BAB I
PENDAHULUAN

1.1 Latar Belakang

Manajemen Sumber Daya Manusia (MSDM) memegang peranan strategis dalam menciptakan keunggulan kompetitif organisasi yang berkelanjutan (Dessler, 2017). Dalam ekosistem MSDM, rekrutmen dan seleksi berfungsi sebagai gerbang fundamental untuk mengakuisisi talenta yang tidak hanya memenuhi tuntutan kompetensi teknis (Person-Job Fit), tetapi juga memiliki keselarasan nilai dengan budaya perusahaan (Person-Organization Fit) (Kristof-brown et al., 2005). Kegagalan pada tahap ini dapat berdampak fatal, mulai dari penurunan produktivitas tim hingga eskalasi biaya turnover yang merugikan perusahaan (Armstrong, 2005). Oleh karena itu, peningkatan akurasi dan validitas prediktif dalam menilai kinerja kandidat menjadi prioritas utama bagi praktisi HR demi memitigasi risiko *bad hiring* (Schmidt & Hunter, 1998).

Metode wawancara tetap menjadi instrumen yang paling dominan digunakan secara global dalam upaya menyeleksi kandidat terbaik. Data terbaru dari *selectsoftwarereviews* yang dibuat oleh Nguyen (2025) mengindikasikan bahwa 72% perusahaan di Amerika Serikat kini mengadopsi wawancara terstruktur sebagai metode seleksi utama. Tren ini didukung oleh konsensus akademis yang menempatkan wawancara terstruktur sebagai "standar emas" dalam seleksi personel. Sackett et al. (2023), dalam sebuah meta-analisis komprehensif, menegaskan bahwa wawancara terstruktur memiliki validitas prediktif rata-rata tertinggi dibandingkan metode seleksi lainnya, mengungguli tes kemampuan kognitif maupun referensi kerja. Keunggulan ini terletak pada standarisasi pertanyaan dan kriteria penilaian yang ketat, yang dirancang untuk meminimalkan variabilitas yang tidak relevan. Huffcutt dan Murphy (2023) menambahkan bahwa selain korelasi yang tinggi dengan kinerja pekerjaan, wawancara terstruktur menawarkan stabilitas skor yang lebih baik dan kejelasan kriteria, yang secara teoritis menjamin keadilan bagi seluruh pelamar.

Wawancara terstruktur dalam realisasinya di lapangan sering berjalan tidak seideal dengan teori yang ada. Di tengah tuntutan bisnis yang mengharuskan proses seleksi berjalan cepat dan efisien, wawancara manual menghadapi kendala signifikan berupa keterbatasan kognitif manusia (*human cognitive limitations*). Kendala utama yang sering muncul adalah subjektivitas dan bias penilai. Thomas dan Reimann (2023) menyoroti fenomena psikologis di mana praktisi *HR* sering kali terjebak dalam "ilusi objektivitas"; mereka cenderung merasa penilaian naluriah mereka sudah adil, sehingga meremehkan kebutuhan akan alat bantu koreksi seperti rubrik terstandar. Akibatnya, banyak penilai enggan mematuhi panduan struktural secara ketat. Hal ini diperburuk oleh temuan Bergelson et al. (2022), yang menyatakan bahwa tanpa kepatuhan ketat terhadap rubrik penilaian dan pelatihan anti-bias yang berkelanjutan, organisasi akan terus menghasilkan keputusan

rekrutmen yang inkonsisten, yang tidak hanya merugikan secara operasional tetapi juga berisiko secara hukum dan etika.

Masalah subjektivitas dalam wawancara pada proses rekrutmen semakin kompleks ketika dihadapkan pada volume pelamar yang tinggi, seperti yang terjadi pada program rekrutmen massal. Radbruch et al. (2021) dalam penelitian terbarunya mengenai *Interview Sequences* mengungkapkan adanya *sequential contrast effect*, di mana evaluasi seorang kandidat sangat dipengaruhi oleh kualitas kandidat sebelumnya. Efek ini menguat ketika wawancara dilakukan dengan frekuensi tinggi dalam jarak waktu yang pendek. Data menunjukkan bahwa peluang rekomendasi positif bagi seorang kandidat menurun secara signifikan jika evaluator baru saja memberikan penilaian positif kepada kandidat sebelumnya. Fenomena kelelahan penilai (*interviewer fatigue*) ini menegaskan bahwa objektivitas manusia sangat rentan terdistorsi oleh faktor urutan dan beban kerja, menjadikan proses manual sulit diandalkan untuk skala besar.

Tantangan efisiensi dan objektivitas yang sering menjadi kendala dalam proses wawancara mendorong integrasi teknologi kecerdasan buatan (*Artificial Intelligence/AI*) sebagai solusi alternatif. Transformasi ini bukan sekadar mengganti media pencatatan, melainkan merekayasa ulang alur kerja penilaian itu sendiri. Irawan (2024) dalam studinya mengenai operasional rekrutmen membedah struktur wawancara kerja menjadi empat tahapan krusial: (1) pembukaan dan pembangunan hubungan (*building rapport*), (2) eksplorasi latar belakang dan kompetensi, (3) klarifikasi dan pendalaman jawaban, serta (4) penutupan. Dalam metode konvensional, keempat tahapan ini membebankan beban kognitif penuh kepada pewawancara manusia. Namun, dengan implementasi teknologi pemrosesan bahasa alami, beban kerja pada tahap "eksplorasi latar belakang dan kompetensi" serta "klarifikasi" dapat dipangkas secara signifikan. Sistem AI mampu menganalisis transkrip jawaban kandidat secara instan untuk mengekstrak indikator kompetensi, membandingkannya dengan kamus kompetensi, dan memberikan penilaian awal. Hal ini memungkinkan pewawancara manusia untuk memfokuskan energi mereka hanya pada aspek intuitif yang sulit digantikan, seperti penilaian cultural fit dan pembangunan rapport, sementara tugas analitis yang berulang dan melelahkan didelegasikan kepada sistem cerdas.

Teknologi utama yang mendasari kemampuan AI untuk melakukan analisis dan penilaian terhadap transkrip jawaban adalah *Large Language Models* (LLM). LLM dibangun di atas arsitektur Transformer yang diperkenalkan oleh Vaswani et al. (2017), yang menggunakan mekanisme *self-attention* untuk memproses urutan teks dalam jumlah masif. Berbeda dengan model NLP generasi sebelumnya, LLM memiliki kemampuan *emergent properties* yang memungkinkannya memahami konteks linguistik yang kompleks, nuansa semantik, hingga penalaran logis (Wei, Tay, et al., 2022). Dalam survei komprehensif oleh Zhao et al. (2023), disebutkan bahwa LLM modern seperti seri GPT atau Llama telah mencapai performa yang mendekati atau bahkan melampaui manusia dalam berbagai tugas pemahaman teks (*natural language understanding*), menjadikannya kandidat ideal untuk mesin penilai otomatis (*automated scoring engine*). Namun demikian, walau memiliki kapabilitas

linguistik yang superior, pemanfaatan LLM secara langsung (*metode direct prompting*) untuk penilaian wawancara memiliki risiko fundamental. LLM bekerja berdasarkan probabilitas statistik kata, bukan pemahaman fakta yang deterministik. Hal ini memicu fenomena "halusinasi" (*hallucination*). Ji et al. (2023) mendefinisikan halusinasi pada LLM sebagai kecenderungan model untuk menghasilkan teks yang fasih dan terdengar logis, namun secara faktual salah atau tidak berlandaskan pada sumber yang diberikan (*source-reference divergence*). Dalam konteks penilaian wawancara, halusinasi ini sangat berbahaya; model bisa saja memberikan skor sempurna pada jawaban kandidat yang kosong hanya karena kandidat menggunakan kata-kata kunci yang umum, atau model mengarang rubrik penilaian sendiri yang tidak sesuai standar perusahaan. Tanpa mitigasi, halusinasi ini merusak validitas instrumen seleksi.

Tantangan halusinasi dan keterbatasan memori parametrik yang statis dari LLM dapat dijawab dengan metode *Retrieval-Augmented Generation* (RAG) diperkenalkan oleh Lewis et al. (2020). Secara teknis, RAG menggabungkan generator parametrik (LLM) dengan memori non-parametrik (indeks vektor dokumen). Dalam skenario penilaian wawancara, ketika sistem diminta menilai jawaban kandidat, RAG terlebih dahulu melakukan pencarian semantik (*semantic search*) ke dalam basis data yang berisi panduan wawancara dan kunci jawaban resmi. Dokumen yang relevan kemudian disisipkan ke dalam prompt sebagai konteks. Pendekatan ini terbukti secara empiris meningkatkan faktualitas jawaban model. Namun, RAG standar (atau sering disebut *Naive* RAG) memiliki kelemahan kritis: sistem ini bekerja secara buta (*blind retrieval*). Jika modul pencarian mengambil dokumen yang salah (*retrieval error*), LLM akan tetap memprosesnya, menghasilkan penilaian yang bias sampah-masuk-sampah-keluar (*garbage in, garbage out*).

Tantangan halusinasi pada LLM sebenarnya telah coba diatasi melalui berbagai pendekatan. Studi oleh Mizumoto dan Eguchi (2023) menunjukkan bahwa metode Direct Prompting pada ChatGPT mampu menilai esai, namun rentan inkonsistensi tanpa konteks eksternal. Pendekatan selanjutnya, seperti yang dilakukan oleh Qiu et al. (2025), mengintegrasikan Retrieval-Augmented Generation (RAG) untuk menyisipkan referensi penilaian. Meskipun terbukti meningkatkan faktualitas, pendekatan Qiu tersebut masih terbatas pada arsitektur Standard RAG (Naive RAG) yang memiliki kelemahan kritis berupa blind retrieval; jika sistem mengambil dokumen referensi yang salah, penilaian akhir akan tetap bias.

Penyempurnaan terhadap kelemahan arsitektur RAG konvensional kini ditawarkan melalui pendekatan mutakhir bernama *Self-Reflective Retrieval-Augmented Generation* (*Self*-RAG). Asai et al. (2023) dalam penelitiannya memaparkan bahwa *Self*-RAG merombak paradigma sistem dari sekadar "mengambil dan menjawab" menjadi siklus cerdas "mengambil, mengkritik, dan menjawab". Model dalam arsitektur ini dilatih secara khusus untuk menghasilkan token refleksi (*reflection tokens*) selama proses inferensi, yang berfungsi memverifikasi relevansi dokumen dan dukungan fakta secara mandiri, sehingga risiko halusinasi dapat diminimalisir secara signifikan.

Mengisi celah halusinasi dalam membuat output penilaian berbasis AI, penelitian ini mengusulkan pendekatan hibrida dengan mengadaptasi mekanisme Self-Correction menggunakan teknik Iterative Prompting—sebagaimana divalidasi oleh Madaan et al. (2023)dan Shinn et al. (2023). Strategi ini memungkinkan sistem untuk memvalidasi relevansi rubrik dan konsistensi skor secara mandiri pada Local LLM, menawarkan solusi yang tidak hanya akurat dan transparan, tetapi juga efisien dari segi biaya dan privasi data. Sinergi antara kemampuan linguistik *Large Language Models* (LLM) dengan arsitektur hibrida RAG dan Self-RAG menawarkan jawaban komprehensif atas masalah objektivitas, bias kognitif, dan inkonsistensi penilaian wawancara. RAG berperan vital dalam memperluas basis pengetahuan (*knowledge boundaries*) LLM dengan menyuplai konteks spesifik berupa standar penilaian perusahaan. Selanjutnya, lapisan *Self*-RAG bertindak sebagai mekanisme kontrol kualitas yang memastikan output penilaian selaras dengan standar tersebut dan memberikan evaluasi yang tepat sasaran. Berlandaskan pada urgensi efisiensi proses seleksi dan kebutuhan akan akurasi penilaian yang tinggi, penelitian ini disusun dengan judul: "Implementasi *Large Language Model* dengan Pendekatan *Self-RAG* pada Dashboard Sistem Penilaian Wawancara (Studi Kasus: Seleksi Wawancara Local Project AIESEC in UNHAS)"

1.2 Rumusan Masalah

Rumusan masalah dari penelitian ini, yaitu :

1. Bagaimana merancang dan membangun sistem penilaian wawancara menggunakan *Large Language Model* (LLM) dengan paradigma *Self*-RAG pada studi kasus Local Project AIESEC in UNHAS?
2. Bagaimana performa sistem penilaian otomatis yang dikembangkan dalam memberikan skor dan umpan balik dalam proses penilaian wawancara?

1.3 Batasan Penelitian

Batasan dari penelitian ini, yaitu :

1. Studi kasus dilakukan pada proses rekrutmen relawan Local Project AIESEC in UNHAS. Data yang digunakan terbatas pada rubrik penilaian dan contoh jawaban wawancara dari proyek tersebut.
2. Sistem hanya dilatih dan diuji untuk menilai pertanyaan terkait Analisis Personal (*Personal Analysis*) dan Kemampuan Kepemimpinan (*Leadership Capability*), tidak mencakup pertanyaan teknis spesifik lainnya.
3. Sistem berbasis teks (*text-based*). Jawaban wawancara berupa audio harus ditranskrip terlebih dahulu menjadi teks sebelum diproses oleh sistem (tidak memproses audio secara real-time).
4. Penilaian dilakukan per satu pasang pertanyaan dan jawaban (*single-turn QA evaluation*), bukan penilaian holistik dari keseluruhan sesi wawancara secara sekaligus.

5. Keluaran sistem terbatas pada Skor Kuantitatif, Alasan Penilaian (*Reasoning*), dan Saran Perbaikan (*Improvement*) berdasarkan *knowledge base* rubrik yang disediakan.
6. Penelitian berfokus pada implementasi teknik RAG dan Self-RAG menggunakan *Open Source* LLM (Qwen2.5-7B) dan dijalankan pada lingkungan lokal/terbatas (*prototype*), bukan aplikasi skala produksi (*production-ready*).

1.4 Tujuan Penelitian

Tujuan dari dilaksanakannya penelitian ini adalah :
1. Merancang sistem penilaian wawancara berbasis LLM dengan pendekatan *Self*-RAG untuk mengevaluasi jawaban wawancara pada Local Project AIESEC in UNHAS.
2. Mengukur performa penilaian yang dihasilkan oleh sistem penilaian wawancara menggunakan LLM dan paradigma *Self*-RAG.

1.5 Manfaat Penelitian

Manfaat dari dilakukannya penelitian ini adalah :
1. Memberikan kontribusi pada pengembangan ilmu *Natural Language Processing* (NLP), khususnya dalam penerapan metode *Self-Reflective* RAG (Self-RAG) untuk kasus penggunaan penilaian otomatis (*automated grading*) dalam Bahasa Indonesia.
2. Memberikan alternatif alat bantu yang dapat mempercepat proses penyaringan awal (*screening*) relawan dan menjaga standar objektivitas penilaian di setiap rekrutmen.
3. Sebagai referensi dalam mengadopsi teknologi AI untuk meminimalisir subjektivitas dan bias kognitif manusia (seperti *halo effect*) dalam proses wawancara.

1.6 Landasan Teori

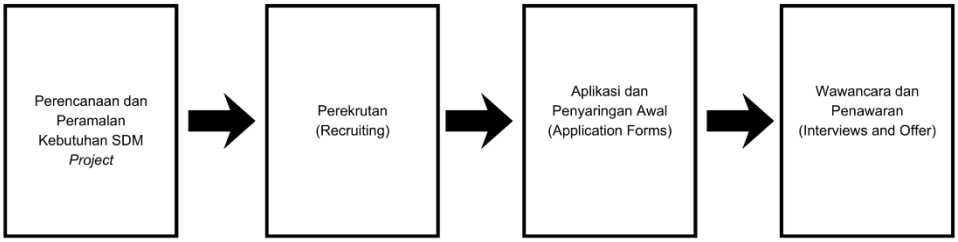
1.6.1 Manajemen Sumber Daya Manusia & Rekrutmen

Manajemen Sumber Daya Manusia (MSDM) merupakan bagian krusial dalam organisasi yang berfokus pada kebijakan dan praktik pengelolaan tenaga kerja. Menurut Dessler (2017), MSDM didefinisikan sebagai proses memperoleh, melatih, menilai, dan memberikan kompensasi kepada karyawan, serta memperhatikan hubungan kerja, kesehatan, keselamatan, dan keadilan. Dalam konteks organisasi nirlaba seperti AIESEC, manajemen relawan memegang prinsip serupa untuk memastikan keberlanjutan proyek sosial.

1.6.1.1 Rekrutmen dan Seleksi

Rekrutmen adalah proses menemukan dan menarik pelamar yang mampu untuk bekerja. Proses ini dimulai ketika pelamar dicari dan berakhir ketika lamaran mereka

diserahkan (Robbins & Coulter, 2017). Tahap selanjutnya adalah seleksi, yaitu proses menyaring pelamar kerja untuk memastikan bahwa kandidat yang paling tepatlah yang diterima. Kesalahan dalam tahap seleksi dapat berdampak pada kinerja organisasi dan biaya yang terbuang. Dalam konteks penelitian ini, tahapan tersebut diadaptasi ke dalam alur penerimaan relawan pada *Local Project AIESEC*. Alur proses rekrutmen dan seleksi yang digunakan dapat dilihat secara rinci pada **Gambar 1** berikut.



Gambar 1. Alur Proses Rekrutmen dan Seleksi *Local Project AIESEC* Sumber: Diolah Penulis dari Dessler (2017)

1.6.1.2 Wawancara sebagai Alat Seleksi

Wawancara merupakan metode seleksi yang paling umum digunakan. Menurut Mondy & Martocchio (2016), wawancara adalah percakapan yang berorientasi pada tujuan di mana pewawancara dan pelamar bertukar informasi. Terdapat beberapa jenis wawancara, namun penelitian ini berfokus pada Wawancara Terstruktur Berbasis Kompetensi (*Structured Competency-Based Interview*). Dalam metode ini, semua kandidat ditanyakan serangkaian pertanyaan standar yang sama, yang dinilai berdasarkan kriteria yang telah ditetapkan sebelumnya untuk meningkatkan reliabilitas.

1.6.1.3 Bias Kognitif dalam Wawancara

Wawancara cukup popular, namun rentan terhadap subjektivitas pewawancara. Beberapa bias psikologis yang sering terjadi menurut riset Highhouse (2008) antara lain:

- ***Halo Effect***
  Menilai kandidat secara keseluruhan positif hanya karena satu karakteristik yang menonjol.
- ***Confirmation Bias***
  Pewawancara cenderung mencari jawaban yang mendukung kesan pertama mereka terhadap kandidat.
- ***Inconsistency***
  Pewawancara yang berbeda dapat memberikan penilaian yang jauh berbeda untuk jawaban yang sama karena tidak adanya standar penilaian yang baku.

Permasalahan subjektivitas inilah yang mendasari perlunya intervensi teknologi berupa sistem penilaian otomatis untuk menjaga objektivitas dan

konsistensi. Untuk memperjelas urgensi pengembangan sistem ini, perbedaan mendasar antara penilaian wawancara secara manual dengan sistem berbasis AI dirangkum pada **Tabel 1** berikut.

Tabel 1. Perbandingan Metode Penilaian Wawancara Manual dan Berbasis AI

Aspek Pembanding	Penilaian Wawancara	Penilaian Berbasis AI (Sistem Usulan)
Objektifitas	Rentan terhadap bias kognitif (*Halo Effect, Confirmation Bias*) dan preferensi pribadi pewawancara.	Konsisten dan objektif karena penilaian didasarkan sepenuhnya pada rubrik yang telah diprogram tanpa dipengaruhi emosi.
Konsistensi	Standar penilaian dapat berubah-ubah (*inconsistent*) antar kandidat atau antar pewawancara yang berbeda.	Menerapkan standar penilaian yang identik (*standardized*) untuk setiap jawaban kandidat, kapanpun penilaian dilakukan.
Efisiensi Waktu	Membutuhkan waktu lama untuk merekap catatan, berdiskusi, dan menentukan skor akhir secara manual.	Penilaian, skor, dan umpan balik dihasilkan secara instan (*near real-time*) segera setelah jawaban diinput.
Umpan Balik (*Feedback*)	Kandidat seringkali tidak mendapatkan *feedback* detail karena keterbatasan waktu tim rekrutmen.	Mampu menghasilkan alasan penilaian (*reasoning*) dan saran perbaikan (*improvement*) secara otomatis dan rinci.
Kapasitas Pemrosesan	Terbatas pada stamina dan fokus manusia; rentan mengalami kelelahan (*fatigue*) jika menilai banyak kandidat.	Dapat memproses data dalam jumlah besar (*scalable*) secara terus-menerus tanpa penurunan kualitas akibat kelelahan.

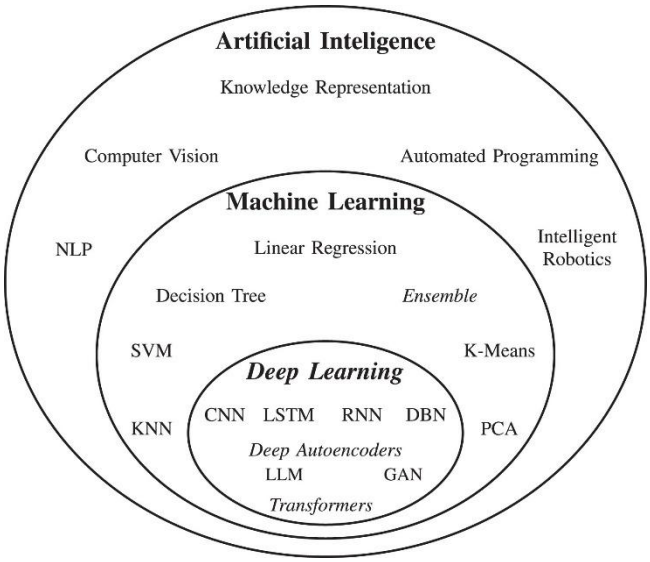
Sumber : Diolah Penulis dari Highhouse (2008) dan Chamorro-Premuzic et al. (2016)

1.6.2 *Artificial Intelligence* dan *Natural Language Processing*

Perkembangan teknologi informasi yang pesat dalam dekade terakhir telah membawa dampak signifikan pada berbagai sektor industri, termasuk dalam tata kelola sumber daya manusia. Pemanfaatan teknologi komputasi kini tidak lagi sebatas pada otomatisasi administrasi, melainkan telah merambah pada pengambilan keputusan strategis. Salah satu terobosan teknologi yang paling berpengaruh adalah Kecerdasan Buatan atau *Artificial Intelligence* (AI). Penerapan AI, khususnya dalam pemrosesan bahasa alami, menawarkan solusi untuk meningkatkan efisiensi dan objektivitas yang sulit dicapai oleh metode konvensional.

1.6.2.1 *Artificial Intelligence* (AI)

Artificial Intelligence atau Kecerdasan Buatan adalah cabang ilmu komputer yang berfokus pada pengembangan sistem yang mampu melakukan tugas-tugas yang biasanya memerlukan kecerdasan manusia. Dapat dilihat pada **Gambar 2** mengenai hierarki *Artificial Intelligence* dan bidang bidang yang ada di bawahnya. Russell & Norvig (2020) dalam buku standard mereka, "*Artificial Intelligence: A Modern Approach*", mendefinisikan AI sebagai studi tentang agen yang menerima persepsi dari lingkungan dan melakukan tindakan. Dalam konteks rekrutmen, AI digunakan bukan untuk menggantikan manusia sepenuhnya, melainkan sebagai *Decision Support System* (Sistem Pendukung Keputusan) untuk mengolah data kandidat secara lebih efisien.



Gambar 2. Hierarki Artificial Intelligence, Machine Learning, dan Deep Learning
Sumber: Digambar ulang dari Farias Junior et al. (2025)

1.6.2.2 *Natural Language Processing* (NLP)

Natural Language Processing (NLP) adalah bidang irisan antara ilmu komputer, kecerdasan buatan, dan linguistik komputasi. Jurafsky & Martin (2024) mendefinisikan NLP sebagai teori dan teknik komputasi yang memungkinkan komputer untuk memproses, memahami, dan menghasilkan bahasa manusia. Tugas utama NLP dalam konteks penelitian ini meliputi:

- ***Text Understanding***
  Kemampuan mesin memahami konteks dan makna semantik dari jawaban kandidat.
- ***Information Retrieval***
  Menemukan informasi relevan (rubrik penilaian) untuk mencocokkan dengan jawaban.
- ***Text Generation***

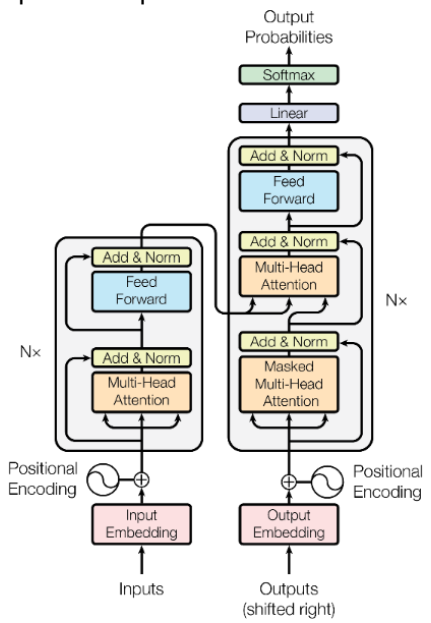
Menghasilkan teks baru yang koheren, yang digunakan sistem untuk memberikan alasan penilaian (*reasoning*) dan saran (*improvement*).

1.6.3 *Large Language Model* (LLM)

Perkembangan teknologi *Natural Language Processing* (NLP) mengalami lonjakan signifikan dengan hadirnya *Large Language Model* (LLM). Model bahasa ini dirancang untuk memahami, menghasilkan, dan memanipulasi bahasa manusia dengan cara memprediksi probabilitas kata selanjutnya dalam sebuah urutan teks. Zhao et al. (2023) menjelaskan bahwa LLM berbeda dari model bahasa tradisional karena skala pelatihannya yang masif dan kemampuannya untuk menyelesaikan berbagai tugas (*general-purpose*) tanpa perlu pelatihan ulang yang spesifik. Dalam penelitian ini, LLM bertindak sebagai mesin inti yang memproses logika penilaian jawaban wawancara.

1.6.3.1. Arsitektur Transformer

Kemampuan LLM dalam menangani konteks teks yang panjang tidak lepas dari arsitektur dasar yang digunakannya. Vaswani et al. (2017) dalam penelitian inovatif mereka memperkenalkan arsitektur Transformer yang menggantikan metode lama seperti RNN (*Recurrent Neural Networks*). Kunci utama dari arsitektur ini adalah mekanisme atensi (*Self-Attention Mechanism*), yang memungkinkan model untuk menimbang pentingnya setiap kata dalam kalimat terhadap kata lainnya, terlepas dari jarak antar kata tersebut. Struktur dasar Transformer yang terdiri dari komponen *Encoder* dan *Decoder* dapat dilihat pada **Gambar 3** berikut.



Gambar 3. Arsitektur Transformer Sumber: Vaswani et al. (2017)

Kekuatan utama arsitektur Transformer terletak pada penggunaan mekanisme atensi (*Attention Mechanism*), spesifiknya *Scaled Dot-Product Attention*. Mekanisme ini memungkinkan model untuk memproses seluruh urutan input secara paralel dan menentukan tingkat kepentingan (weight) antar kata dalam satu kalimat. Vaswani et al. (2017) merumuskan perhitungan atensi sebagai berikut:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (1)$$

Dimana :

- Q**(Query)** : Representasi kata yang sedang dicari konteksnya.
- K **(*Key*)** : Representasi kata-kata lain dalam kalimat yang menjadi referensi.
- V **(*Value*)** : Isi informasi aktual dari kata tersebut.
- $\sqrt{d_k}$: Faktor penskalaan untuk menjaga kestabilan gradien saat pelatihan.

Secara sederhana, persamaan di atas bekerja layaknya sistem pencarian (*search engine*): model mencocokkan *Query* dengan *Key* yang paling relevan, lalu mengambil Value-nya. Dalam konteks penelitian ini, mekanisme inilah yang memungkinkan LLM memahami hubungan semantik yang kompleks antara pertanyaan wawancara dan jawaban kandidat, meskipun kalimatnya panjang dan berbelit-belit.

1.6.3.2. Kuantisasi Model (*Quantization*)

Implementasi LLM seringkali terkendala oleh ukuran parameter model yang sangat besar, yang menuntut kapasitas memori dan daya komputasi yang tinggi. Untuk mengatasi hal ini, diterapkan teknik kuantisasi (*quantization*). Gholami et al. (2022) mendefinisikan kuantisasi sebagai proses mereduksi presisi representasi numerik bobot dan aktivasi model dari format *floating-point* presisi tinggi (biasanya FP32 atau 32-bit) menjadi format *integer* presisi rendah (seperti INT8 atau 8-bit). Proses ini secara signifikan mengurangi kebutuhan memori dan mempercepat latensi inferensi dengan dampak minimal terhadap akurasi model.

Secara matematis, jenis kuantisasi yang paling umum digunakan adalah Kuantisasi Seragam (*Uniform Quantization*). Gholami et al. (2022) merumuskan pemetaan dari nilai riil r (FP32) ke nilai terkuantisasi q (INT8) menggunakan fungsi transformasi afin sebagai berikut:

$$q = \text{clip}(\text{round}\left(\frac{r}{S}+Z\right), q_{min}, q_{max}) \quad (2)$$

Dimana :

- q : Nilai terkuantisasi (integer)
- r : Nilai input asli dalam format *floating-point* (FP32).
- S **(*Scale Factor*)** : Faktor skala bilangan riil yang menentukan langkah ukuran (*step size*) kuantisasi.

- Z (***Zero Point***) : Nilai *integer* yang memetakan nilai nol pada domain *floating-point* ke domain *integer*. Ini memungkinkan representasi asimetris dari data.
- $\mathrm{round}(.)$: Operasi pembulatan ke bilangan bulat terdekat.
- $\mathrm{clip}(.)$: Fungsi pemangkasan untuk memastikan nilai q tetap berada dalam rentang bit yang tersedia (misalnya $[-128, 127]$ untuk INT8 bertanda).

Sebaliknya, untuk mengembalikan nilai ke domain aslinya saat proses komputasi (dekuantisasi), digunakan rumus:

$$\tilde{r} = S(q - Z) \tag{3}$$

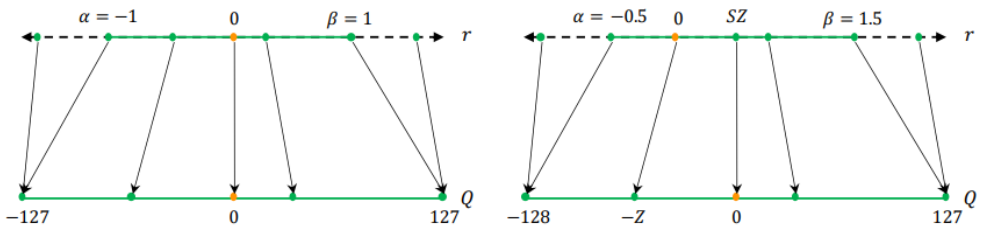
Dimana :

- $\tilde{r}$: Nilai dekuantisasi (floating point)
- q : Nilai terkuantisasi (integer)
- S (***Scale Factor***) : Faktor skala bilangan riil yang menentukan langkah ukuran (*step size*) kuantisasi.
- Z (***Zero Point***) : Nilai *integer* yang memetakan nilai nol pada domain *floating-point* ke domain *integer*. Ini memungkinkan representasi asimetris dari data.

Terdapat dua skema utama berdasarkan penentuan nilai nol (Z) pada penerapannya, sebagaimana diilustrasikan pada **Gambar 4**:

1. Kuantisasi Simetris (*Symmetric Quantization*)
   Menetapkan $Z = 0$, sehingga rentang nilai dipetakan secara simetris di sekitar nol. Metode ini lebih sederhana secara komputasi.
2. Kuantisasi Asimetris (*Asymmetric Quantization*)
   Menggunakan nilai $Z \neq 0$ untuk menangani distribusi data yang tidak seimbang (misalnya output fungsi aktivasi ReLU yang selalu positif). Metode ini memberikan presisi yang lebih baik karena memanfaatkan rentang bit secara penuh.

Penulis menggunakan model Qwen-7B pada penelitian ini yang dikonversi menggunakan format GGUF yang menerapkan teknik kuantisasi blok (*block-wise quantization*) berbasis prinsip-prinsip di atas untuk menjaga keseimbangan antara kompresi ukuran dan kualitas output teks.



Gambar 4. Ilustrasi Kuantisasi Simetris dan Asimetris pada INT8 Sumber: Gholami et al. (2022)

1.6.3.3. Model Spesifik: Qwen-2.5

Penelitian pengembangan sistem menggunakan pendekatan RAG dan *Self-RAG ini* mengggunakan model Qwen-2.5-7B-Instruct. Qwen merupakan seri model bahasa besar yang dikembangkan oleh tim Alibaba Cloud. Bai et al. (2023) menjelaskan bahwa Qwen dilatih menggunakan data multibahasa berskala masif yang mencakup teks, kode pemrograman, dan matematika.

Pemilihan Qwen-2.5, khususnya varian *Instruction Tuned*, didasarkan pada karakteristik utamanya yang unggul dibandingkan model sejenis (seperti Llama-2 atau Mistral) dalam hal pemahaman instruksi yang kompleks dan kemampuan multibahasa yang lebih baik, termasuk keluwesan dalam Bahasa Indonesia. Model ini memiliki arsitektur Transformer *Decoder-only* dengan 7 miliar parameter, yang dinilai memiliki keseimbangan terbaik antara kinerja penalaran (*reasoning*) dan efisiensi komputasi saat dijalankan pada perangkat lokal dengan sumber daya terbatas.

1.6.4 *Retrieval Augmented Generation* (RAG)

Model bahasa besar (LLM) telah menunjukkan kemampuan luar biasa dalam memahami dan menghasilkan teks. Namun, LLM konvensional memiliki keterbatasan inheren, yaitu ketergantungan pada pengetahuan parametrik statis yang dipelajari selama masa pelatihan (*pre-training*). Ji et al. (2023) mengidentifikasi bahwa keterbatasan ini sering menyebabkan dua masalah utama: (1) Ketidakmampuan memperbarui pengetahuan (*knowledge cut-off*) tanpa pelatihan ulang yang mahal, dan (2) Halusinasi (*hallucination*), di mana model menghasilkan fakta yang salah atau tidak berdasar dengan penuh keyakinan.

Kelemahan yang dimiliki oleh LLM coba diatasi oleh Lewis et al. (2020) dengan memperkenalkan paradigma *Retrieval Augmented Generation* (RAG). RAG menggabungkan kekuatan model generatif parametrik (seperti Qwen atau GPT) dengan komponen penelusuran (*retrieval*) non-parametrik yang mengakses basis data eksternal. Mekanisme kerja RAG dapat dianalogikan seperti ujian "buku terbuka" (*open-book exam*). Tanpa RAG, LLM harus menjawab soal hanya dengan mengandalkan ingatan belajarnya (*training* data). Dengan RAG, LLM diizinkan untuk "mencontek" atau merujuk pada buku referensi tertentu sebelum menjawab. Dalam konteks penelitian ini, "buku referensi" tersebut adalah rubrik penilaian standar AIESEC. Hal ini mencegah model memberikan penilaian subjektif atau mengarang kriteria sendiri. Berdasarkan arsitektur yang dijelaskan oleh Gao et al. (2023), sistem RAG terdiri dari tiga komponen utama yang saling berinteraksi:

1. *Retriever* (Pencari)

   Komponen ini bertugas menelusuri basis data vektor untuk menemukan dokumen yang paling relevan dengan *query* (pertanyaan wawancara dan jawaban kandidat). *Retriever* membandingkan makna semantik antara input pengguna dengan seluruh isi rubrik penilaian.
2. *Augmenter* (Penggabung)

   Setelah dokumen relevan ditemukan (misalnya: indikator penilaian untuk pertanyaan tentang "Kepemimpinan"), komponen ini menggabungkan dokumen

tersebut dengan pertanyaan asli kandidat menjadi satu kesatuan konteks (*prompt*).

3. *Generator* (Pembuat Jawaban)

   Komponen terakhir adalah LLM itu sendiri (Qwen-2.5). Berbekal prompt yang sudah diperkaya (*augmented*) dengan data rubrik, Generator kemudian memproduksi teks penilaian akhir yang berisi skor, alasan, dan saran perbaikan yang akurat sesuai referensi yang diberikan.

1.6.4.1 Formulasi RAG

Secara formal, Lewis et al. (2020) merumuskan proses pembangkitan jawaban dalam RAG sebagai berikut. Diberikan sebuah input pertanyaan x, model mengambil serangkaian dokumen relevan z dari korpus pengetahuan eksternal, dan kemudian menghasilkan target respons y.

1.6.4.2 Arsitektur Sistem RAG

Gao et al. (2023) menjelaskan bahwa arsitektur RAG terdiri dari tiga komponen utama yang bekerja secara sekuensial:

1. *Indexing* (Pengindeksan)

   Dokumen eksternal (dalam penelitian ini adalah rubrik penilaian dan panduan wawancara) diubah menjadi representasi vektor (*vector embeddings*) menggunakan model embedding dan disimpan dalam basis data vektor (*vector database*) (Gao et al., 2023).

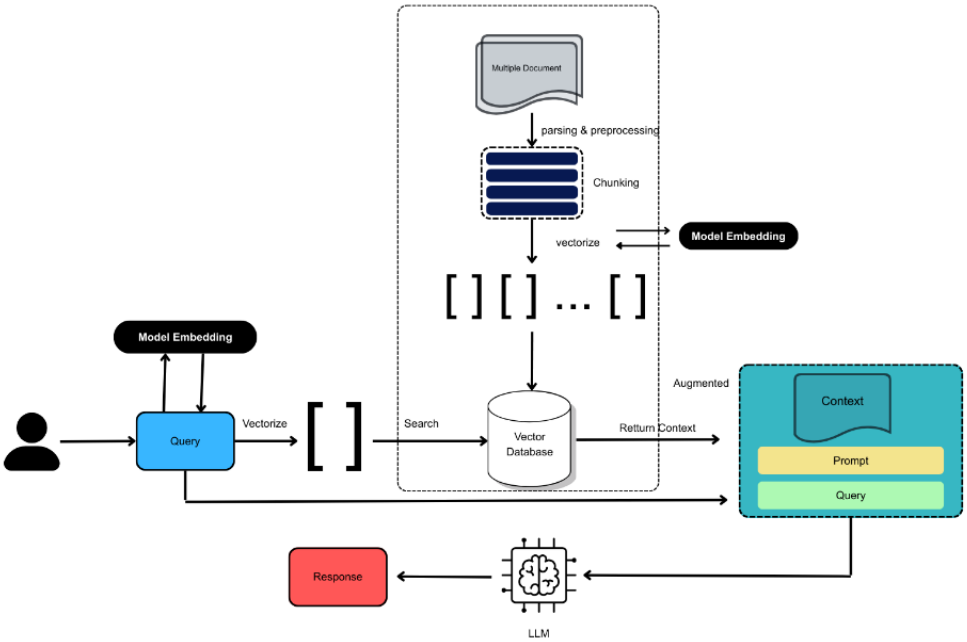
2. Retrieval **(**Pencarian**)**

   Saat pertanyaan masuk, sistem mencari potongan dokumen (*chunks*) yang memiliki kemiripan semantik tertinggi dengan pertanyaan tersebut menggunakan algoritma pencarian tetangga terdekat (*Nearest Neighbor Search*) seperti *Cosine Similarity*.

3. Generation **(**Pembangkitan**)**

   Potongan dokumen yang relevan digabungkan dengan pertanyaan asli sebagai konteks (*prompt*), kemudian diteruskan ke LLM untuk menghasilkan jawaban yang didasarkan pada fakta yang ditemukan.

Dalam konteks penilaian wawancara otomatis, penerapan RAG sangat krusial. Sistem penilaian tidak boleh mengandalkan pengetahuan umum LLM, melainkan harus patuh secara ketat pada rubrik standar organisasi. Dengan RAG, LLM dipaksa untuk "membaca" rubrik spesifik AIESEC sebelum memberikan skor, sehingga meminimalisir subjektivitas model. Ilustrasi arsitektur sistem RAG standar ditunjukkan pada **Gambar 5** berikut.



Gambar 5. Arsitektur Retrieval Augmented Generation (RAG) Sumber: Diolah Penulis dari Lewis et al. (2020)

1.6.5 *Self-Reflective Retrieval Augmented Generation* (*Self*-RAG)

Metode RAG konvensional memiliki kelemahan di mana komponen *Generator* (LLM) dipaksa untuk menjawab berdasarkan dokumen yang diambil (*retrieved context*), terlepas dari apakah dokumen tersebut relevan atau tidak. Hal ini dapat menyebabkan "halusinasi yang dipicu oleh pengambilan" (*retrieval-induced hallucination*) jika sistem pencari gagal menemukan konteks yang tepat. Untuk mengatasi masalah ini, Asai et al. (2023) mengembangkan *Self-Reflective* RAG (*Self*-RAG).

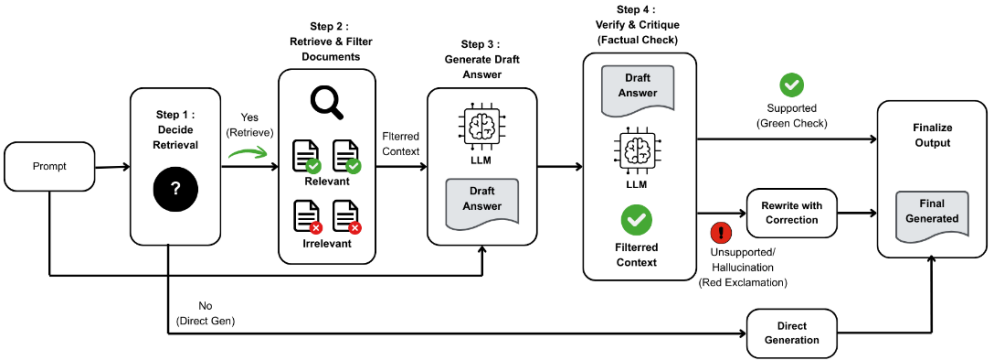
Self-RAG meningkatkan kualitas dan faktualitas luaran dengan melatih LLM untuk melakukan kritik diri (*self-critique*) melalui token refleksi khusus. Berbeda dengan RAG standar yang linier, *Self*-RAG memungkinkan model untuk mengevaluasi kualitas prosesnya sendiri secara adaptif.

1.6.5.1 Arsitektur dan Alur Kerja *Self*-RAG

Arsitektur *Self*-RAG dibangun di atas kerangka kerja RAG standar namun menambahkan lapisan evaluasi metakognitif. Yan et al. (2023) menjelaskan bahwa mekanisme korektif ini sangat penting untuk aplikasi berisiko tinggi seperti penilaian kandidat. Dalam penelitian ini, alur kerja *Self*-RAG diimplementasikan melalui mekanisme validasi bertahap sebagai berikut:

1. *Retrieval on Demand* (Pencarian Sesuai Kebutuhan)
   Sistem pertama-tama menilai apakah pertanyaan input memerlukan informasi eksternal. Jika ya, sistem memanggil modul *retriever* untuk mengambil rubrik yang relevan.
2. *Relevance Critique* (Kritik Relevansi)
   Segera setelah dokumen diambil, model melakukan evaluasi mandiri untuk menentukan apakah dokumen tersebut relevan dengan pertanyaan. Jika dokumen dinilai tidak relevan, sistem dapat mengabaikannya atau mengurangi bobot pengaruhnya.
3. *Generation & Grounding Critique* (Pembangkitan & Kritik Dukungan Fakta)
   Setelah menghasilkan draf penilaian awal, model melakukan verifikasi konsistensi (*consistency check*). Model memeriksa apakah setiap klaim dalam penilaian (misalnya, pemberian skor 3) didukung sepenuhnya oleh bukti dalam teks jawaban kandidat dan indikator rubrik.
4. *Utility Critique* (Kritik Utilitas)
   Tahap akhir di mana model menilai apakah respons yang dihasilkan bermanfaat dan menjawab pertanyaan pengguna secara utuh sebelum ditampilkan sebagai hasil akhir.

Alur kerja adaptif ini memastikan bahwa sistem penilaian tidak hanya "menjawab", tetapi juga "memverifikasi" jawabannya sendiri sebelum disajikan. Ilustrasi alur kerja *Self-RAG* dalam sistem ini ditunjukkan pada **Gambar 6** berikut.



Gambar 6. Alur Kerja *Self-Reflective* RAG dengan Mekanisme Validasi Bertahap
Sumber: Diolah Penulis dari Asai et al. (2023)

1.6.5.2 Prompt-Based Self-Correction

Selain pendekatan berbasis pelatihan (*training-based*) yang memodifikasi bobot internal model, perkembangan terkini dalam *Large Language Models* menunjukkan efektivitas pendekatan berbasis inferensi (*inference-only*) untuk mencapai kemampuan kritik mandiri. Pendekatan ini memanfaatkan kemampuan penalaran intrinsik model melalui rekayasa instruksi (*prompt engineering*) tanpa memerlukan sumber daya komputasi masif untuk *fine-tuning*.

Madaan et al. (2023) dalam penelitiannya yang berjudul *"Self-Refine: Iterative Refinement with Self-Feedback"* memperkenalkan paradigma Iterative Refinement. Dalam metode ini, LLM tidak berhenti pada luaran awal (*initial generation*), melainkan melakukan siklus umpan balik (*feedback loop*). Model diinstruksikan untuk bertindak sebagai kritikus terhadap luaran buatannya sendiri, mengidentifikasi kekurangan, dan kemudian melakukan revisi berdasarkan kritik tersebut. Madaan membuktikan bahwa LLM sering kali memiliki kemampuan yang lebih baik dalam mengkritik dan memperbaiki teks daripada menghasilkan teks sempurna dalam satu kali percobaan (*one-pass generation*).

Kemampuan LLM dalam mengkritik mendorong hadirnya kerangka dalam refleksi dan mengkritik yang dikeluarkan LLM, salah satunya adalah kerangka kerja *Reflexion* yang diajukan Shinn et al. (2023). Kerangka *reflexion* mendefinisikan agen AI dengan kemampuan pembelajaran penguatan verbal (*verbal reinforcement learning*). Alih-alih memperbarui bobot model (seperti pada *Reinforcement Learning* tradisional), Reflexion menggunakan sinyal linguistik atau teks reflektif sebagai memori jangka pendek untuk mengarahkan model agar tidak mengulangi kesalahan yang sama. Shinn menekankan bahwa mekanisme refleksi ini memungkinkan model untuk melakukan evaluasi mandiri terhadap penalaran yang digunakannya.

Penerapan konsep *Self-Refine* dan *Reflexion* dalam arsitektur RAG memungkinkan terciptanya sistem yang mampu memvalidasi relevansi dokumen dan konsistensi jawaban secara mandiri. Keunggulan utama pendekatan ini adalah efisiensi sumber daya; organisasi dapat menerapkan mekanisme *Self-Correction* yang canggih menggunakan *Local LLM* tanpa perlu menyediakan infrastruktur server GPU kelas atas untuk melatih model khusus.

1.6.6 Vector Database dan Embedding

Implementasi metode RAG dan pencarian semantik membutuhkan representasi data yang dapat dipahami oleh mesin secara matematis. Oleh karena itu, diperlukan teknologi *Text Embedding* dan basis data vektor (*Vector Database*).

1.6.6.1 Text Embedding

Komputer tidak dapat memahami teks (kata atau kalimat) secara langsung. Oleh karena itu, teks harus dikonversi menjadi representasi numerik yang disebut vektor. Reimers dan Gurevych (2019) mendefinisikan *Embedding* sebagai proses pemetaan kata, frasa, atau dokumen ke dalam ruang vektor kontinu berdimensi tinggi. Dalam ruang vektor ini, kalimat yang memiliki makna semantik serupa akan diposisikan berdekatan, meskipun kata-kata penyusunnya berbeda. Dalam penelitian ini, model *embedding* yang digunakan adalah BAAI/bge-m3, yang mampu mengubah teks rubrik dan jawaban wawancara menjadi deretan angka (vektor) yang merepresentasikan makna konteksnya.

1.6.6.2 Cosine Similarity

Teks yang telah diubah menjadi *vector* perlu dilakukan pengukuran, yakni mengukur seberapa mirip jawaban kandidat dengan indikator dalam rubrik. Metode pengukuran yang paling umum digunakan adalah *Cosine Similarity*. Han et al. (2012) menjelaskan bahwa *Cosine Similarity* mengukur kosinus sudut antara dua vektor dalam ruang multidimensi. Nilai kemiripan dihitung dengan rumus:

$$similarity(A,B) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \quad (4)$$

Nilai 1 menunjukkan kedua dokumen sangat mirip (sudut 0 derajat), dan nilai 0 menunjukkan tidak ada kemiripan (tegak lurus). Metode ini dipilih karena efisien dalam mengukur kemiripan orientasi dokumen tanpa terpengaruh oleh panjang pendeknya teks.

1.6.6.3 FAISS (*Facebook AI Similarity Search*)

Metode retrieval dalam RAG mendorong untuk melakukan pencarian vektor dalam jumlah besar secara cepat, metode pencarian konvensional seringkali terlalu lambat. Johnson et al. (2017) dari tim riset Facebook AI memperkenalkan FAISS, sebuah pustaka (*library*) yang dirancang khusus untuk pencarian kesamaan (*similarity search*) yang efisien pada vektor padat (*dense vectors*).

FAISS bekerja dengan cara mengindeks vektor-vektor data menggunakan teknik klastering dan kuantisasi produk, sehingga sistem tidak perlu membandingkan query dengan seluruh data satu per satu (*brute force*). Dalam sistem ini, FAISS digunakan sebagai mesin pencari utama (*vector store*) yang memungkinkan sistem menemukan rubrik penilaian yang tepat dalam hitungan milidetik, bahkan jika basis data rubrik bertambah besar di masa depan.

1.6.7 Evaluasi Sistem RAG

Evaluasi terhadap sistem *Retrieval Augmented Generation* (RAG) memerlukan pendekatan yang berbeda dibandingkan evaluasi model klasifikasi konvensional. Kinerja sistem RAG tidak hanya ditentukan oleh ketepatan skor akhir, melainkan juga sangat bergantung pada kualitas dua komponen utamanya: kemampuan mesin pencari (*retriever*) dalam menemukan dokumen yang relevan dan kemampuan model generatif (*generator*) dalam menyusun jawaban berdasarkan dokumen tersebut. Oleh karena itu, diperlukan metrik evaluasi terpisah untuk mengisolasi dan mengukur kinerja masing-masing komponen agar dapat dipastikan bahwa sistem bekerja sesuai logika yang dirancang, bukan karena kebetulan.

1.6.7.1 *Quadratic Weighted Kappa* (QWK)

Data nilai wawancara memiliki sifat ordinal, di mana urutan angka memiliki makna tingkatan kualitas. Kesalahan prediksi model yang memiliki selisih besar (misalnya, ahli memberi nilai 0 tetapi model memprediksi 4) dianggap lebih fatal daripada kesalahan dengan selisih kecil (misalnya, ahli memberi nilai 3 tetapi model

memprediksi 4). Oleh karena itu, metrik akurasi standar dinilai kurang memadai untuk mengukur reliabilitas penilaian bertingkat.

Cohen (1968) memperkenalkan metrik *Quadratic Weighted Kappa* (QWK) atau *Cohen's Kappa* untuk mengukur tingkat kesepakatan antar-penilai (*inter-rater reliability*) dengan memperhitungkan peluang kesepakatan yang terjadi secara acak. Keunggulan utama QWK terletak pada penggunaan bobot kuadratik yang memberikan "hukuman" (*penalty*) eksponensial terhadap kesalahan prediksi. Semakin besar selisih antara skor prediksi AI dan skor ahli, semakin besar pengurangan nilai akurasinya.

Nilai koefisien Kappa (κ) berkisar antara -1 hingga 1. Landis dan Koch (1977) mengklasifikasikan interpretasi nilai Kappa sebagai berikut: nilai di bawah 0.00 menunjukkan kesepakatan buruk, 0.00–0.20 lemah, 0.21–0.40 cukup, 0.41–0.60 sedang, 0.61–0.80 substansial, dan 0.81–1.00 menunjukkan kesepakatan yang hampir sempurna. Rumus perhitungan QWK adalah:

$$\kappa = 1 - \frac{\sum_{i,j} w_{i,j} O_{i,j}}{\sum_{i,j} w_{i,j} E_{i,j}} \quad (5)$$

Dimana $O_{i,j}$ adalah matriks observasi kesepakatan, $E_{i,j}$ adalah matriks ekspektasi kesepakatan secara acak, dan $w_{i,j}$ adalah bobot kuadratik yang diberikan untuk setiap selisih skor.

1.6.7.2 *Retrieval Accuracy* (*Hit Rate*)

Kinerja komponen Retriever merupakan fondasi utama dari keberhasilan sistem RAG. Kegagalan dalam tahap ini akan menyebabkan masalah *garbage in*, *garbage out*, di mana model bahasa menerima konteks yang salah sehingga menghasilkan penilaian yang tidak valid. Untuk mengukur efektivitas pencarian ini, digunakan metrik *Retrieval Accuracy* atau yang secara teknis dikenal sebagai *Hit Rate*.

Chen et al. (2023) mendefinisikan *Hit Rate* sebagai proporsi keberhasilan sistem dalam menampilkan setidaknya satu dokumen yang benar-benar relevan (*ground truth*) di dalam daftar k dokumen teratas yang diambil (*top-k retrieval*). Dalam konteks penelitian ini, *Hit Rate* mengukur seberapa sering sistem berhasil menarik indikator rubrik yang tepat sesuai dengan pertanyaan wawancara yang sedang dianalisis. Jika sistem gagal mengambil rubrik yang benar, maka penilaian AI dipastikan cacat prosedural. Perhitungan *Hit Rate* dapat dirumuskan sebagai berikut:

$$\text{Hit Rate} = \frac{Jumlah\ Kueri\ dengan\ minimal\ 1\ Dokumen\ Relevan\ yang\ Diambil}{Total\ Jumlah\ Kueri} \times 100\% \quad (6)$$

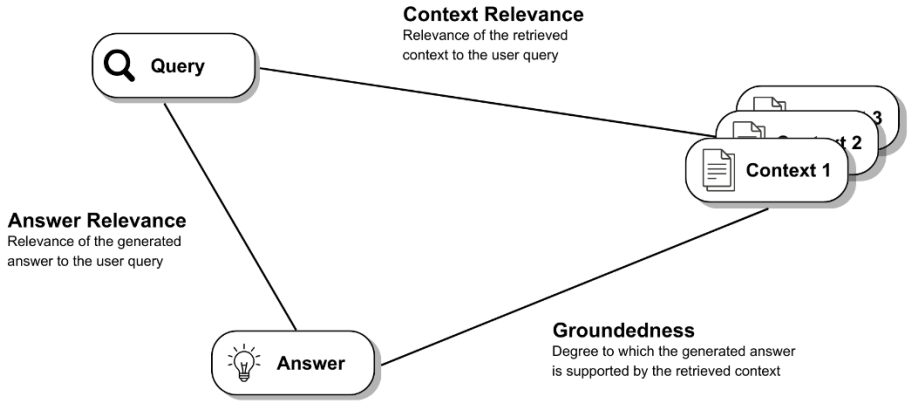
1.6.7.3 *Faithfulness* (Kesetiaan pada Konteks)

Tantangan terbesar dalam penggunaan *Large Language Model* (LLM) adalah risiko halusinasi, yaitu kondisi ketika model menghasilkan pernyataan yang tidak didukung oleh fakta. Dalam sistem penilaian wawancara, halusinasi sangat berbahaya karena

dapat memunculkan kriteria penilaian fiktif yang tidak ada dalam standar organisasi. Untuk memitigasi risiko ini, digunakan metrik *Faithfulness*.

Es et al. (2025), dalam kerangka kerja evaluasi RAGAS (*Retrieval Augmented Generation Assessment*), menjelaskan bahwa *Faithfulness* mengukur konsistensi faktual antara jawaban yang dihasilkan oleh Generator dengan konteks yang diambil (*retrieved context*). Metrik ini menilai apakah setiap klaim atau argumen yang dibuat oleh AI dalam alasan penilaian (*reasoning*) dapat ditelusuri kembali buktinya pada rubrik dan transkrip jawaban kandidat.

Nilai *Faithfulness* dihitung dengan membagi jumlah klaim yang didukung oleh bukti dengan total jumlah klaim yang dibuat oleh sistem. Nilai yang tinggi (mendekati 1) menunjukkan bahwa sistem "setia" pada rubrik dan tidak mengarang informasi, yang merupakan syarat mutlak untuk objektivitas penilaian. Ilustrasi mengenai kerangka kerja evaluasi komponen RAG ditunjukkan pada **Gambar 7**.



Gambar 7. Triad Evaluasi Sistem RAG (*Context Relevance, Answer Relevance, dan Groundedness*) Sumber: Stankov dan Velinova (2025)

1.6.8 Kerangka Kerja Pengembangan (*Development Frameworks*)

Realisasi sistem penilaian otomatis ini tidak hanya bergantung pada kecanggihan model bahasa, melainkan juga pada keandalan arsitektur perangkat lunak yang menopangnya. Integrasi antara model kecerdasan buatan (*backend*) dengan antarmuka pengguna (*frontend*) memerlukan serangkaian alat pengembangan yang mampu menangani komputasi berat secara efisien namun tetap responsif terhadap input pengguna. Pemilihan kerangka kerja (*framework*) yang tepat sangat krusial untuk memastikan bahwa latensi yang dihasilkan oleh proses inferensi LLM tidak mengganggu pengalaman pengguna.

1.6.8.1 Bahasa Pemograman Python

Bahasa pemrograman *Python* dipilih sebagai fondasi utama dalam pengembangan seluruh komponen sistem, baik sisi server maupun logika pemrosesan data. Rossum & Drake (2009) merancang *Python* sebagai bahasa tingkat tinggi yang berorientasi objek dengan filosofi desain yang menekankan pada keterbacaan kode (*code*

readability). Popularitas Python dalam domain sains data dan kecerdasan buatan didukung oleh ekosistem pustaka (*libraries*) yang masif dan matang.

Penelitian ini memanfaatkan berbagai pustaka spesifik Python untuk mendukung fungsi RAG dan LLM. Pustaka `Sentence-Transformers` digunakan untuk menghasilkan *embedding* vektor dari teks rubrik, sementara FAISS (*Facebook AI Similarity Search*) digunakan untuk manajemen basis data vektor yang memungkinkan pencarian semantik berkecepatan tinggi. Selain itu, pustaka llama-cpp-python menjadi komponen vital yang memungkinkan model bahasa besar terkuantisasi (format GGUF) dapat dijalankan secara efisien pada perangkat keras lokal dengan memanfaatkan akselerasi CPU dan GPU secara hibrida. Fleksibilitas Python dalam mengintegrasikan berbagai modul heterogen inilah yang memungkinkannya menjadi perekat utama dalam arsitektur sistem ini.

1.6.8.2 FastAPI (*Backend Framework*)

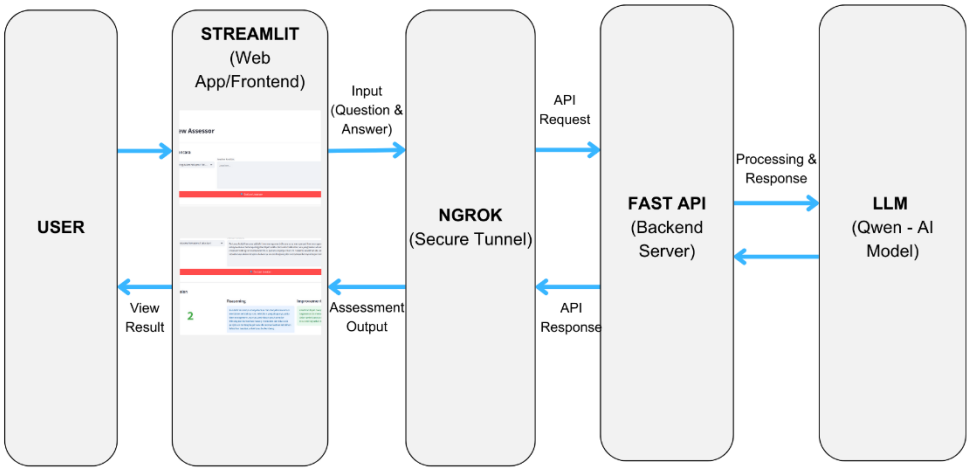
Pemisahan logika pemrosesan AI dengan antarmuka pengguna diimplementasikan melalui arsitektur *Client-Server* menggunakan FastAPI. FastAPI adalah kerangka kerja web modern untuk membangun API (*Application Programming Interface*) dengan *Python* yang berbasis pada standar tipe data Python. Ramírez (2018) menyoroti bahwa FastAPI dirancang untuk memberikan kinerja eksekusi yang sangat tinggi, setara dengan NodeJS dan Go, berkat penggunaan *Starlette* untuk fitur web dan *Pydantic* untuk validasi data.

Implementasi FastAPI dalam penelitian ini tidak sekadar sebagai penghubung data, melainkan sebagai pengelola konkurensi. Fitur utama yang dimanfaatkan adalah dukungan bawaan untuk pemrograman asinkron (*asynchronous programming*) menggunakan sintaks `async` dan `await`. Mengingat proses inferensi LLM bersifat *CPU-bound* dan memakan waktu lama (30-60 detik per jawaban), penggunaan server tradisional yang bersifat sinkron akan menyebabkan pemblokiran (*blocking*) di mana permintaan baru tidak dapat diproses sebelum permintaan sebelumnya selesai. Dengan fitur asinkron dan mekanisme penguncian (*locking mechanism*) yang disediakan oleh FastAPI, sistem dapat mengelola antrian permintaan evaluasi (*inference queue*) secara tertib, mencegah terjadinya timeout atau kegagalan koneksi saat beberapa pengguna mengakses sistem secara bersamaan.

1.6.8.3 Streamlit (*Frontend Framework*)

Sisi antarmuka pengguna (*Front-End*) dikembangkan menggunakan kerangka kerja Streamlit. Streamlit Inc.(2019) mendeskripsikan Streamlit sebagai kerangka kerja *open-source* yang dirancang khusus untuk mempercepat pembuatan aplikasi web berbasis data (*data apps*). Berbeda dengan pengembangan web tradisional yang memerlukan penguasaan HTML, CSS, dan JavaScript secara mendalam, Streamlit memungkinkan pengembang untuk mentransformasi skrip data Python menjadi aplikasi web interaktif secara langsung.

Penggunaan Streamlit dalam penelitian ini didasari oleh kebutuhan akan *prototyping* yang cepat untuk memvisualisasikan hasil penilaian. Streamlit menyediakan komponen antarmuka siap pakai (*widgets*) yang responsif untuk menerima input teks panjang dari jawaban kandidat dan menampilkan hasil evaluasi JSON dari server dalam format kartu skor yang mudah dibaca. Pendekatan ini memungkinkan fokus pengembangan lebih tertuju pada akurasi logika penilaian AI daripada kompleksitas desain antarmuka, namun tetap menghasilkan dashboard yang fungsional dan estetis untuk kebutuhan pengujian pengguna (*User Acceptance Test*). Untuk memberikan gambaran menyeluruh mengenai integrasi antar komponen yang telah dijelaskan, arsitektur teknis sistem penilaian otomatis ini diilustrasikan pada **Gambar 8**.



Gambar 8. Arsitektur Sistem Penilaian Otomatis Berbasis Web Sumber: Olahan Penulis

1.6.8.4 Ngrok (*Secure Tunneling*)

Aksesibilitas sistem dalam tahap pengujian menjadi tantangan tersendiri ketika server AI dijalankan pada lingkungan lokal (*local environment*) yang berada di belakang *firewall* atau NAT (*Network Address Translation*). Untuk mengatasi hal ini tanpa harus menyewa *server cloud* GPU yang mahal, digunakan teknologi tunneling melalui Ngrok. Shreve (2016) menjelaskan Ngrok sebagai alat yang menciptakan terowongan aman (*secure tunnel*) dari internet publik ke aplikasi yang berjalan di *localhost*.

Ngrok bertugas mengekspos port lokal tempat FastAPI berjalan (misalnya port 8000) ke internet melalui URL publik yang diamankan dengan enkripsi HTTPS dalam implementasi sistem ini. Hal ini memungkinkan aplikasi Streamlit, yang mungkin di-hosting di layanan cloud terpisah atau diakses dari perangkat berbeda, untuk dapat mengirimkan data jawaban kandidat ke server lokal peneliti seolah-olah server tersebut berada di jaringan publik. Mekanisme ini menjembatani kesenjangan

antara kebutuhan komputasi berat yang hanya bisa dilakukan di komputer lokal berspesifikasi tinggi dengan kebutuhan aksesibilitas berbasis web.

BAB II
METODOLOGI PENELITIAN

2.1 Tempat dan Waktu Penelitian

2.1.1 Tempat Penelitian

Seluruh proses penelitian dilaksanakan di Rumah Peneliti yang berlokasi di Tamalanera, Kota Makassar dan di Laboratorium Rekayasa Perangkat Lunak Program Studi Sistem Informasi, Departemen Matematika, Fakultas MIPA, Universitas Hasanuddin.

2.1.2 Waktu Penelitian

Penelitian dilaksanakan selama 12 pekan dengan tahapan pelaksanaan yang mengacu pada tahapan penelitian yang akan dijelaskan pada bagian **2.3.1** di bab ini. Timeline penelitian bisa dilihat lebih detail pada **Tabel 2** berikut.

Tabel 2. Tabel waktu pelaksanaan penelitian

No	Tahapan Penelitian	Pekan/Bulan											
		Oktober		November				Desember				January	
		1	2	1	2	3	4	1	2	3	4	1	2
1	Studi Literatur	■	■										
2	Pengumpulan Data		■	■									
3	Perancangan Sistem & Arsitektur *Self*-RAG				■	■	■	■	■				
4	Implementasi Sistem									■	■		
5	Pengujian Sistem										■	■	
6	Analisis & Evaluasi Hasil Pengujian												■

2.2 Instrumen Penelitian

Penelitian dilaksanakan menggunakan beberapa instrument berupa perangkat keras dan perangkat lunak yang digunakan dalam proses pengembangan dan evaluasi sistem

2.2.1 Perangkat Keras

Selama proses pengembangan pipeline RAG, pengembangan sistem penilaian wawancara, dan evaluasi sistem, peneliti menggunakan Laptop dengan spesifikasi pada **Tabel 3** berikut

Tabel 3 Spesifikasi perangkat laptop yang digunakan untuk pengembangan sistem dan pipeline RAG

Komponen	Spesifikasi
Sistem Operasi	Microsoft Windows 11 Home Single Language
Processor	AMD Ryzen 7 5800H
RAM	8 GB
Penyimpanan	512gb
GPU	AMD Radeon RX 5500M

Selain laptop yang digunakan sebagai media pengembangan sistem dan pipeline RAG, penulis juga menggunakan *Personal Computer* (PC) sebagai media pengembangan server, integrasi model LLM, dan *deployment* model LLM pada API *service.* Peneliti menggunakan PC dengan spesifikasi pada **Tabel 4** berikut

Tabel 4. Spesifikasi perangkat PC yang digunakan sebagai server dari sistem

Komponen	Spesifikasi
Sistem Operasi	Microsoft Windows 11 Home Single Language
Processor	12th Gen Intel® Core™ i7-12700
RAM	16 GB
Penyimpanan	512gb
GPU	NVIDIA GeForce RTX 4060 Ti

2.2.2 Perangkat Lunak dan Pustaka

Tidak hanya perangkat keras yang digunakan oleh peneliti dalam pengembangan penilaian wawancara ini. Peneliti juga menggunakan beberapa perangkat lunak dan *library* yang digunakan dalam proses vektorisasi, *deployment* model, *indexing*, chunking, dan proses lainnya dalam pengembangan sistem ini. Beberapa perangkat lunak dan pustaka yang penulis gunakan selama proses pengembangan sistem ini dijelaskan pada **Tabel 5** berikut.

Tabel 5. Daftar perangkat lunak dan pustaka yang digunakan dalam pengembangan sistem

Perangkat Lunak/Pustaka	Fungsi/Deskripsi
Visual Studio Code (VS Code)	*Integrated Development Environment* (IDE) utama yang digunakan untuk menulis, menyunting, dan men-debug seluruh kode sumber aplikasi (*frontend*) dan server (*backend*).

Google Colaboratory (Colab)	Lingkungan komputasi berbasis *cloud* yang digunakan untuk tahap eksperimen awal, pengujian *pipeline* RAG, dan validasi performa model LLM memanfaatkan akses GPU gratis (T4).
Google Sheets	Alat pengolah data yang digunakan untuk manajemen dataset jawaban wawancara, merekapitulasi perbandingan skor (AI vs. Manusia), serta menghitung metrik evaluasi statistik (*Accuracy, F1-Score, Kappa*).
Google Forms	Platform survei digital yang digunakan sebagai instrumen pengumpulan data kualitatif untuk validasi ahli (*Expert Judgment*).
Python 3.11	Bahasa pemrograman utama yang digunakan untuk mengembangkan seluruh logika sistem, mulai dari pemrosesan data, manajemen model AI, hingga antarmuka pengguna.
FastAPI	Kerangka kerja (*framework*) *backend* untuk membangun REST API yang menangani permintaan evaluasi dari pengguna secara asinkron (*asynchronous*) dan efisien.
Streamlit	Kerangka kerja *frontend* untuk membangun antarmuka pengguna (*dashboard*) berbasis web yang interaktif, tempat pengguna menginput jawaban dan melihat hasil penilaian.
Llama-cpp-python	Pustaka untuk memuat dan menjalankan *Large Language Model* (LLM) berformat GGUF (Qwen-2.5) pada perangkat keras lokal (CPU/GPU) dengan latensi rendah.
Sentence-Transformers	Pustaka yang digunakan untuk memuat model *embedding* (BAAI/bge-m3) guna mengubah teks rubrik dan jawaban wawancara menjadi representasi vektor numerik.
FAISS (Facebook AI Similarity Search)	Pustaka basis data vektor (*vector store*) yang digunakan untuk menyimpan indeks vektor rubrik dan melakukan pencarian kesamaan semantik (*semantic search*) berkecepatan tinggi.
Ngrok	Layanan secure tunneling untuk mengekspos server lokal (FastAPI) ke internet publik agar dapat diakses oleh aplikasi frontend atau pengguna dari jarak jauh selama pengujian.
PyTorch	Kerangka kerja deep learning dasar yang menjadi dependensi utama untuk menjalankan *Sentence-Transformers* dan operasi tensor lainnya.

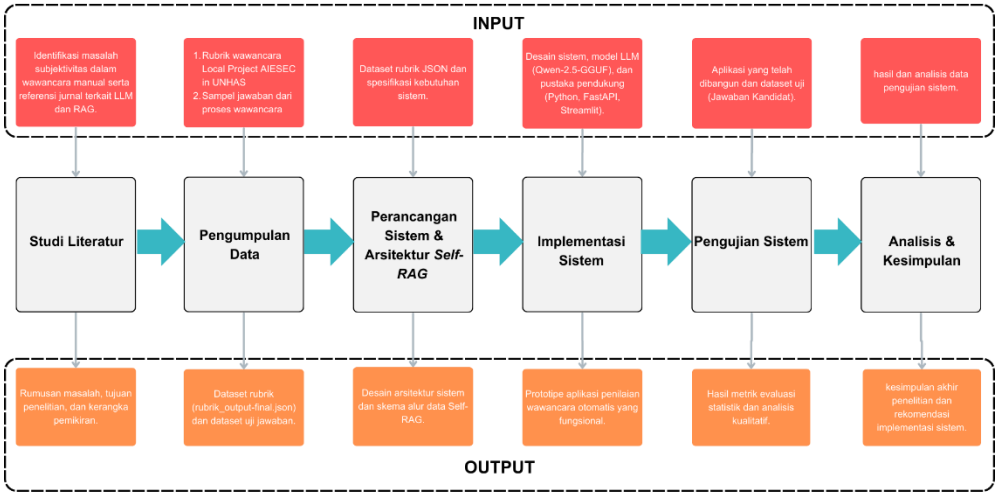
Hugging Face Hub	Pustaka utilitas untuk mengunduh model LLM (.gguf) dan model *embedding* secara otomatis dari repositori Hugging Face.
Requests	Pustaka HTTP client yang digunakan pada sisi aplikasi Streamlit untuk mengirimkan data input (JSON) ke server API *backend*.
NumPy	Pustaka komputasi numerik untuk memanipulasi data array vektor hasil *embedding* sebelum diproses oleh FAISS.
Graphviz	Pustaka visualisasi yang digunakan dalam tahap analisis untuk memetakan alur logika atau *decision tree* sistem (berdasarkan impor kode Anda).
Uvicorn	Server *gateway interface* (ASGI) berkinerja tinggi yang digunakan untuk menjalankan aplikasi FastAPI agar dapat menangani koneksi web secara *real-time*.
Pydantic	Pustaka validasi data yang digunakan untuk mendefinisikan skema input (JSON) pada API, memastikan data pertanyaan dan jawaban yang dikirim ke server memiliki format yang benar.
Asyncio	Pustaka bawaan Python untuk pemrograman asinkron, digunakan untuk mengelola antrian (*queue*) dan proses konkuren agar server tidak macet (*blocking*) saat menangani banyak permintaan.
Pyngrok	*Wrapper* Python untuk aplikasi Ngrok, memungkinkan pembuatan *secure tunnel* secara otomatis langsung dari dalam kode Python (seperti di Google Colab) tanpa perlu konfigurasi manual.
Textwrap	Pustaka utilitas Python yang digunakan untuk merapikan format teks output pada terminal atau log agar mudah dibaca saat proses debugging atau analisis manual.
JSON	Pustaka standar untuk memproses pertukaran data antara aplikasi *frontend* dan server *backend* dalam format JavaScript *Object Notation* yang ringan.
RE (Regular Expression)	Pustaka pengolahan teks yang digunakan untuk mengekstraksi pola tertentu (seperti angka skor atau format JSON yang tercampur teks) dari output mentah model LLM.
OS & Sys	Pustaka sistem operasi yang digunakan untuk manajemen jalur direktori (*file path*), variabel lingkungan, dan konfigurasi sistem dasar server.

Transformers	Pustaka dari Hugging Face yang menyediakan arsitektur dasar (AutoTokenizer, AutoModel) yang diperlukan jika sistem perlu memuat model versi penuh (non-GGUF) atau komponen tokenizer tertentu.

2.3 Prosedur Penelitian

Tahap pertama dalam melaksanakan sebuah penelitian setelah ditentukan masalah apa yang ingin diangkat adalah merancang prosedur penelitian guna terciptanya lingkungan kerja penelitian yang rapi dan terstruktur. Prosedur penelitian dirancang sebagai panduan teknis utama dalam melaksanakan penelitian dan juga sebagai prasyarat pelaksanaan penelitian yang menunjukkan instrument dan perangkat yang akan digunakan selama proses penelitian.

2.3.1 Tahapan/Alur Penelitian



Gambar 9. Diagram tahapan, masukan, dan luaran penelitian

Gambar 9 menunjukkan tahapan atau alur dari penelitian yang akan dilaksanakan beserta masukan dan luaran dari setiap tahap penelitian. Selain menjadi pedoman teknis dalam melaksanakan penelitian, diagram diagram tersebut akan menentukan dalam penyusunan hasil penelitian yang akan dipaparkan pada bagian berikutnya.

2.3.2 Persiapan Perangkat Penelitian

Penelitian dilaksanakan ketika semua prasyarat dari penelitian telah siap dan terpenuhi agar nantinya tidak ada tahap atau proses yang terhambat dalam pelaksanaan penelitian. Penelitian ini mempersyaratkan berbagai instrument

penelitian yang telah dipaparkan pada bagian sebelumnya. Instrumen penelitian meliputi perangkat lunak dan perangkat keras yang sudah harus tersedia sebelum penelitian dimulai dan dipastikan fungsionalitasnya agar ketika penilitian telah dilaksanakan, instrumen tersebut bisa bekerja dengan baik

2.4 Pengumpulan Data

Proses pengumpulan data merupakan tahapan fundamental dalam penelitian ini untuk memastikan ketersediaan informasi yang valid dan reliabel bagi pengembangan serta evaluasi sistem. Penelitian ini menerapkan pendekatan pengumpulan data yang komprehensif dengan memadukan data tekstual naratif (jawaban wawancara) dan data terstruktur (skor penilaian). Sumber data diklasifikasikan menjadi dua kategori utama, yaitu data primer yang diperoleh langsung dari subjek penelitian dan data sekunder yang bersumber dari dokumen pedoman organisasi. Keseluruhan data yang dihimpun kemudian melalui proses pra-pemrosesan agar dapat dibaca dan dianalisis oleh model bahasa besar (Large Language Model).

2.4.1 Data Primer

Data primer dalam penelitian ini merujuk pada data mentah yang diperoleh secara langsung dari aktivitas seleksi wawancara *Local Project* AIESEC in Unhas pada periode semester 1 tahun 2023. Sumber data berasal dari *tools* atau lembar kerja wawancara (*interview tools*) yang digunakan oleh para perekrut (*interviewer*) saat melakukan seleksi kandidat. Data ini memegang peranan krusial karena merepresentasikan kondisi nyata (*real-world data*) dari interaksi antara penanya dan penjawab, yang memiliki variasi linguistik yang kompleks dan beragam.

Kumpulan data primer ini terdiri dari empat komponen informasi utama yang saling terkait. Pertama adalah Identitas Kandidat (yang telah dianonimisasi untuk menjaga privasi). Kedua adalah Pertanyaan Wawancara yang diajukan. Ketiga adalah Jawaban Kandidat berupa teks transkrip respons yang diberikan oleh pelamar. Keempat, dan yang paling kritikal untuk tahap evaluasi, adalah Nilai Historis yang diberikan oleh *interviewer* manusia. Kemudian data primer ini disederhanakan dengan hanya berisi pertanyaan dan jawaban dari sampel kandidat yang diwawancara. Total ada 4 kandidat yang jawabannya dijadikan sampel. Kemudian dari setiap kandidat tersebut diambil 10 pasang pertanyaan dan jawaban (menggunakan 10 pertanyaan yang sama) sehingga total ada 40 pasang pertanyaan dan jawaban yang digunakan. Sampel data pertanyaan dan jawaban bisa dilihat pada **Tabel 6**, untuk sample data pertanyaan dan jawaban lengkap dapat dilihat pada **Error! Reference source not found.**.

Mengingat data wawancara bersifat rahasia, seluruh data primer yang digunakan dalam penelitian ini telah melalui proses anonimisasi (penghapusan identitas pribadi kandidat seperti nama dan kontak). Selain itu, pemrosesan data dilakukan sepenuhnya pada lingkungan lokal (Local Environment) menggunakan model LLM yang berjalan di perangkat peneliti (On-Premise), tanpa pengiriman data

ke server pihak ketiga (seperti OpenAI API), guna menjamin keamanan dan kerahasiaan data kandidat.

Tabel 6. Sampel data pertanyaan dan jawaban wawancara *Local Project* AIESEC in UNHAS

Pertanyaan	Jawaban
Apa nilai nilai (values) yang penting dalam hidupmu? Jelaskan!	Saya punya kalimat yang menjadi motivasi saya dan juga menjadi nilai hidup saya, yaitu "Orang terbaik di dunia ini adalah orang yang paling bermanfaat bagi banyak orang", jadi saya ingin berkontribusi utnuk menberi dampak positif ke orang orang di sekitar saya dan mendorong saya untuk menjadi lebih baik dari hari ke hari
Apa tiga kelebihan teratasmu/terkuatmu? Jelaskan!	Kelibahan yang saya miliki adalah pertama saya merupakan orang yang gampang beradaptasi. Berdasarkan pengalaman saya sebelumnya saat bekerja secara tim di sebuah project saya bisa beradaptasi dengan mudah dengan orang orang baru yang saya temui saat mengerjakan project itu, itu menunjukkan bahwa saya mudah beradaptasi. Selanjutnya, kelebihan saya adalah orang yang bertanggung jawab. Saya pernah mengambil beberapa tanggung jawab secara bersamaan namun saya dapat mengatur dan menghandle semuanya dengan cukup baik. Kelebihan Ketiga saya adalah Personality saya. Saya bisa adjust personality saya berdasarkan situasi dimana saya berada, misalnya saya berhadapan dengan orang yang introvert, maka saya melakukan personal approach ke orang itu
Apa tiga kelemahan/kekurangan teratasmu dan bagaimana kamu mengatasinya? Jelaskan!	pertama adalah Time management, saya cukup keteteran dalam mengatur time managementnya, saat saya mendapat banyak tugas dan tanggung jawab, saya terpaksa mengerjakan beberapa tanggung jawab atau tugasnya mendekati deadline. Namun, saya berusaha untuk mengatur waktunya menjadi lebih efisien. Kekurangan saya selanjutnya adalah mengendalikan mood saya, teman teman saya bisa mengetahui perasaan dan apa yang saya katakan tanpa saya mengatakannya, karena teman saya bisa membaca perasaan saya lewat ekspresi saya

2.4.2 Data Sekunder

Data sekunder berfungsi sebagai basis pengetahuan (*knowledge base*) yang menjadi acuan utama bagi sistem *Retrieval Augmented Generation* (RAG) dalam melakukan penilaian. Sumber data sekunder penelitian ini adalah dokumen Pedoman Wawancara (*Interview Guideline*) resmi untuk *Local Project* AIESEC.

Dokumen tersebut memuat standar kompetensi, indikator perilaku, dan kriteria penilaian yang baku.

Mengingat format asli dokumen pedoman yang berupa teks naratif panjang (PDF/Doc), dilakukan proses transformasi data agar dapat diproses secara komputasional oleh mesin. Proses ini diawali dengan penyederhanaan (*simplification*) dan strukturisasi rubrik ke dalam format tabel menggunakan *Google Spreadsheet*. Langkah selanjutnya adalah konversi data tabel tersebut menjadi format *JavaScript Object Notation* (JSON) yang terstruktur. Format JSON dipilih karena kompatibilitasnya yang tinggi dengan pustaka *vector database* dan kemudahan pembacaan (*parsing*) oleh bahasa pemrograman Python.

Struktur data JSON rubrik penilaian ini dirancang secara hierarkis yang terdiri dari lima elemen kunci pada setiap entitasnya:

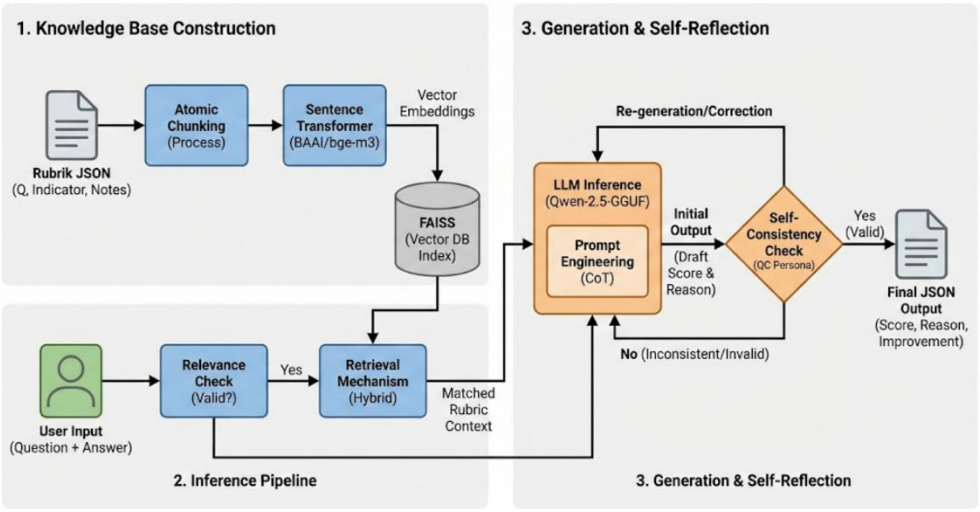
1. Dimensi
   Aspek kompetensi besar yang dinilai (misalnya: *Activating Leadership*).
2. Pertanyaan
   Teks pertanyaan spesifik yang diajukan kepada kandidat.
3. Indikator
   Penjelasan mengenai perilaku atau poin kunci yang harus muncul dalam jawaban kandidat.
4. Catatan
   Konteks tambahan atau definisi istilah khusus untuk membantu pemahaman penilai.
5. Level Skor (0-4)
   Deskripsi spesifik yang membedakan kualitas jawaban untuk setiap tingkatan nilai, mulai dari nilai 0 (terendah) hingga 4 (tertinggi).

Transformasi data sekunder ini memastikan bahwa setiap butir penilaian memiliki representasi vektor yang unik, sehingga memungkinkan sistem pencari (*retriever*) menemukan referensi rubrik yang paling relevan dengan presisi tinggi. Potongan kode rubrik yang telah ditransformasikan menjadi format JSON dapat dilihat pada **Lampiran 7**.

2.5 Perancangan Sistem

Perancangan sistem merupakan tahapan yang sangat penting dalam proses pengembangan teknologi yang bertujuan untuk mengubah kebutuhan-kebutuhan yang telah diidentifikasi melalui analisis mendalam dan data yang dikumpulkan menjadi sebuah rancangan teknis yang matang dan siap untuk diimplementasikan secara praktis. Dalam konteks ini, sistem penilaian wawancara otomatis dikembangkan dengan menggunakan pendekatan arsitektur Client-Server, yang secara efektif terintegrasi dengan metode *Self-Reflective Retrieval Augmented Generation* (*Self*-RAG), sebuah teknik canggih yang memadukan kemampuan retrieval informasi dengan generasi respons yang reflektif. Fokus utama dari perancangan ini adalah pembentukan mekanisme inferensi yang cerdas, yang dirancang untuk secara mendalam memahami konteks spesifik dari pertanyaan-

pertanyaan wawancara, mengambil acuan-acuan penilaian yang paling relevan dari basis pengetahuan yang tersedia, serta melakukan evaluasi mandiri terhadap hasil penilaian yang dihasilkan, dengan tujuan utama untuk meminimalkan risiko halusinasi atau kesalahan interpretasi yang mungkin timbul dari model kecerdasan buatan. Pendekatan ini tidak hanya meningkatkan akurasi dan keandalan sistem, tetapi juga memastikan bahwa proses penilaian berjalan secara objektif dan dapat diandalkan dalam berbagai skenario aplikasi praktis. Arsitektur *Self*-RAG dapat dilihat pada **Gambar 10** berikut.



Gambar 10. Diagram Arsitektur Pipeline Self-RAG

2.5.1 Arsitektur Sistem

Arsitektur sistem ini didasarkan pada fondasi pemrosesan bahasa alami (*Natural Language Processing*), yang secara inovatif mengintegrasikan kemampuan model bahasa besar (*Large Language Model*) dengan basis data vektor untuk mencapai pemahaman dan pemrosesan informasi yang lebih akurat serta efisien. Secara keseluruhan, alur kerja sistem ini terstruktur dalam tiga blok proses utama yang saling terkait, yaitu: (1) Konstruksi Basis Pengetahuan (*Knowledge Base Construction*), yang bertanggung jawab atas pembentukan dan pengorganisasian pengetahuan dasar yang diperlukan; (2) Mekanisme Temu Kembali (*Retrieval Mechanism*), yang memfasilitasi pencarian dan pengambilan informasi relevan dari basis data yang tersedia; serta (3) Pembangkitan Jawaban dengan Refleksi Diri (*Generative with Self-Reflection*), yang melibatkan proses generasi respons yang disertai dengan evaluasi internal untuk memastikan keakuratan dan keandalan hasil akhir. Pendekatan ini tidak hanya meningkatkan kemampuan sistem dalam menangani kompleksitas bahasa manusia, tetapi juga memastikan bahwa setiap tahapan proses berjalan secara sistematis dan terintegrasi, sehingga mendukung pengembangan solusi yang lebih canggih dalam bidang kecerdasan buatan.

1. Konstruksi Basis Pengetahuan (*Knowledge Base Construction*)

   Proses pembentukan basis pengetahuan ini dimulai dengan pengolahan dokumen rubrik penilaian yang telah dikonversi ke dalam format JSON untuk memfasilitasi manipulasi data yang lebih efisien. Mengingat keterbatasan jendela konteks (*context window*) yang melekat pada model bahasa besar (*Large Language Model*), data rubrik tidak diintegrasikan secara keseluruhan ke dalam prompt, melainkan melalui teknik *Atomic Chunking* yang cermat. Teknik ini melibatkan pemisahan setiap entitas pertanyaan dalam rubrik menjadi unit-unit terkecil (chunk) yang komprehensif, masing-masing mencakup pertanyaan spesifik, indikator penilaian, serta catatan tambahan yang relevan, sehingga memungkinkan pemrosesan yang lebih granular dan akurat.

   Selanjutnya, setiap chunk teks tersebut dikonversi menjadi representasi vektor (*vector embeddings*) dengan menggunakan model BAAI/bge-m3, yang dipilih berdasarkan kemampuan unggulnya dalam menangani representasi semantik multibahasa, termasuk Bahasa Indonesia, serta dukungannya terhadap panjang sekuens yang memadai untuk menjaga integritas informasi. Vektor-vektor yang dihasilkan dari proses ini kemudian disimpan dan diindeks secara sistematis menggunakan FAISS (Facebook AI *Similarity Search*), sebuah alat yang, sebagaimana dijelaskan oleh Johnson et al. (2017), memungkinkan pencarian kesamaan (*similarity search*) pada data berdimensi tinggi dengan efisiensi komputasi yang optimal, sehingga memungkinkan sistem untuk mengidentifikasi rubrik yang relevan dalam waktu yang sangat singkat, yaitu dalam hitungan milidetik. Pendekatan ini tidak hanya meningkatkan kecepatan akses data tetapi juga memastikan bahwa basis pengetahuan dapat diandalkan sebagai sumber acuan yang akurat dan responsif dalam konteks aplikasi praktis.

2. Alur Pemrosesan Permintaan (*Inference Pipeline*)

   Sistem ini menerima masukan dalam bentuk pasangan teks, yaitu Pertanyaan Wawancara dan Jawaban Kandidat, yang dikirimkan melalui antarmuka pengguna. Sebelum memasuki tahap penilaian inti, sistem menjalankan serangkaian validasi dan pengambilan konteks yang ketat untuk memastikan integritas proses, yang meliputi langkah-langkah berikut:

   - Pemeriksaan Relevansi (*Relevance Check*)

     Pada tahap ini, sistem melakukan validasi untuk menentukan apakah jawaban kandidat menunjukkan korelasi logis yang kuat dengan pertanyaan yang diajukan. Langkah pencegahan ini dirancang untuk menghindari model memberikan skor pada jawaban yang tidak relevan atau keluar dari topik (*out of topic*), sehingga menjaga objektivitas dan akurasi penilaian secara keseluruhan.

   - Proses *Retrieval*

     Sistem melaksanakan pencarian rubrik yang paling sesuai dengan pertanyaan input melalui mekanisme hibrida yang canggih. Mekanisme ini

memprioritaskan pencarian kesamaan teks eksak (*exact match*) sebagai langkah awal, yang memungkinkan identifikasi langsung jika ada kecocokan sempurna. Apabila tidak ditemukan, sistem secara otomatis beralih ke pencarian semantik dengan memanfaatkan jarak vektor (*cosine similarity*) pada indeks FAISS untuk menemukan rubrik dengan makna terdekat. Konsep ini mengadopsi prinsip-prinsip *Retrieval Augmented Generation* (RAG) yang diperkenalkan oleh Lewis et al. (2020), di mana model generatif diperkaya dengan fakta eksternal sebelum menghasilkan respons, sehingga meningkatkan kedalaman dan keandalan inferensi.

- Penyaringan Konteks (*Context Filtering*)

  Setelah rubrik yang relevan diidentifikasi, sistem mengevaluasi secara kritis apakah catatan tambahan (*notes*) atau indikator perlu disertakan dalam prompt. Evaluasi ini dilakukan untuk menjaga efisiensi penggunaan token dan memfokuskan perhatian model secara eksklusif pada informasi yang paling krusial, sehingga mengoptimalkan performa komputasi tanpa mengorbankan kualitas analisis.

3. Pembangkitan Jawaban dan Refleksi Diri (*Generation & Self-Correction*)

   Tahap ini mewakili inti dari kecerdasan adaptif sistem, di mana model bahasa Qwen-2.5-7B-Instruct yang telah dikuantisasi dalam format GGUF berfungsi sebagai mesin inferensi utama. Proses penilaian dilaksanakan melalui dua fase yang saling melengkapi:

   - Penilaian Awal (Initial Scoring dengan CoT)

     Sistem merancang instruksi (*prompt*) yang menempatkan model dalam persona "Senior HR *Recruiter*", sehingga memungkinkan pendekatan yang lebih profesional dan kontekstual. Teknik *Chain-of-Thought* (CoT), sebagaimana diteliti oleh Wei et al. (2022), diterapkan dengan meminta model untuk melakukan analisis langkah demi langkah (*reasoning*) secara mendalam sebelum menetapkan skor numerik. Dalam proses ini, model diinstruksikan untuk membandingkan jawaban kandidat dengan indikator rubrik secara kritis, memastikan bahwa setiap langkah evaluasi didasarkan pada analisis yang logis dan terstruktur.

   - Pemeriksaan Konsistensi Mandiri (Mekanisme *Self-Correction*)

     Hasil penilaian awal, yang mencakup skor dan alasan, tidak diterima secara langsung tanpa verifikasi tambahan. Sistem mengimplementasikan proses *Self*-RAG dengan mengaktifkan persona "*Quality Control*", di mana model diminta untuk mengevaluasi outputnya sendiri secara reflektif: apakah skor yang diberikan selaras dengan alasan yang dikemukakan? Apakah alasannya didukung oleh rubrik yang relevan? Jika terdeteksi inkonsistensi, model akan merevisi penilaian tersebut secara otomatis, sehingga meningkatkan validitas dan reliabilitas hasil akhir. Pendekatan reflektif ini dirancang untuk meminimalkan bias dan kesalahan,

memastikan bahwa penilaian tidak hanya akurat tetapi juga konsisten dalam berbagai skenario.

Keluaran akhir dari sistem ini berupa objek JSON yang komprehensif, yang mencakup skor final, alasan penilaian yang terperinci, serta saran perbaikan (improvement) yang konstruktif, yang kemudian dikirimkan kembali ke antarmuka pengguna melalui API untuk memfasilitasi interaksi yang mulus dan informatif. Pendekatan ini secara keseluruhan menekankan integrasi teknologi canggih dengan prinsip-prinsip evaluasi yang ketat, sehingga menghasilkan sistem yang tidak hanya efisien tetapi juga dapat diandalkan dalam konteks penilaian wawancara otomatis.

2.5.2 Perancangan Basis Data Vektor

Perancangan basis data vektor dirancang dengan tujuan utama untuk mengubah data rubrik yang awalnya bersifat tekstual menjadi format representasi numerik yang dapat diinterpretasikan secara efektif oleh sistem komputasi. Transformasi ini memainkan peran yang sangat penting dalam arsitektur *Retrieval Augmented Generation* (RAG), di mana keberhasilan proses *retrieval* secara signifikan ditentukan oleh cara data diorganisasikan dan diposisikan dalam ruang vektor multidimensi. Dengan demikian, kualitas pengambilan informasi yang relevan sangat bergantung pada akurasi dan efisiensi organisasi data tersebut, yang memungkinkan mesin untuk melakukan pencarian kesamaan semantik dengan presisi tinggi. Tahapan ini secara keseluruhan terstruktur dalam dua proses utama yang saling melengkapi, yaitu *Atomic Chunking*, yang melibatkan pemecahan data tekstual menjadi unit-unit terkecil untuk memfasilitasi pemrosesan yang lebih granular, dan *Vector Embedding*, yang mengonversi unit-unit tersebut menjadi vektor numerik yang mewakili makna semantiknya. Pendekatan ini tidak hanya meningkatkan kemampuan sistem dalam menangani kompleksitas data tekstual tetapi juga memastikan bahwa arsitektur RAG dapat beroperasi dengan optimal, sehingga mendukung inferensi yang lebih andal dan akurat dalam konteks aplikasi kecerdasan buatan.

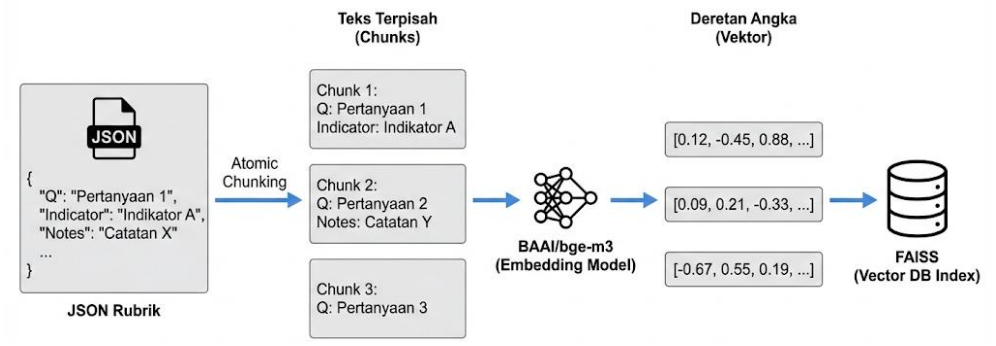
1. *Atomic Chunking* Data Rubrik

   Data rubrik yang diperoleh dari file JSON tidak diperlakukan sebagai satu kesatuan dokumen yang monolitik dan ekstensif, melainkan dipecah secara metodis menjadi unit-unit terkecil yang berdiri sendiri, yang dikenal sebagai chunk. Strategi *atomic chunking* ini diimplementasikan untuk menjamin bahwa setiap segmen data secara eksklusif mencakup satu konteks penilaian yang spesifik, sehingga menghindari potensi tumpang tindih atau interpretasi yang tidak jelas. Berdasarkan kode implementasi sistem yang telah dikembangkan, setiap chunk dibangun dengan format gabungan yang mencakup tiga elemen utama: teks pertanyaan (*Question*), indikator penilaian (*Indicator*), dan catatan tambahan (*Notes*). Integrasi elemen-elemen ini dirancang untuk memastikan

bahwa setiap vektor yang dihasilkan selanjutnya memiliki konteks semantik yang lengkap, koheren, dan bebas dari ambiguitas, yang pada akhirnya meningkatkan keakuratan dalam proses transformasi dan analisis data. Pendekatan ini tidak hanya memfasilitasi pemrosesan yang lebih efisien dan terstruktur tetapi juga memungkinkan sistem untuk menangani variasi kompleksitas dalam data rubrik dengan presisi yang lebih tinggi, memastikan bahwa setiap aspek evaluasi dapat dieksplorasi secara independen tanpa intervensi dari konteks eksternal.

2. *Vector Embedding* dan Penyimpanan Indeks

Transformasi teks ke dalam representasi vektor dilakukan melalui pemanfaatan model embedding BAAI/bge-m3, yang dikembangkan oleh Beijing Academy of Artificial Intelligence (BAAI) dengan varian M3. Xiao et al. (2024) dalam dokumentasi teknis mereka menjelaskan bahwa BGE-M3 menonjol dalam kemampuan *multi-linguality* dan *multi-granularity*, yang berarti model ini dapat menangkap nuansa makna dalam kalimat yang bervariasi—baik yang singkat maupun yang panjang—dengan tingkat presisi yang luar biasa, sehingga sangat sesuai untuk aplikasi yang melibatkan bahasa dan konteks yang heterogen. Dalam penelitian ini, model tersebut digunakan untuk mengonversi setiap chunk rubrik menjadi vektor padat (*dense vector*), yang merepresentasikan esensi semantik dalam bentuk numerik yang dapat dioperasikan oleh algoritma komputasi. Proses embedding dan indexing rubrik dapat dilihat pada **Gambar 11** berikut.



Gambar 11. Proses Embedding dan Indexing Rubrik

Vektor-vektor yang dihasilkan dari proses embedding ini selanjutnya dinormalisasi (*normalize embeddings*) untuk menjaga konsistensi skala dan mencegah distorsi dalam perhitungan kesamaan, sebelum akhirnya disimpan ke dalam basis data vektor. Penyimpanan dan pengindeksan dilakukan dengan memanfaatkan pustaka FAISS (*Facebook AI Similarity Search*) menggunakan tipe indeks `IndexFlatL2`, yang dirancang untuk performa yang optimal. Johnson et al. (2017) menyatakan bahwa `IndexFlatL2` mengandalkan perhitungan jarak Euclidean (L2) yang sangat akurat untuk mengukur kedekatan antar vektor, sehingga

memungkinkan evaluasi yang tepat terhadap hubungan spasial dalam ruang multidimensi. Dengan normalisasi yang telah diterapkan terlebih dahulu, perhitungan jarak L2 ini secara matematis ekuivalen dengan *Cosine Similarity*, yang terbukti efektif dalam menentukan derajat kemiripan semantik antara pertanyaan pengguna dan rubrik yang tersimpan dalam basis data. Pendekatan ini secara keseluruhan memperkuat kemampuan sistem untuk melakukan retrieval yang cepat, akurat, dan andal, memastikan bahwa proses pencarian informasi berlangsung dengan efisiensi maksimal dalam lingkungan komputasi yang kompleks dan dinamis.

2.5.3 Mekanisme *Retrieval Augmented Generation* (RAG)

Mekanisme *Retrieval Augmented Generation* (RAG) dikembangkan dengan tujuan utama untuk menjembatani kesenjangan yang signifikan antara pengetahuan umum yang dimiliki oleh *Large Language Model* (LLM) dengan standar penilaian yang spesifik dan unik milik suatu organisasi. Pendekatan ini tidak hanya berfokus pada pencarian data secara sederhana, melainkan melibatkan serangkaian proses validasi konteks yang ketat sebelum menghasilkan penilaian akhir, sehingga memastikan bahwa output yang dihasilkan lebih akurat, relevan, dan selaras dengan kriteria evaluasi yang ditetapkan. Alur kerja mekanisme ini secara sistematis mencakup tiga tahap utama: *Hybrid Retrieval*, yang mengintegrasikan metode pencarian eksak dan semantik untuk mengoptimalkan pengambilan informasi; *Dynamic Context Selection*, yang secara adaptif memilih dan menyaring konteks yang paling krusial; serta *Chain-of-Thought Prompting*, yang memandu model untuk melakukan reasoning langkah demi langkah guna meningkatkan kedalaman analisis. Dengan demikian, mekanisme ini memperkuat integritas proses penilaian dalam konteks aplikasi kecerdasan buatan yang kompleks.

1. *Hybrid Retrieval* (Pencarian Hibrida)

   Sistem menerima masukan berupa pasangan pertanyaan wawancara dan jawaban kandidat. Proses pencarian rubrik dilakukan dengan strategi hibrida untuk memaksimalkan akurasi. Pertama, sistem melakukan pencarian pencocokan eksak (*exact match*) dengan menormalisasi teks pertanyaan (menghapus simbol non-alfanumerik) untuk menemukan rubrik yang memiliki redaksi pertanyaan yang identik. Jika pencarian eksak tidak menemukan hasil, sistem beralih ke pencarian semantik (*semantic search*) menggunakan basis data vektor FAISS. Metode ini memungkinkan sistem tetap menemukan rubrik yang relevan meskipun terdapat sedikit perbedaan kata (*typo* atau sinonim) antara input pengguna dan database rubrik.
2. Seleksi Konteks Dinamis (*Dynamic Context Selection*)

   Setelah rubrik yang paling relevan ditemukan, sistem tidak serta-merta memasukkan seluruh informasi rubrik ke dalam *prompt*. Sistem menjalankan fungsi cerdas `need_context_fields` yang memanfaatkan LLM kecil untuk menganalisis apakah jawaban kandidat memerlukan konteks tambahan berupa "Notes" atau "Indicator". Langkah ini dilakukan untuk efisiensi token dan mengurangi *noise*. Jika jawaban kandidat sudah sangat jelas, indikator

tambahan mungkin tidak diperlukan. Sebaliknya, jika jawaban mengandung ambiguitas, sistem akan menyertakan "Notes" dari rubrik untuk memperluas pemahaman (*context widening*) bagi model penilai.

3. Konstruksi Prompt dengan *Chain-of-Thought* (CoT)

   Tahap akhir dari mekanisme ini adalah penyusunan instruksi (*prompt building*) yang akan dikirim ke model utama (Qwen-2.5). Teknik *Chain-of-Thought* (CoT) diterapkan dalam struktur *prompt* untuk memandu model melakukan penalaran bertahap. Wei dkk. (2022) membuktikan bahwa CoT secara signifikan meningkatkan kemampuan model dalam menyelesaikan tugas penalaran kompleks. Dalam implementasinya, *prompt* dirancang agar model mengeluarkan tag `<analisis>` terlebih dahulu—di mana model membedah jawaban kandidat kata demi kata—sebelum akhirnya memutuskan skor dalam format JSON. Keluaran teks dari model kemudian di-*parsing* menggunakan ekspresi reguler (*Regex*) untuk mengekstraksi objek JSON yang berisi skor, alasan, dan saran, yang siap ditampilkan kepada pengguna.

2.6 Pengujian dan Evaluasi Sistem

Evaluasi sistem dilakukan untuk menilai tingkat efektivitas arsitektur *Self*-RAG yang dikembangkan dalam mereplikasi kemampuan penilaian yang dilakukan oleh pakar manusia (*human expert*). Proses evaluasi ini disusun ke dalam tiga aspek utama, yaitu perancangan skenario pengujian yang melibatkan pakar, pengukuran kinerja sistem berdasarkan metrik klasifikasi statistik, serta evaluasi terhadap kinerja komponen *retrieval* dan *generation* yang membentuk keseluruhan sistem.

2.6.1 Skenario Pengujian

Pengujian sistem dilakukan melalui pendekatan *expert judgment* dengan melibatkan perekrut senior dari organisasi AIESEC sebagai evaluator. Pendekatan ini bertujuan untuk memperoleh label kebenaran dasar (*ground truth*) yang valid serta masukan kualitatif terkait kesesuaian dan ketepatan alasan penilaian (*reasoning*) yang dihasilkan oleh sistem berbasis kecerdasan buatan.

Pelaksanaan pengujian menggunakan instrumen berupa lembar kerja kolaboratif (Google Sheets) yang memuat 40 sampel jawaban kandidat hasil pemrosesan sistem. Rancangan mekanisme pengujian disusun ke dalam beberapa tahapan sebagai berikut:

1. Distribusi Sampel

   Data uji terdiri atas jawaban kandidat yang mencakup berbagai variasi pertanyaan wawancara. Setiap pertanyaan disertai sejumlah sampel jawaban untuk merepresentasikan rentang kualitas jawaban yang beragam, mulai dari kategori kurang memuaskan hingga sangat memuaskan. Sampel data uji untuk *expert* bisa dilihat pada **Tabel 7** dan versi lengkap dapat dilihat di **Error! Reference source not found.**.

Tabel 7. Instrumen pengujian *expert*

ID Sample Jawaban		**SJ-01**		
Pertanyaan	Jawaban	Skor	*Reason*	*Improvement*
Apa nilai nilai (values) yang penting dalam hidupmu? Jelaskan!	Saya punya kalimat yang menjadi motivasi saya dan juga menjadi nilai hidup saya, yaitu "Orang terbaik di dunia ini adalah orang yang paling bermanfaat bagi banyak orang", jadi saya ingin berkontribusi utnuk menberi dampak positif ke orang orang di sekitar saya dan mendorong saya untuk menjadi lebih baik dari hari ke hari	3	Jawaban kandidat menjelaskan dengan jelas nilai-nilai pribadi dan memberikan contoh nyata bagaimana nilai tersebut diterapkan dalam kehidupan sehari-hari.	Kandidat bisa menambahkan lebih banyak contoh konkret tentang bagaimana nilai tersebut mempengaruhi keputusan dan sikapnya dalam kehidupan sehari-hari.
Apa tiga kelemahan/ kekurangan teratasmu dan bagaimana kamu mengatasinya? Jelaskan!	pertama adalah Time management, saya cukup keteteran dalam mengatur time managementnya, saat saya mendapat banyak tugas dan tanggung jawab, saya terpaksa mengerjakan beberapa tanggung jawab atau tugasnya mendekati deadline. Namun, saya berusaha untuk mengatur waktunya menjadi lebih efisien. Kekurangan saya selanjutnya adalah mengendalikan mood saya, teman teman saya bisa mengetahui perasaan dan apa yang saya katakan tanpa saya mengatakannya, karena teman saya bisa membaca perasaan saya lewat ekspresi saya	2	Kandidat menyebutkan dua kelemahan dan menjelaskannya dengan baik, yaitu kelemahan dalam manajemen waktu dan kemampuan mengendalikan mood. Namun, jawaban kurang mencakup semua tiga kelemahan yang diminta.	Kandidat dapat menambahkan satu kelemahan lainnya dan menjelaskannya secara lebih detail.

2. Validasi Skor Otomatis

   Evaluator diminta untuk meninjau skor yang dihasilkan oleh sistem (Skor AI). Apabila evaluator menyetujui skor tersebut, kolom koreksi tidak perlu diisi. Sebaliknya, apabila skor dinilai tidak sesuai, evaluator diminta memberikan nilai yang dianggap tepat pada kolom Skor Koreksi. Mekanisme ini dirancang untuk meningkatkan efisiensi waktu penilaian sekaligus mengidentifikasi perbedaan penilaian antara sistem dan pakar manusia.

3. Validasi Alasan Penilaian

   Selain penilaian numerik, evaluator juga melakukan penilaian terhadap narasi alasan penilaian (*reasoning*) yang dihasilkan sistem. Evaluator memberikan status "Ya" apabila alasan dinilai telah sesuai dengan rubrik penilaian, dan "Tidak" apabila alasan dianggap tidak relevan atau mengandung halusinasi, disertai dengan catatan perbaikan pada kolom yang telah disediakan.

4. Kuesioner Akhir

   Setelah seluruh sampel dinilai, evaluator mengisi kuesioner evaluasi kinerja sistem yang bertujuan untuk mengukur aspek kegunaan (*usability*) dan tingkat kepuasan pengguna secara kualitatif.

5. Studi Ablasi

   Untuk memvalidasi efektivitas mekanisme Self-Correction yang menjadi kebaruan (novelty) dalam penelitian ini, dilakukan uji komparatif melalui metode Studi Ablasi. Pengujian ini bertujuan mengisolasi kontribusi spesifik dari modul kritik mandiri terhadap akurasi akhir. Skenario pengujian dibagi menjadi dua konfigurasi sistem yang dijalankan pada dataset yang sama:

   - Konfigurasi A (Baseline - Naive RAG)

     Pada skenario ini, modul *self-corection* dinonaktifkan. Sistem beroperasi secara linear (Input → Retrieval → Generation → Output). Output pertama yang dihasilkan oleh LLM langsung dianggap sebagai nilai akhir tanpa melalui proses verifikasi atau revisi. Konfigurasi ini merepresentasikan performa sistem RAG standar tanpa mekanisme refleksi. Secara teknis, fungsi self_consistency_check dalam kode program dinonaktifkan (bypassed), sehingga output JSON yang dihasilkan pada tahap generasi awal langsung dicatat sebagai hasil akhir tanpa melalui proses verifikasi oleh node auditor. Konfigurasi ini merepresentasikan performa sistem RAG standar tanpa mekanisme refleksi, yang digunakan sebagai data pembanding (baseline).

   - Konfigurasi B (Proposed Method - Self-Reflective RAG).

     Pada skenario ini, sistem dijalankan secara utuh dengan mengaktifkan modul *self-corection*. Model bertindak sebagai auditor yang melakukan kritik terhadap draf penilaian awal dan melakukan perbaikan (iterative refinement) jika ditemukan inkonsistensi antara skor dan alasan.

   Perbandingan nilai akurasi (QWK) antara Konfigurasi A dan Konfigurasi B akan menunjukkan secara kuantitatif seberapa besar mekanisme *Self-*

Correction berkontribusi dalam mengurangi halusinasi dan meningkatkan validitas penilaian.

2.6.2 Evaluasi Kinerja Klasifikasi

Evaluasi kinerja klasifikasi diarahkan untuk menilai tingkat kesesuaian antara hasil penilaian yang dihasilkan oleh sistem otomatis dan penilaian yang diberikan oleh pakar. Mengingat skor wawancara direpresentasikan dalam bentuk data ordinal dengan skala bertingkat (0, 1, 2, 3, dan 4), penggunaan metrik akurasi konvensional dinilai kurang memadai karena tidak mempertimbangkan jarak antar tingkat penilaian. Oleh sebab itu, penelitian ini menerapkan metrik *Quadratic Weighted Kappa* (QWK), yaitu pengembangan dari *Cohen's Kappa* dengan skema pembobotan kuadratik.

Pemilihan metrik QWK didasarkan pada kemampuannya dalam memberikan bobot kesalahan yang proporsional terhadap jarak perbedaan skor. Sebagai ilustrasi, prediksi sistem dengan skor 4 terhadap nilai sebenarnya 3 dikenakan penalti yang lebih rendah dibandingkan prediksi skor 4 terhadap nilai sebenarnya 0. Cohen (1968) menegaskan bahwa pembobotan kuadratik sangat sesuai digunakan dalam pengukuran reliabilitas antar penilai (*inter-rater reliability*) pada data berskala ordinal, karena mampu merepresentasikan tingkat keparahan kesalahan klasifikasi secara lebih akurat.

Sebelum perhitungan dilakukan, data hasil pengujian melalui proses pra-pemrosesan (*data cleaning*) untuk menetapkan nilai *Ground Truth* ($\boldsymbol{y_{true}}$) dan Prediksi ($\boldsymbol{y_{pred}}$) dengan logika sebagai berikut:

- Data Prediksi ($\boldsymbol{y_{pred}}$): Diambil langsung dari kolom "Skor AI".
- Data Kebenaran ($\boldsymbol{y_{true}}$): Diambil dari kolom "Skor Koreksi". Apabila kolom "Skor Koreksi" kosong (*Expert* setuju), maka nilai ($\boldsymbol{y_{true}}$) akan disamakan dengan nilai ($\boldsymbol{y_{pred}}$). Apabila terisi, maka nilai koreksi tersebut yang digunakan.

Untuk memperjelas mekanisme penentuan nilai kebenaran tersebut, ilustrasi logika konversi data disajikan pada **Tabel 8**.

Tabel 8. Logika Penentuan Ground Truth

ID Sampel	ID Expert	Pertanyaan	Skor AI (y_{pred})	Input *Expert* (Skor Koreksi)	*Ground Truth* (y_{pred})	Keterangan Logika
SJ-05	EP-01	Bagaimana caramu menghadapi orang-orang yang memiliki keyakinan, pendapat, atau prinsip yang	3	-	3	Expert setuju, nilai diambil dari Skor AI.

		berbeda darimu?				
SJ-01	EP-04	Apa nilai nilai (values) yang penting dalam hidupmu? Jelaskan!	3	-	3	Expert setuju, nilai diambil dari Skor AI.
SJ-02	EP-04	Apa tiga kelebihan teratasmu/terkuatmu? Jelaskan!	2	0	0	*Expert* mengoreksi, nilai diambil dari Input Expert.

Nilai QWK berkisar antara -1 hingga 1, di mana nilai di atas 0.81 menunjukkan kesepakatan yang "hampir sempurna" (*almost perfect agreement*), sedangkan nilai di bawah 0.40 menunjukkan kesepakatan yang buruk. Rumus implementasi menggunakan pustaka Scikit-learn adalah sebagai berikut:

$$\kappa = 1 - \frac{\sum_{i,j} w_{i,j} O_{i,j}}{\sum_{i,j} w_{i,j} E_{i,j}} \quad (7)$$

Dimana $w_{i,j}$ adalah bobot kuadratik yang diberikan untuk selisih antara kategori i dan j.

Penginterpretasian nilai koefisien Kappa (κ) yang diperoleh dari perbandingan penilaian sistem dan ahli akan mengacu pada standar interpretasi yang dikemukakan oleh Landis dan Koch (1977). Standar ini membagi tingkat kesepakatan (agreement) menjadi enam tingkatan hierarkis yang dapat dilihat pada **Tabel 9** berikut.

Tabel 9. Interpretasi Nilai Koefisien Kappa

Nila Kappa (κ)	**Tingkat Kesepakatan** (*Strength of Agreement*)
$\kappa < 0.00$	Poor (Buruk / Tidak Ada Kesepakatan)
$0.00 \leq \kappa \leq 0.20$	Slight (Sangat Lemah)
$0.20 < \kappa \leq 0.40$	Fair (Cukup / Lemah)
$0.40 < \kappa \leq 0.60$	Moderate (Sedang)
$0.60 < \kappa \leq 0.80$	Substantial (Kuat / Bagus)
$0.80 < \kappa \leq 1.00$	Almost Perfect (Hampir Sempurna)

Sumber 1 : (Landis & Koch, 1977)

2.5.4 Implementasi *Pipeline Self*-RAG

Implementasi pipeline *Self*-RAG (*Self-Reflective Retrieval Augmented Generation*) merupakan elemen fundamental dalam sistem yang dirancang untuk memastikan tingkat validitas dan reliabilitas yang tinggi pada proses penilaian otomatis. Berbeda dengan pendekatan *Retrieval Augmented Generation* (RAG) konvensional yang umumnya hanya berfokus pada pengambilan informasi relevan dan menghasilkan

jawaban secara langsung, pendekatan *Self*-RAG mengintegrasikan tahapan evaluasi internal yang lebih komprehensif. Mekanisme ini dirancang untuk meniru proses kognitif manusia, khususnya pada aspek *metacognition*, yaitu kemampuan sistem untuk merefleksikan, menilai, dan memverifikasi kembali hasil pemikirannya sendiri sebelum menghasilkan keluaran akhir.

Secara konseptual, alur implementasi *Self*-RAG dibangun di atas dua strategi utama. Strategi pertama adalah rekayasa instruksi berbasis *Chain-of-Thought*, yang memungkinkan model menalar secara bertahap dan terstruktur dalam menyusun jawaban berdasarkan konteks yang diperoleh dari proses *retrieval*. Strategi kedua adalah penerapan mekanisme *self-corection*, yaitu proses pemeriksaan konsistensi mandiri yang berfungsi untuk mengevaluasi keselarasan antara berbagai hasil penalaran yang dihasilkan model. Dengan mengombinasikan kedua strategi tersebut, *Self*-RAG tidak hanya meningkatkan akurasi jawaban, tetapi juga memperkuat keandalan sistem melalui proses refleksi dan verifikasi internal yang sistematis.

1. **Rekayasa Instruksi (*Prompt Engineering*) dengan *Chain-of-Thought***

   Kualitas keluaran yang dihasilkan oleh model bahasa besar sangat ditentukan oleh perancangan instruksi atau prompt yang digunakan. Dalam penelitian ini, diterapkan teknik *Chain-of-Thought* (CoT) sebagaimana diperkenalkan oleh Wei et al. (2022), yang mendorong model untuk mengekspresikan tahapan penalaran secara eksplisit sebelum menghasilkan jawaban akhir. Pendekatan ini telah dibuktikan secara empiris mampu meningkatkan kemampuan penalaran logis model dalam menyelesaikan permasalahan yang bersifat kompleks, jika dibandingkan dengan metode standard prompting yang hanya mengandalkan satu tahap respons langsung.

   Implementasi *Chain-of-Thought* pada sistem ini diawali dengan penetapan persona (*role-play*) sebagai landasan konteks penalaran model. Melalui System Prompt pada kode program, model diarahkan untuk berperan sebagai seorang “Senior *HR Recruiter*” yang bersikap objektif, analitis, dan kritis dalam melakukan penilaian. Selanjutnya, struktur prompt dirancang secara terstruktur dan sistematis dengan memuat tiga komponen utama yang dapat dilihat pada **Tabel 10**, Dimana ketiga komponen tersebut berperan penting dalam membimbing proses penalaran model secara bertahap dan terkontrol, yaitu :

   - Konteks Data

     Menyajikan pertanyaan wawancara, jawaban kandidat, dan indikator rubrik yang telah diambil dari basis data vektor.
   - Instruksi Berjenjang

     Meminta model untuk tidak langsung memberikan skor. Model diwajibkan melakukan analisis naratif terlebih dahulu—membandingkan jawaban kandidat dengan indikator kata demi kata—baru kemudian mengonversi hasil analisis tersebut menjadi angka skor.
   - Format Luaran Terstruktur

Model diminta membungkus proses berpikirnya dalam tag XML `<analisis>` dan hasil akhirnya dalam format JSON. Pemisahan ini memudahkan proses parsing dan memastikan bahwa skor yang dihasilkan didasarkan pada logika yang tertulis, bukan tebakan acak.

Tabel 10. Struktur Prompt Chain-of-Thought untuk Proses Penilaian Pertama

Komponen Prompt	Fungsi Logis	Contoh Instruksi dalam Kode
Penetapan Persona (*Role definition*)	Memberikan identitas dan keahlian spesifik kepada model agar penilaian bersifat objektif dan profesional.	"Anda adalah Senior HR Recruiter yang ahli menilai wawancara kerja secara objektif, kritis, dan adil."
Injeksi Konteks (*Context Injection*)	Menyajikan data dinamis yang diambil dari basis data (RAG) sebagai acuan penilaian yang valid.	"### KONTEKS Pertanyaan: {question}... Indikator Utama: {indicator}... Catatan Tambahan: {notes}"
Instruksi Berjenjang (*Step-by-step Reasoning*)	Menerapkan teknik *Chain-of-Thought* dengan memaksa model untuk menganalisis terlebih dahulu sebelum memberikan nilai angka.	"Lakukan langkah berikut: 1. ANALISIS: Bandingkan kata per kata... 2. PENENTUAN SKOR: Tentukan skor... 3. FORMAT OUTPUT: Buat kesimpulan..."
Batasan Luaran (*Output Constraint*)	Memastikan format jawaban model terstruktur agar dapat dibaca oleh sistem (*parsing*) tanpa error.	"Mulailah dengan tag <analisis> untuk coretan pemikiranmu, lalu akhiri dengan blok JSON... { "score": <angka>, ... }"

Struktur lengkap *prompt* dapat dilihat pada **Lampiran 6.**

Selain prompt penilai, sistem juga menggunakan prompt khusus untuk mekanisme *Self-Correction* yang menempatkan model dalam peran sebagai auditor. Struktur instruksi untuk tahap ini dirancang sebagai berikut:

Tabel 11. Struktur Prompt Chain-of-Thought untuk Proses *Self-Reflection*

Komponen Prompt	Fungsi Logis	Contoh Instruksi dalam Kode
Penetapan Persona (*Role definition*)	Mengubah peran model dari penilai menjadi pemeriksa kualitas.	"Anda adalah Quality Control (QC) untuk penilaian AI. Tugas Anda mengecek konsistensi penilaian."
Injeksi Konteks (*Context Injection*)	Menyajikan data dinamis yang diambil dari basis data dan hasil *generate output* penilaian pertama sebagai sebagai dasar dalam proses *Self-Reflection* dan *Self-Critique*.	“DATA PENILAIAN: - Pertanyaan: {question} - Jawaban: {answer} - Skor Diberikan: {initial_json.get('score')} - Alasan: "{initial_json.get('reason')}"

		- Indikator Rubrik: {rub.get('indicator', '-')}"
Instruksi Berjenjang (*Step-by-step Reasoning*)	Meminta model membandingkan skor dengan narasi alasan.	"CEK KONSISTENSI: 1. Apakah skor {initial_json.get('score')} masuk akal untuk alasan tersebut? 2. Apakah skor dan alasan sudah mengacu pada Indikator dan Skor Level Rubrik? Jika SUDAH BAGUS, jawab hanya dengan kata: VALID Jika ADA MASALAH, perbaiki dan berikan JSON baru. "
Trigger Koreksi	Memberikan opsi biner untuk memaksa model mengambil keputusan tegas.	" Jika SUDAH BAGUS, jawab hanya dengan kata: VALID Jika ADA MASALAH, perbaiki dan berikan JSON baru."

Struktur lengkap *prompt* dapat dilihat pada **Lampiran 6.**

2. Mekanisme Pemeriksaan Konsistensi (*Consistency Check Loop*)

   Halusinasi dan kesalahan penalaran logis merupakan permasalahan krusial dalam pemanfaatan *Large Language Models* (LLM). Asai et al. (2023), melalui kajiannya mengenai *Self*-RAG, merekomendasikan penerapan mekanisme reflektif yang memungkinkan model melakukan evaluasi kritis terhadap keluarannya sendiri (*self-critique*). Konsep ini dalam penelitian ini diimplementasikan ke dalam sebuah fungsi bernama `self_consistency_check`, yang berperan sebagai mekanisme pengendalian mutu (*quality control*) dan dijalankan secara otomatis setelah model menghasilkan penilaian awal. Struktur pemeriksaan konsistensi bisa dilihat pada struktur prompt *Chain-of-Thought* di **Gambar 12**.

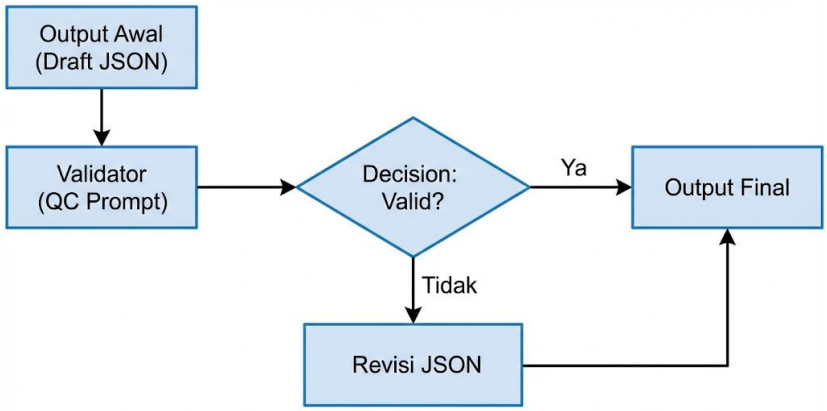
   Secara operasional, mekanisme tersebut dilakukan melalui pemanggilan ulang LLM menggunakan *prompt* yang berbeda dari tahap sebelumnya. Pada fase ini, peran model dialihkan dari sebagai penilai menjadi "Auditor Kualitas" yang bertugas melakukan verifikasi. Sistem mengirimkan kembali skor serta argumentasi yang dihasilkan pada tahap evaluasi awal, kemudian meminta model untuk melakukan validasi terhadap dua aspek utama, yaitu:

   - Validitas Format

     Sistem memeriksa apakah luaran dapat dibaca sebagai objek JSON yang valid menggunakan pustaka *Regular Expression* (Regex). Jika format rusak, sistem akan meminta regenerasi.

- Konsistensi Logika

  Model auditor mengevaluasi apakah skor yang diberikan (misalnya: skor 4) selaras dengan alasan yang ditulis (misalnya: "Jawaban kandidat sempurna"). Jika ditemukan ketidaksesuaian—seperti alasan yang memuji tapi skor rendah, atau sebaliknya—model akan melakukan revisi otomatis (*self-correction*) dan menghasilkan objek JSON baru yang lebih akurat.



Gambar 12. Struktur Prompt Chain-of-Thought

Mekanisme looping perbaikan ini memastikan bahwa hasil akhir yang diterima pengguna telah melalui dua lapisan verifikasi, sehingga meminimalkan risiko penilaian yang bias atau tidak rasional.

2.6.3 Evaluasi Komponen *Self*-RAG

Selain melakukan penilaian terhadap keluaran akhir berupa skor, penelitian ini turut mengevaluasi kinerja internal dari arsitektur *Self-RAG (Self-Reflective Retrieval Augmented Generation)* guna memastikan bahwa sistem beroperasi sesuai dengan mekanisme dan logika yang telah dirancang. Evaluasi terhadap komponen ini dilakukan dengan menggunakan dua metrik utama, yaitu *Hit Rate* dan *Faithfulness*, yang masing-masing merepresentasikan kinerja proses pengambilan informasi dan kesesuaian hasil generasi dengan sumber rujukan.

1. *Hit Rate* (Tingkat Keberhasilan Temu Kembali)

   Hit Rate digunakan sebagai indikator untuk menilai kinerja modul *retriever* dalam mengidentifikasi dan mengambil dokumen rubrik penilaian yang relevan. Dalam konteks penelitian ini, *Hit Rate* didefinisikan sebagai persentase keberhasilan sistem dalam melakukan proses pengambilan (*retrieval*) rubrik penilaian yang sesuai dengan identitas pertanyaan (ID) yang sedang diproses.

   Suatu proses *retrieval* dinyatakan berhasil (*hit*) apabila dokumen rubrik yang digunakan sebagai konteks dalam *prompt* memiliki kesesuaian, baik dari segi ID maupun konten pertanyaan, dengan pertanyaan masukan. Sebaliknya, kegagalan pada tahap ini berpotensi menimbulkan kesalahan berantai

(*cascading error*), di mana hasil penilaian menjadi tidak akurat akibat penggunaan acuan rubrik yang tidak sesuai.

2. *Faithfullness* (Kesetiaan Penalaran)

   Faithfulness digunakan untuk menilai tingkat ketepatan sistem dalam menghasilkan alasan penilaian (*reasoning*) yang benar-benar didasarkan pada konteks rubrik yang disediakan, bukan berasal dari pengetahuan internal model yang berpotensi menimbulkan bias atau halusinasi. Es *dkk.* (2025), melalui kerangka kerja RAGAS, mendefinisikan *faithfulness* sebagai tingkat konsistensi antara keluaran yang dihasilkan oleh sistem dan konteks sumber yang digunakan. Dalam penelitian ini, nilai *faithfulness* dihitung menggunakan rumus sebagai berikut:

$$Faithfullness = \frac{Jumlah\ Alasan\ Valid}{Total\ Sampel\ Data} \times 100\% \quad (8)$$

Mengingat belum adanya standar baku industri untuk nilai ambang batas (*threshold*) *Self*-RAG, penelitian ini mengadopsi standar akurasi umum dalam sistem *Information Retrieval* dan NLP. Kinerja komponen Self-RAG dievaluasi berdasarkan persentase keberhasilan dengan kriteria interpretasi pada **Tabel 12** berikut:

Tabel 12. Kriteria performa komponen *Self*-RAG

Rentang Presentase	Kategori Performa	Interpretasi
$80\% < x \leq 100\%$	Sangat Baik (Excellent)	Sistem sangat dapat diandalkan dalam mengambil konteks dan menjaga konsistensi fakta.
$60\% < x \leq 80\%$	Baik (Good)	Sistem bekerja dengan baik namun memiliki sedikit ruang kesalahan yang dapat ditoleransi.
$40\% \leq x \leq 60\%$	Cukup (Fair)	Kinerja sistem pas-pasan, memerlukan perbaikan pada chunking atau prompting.
$< 40\%$	Kurang (Poor)	Sistem gagal menjalankan fungsi Self-RAG secara efektif.

Penelitian ini menargetkan *Hit Rate* minimal berada pada kategori **Sangat Baik (>80%)** karena kesalahan pengambilan rubrik akan berakibat fatal pada penilaian. Sedangkan untuk *Faithfulness*, target minimal adalah **Baik (>60%)**.

2.6.4 Penilaian Ahli (*Expert Judgment*)

Tahapan terakhir dalam evaluasi sistem dilaksanakan melalui pendekatan expert judgment. Evaluasi ini bertujuan untuk menilai kualitas keluaran sistem berdasarkan

sudut pandang profesional perekrut yang memiliki keahlian dan pengalaman dalam proses seleksi wawancara. Berbeda dengan pengujian fungsional, penilaian ahli menekankan pada aspek validitas konten, tingkat kepercayaan terhadap keputusan yang dihasilkan oleh sistem berbasis kecerdasan buatan (AI trust), serta kelayakan sistem untuk diimplementasikan dalam konteks operasional nyata.

Instrumen evaluasi yang digunakan berupa kuesioner digital berbasis Google Form yang diisi oleh para pakar setelah mereka menyelesaikan proses verifikasi terhadap seluruh sampel jawaban. Perancangan struktur kuesioner difokuskan untuk mengukur tiga dimensi utama kualitas sistem cerdas, yaitu kinerja sistem dalam melakukan penilaian, kualitas komponen Retrieval Augmented Generation (retrieval dan generation), serta tingkat utilitas dan dampak implementasi sistem dalam mendukung proses seleksi wawancara.

1. Desain instrument Kuesioner

   Instrumen evaluasi dirancang dengan mengombinasikan beberapa jenis pertanyaan, yaitu pertanyaan tertutup menggunakan Skala Likert empat tingkat (1–4) guna mendorong responden memberikan penilaian yang tegas tanpa opsi netral, pertanyaan kategorikal untuk mengidentifikasi frekuensi kejadian tertentu, serta pertanyaan terbuka yang bertujuan untuk menghimpun masukan dan evaluasi kualitatif. Berdasarkan instrumen yang disusun, indikator pengukuran selanjutnya dikelompokkan ke dalam beberapa kategori sebagai berikut:

   - Dimensi Kinerja Penilaian

     Dimensi ini bertujuan untuk menilai persepsi pakar terhadap tingkat akurasi skor yang dihasilkan oleh sistem jika dibandingkan dengan standar penilaian profesional yang mereka terapkan. Selain itu, dimensi ini juga mengevaluasi tingkat kesepakatan (*agreement rate*) secara umum, yakni seberapa sering pakar menerima hasil penilaian sistem tanpa perlu melakukan penyesuaian atau koreksi secara manual.

   - Dimensi Kualitas *Self*-RAG

     Dimensi ini digunakan untuk menilai transparansi dan validitas proses *Self-Retrieval Augmented Generation* (*Self*-RAG). Indikator yang dievaluasi mencakup kualitas logika penalaran (*reasoning*), yaitu apakah alasan penilaian yang dihasilkan bersifat rasional, relevan, dan mudah dipahami, serta ketepatan pengambilan konteks (*context retrieval*), yakni kemampuan sistem dalam mengaitkan jawaban kandidat dengan indikator rubrik yang sesuai. Selain itu, dimensi ini juga mencakup evaluasi terhadap aspek *faithfulness* dengan mengidentifikasi adanya potensi halusinasi atau informasi yang tidak didukung oleh konteks sumber dalam narasi yang dihasilkan sistem.

   - Dimensi Utilitas

     Dimensi utilitas berfokus pada penilaian tingkat kebermanfaatan sistem dalam konteks operasional. Aspek yang dievaluasi meliputi kualitas saran perbaikan (*improvement suggestion*), khususnya apakah

rekomendasi yang diberikan bersifat konstruktif dan dapat ditindaklanjuti (*actionable*). Selain itu, pakar juga diminta untuk menilai efisiensi waktu yang ditawarkan oleh sistem serta memberikan pandangan mengenai tingkat implementasi yang paling sesuai, mulai dari penggunaan sebagai alat bantu pendukung keputusan (*second opinion*) hingga penerapan dalam bentuk otomatisasi penuh.

2. Teknik Analisis Data

Data yang diperoleh dari penilaian ahli dianalisis menggunakan metode campuran (mixed-method) untuk mendapatkan gambaran komprehensif mengenai kinerja sistem.

a. Analisis Kuantitatif (Skala Likert)

Data dari pertanyaan berskala 1 sampai 4 (seperti Akurasi, Logika Penalaran, dan Kualitas Saran) dianalisis menggunakan teknik Persentase Kelayakan. Teknik ini mengonversi skor mentah menjadi persentase untuk menentukan tingkat kelayakan sistem berdasarkan kriteria yang telah ditetapkan. Rumus perhitungan yang digunakan adalah:

$$P = \frac{\sum x}{N \times i} \times 100\% \quad (9)$$

Keterangan :

- P = Persentase Kelayakan
- $\sum x$ = Total jumlah skor jawaban responden
- N = Jumlah responden
- i = Skor maksimal ideal (4)

Hasil perhitungan persentase selanjutnya dikoversikan ke dalam kategori kelayakan sebagaimana disajikan pada **Tabel 13**. Nilai rata-rata (Mean) juga dihitung untuk mengetahui kecenderungan pusat penilaian pada setiap indikator.

Tabel 13. Kriteria Interpretasi Skor Kelayakan

Persentase (%)	Kriteria Penilaian
$81.25 < x \leq 100$	Sangat Layak / Sangat Baik
$62.50 < x \leq 81.25$	Layak / Baik
$43.75 < x \leq 62.50$	Kurang Layak / Cukup
$25.00 \leq x \leq 43.75$	Tidak Layak / Kurang

b. Analisis Frekuensi (Pilihan Ganda)

Pertanyaan non-skala yang bersifat kategorikal, seperti tingkat halusinasi (Tidak Ada, Sedikit, atau Banyak) serta rekomendasi penerapan sistem, dianalisis dengan menggunakan distribusi frekuensi relatif. Pendekatan ini digunakan untuk menghitung persentase pakar pada setiap pilihan jawaban. Hasil analisis tersebut dimanfaatkan untuk menjawab pertanyaan evaluatif tertentu, seperti proporsi pakar yang merekomendasikan sistem untuk digunakan sebagai alat bantu penilaian

dibandingkan dengan penerapan dalam bentuk otomatisasi penuh. Analisis ini tidak menggunakan skor kelayakan, melainkan Tingkat Konsensus Mayoritas. Interpretasi didasarkan pada modus (pilihan terbanyak) dari jawaban para ahli. Interpretasi tingkat consensus ahli bisa dilihat pada **Tabel 14** berikut.

Tabel 14. Interpretasi Tingkat Konsensus Ahli

Persentase Pemilih Opsi	Interpretasi Konsensus
> 50% (Mayoritas)	Konsensus Tercapai. Opini tersebut mewakili pandangan umum para ahli.
≤ 50% (Minoritas)	Divergen / Variatif. Tidak ada kesepakatan tunggal, menunjukkan adanya preferensi yang beragam antar ahli.

c. Analisis Kualitatif (Isian Teks)

Masukan berupa data tekstual yang diperoleh dari kolom uraian terkait kelebihan dan kelemahan sistem dianalisis dengan menggunakan model analisis interaktif yang dikemukakan oleh Miles, Huberman, dan Saldaña (2014). Proses analisis tersebut mencakup tiga alur kegiatan utama yang berlangsung secara simultan, yaitu:

- Reduksi Data (Data Condensation)

  Proses pemilihan, pemusatan, penyederhanaan, dan transformasi data kasar dari catatan lapangan (jawaban *Google Form*). Data yang tidak relevan dibuang, sementara poin-poin kunci mengenai kinerja AI dipertahankan.
- Penyajian Data (Data Display)

  Menyusun sekumpulan informasi yang telah direduksi ke dalam matriks atau narasi tematik untuk memudahkan penarikan kesimpulan.
- Penarikan Kesimpulan (Conclusion Drawing)

  Menarik intisari dari umpan balik pakar untuk merumuskan rekomendasi perbaikan sistem di masa depan.