retrieval Augmented Generation (RAG)

Retrieval-Augmented Generation (RAG) didefinisikan sebagai sebuah paradigma atau kerangka kerja dalam pengembangan kecerdasan buatan yang mengintegrasikan kemampuan generatif model bahasa dengan mekanisme pengambilan informasi dari sumber pengetahuan eksternal. Dalam metodologi ini, sistem tidak langsung menghasilkan jawaban secara mandiri, melainkan

terlebih dahulu melakukan proses pencarian informasi yang relevan dari repositori data luar, seperti koleksi dokumen digital atau basis data terstruktur. Informasi yang berhasil ditarik kemudian diproses sebagai konteks tambahan untuk memperkaya (*augment*) input bagi model bahasa besar (*Large Language Model*), sehingga proses sintesis jawaban didasarkan pada referensi yang nyata dan spesifik [26].



Gambar 2.1 Gambaran skema RAG [26]

Sebagaimana diilustrasikan pada Gambar 2.1, proses ini dimulai dengan tahap ingesti data di mana dokumen mentah dalam format PDF dikonversi melalui mesin *parsing* menjadi format teks terstruktur seperti *Markdown* untuk mempertahankan integritas konten. Teks tersebut kemudian diproses oleh *embedding model* menjadi representasi vektor numerik yang menangkap makna semantik data, lalu disimpan di dalam *vector database* sebagai basis pengetahuan. Saat pengguna memberikan input kueri, sistem akan melakukan pencarian pada basis data vektor untuk mengambil potongan konteks yang paling relevan (*retrieval*). Tahap akhir melibatkan penggabungan kueri pengguna dengan konteks yang berhasil diambil untuk diproses oleh *Large Language Model* (LLM), yang kemudian menyintesis informasi tersebut menjadi sebuah jawaban akhir yang faktual dan relevan terhadap *query* awal.

2.3.5 Vector Database

Basis data vektor (*Vector Database*) merupakan sistem yang dirancang secara khusus untuk melakukan penyimpanan dan pengelolaan vektor berdimensi tinggi secara efisien. Vektor tersebut bertindak sebagai representasi matematis dari fitur-fitur data yang digunakan sebagai parameter utama dalam melakukan analisis komparasi kemiripan (*similarity comparison*) antar objek data [28].

2.3.6 Approximate Nearest Neighbour (ANN)

Pencarian tetangga terdekat aproksimasi (*Approximate Nearest Neighbor Search* atau ANN) merupakan metode yang umum digunakan dalam proses *information retrieval*, khususnya dalam konteks pemanfaatan *Large Language Models* (LLM) dan arsitektur *Retrieval-Augmented Generation* (RAG). Salah satu algoritma ANN yang paling luas diimplementasikan didasarkan pada pembangunan graf multi-lapis di atas kumpulan data, yang dikenal sebagai *Hierarchical Navigable Small World* (HNSW) [29].

2.4 Tools/software yang digunakan

2.4.1 Python

Python merupakan bahasa pemrograman tingkat tinggi yang bersifat umum (*general-purpose*) dan berorientasi objek (*object-oriented*), dikembangkan oleh Guido van Rossum. Bahasa ini dikenal karena sintaksnya yang sederhana, mudah dibaca, serta fleksibilitas penggunaannya, sehingga cocok digunakan baik oleh pemula maupun profesional. *Python* banyak diterapkan dalam berbagai bidang, termasuk *data science*, pengembangan web, kecerdasan buatan (*artificial intelligence*), otomasi, serta pendidikan. Selain itu, ekosistem pustaka (*library*) yang kaya dan kemampuan lintas platform menjadikan *Python* salah satu bahasa pemrograman paling populer dan berkembang pesat dalam industri perangkat lunak modern [30].

2.4.2 Streamlit

Streamlit merupakan sebuah pustaka sumber terbuka (*open-source library*) berbasis bahasa pemrograman Python yang dirancang untuk memfasilitasi pembangunan aplikasi berbasis web secara efisien tanpa memerlukan keahlian khusus di bidang pemrograman antarmuka (*front-end coding*). Pustaka ini telah banyak diimplementasikan dalam pengembangan sistem informasi dan riset ilmiah untuk menciptakan aplikasi web interaktif yang menyederhanakan eksplorasi kumpulan data berskala besar secara intuitif. Guna mendukung penyajian informasi yang komprehensif, Streamlit menyediakan fitur modular yang memudahkan integrasi visualisasi data grafik dasar, seperti diagram batang (*bar chart*) dan diagram pencar (*scatter plot*), maupun representasi data dalam format tabular seperti *data frame* dan tabel terstruktur [31].