4. PENGUJIAN

Bagian ini akan membahas hasil pengujian terhadap sistem *chatbot* yang sudah dikembangkan. Pengujian dilakukan secara bertahap agar evaluasi sistem bisa mencakup lebih dari satu sisi. Ada dua tahap yang digunakan. Masing-masing tahap dirancang untuk menilai bagian yang berbeda dari sistem. Tahap pertama lebih fokus ke cara sistem mengambil dokumen dari *database*. Dokumen-dokumen ini sebelumnya dibagi menjadi bagian-bagian kecil (*chunk*). Ukuran potongan ini perlu diuji karena bisa mempengaruhi hasil pencarian. Kalau potongannya terlalu kecil, informasi penting bisa terlewat. Sebaliknya, kalau terlalu besar, pencarian jadi tidak akurat. Maka dari itu, dilakukan pengujian terhadap 5 ukuran *chunk* untuk menentukan *chunk* mana yang terbaik untuk digunakan untuk *database* yang akan digunakan dalam proses RAG. Tahap kedua lebih ke penggunaan *chatbot* oleh mahasiswa. Beberapa mahasiswa yang sedang atau pernah mengambil mata kuliah *Artificial Intelligence and Machine Learning* diminta mencoba chatbot dan bertanya sesuai materi kuliah. Setelah mencoba, mereka diminta menjawab beberapa pertanyaan dalam Google Form agar diketahui kepuasan user terhadap chatbot dalam berbagai aspek. Kedua tahap pengujian ini dipakai agar hasilnya tidak berat sebelah. Penilaian teknis saja tidak cukup, karena *chatbot* ini akan digunakan langsung oleh mahasiswa, sehingga penting juga untuk melihat bagaimana pengalaman mereka. Dengan adanya pengujian-pengujian ini, diharapkan hasil yang diperoleh bisa menjadi dasar untuk perbaikan sistem di masa yang akan datang.

4.1 Pengujian Dataset

Tahap awal pengujian dilakukan untuk mengevaluasi ukuran *chunk* agar mendapatkan ukuran yang terbaik untuk digunakan dalam proses pencarian dokumen dalam sistem *chatbot*. Ukuran *chunk* merujuk pada panjang karakter dari tiap potongan dokumen yang dimasukkan ke dalam *vector database*. Dalam sistem berbasis *Retrieval-Augmented Generation* (RAG), proses *retrieval* sangat bergantung pada representasi semantik dari setiap potongan dokumen. Oleh karena itu, ukuran *chunk* dapat memengaruhi kedalaman konteks yang ditangkap, serta ketepatan sistem dalam mencocokkan dokumen dengan pertanyaan yang diajukan.

Lima ukuran *chunk* dipilih untuk diuji, yaitu 2000, 3000, 4000, 5000, dan 6000 karakter. Rentang ini dipilih karena secara umum mewakili variasi dari *chunk* pendek, sedang, hingga panjang. Ukuran 2000 karakter dianggap sebagai *baseline* karena banyak digunakan dalam pengujian awal model *retrieval*, sementara ukuran 6000 karakter mendekati batas maksimal input model dalam satu kali proses pencarian. Dengan lima variasi ini, diharapkan dapat diperoleh konfigurasi yang seimbang antara kedalaman informasi dan kecepatan pemrosesan. Untuk menguji performa *retrieval*, digunakan lima pertanyaan teoretis yang masing-masing mewakili satu dari lima pokok bahasan utama dalam mata kuliah *Artificial Intelligence and Machine Learning* (AIML). Pemilihan pertanyaan berbasis teori dilakukan karena pertanyaan jenis ini biasanya membutuhkan jawaban berbentuk penjelasan konsep atau definisi, yang secara umum lebih bergantung pada relevansi semantik dibanding pencocokan literal. Adapun kelima pertanyaan yang digunakan adalah sebagai berikut:

1. *What is the definition of Unsupervised Learning?*
2. *What are the main advantages of Particle Swarm Optimization compared to traditional optimization algorithms?*
3. *How is AI used in everyday life?*
4. *What are some common heuristic search algorithms?*
5. *What is an artificial neural network* (ANN)*?*

Masing-masing pertanyaan diajukan terhadap setiap konfigurasi ukuran *chunk*, sehingga total terdapat 25 skenario pengujian (5 pertanyaan × 5 ukuran *chunk*). Setiap skenario menghasilkan lima dokumen hasil *retrieval* yang kemudian dievaluasi menggunakan tiga metrik utama. Evaluasi performa *retrieval* dilakukan menggunakan tiga metrik: *cosine similarity*, *Normalized Discounted Cumulative Gain* (NDCG), dan waktu *retrieval*. *Cosine similarity* dihitung secara otomatis oleh sistem untuk mengukur kedekatan semantik antara query dan setiap dokumen hasil pencarian, menggunakan representasi embedding dari model. Semakin tinggi nilainya, semakin mirip secara makna antara query dan dokumen. Sedangkan *retrieval time* dicatat menggunakan fungsi time.time() dari Python, yang mengukur selisih waktu dari awal proses pencarian hingga dokumen dikembalikan, guna menilai efisiensi sistem.

Sementara itu, proses evaluasi menggunakan NDCG memerlukan langkah tambahan karena melibatkan penilaian manual terhadap relevansi dokumen. Untuk setiap *query*, lima

dokumen hasil *retrieval* dari sistem disusun berdasarkan urutan yang diberikan oleh sistem (*ranking* prediksi). Peneliti kemudian melakukan penilaian relevansi terhadap setiap dokumen dengan memberikan skor berbasis skala 0 hingga 3, di mana: skor 3 menunjukkan dokumen sangat relevan, 2 cukup relevan, 1 kurang relevan, dan 0 tidak relevan. Penilaian ini dilakukan dengan mengevaluasi sejauh mana isi dokumen menjawab inti pertanyaan yang diajukan. Skor awal diperoleh menggunakan bantuan LLM untuk memperkirakan kedekatan semantik, lalu diverifikasi ulang secara manual agar sesuai dengan konteks akademik dan cakupan materi. Hasil penilaian dapat dilihat pada bagian Lampiran 1. Setelah seluruh skor disusun, perhitungan nilai NDCG dilakukan menggunakan fungsi ndcg_score(y_true, y_score) dari pustaka *scikit-learn*, di mana y_true berisi skor relevansi aktual (*ground truth*) dan y_score merupakan urutan ranking dari sistem. Nilai NDCG dari setiap skenario kemudian dirata-ratakan untuk memperoleh nilai NDCG akhir dari setiap konfigurasi ukuran *chunk*.

Dengan menggunakan tiga metrik tersebut secara bersamaan, pengujian ini bertujuan untuk menemukan konfigurasi *chunk* yang paling seimbang antara akurasi pencarian informasi dan kecepatan respons sistem. Konfigurasi yang optimal diharapkan mampu menghasilkan nilai *cosine similarity* dan NDCG yang tinggi, dengan waktu *retrieval* yang efisien, sehingga sistem dapat memberikan hasil yang relevan dan cepat saat digunakan oleh pengguna. Berikut adalah hasil pengujian terhadap lima konfigurasi *chunking* disajikan dalam Tabel 4.1.

Tabel 4.1

Hasil Evaluasi Pengujian Konfigurasi Ukuran *Chunk*

Ukuran *Chunk*	*Average* NDCG	*Average Cosine Similarity*	*Average Retrieval Time* (detik)
2000 karakter	0.8835	0.6337	10.0395
3000 karakter	0.8518	0.6016	8.4920
4000 karakter	0.8591	0.6164	8.2290
5000 karakter	0.8975	0.6332	8.6575
6000 karakter	0.8612	0.6311	8.4716

Berdasarkan hasil yang diperoleh, ukuran *chunk* 5000 karakter menunjukkan performa terbaik dibandingkan konfigurasi lainnya. Konfigurasi ini menghasilkan nilai NDCG tertinggi sebesar 0.8975, yang menunjukkan bahwa sistem mampu menempatkan dokumen-dokumen paling relevan di posisi atas dalam hasil retrieval. Selain itu, nilai rata-rata *cosine similarity* sebesar 0.6332 juga tergolong tinggi dan sebanding dengan konfigurasi 2000 karakter, namun dengan waktu *retrieval* yang lebih rendah, yaitu 8.6575 detik dibandingkan 10.0395 detik pada ukuran 2000 karakter. Meskipun ukuran *chunk* 6000 karakter memiliki waktu *retrieval* yang sedikit lebih cepat (8.4716 detik), nilai NDCG dan *cosine similarity* yang diperoleh lebih rendah dibandingkan dengan ukuran 5000 karakter. Oleh karena itu, konfigurasi dengan ukuran *chunk* 5000 karakter dipilih sebagai konfigurasi optimal dalam sistem *chatbot* ini, karena mampu memberikan keseimbangan terbaik antara relevansi hasil *retrieval* dan efisiensi waktu pencarian.

4.2 Pengujian Chatbot

Sebelum tahap pengujian *chatbot* dilakukan, sejumlah penyesuaian teknis diterapkan untuk memastikan sistem dapat berjalan secara optimal di lingkungan *cloud*. Pengujian chatbot direncanakan dilakukan setelah sistem berhasil di-*deploy* menggunakan Google Cloud Run sebagai platform layanan serverless dari Google Cloud Platform (GCP). *Deployment* ini dilakukan bukan sebagai bagian utama dari sistem, melainkan untuk mendukung pelaksanaan pengujian kualitatif secara profesional dan efisien. Dengan menggunakan *platform cloud*, chatbot dapat diakses melalui antarmuka web berbasis Streamlit, sehingga responden tidak perlu menggunakan *terminal* atau *command line interface*. Hal ini sangat penting untuk memastikan kenyamanan penggunaan bagi mahasiswa dan memungkinkan pelaksanaan pengujian dalam kondisi nyata. Chatbot dijalankan dalam infrastruktur *cloud* untuk memastikan aksesibilitas dan kestabilan ketika digunakan secara bersamaan oleh banyak pengguna. Model bahasa yang digunakan untuk pengujian adalah Nusantara-0.8b-Indo-Chat, yang dipilih karena efisiensi dan kecepatan inferensinya, sehingga lebih sesuai untuk konteks *deployment cloud* yang membutuhkan waktu respons cepat. Bahasa yang digunakan adalah Bahasa Indonesia karena chatbot ini diperuntukkan bagi mahasiswa Indonesia.

Seluruh dokumen pembelajaran yang digunakan untuk proses retrieval disimpan dalam format *vector embedding* dan diletakkan di *bucket* Google Cloud Storage (GCS). Saat inisialisasi aplikasi, sistem akan secara otomatis mengunduh database vektor dari GCS untuk memastikan ketersediaan data yang konsisten di setiap sesi. Proses *retrieval* dilakukan menggunakan pendekatan *hybrid* yang menggabungkan BM25 dan ChromaDB *retriever*, dilengkapi dengan *reranker* bge-reranker-base. Sistem ini dirancang untuk mengambil satu dokumen paling relevan berdasarkan hasil *reranking*, yang kemudian digunakan sebagai konteks dalam pembuatan jawaban oleh LLM. Pengujian internal *chatbot* kemudian dilakukan untuk mengevaluasi bagaimana sistem merespons pertanyaan secara aktual. Dalam proses ini ditemukan beberapa kendala yang memengaruhi kualitas respons *chatbot*. Untuk itu, dilakukan serangkaian penyesuaian guna meningkatkan kinerja sistem sebelum diuji langsung oleh pengguna.

4.2.1 Masalah Overgeneration dan Strategi Pemrosesan Jawaban

Salah satu permasalahan yang muncul selama pengujian adalah fenomena *overgeneration*, yaitu ketika model menghasilkan jawaban yang terlalu panjang dan berisi pengulangan informasi yang tidak perlu. Pola pengulangan ini umumnya muncul di bagian akhir jawaban, baik dalam bentuk frasa yang diulang beberapa kali, maupun item dalam daftar yang bersifat redundan. Hal ini membuat jawaban menjadi tidak efektif, membingungkan, dan kurang efisien untuk dibaca. Untuk mengatasi masalah ini, dirancang sebuah fungsi pemrosesan lanjutan bernama preprocess_final_response, yang berfungsi membersihkan dan menyusun ulang hasil jawaban sebelum ditampilkan ke pengguna. Strategi yang diterapkan dalam fungsi ini dibagi ke dalam tiga langkah utama:

1. Pemisahan jawaban menjadi unit kalimat yang lebih terstruktur

   Kalimat-kalimat dalam jawaban dipisah berdasarkan tanda baca akhir seperti titik, tanda tanya, atau tanda seru menggunakan ekspresi reguler. Tujuannya adalah agar sistem dapat mengevaluasi setiap kalimat secara mandiri untuk mendeteksi kemungkinan pengulangan atau kalimat yang belum selesai. Pemisahan ini tetap mempertimbangkan struktur daftar bernomor (misalnya “2.” atau “3.”), agar tidak keliru dianggap sebagai akhir kalimat biasa.

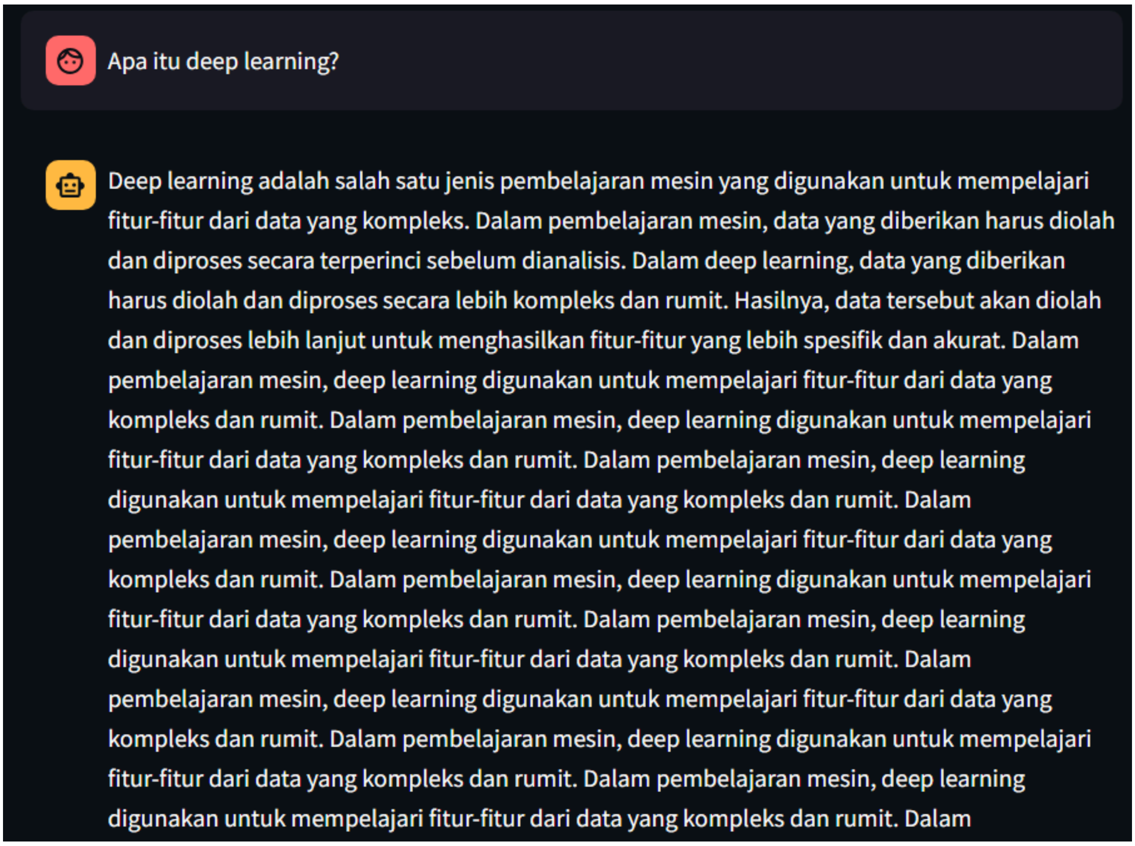
2. Identifikasi dan penghapusan kalimat yang redundan

   Setelah dipisah, setiap kalimat dibandingkan secara semantik untuk mendeteksi apakah isinya serupa atau identik. Kalimat yang memiliki struktur atau makna yang sama akan dianggap sebagai duplikat dan dihapus. Pendekatan ini mencegah informasi yang sama diulang beberapa kali, dan memastikan setiap bagian jawaban memberikan nilai informasi yang baru.

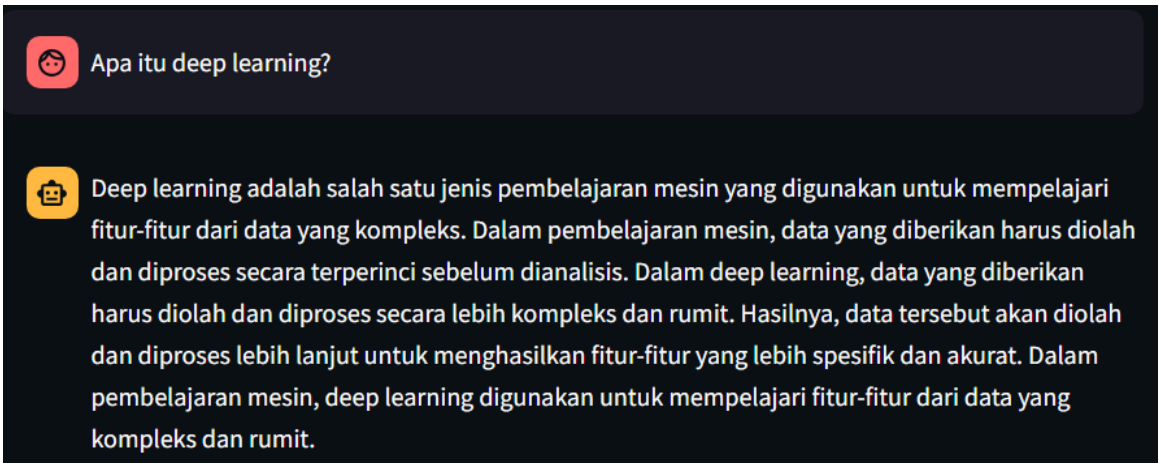
3. Rekonstruksi daftar dan penyaringan isi yang tidak informatif

   Jika jawaban mengandung daftar bernomor, sistem akan mendeteksi item-item dalam daftar tersebut dan memeriksa adanya duplikasi atau informasi yang terlalu umum. Item yang terlalu pendek (misalnya hanya satu frasa generik) atau terlalu mirip dengan item lain akan disaring. Penyusunan ulang daftar dilakukan dengan menjaga urutan alami dan numerik.

Dengan strategi ini, sistem mampu mengurangi pengulangan yang tidak perlu dan meningkatkan keterbacaan serta kualitas jawaban. Efektivitas pemrosesan ini dapat diamati pada contoh hasil yang ditunjukkan dalam Gambar 4.1 dan Gambar 4.2, yang memperlihatkan perbedaan antara jawaban sebelum dan sesudah melalui proses penyaringan. Gambar tersebut memperlihatkan bagaimana bagian akhir respons yang awalnya berulang-ulang dapat disederhanakan menjadi satu pernyataan yang padat dan informatif.



Gambar 4.1 Contoh jawaban *chatbot* yang *overgeneration*

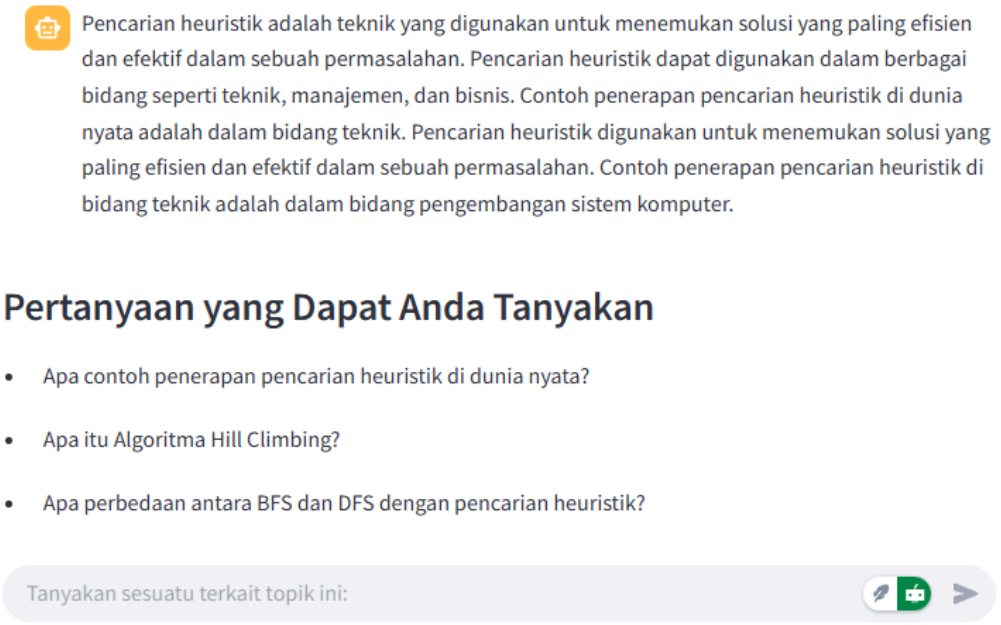


Gambar 4.2 Contoh jawaban *chatbot* yang sudah dilakukan penyuntingan

4.2.2 Penyesuaian *Prompt* untuk Mengurangi Pengulangan Ide

Setelah dilakukan pemrosesan untuk mengatasi *overgeneration*, ditemukan bahwa beberapa jawaban model masih mengandung kalimat yang memiliki makna serupa atau

berulang dalam bentuk yang lebih halus. Kalimat-kalimat ini tidak sepenuhnya identik secara struktur, tetapi menyampaikan informasi yang sama dalam dua cara berbeda. Salah satu contohnya dapat dilihat pada Gambar 4.3, di mana model menjawab pertanyaan *"Apa itu Heuristic Search?"* dengan dua kalimat yang secara esensial menjelaskan konsep yang sama.



Gambar 4.3 Jawaban model sebelum penyesuaian *prompt* yang masih mengandung pengulangan makna

Permasalahan ini menunjukkan bahwa meskipun pengulangan eksplisit telah berkurang, model masih menghasilkan jawaban yang terlalu luas atau kurang terfokus pada inti pertanyaan. Hal ini diduga terjadi karena struktur instruksi (*prompt*) yang digunakan belum cukup mengarahkan model untuk menjawab secara spesifik, terutama ketika konteks dokumen yang diberikan cukup panjang dan memuat berbagai topik terkait. Untuk mengatasi hal ini, dilakukan penyusunan ulang struktur *prompt* agar model lebih terarah dalam menjawab. Penyesuaian ini mencakup beberapa langkah:

1. Menetapkan peran model secara eksplisit

   Model diarahkan untuk bertindak sebagai asisten akademik yang hanya boleh menjawab berdasarkan konteks dokumen yang diberikan, serta menjelaskan ulang dengan gaya bahasa sendiri.

2. Menambahkan instruksi spesifik berdasarkan jenis pertanyaan

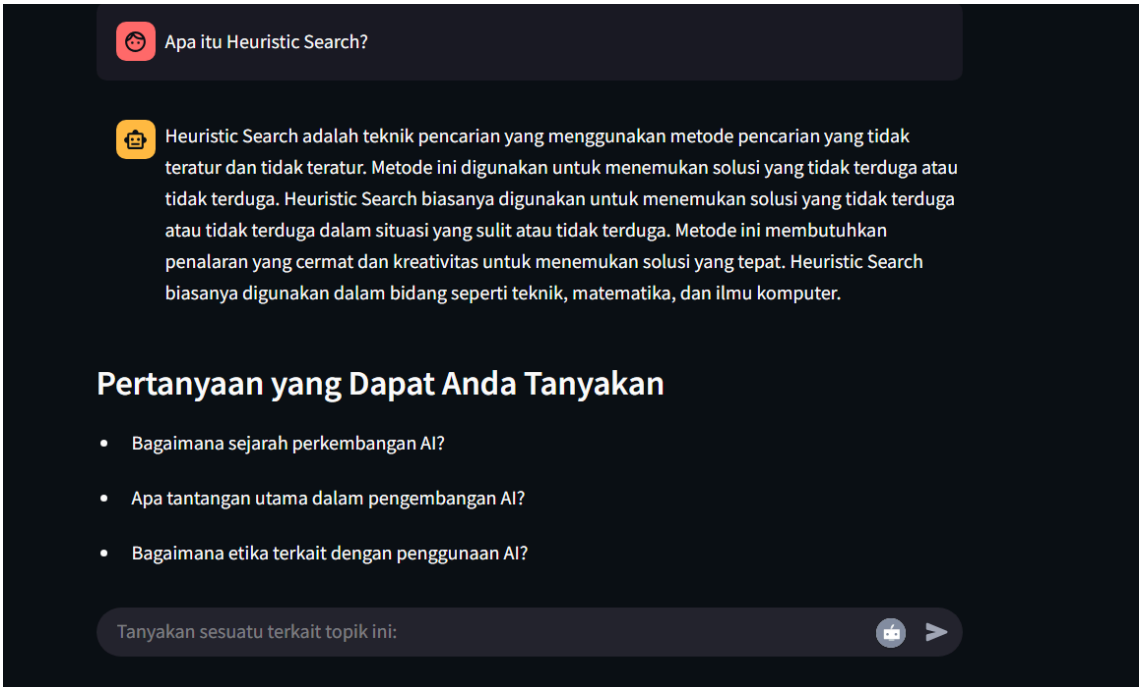
   Setiap jenis pertanyaan, seperti permintaan definisi, contoh, alasan, atau proses kerja, diberi instruksi yang berbeda. Misalnya:

   a. Jika pertanyaan berupa definisi, model diminta menjelaskan deinisi secara ringkas dan fokus.
   b. Jika pertanyaan meminta alasan, model diminta menjawab dengan penjelasan sebab-akibat.
   c. Jika pertanyaan meminta contoh, model diminta memberikan contoh yang relevan berdasarkan konteks.
   d. Jika pertanyaan meminta proses, model diminta menjelaskan langkah-langkah secara runtut.
   e. Jika pertanyaan meminta penejelasan perbedaan antara 2 hal, model diminta menjelaskan perbedaannya secara ringkas dan jelas.

3. Melarang penggunaan informasi di luar dokumen

   Prompt secara eksplisit meminta model untuk tidak menggunakan pengetahuan dari luar konteks (hallucination), melainkan hanya merujuk pada dokumen hasil retrieval.

Sebagai ilustrasi perbandingan antara sebelum dan sesudah penyesuaian *prompt*, Gambar 4.3 menunjukkan contoh jawaban model terhadap pertanyaan yang sama (*"Apa itu Heuristic Search?"*) sebelum *prompt* disesuaikan. Terlihat bahwa meskipun jawaban sudah tidak mengulang secara eksplisit, model masih memberikan penjelasan yang terlalu luas dan tidak langsung menjawab inti pertanyaan. Di sisi lain, Gambar 4.4 menampilkan respons model setelah diterapkannya *prompt* baru. Jawaban terlihat lebih ringkas, terfokus, dan tidak melebar ke informasi yang kurang relevan. Walaupun respons dengan *prompt* baru menunjukkan peningkatan, tetap ditemukan beberapa kasus di mana jawaban model belum sepenuhnya optimal. Penggunaan *prompt* yang lebih terarah berhasil mengurangi kecenderungan model untuk menghasilkan jawaban yang terlalu umum, tetapi masih belum menjamin bahwa jawaban akan tepat sasaran dalam setiap interaksi.



Gambar 4.4 Jawaban model setelah penyesuaian *prompt*, lebih fokus dan padat

4.2.3 Penyaringan Konteks untuk Meningkatkan Relevansi Jawaban dan Mempersempit Fokus Jawaban

Setelah dilakukan penyesuaian pada struktur *prompt*, hasil jawaban menunjukkan peningkatan fokus dalam menjawab inti pertanyaan. Namun demikian, masih ditemukan beberapa kasus di mana model tidak menjawab secara tepat sasaran. Masalah ini muncul terutama ketika model menerima konteks yang terlalu panjang dan kompleks, sehingga bagian konteks yang seharusnya relevan justru tertutup oleh bagian lain yang lebih dominan secara posisi atau jumlah kata. Akibatnya, jawaban menjadi melebar, mencakup penjelasan umum yang tidak secara langsung menjawab pertanyaan, atau bahkan fokus pada informasi yang kurang relevan. Masalah ini diduga berasal dari sifat *chunking* dokumen yang cukup besar, yaitu 5000 karakter per *chunk*. Meskipun *chunk* besar mampu menangkap konteks lebih luas, namun hal ini juga menyebabkan informasi penting tenggelam di antara paragraf lain yang kurang relevan. Oleh karena itu, dilakukan penyaringan (*filtering*) terhadap isi konteks sebelum diberikan ke model, dengan tujuan agar model hanya menerima bagian yang memiliki relevansi semantik tinggi terhadap pertanyaan yang diajukan. Proses penyaringan konteks dilakukan dalam beberapa tahap:

1. Pemisahan konteks ke dalam kalimat pendek

   Seluruh dokumen hasil retrieval dipecah menjadi kalimat-kalimat pendek. Pemisahan dilakukan menggunakan ekspresi reguler yang mempertimbangkan tanda baca (., ?, !) sekaligus struktur angka bernomor agar daftar tidak terpecah secara tidak tepat. Pendekatan ini juga meminimalkan risiko pemotongan kalimat penting secara tidak utuh.

2. Penggabungan kalimat menjadi *chunk* kecil (*window*)

   Kalimat-kalimat tersebut kemudian dikelompokkan menjadi blok-blok kecil berdasarkan *window_size* (misalnya 2 kalimat per blok). Setiap blok diperlakukan sebagai satu unit konteks yang akan dibandingkan dengan pertanyaan.

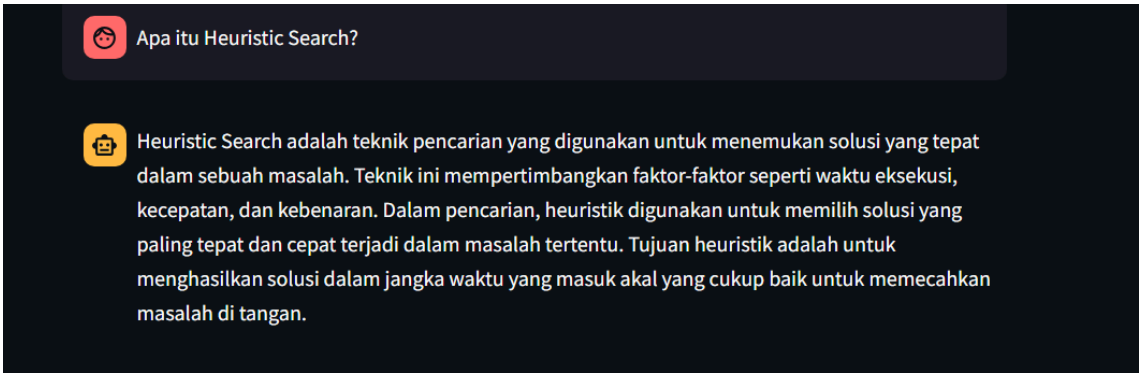
3. Perhitungan kemiripan semantik

   Untuk setiap blok konteks, dihitung skor *cosine similarity* terhadap pertanyaan menggunakan model *embedding* all-MiniLM-L6-v2. Skor ini mencerminkan seberapa dekat makna blok tersebut terhadap maksud pertanyaan.

4. Pemilihan top-k konteks paling relevan

   Blok-blok dengan skor *similarity* tertinggi (top-k) dipilih sebagai konteks akhir yang akan digunakan oleh model. Untuk menjaga alur naratif, blok tersebut disusun kembali berdasarkan urutan aslinya.

Dengan menerapkan pendekatan ini, konteks yang diberikan ke model menjadi jauh lebih fokus terhadap pertanyaan yang diajukan. Model tidak lagi terdistraksi oleh paragraf yang tidak relevan, dan cenderung menjawab lebih tepat sasaran. Gambar 4.5 berikut memperlihatkan contoh hasil jawaban setelah konteks disaring berdasarkan relevansi semantik. Terlihat bahwa jawaban lebih langsung menjawab pertanyaan, tanpa membahas topik yang tidak berkaitan.



Gambar 4.5. Contoh jawaban model setelah dilakukan penyaringan konteks berdasarkan pertanyaan “Apa itu Heuristic Search?”

Namun, dalam beberapa kasus, model masih menghasilkan informasi yang redundan meskipun konteks telah disaring. Hal ini menunjukkan bahwa penyaringan belum sepenuhnya menghilangkan pengulangan, dan perlu eksplorasi lanjutan untuk mengidentifikasi faktor penyebabnya secara pasti.

4.2.4 Pengujian Kualitatif *Chatbot* oleh Mahasiswa

Pengujian tahap kedua bertujuan untuk mengevaluasi efektivitas *chatbot* sebagai alat bantu pembelajaran mahasiswa dalam memahami materi *Artificial Intelligence and Machine Learning* (AIML). Pengujian ini sangat penting karena secara langsung menjawab rumusan masalah dalam penelitian ini. Sebelum dilakukan pengujian, sistem *chatbot* telah dideploy menggunakan layanan Google Cloud Run, yang memungkinkan aplikasi berjalan secara serverless dan dapat diakses oleh banyak pengguna secara bersamaan tanpa konfigurasi server khusus. Penggunaan Cloud Run memberikan tantangan tersendiri, terutama dalam hal kompatibilitas dan efisiensi pemrosesan model bahasa besar (LLM). Oleh karena itu, model yang digunakan disesuaikan agar ringan namun tetap efektif, yakni *Nusantara-0.8b-Indo-Chat*, karena memiliki ukuran parameter yang relatif kecil dan waktu inferensi yang cepat, menjadikannya lebih sesuai untuk lingkungan cloud yang memiliki batasan resource. Pemanfaatan Cloud Run juga memastikan sistem tetap responsif dan stabil selama sesi pengujian berlangsung, karena layanan ini dapat mengelola beban akses dari banyak pengguna tanpa intervensi manual. Hal ini penting untuk menciptakan skenario pengujian yang realistis dan merefleksikan pengalaman penggunaan sesungguhnya.

Dalam pengujian ini, sebanyak 16 mahasiswa Informatika dari Universitas Kristen Petra dilibatkan sebagai responden. Seluruh peserta telah memenuhi kualifikasi sebagai pengguna yang relevan untuk menguji sistem, yakni mahasiswa yang sedang atau telah menempuh mata kuliah *Artificial Intelligence and Machine Learning* (AIML), baik pada kurikulum baru maupun kurikulum lama. Pada kurikulum sebelumnya, materi AIML diajarkan dalam dua mata kuliah terpisah, yaitu Kecerdasan Buatan (AI) dan *Machine Learning*, sehingga mahasiswa yang pernah mengikuti kedua mata kuliah tersebut juga dianggap memiliki kompetensi dan konteks pembelajaran yang setara. Mahasiswa diminta berinteraksi langsung dengan *chatbot* melalui antarmuka yang telah disiapkan, memilih salah satu topik dari modul pembelajaran, dan mengajukan pertanyaan yang mereka anggap sulit atau membingungkan. Setelah berinteraksi, mereka diminta untuk mengisi kuesioner evaluasi menggunakan Skala Likert 1 - 5, untuk menilai beberapa aspek berikut:

1. Pemahaman dan Kejelasan Jawaban: sejauh mana jawaban yang diberikan oleh *chatbot* mudah dipahami dan sesuai dengan pertanyaan yang diajukan.
2. Relevansi Jawaban: tingkat kesesuaian jawaban dengan topik modul dan kebutuhan informasi mahasiswa.
3. Manfaat dan Kepuasan: apakah *chatbot* membantu mahasiswa memahami materi, dan apakah mereka merasa puas setelah menggunakannya.
4. Niat Menggunakan Kembali: apakah mahasiswa ingin menggunakan *chatbot* ini lagi dan merekomendasikannya kepada teman lain.

Setelah seluruh responden menyelesaikan interaksi dengan *chatbot* dan mengisi kuesioner, hasil penilaian mereka kemudian diolah dan dirangkum dalam bentuk skor rata-rata untuk setiap kelompok pertanyaan. Tabel 4.2 menyajikan ringkasan skor rata-rata berdasarkan tema evaluasi yang telah dikelompokkan, seperti relevansi jawaban, kualitas interaksi, hingga niat penggunaan kembali. Hasil ini menjadi dasar untuk menganalisis persepsi mahasiswa terhadap efektivitas chatbot dalam membantu proses pembelajaran, sebagaimana diuraikan dalam penjelasan berikut.

Tabel 4.2

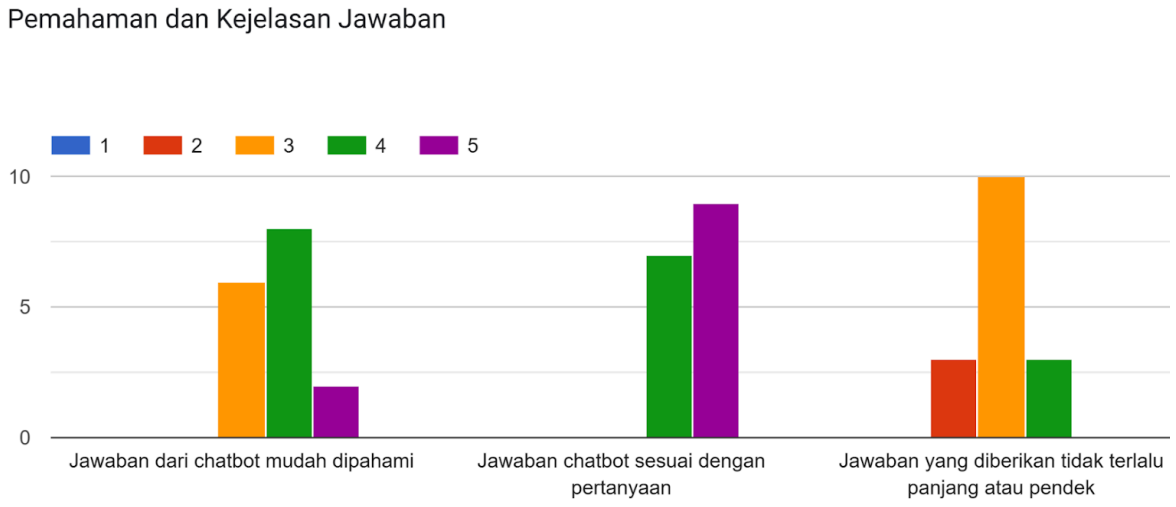
Hasil Pengujian Kualitatif *Chatbot* menggunakan skala likert terhadap mahasiswa

Aspek Penilaian	Pernyataan	Rata-Rata Nilai Skala Likert
Pemahaman Jawaban	Jawaban mudah dipahami	3.81
	Jawaban sesuai dengan pertanyaan	4.25
	Panjang jawaban sesuai	3.00
Relevansi Jawaban	*Chatbot* menjawab sesuai topik modul	4.06
	Jawaban yang diberikan relevan dan berguna	3.75
Manfaat dan Kepuasan	*Chatbot* membantu saya memahami materi	3.88
	Saya puas menggunakan *chatbot* ini	3.75
Niat Menggunakan Kembali	Ingin menggunakan *chatbot* ini lagi	3.62
	Akan merekomendasikan *chatbot* ini ke teman	3.56

Dalam aspek Pemahaman dan Kejelasan Jawaban, terdapat tiga pernyataan, yaitu "Jawaban mudah dipahami", "Jawaban sesuai dengan pertanyaan", dan "Panjang jawaban sesuai". Pernyataan "Jawaban mudah dipahami" memperoleh skor rata-rata 3.81. Nilai ini menunjukkan bahwa sebagian besar mahasiswa merasa cukup terbantu oleh jawaban yang diberikan, tetapi masih terdapat beberapa catatan kritis. Salah satu kendala yang mungkin

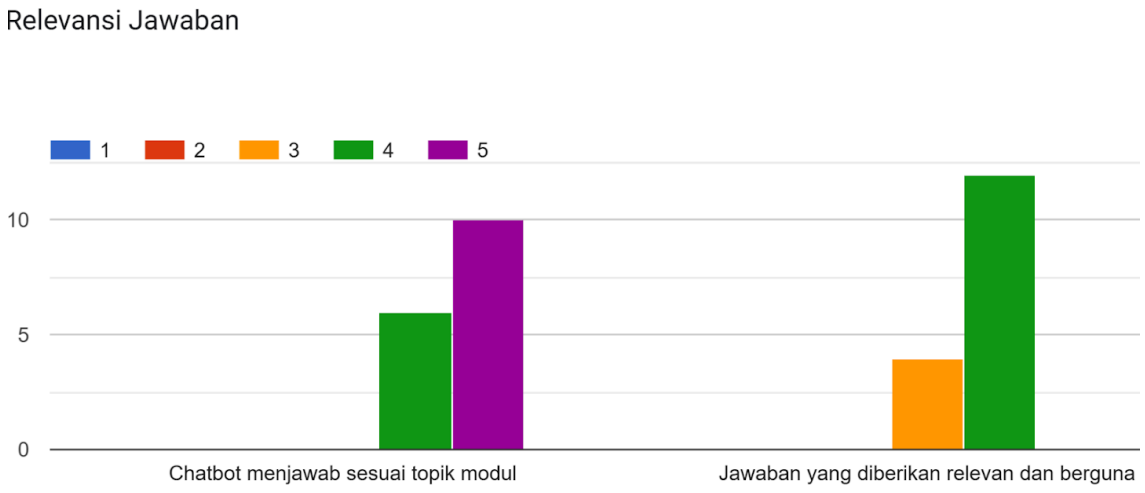
memengaruhi adalah struktur jawaban yang cenderung mengulang inti informasi dalam beberapa bentuk berbeda. Meskipun tidak lagi terdapat kalimat duplikat yang eksplisit (berkat preprocessing untuk menghapus kalimat serupa), pengulangan konsep yang berdekatan masih muncul dan berpotensi menurunkan keterbacaan. Selain itu, meskipun *chatbot* telah dilatih dengan bahasa Indonesia dan konteks sudah diterjemahkan, pemilihan kalimat yang berasal dari konteks beragam masih dapat menyebabkan transisi antar informasi menjadi kurang halus.

Sementara itu, pada pernyataan "Jawaban sesuai dengan pertanyaan" *chatbot* memperoleh skor tinggi yaitu 4.25. Hal ini menunjukkan bahwa pemetaan antara pertanyaan dan informasi yang diberikan sudah cukup baik. Pencapaian ini didukung oleh mekanisme pemfilteran konteks yang memastikan bahwa jawaban berasal dari bagian dokumen yang paling relevan. Sedangkan untuk "Panjang jawaban sesuai" skor berada pada 3.00, yang merupakan nilai terendah dalam dimensi ini. Berdasarkan hasil pengujian, beberapa responden menyatakan bahwa jawaban yang diberikan terkadang terlalu singkat dan tidak menjelaskan secara cukup. Hal ini bisa jadi disebabkan oleh proses filtering konteks yang membuat cakupan informasi menjadi lebih terbatas, terutama jika bagian yang difilter terlalu sempit atau konteks aslinya memang tidak kaya informasi. Fenomena ini mengindikasikan bahwa meskipun filtering berhasil mengurangi jawaban yang terlalu panjang, hal ini perlu diimbangi dengan kemampuan model untuk mengekstrak dan merangkum informasi secara optimal.



Gambar 4.6 Distribusi Skor Likert untuk Aspek Pemahaman dan Kejelasan Jawaban pada Evaluasi *Chatbot*

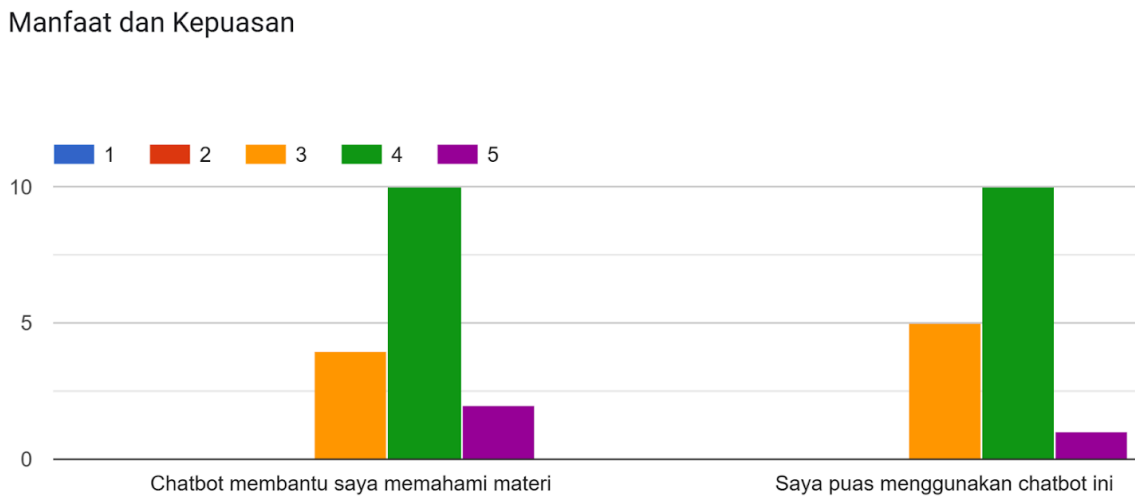
Dalam aspek Relevansi Jawaban, pernyataan "*Chatbot* menjawab sesuai topik modul" meraih skor 4.06, menunjukkan tingkat relevansi yang cukup tinggi. Skor ini cukup masuk akal karena sistem telah membatasi ruang lingkup *retrieval* hanya pada dokumen yang memiliki *metadata* modul sesuai dengan pilihan pengguna. Hal ini tidak hanya mencegah terjadinya pencampuran informasi di luar topik, tetapi juga menunjukkan bahwa *chatbot* berhasil menjaga agar proses pembelajaran tetap terarah sesuai struktur materi yang telah ditentukan. Dengan arah jawaban yang konsisten terhadap topik yang dipilih, *chatbot* mampu membantu mahasiswa fokus pada satu ruang lingkup materi di setiap sesi tanya jawab. Namun, untuk pernyataan "Jawaban yang diberikan relevan dan berguna" nilai yang diperoleh adalah 3.75. Nilai ini menunjukkan bahwa meskipun jawaban sudah berada dalam topik yang tepat, cara penyampaiannya tidak selalu membantu mahasiswa membangun pemahaman yang lebih dalam. Hal ini bisa disebabkan oleh proses filtering yang mengandalkan kesamaan semantik (*cosine similarity*), di mana potongan konteks yang terpilih mungkin memiliki kata kunci yang mirip, tetapi tidak selalu menawarkan penjelasan yang paling komprehensif. Selain itu, karena *chunk* disusun berdasarkan skor tertinggi dan kemudian dirangkai kembali, bisa terjadi potongan-potongan kalimat yang saling tidak berkesinambungan. Hal ini menimbulkan risiko kurangnya kohesi informasi dan penurunan efektivitas penyampaian pesan.



Gambar 4.7 Distribusi Skor Likert untuk Aspek Relevansi Jawaban pada Evaluasi *Chatbot*

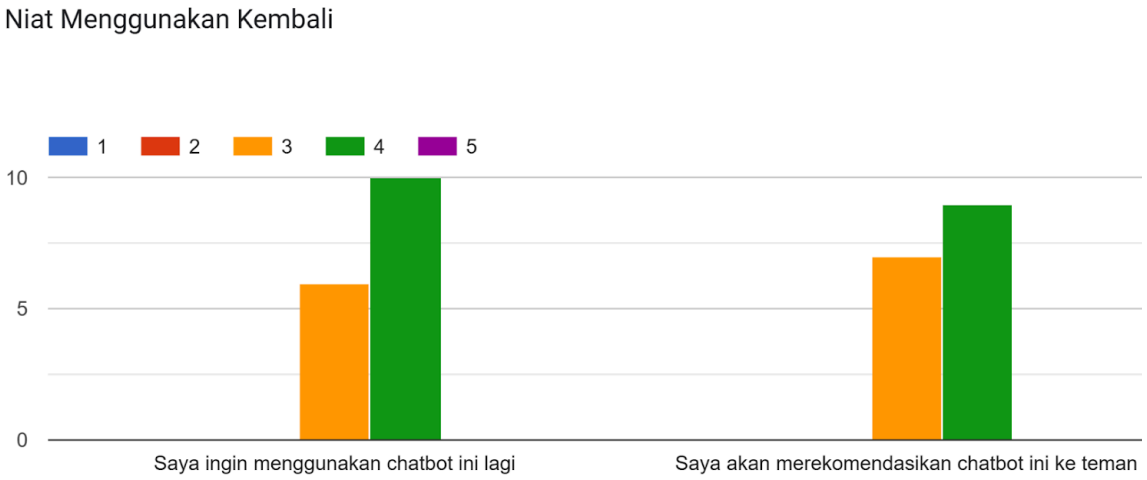
Pada bagian aspek manfaat dan kepuasan, skor 3.88 untuk indikator "*chatbot* membantu saya memahami materi" mencerminkan bahwa chatbot ini secara umum telah memberikan kontribusi positif terhadap proses belajar mahasiswa. Meskipun nilainya tidak mencapai angka sempurna, skor ini menunjukkan bahwa sebagian besar mahasiswa merasa

terbantu dalam memahami topik-topik yang kompleks dalam mata kuliah AIML. Namun, pada indikator "saya puas menggunakan *chatbot* ini", skor sedikit menurun menjadi 3.75, yang menunjukkan bahwa tingkat kepuasan pengguna belum sepenuhnya maksimal. Penurunan ini bisa dikaitkan dengan beberapa kelemahan yang telah dijelaskan sebelumnya, seperti respons model yang terkadang kurang ringkas atau tidak sepenuhnya fokus, serta potensi kurang lengkapnya jawaban dalam beberapa kasus. Meski demikian, skor ini masih berada di atas rata-rata, yang berarti bahwa secara umum *chatbot* telah memenuhi ekspektasi mahasiswa dalam fungsinya sebagai dalam membantu pembelajaran dengan menjawab pertanyaan.



Gambar 4.8 Distribusi Skor Likert untuk Aspek Manfaat dan Kepuasan pada Evaluasi *Chatbot*

Skor 3.62 untuk pernyataan keinginan menggunakan kembali dan 3.56 untuk pernyataan merekomendasikan kepada teman merupakan indikator bahwa sistem telah memberikan kesan positif meskipun belum sepenuhnya “mengesankan”. Nilai ini bisa dipahami sebagai akibat dari faktor-faktor teknis seperti waktu tunggu dan kadang jawaban yang kurang mendalam. Namun, yang penting adalah bahwa tidak ada kecenderungan penolakan dari mahasiswa, dan ini berarti sistem masih memiliki daya tarik serta potensi untuk dikembangkan lebih lanjut menjadi sistem pendukung pembelajaran yang lebih baik.



Gambar 4.9 Distribusi Skor Likert untuk Aspek Niat Menggunakan Kembali pada Evaluasi *Chatbot*