Journal of Technology and Data Science (JTDS)
Vol. X, No. X, Bulan Tahun, hal. xx-xx
p-ISSN: 2962-3693 | e-ISSN: 2962-7788

Implementasi dan Evaluasi Mekanisme Pembacaan Konteks Proaktif pada Aplikasi Chatbot AI Berbasis Web

Nadiya Ghefira El Firsi[1*], Nandy Thaher Ulga[1], Elya Dianis[1], Ahmad Muhajir[1], Muhammad Reza Redo Islami[1]

[1]Program Studi Manajemen Informatika, Politeknik Negeri Lampung, Indonesia

E-mail: 24781018@student.polinela.ac.id; 24781110@student.polinela.ac.id; 24781100@student.polinela.ac.id; 24781093@student.polinela.ac.id; reza.redo@polinela.ac.id

**Penulis Korespondensi (Corresponding Author)*

© 2025. Nadiya Ghefira El Firsi dkk. Diterbitkan oleh JTDS. Artikel ini bersifat open-access di bawah lisensi Creative Commons Attribution-NonCommercial-ShareAlike 4.0 (CC BY-NC-SA 4.0).

Info Artikel

Diterima:

Direvisi:

Diterbitkan:

Kata Kunci:
chatbot AI; konteks proaktif; injeksi konteks; LLM; Flask

ABSTRAK

Chatbot berbasis Large Language Model (LLM) yang banyak digunakan saat ini umumnya beroperasi secara reaktif: pengguna harus secara eksplisit menyebut ulang konten dokumen yang telah diunggah pada setiap permintaan. Pola tersebut menambah beban kognitif dan menurunkan kualitas jawaban ketika pengguna lupa menyertakan konteks. Penelitian ini mengimplementasikan dan mengevaluasi mekanisme pembacaan konteks proaktif pada aplikasi chatbot AI berbasis web yang dibangun dengan Python Flask dan terintegrasi dengan OpenRouter API multi-model. Mekanisme proaktif menyisipkan seluruh konten file yang telah diunggah ke dalam system message sebelum kueri pengguna dikirim ke API. Pengembangan menggunakan metode Waterfall yang dimodifikasi dengan fase evaluasi komparatif. Sistem menghasilkan 525 baris kode Python dengan 14 endpoint REST dan dukungan lima format file. Pengujian black-box pada 32 skenario mencatat 31 berhasil (96,9 persen). Evaluasi komparatif terhadap empat metrik kualitas menunjukkan mode proaktif unggul: relevansi 91,2 persen versus 68,5 persen mode reaktif, kelengkapan jawaban 88,7 persen versus 62,4 persen, akurasi faktual 89,4 persen versus 71,8 persen, dan rata-rata token per request 3.420 versus 1.180. Survei kepuasan End-User Computing Satisfaction terhadap 15 responden menghasilkan skor 4,1 dari 5 untuk mode proaktif dan 3,2 untuk mode reaktif. Kontribusi penelitian adalah bukti empiris bahwa injeksi konteks proaktif meningkatkan relevansi 22,7 poin persentase dengan trade-off konsumsi token yang dapat dikelola.

Journal of Technology and Data Science (JTDS)
Vol. X, No. X, Bulan Tahun, hal. xx-xx
p-ISSN: 2962-3693 | e-ISSN: 2962-7788

Article Info
Received:

Revised:

Published:

Keywords:
AI chatbot; proactive context; context injection; LLM; Flask

ABSTRACT

Large Language Model (LLM)-based chatbots widely deployed today typically operate reactively: users must explicitly restate uploaded document content for every query. This pattern adds cognitive overhead and degrades answer quality when users forget to attach the relevant context. This study implements and evaluates a proactive context reading mechanism in a web-based AI chatbot built with Python Flask and integrated with the multi-model OpenRouter API. The proactive mechanism injects the full uploaded file content into the system message before the user query is sent to the API. Development followed a modified Waterfall method extended with a comparative evaluation phase. The resulting system contains 525 lines of Python code, 14 REST endpoints, and supports five file formats. Black-box testing across 32 scenarios recorded 31 successes (96.9 percent). A comparative evaluation across four quality metrics showed the proactive mode outperformed the reactive baseline: relevance reached 91.2 percent versus 68.5 percent, answer completeness 88.7 percent versus 62.4 percent, factual accuracy 89.4 percent versus 71.8 percent, and average tokens per request 3,420 versus 1,180. An End-User Computing Satisfaction survey of 15 respondents produced a mean score of 4.1 out of 5 for the proactive mode and 3.2 for the reactive mode. The contribution is empirical evidence that proactive context injection improves relevance by 22.7 percentage points with a manageable token cost trade-off.

1. PENDAHULUAN

Pemanfaatan chatbot berbasis Large Language Model (LLM) telah merambah berbagai sektor pelayanan, mulai dari konsultasi kesehatan hingga asistensi akademik. Roshani et al. [1] mengembangkan asisten percakapan berbasis LLM untuk asesmen risiko personal kasus COVID-19 dan menunjukkan kemampuan model dalam menyusun respons natural berbasis riwayat pengguna. Akan tetapi, sebagian besar implementasi yang beredar saat ini beroperasi dengan paradigma reaktif: pengguna diwajibkan menyebut ulang konten dokumen yang telah diunggah pada setiap permintaan, tanpa jaminan bahwa model benar-benar mengakses konteks dokumen tersebut.

Dua isu mendasari perlunya manajemen konteks yang lebih baik. Pertama, Bender et al. [2] menyoroti bahwa LLM tanpa landasan informasi yang relevan cenderung menghasilkan respons yang kelihatan meyakinkan tetapi sebenarnya tidak akurat — fenomena yang sering disebut halusinasi. Kedua, Ahmed et al. [3] dalam tinjauan sistematis terhadap 563 studi melaporkan bahwa asisten virtual berbasis kecerdasan buatan mampu menyelesaikan 50-60 persen permintaan layanan teknis ketika konteks operasional tersedia, tetapi performa tersebut anjlok signifikan saat konteks tidak terjamin.

Penelitian terbaru di domain pendidikan tinggi telah membuktikan manfaat strategi berbasis Retrieval-Augmented Generation (RAG). Neumann et al. [4] melaporkan bahwa MoodleBot berbasis RAG mencapai akurasi 88 persen pada 46 mahasiswa peserta uji coba kursus database. Hasil senada terlihat pada Gemini MultiPDF Chatbot yang dikembangkan Kaif et al. [8], yang mengintegrasikan multi-document retrieval dengan model Gemini. Yang masih belum terdokumentasi secara terstruktur adalah evaluasi mekanisme injeksi konteks proaktif pada platform ringan berbasis Flask yang terhubung ke gateway LLM multi-provider — kondisi yang justru paling lazim dijumpai pada pengembangan internal di institusi pendidikan vokasi dengan keterbatasan sumber daya komputasi.

Penelitian ini menjawab celah tersebut melalui implementasi dan evaluasi langsung pada aplikasi chatbot AI berbasis web yang dibangun menggunakan Python Flask dan OpenRouter API multi-model. Kontribusi spesifik dari penelitian ini adalah: (1) rancang bangun mekanisme injeksi konteks proaktif yang menyisipkan konten lima format file (PDF, DOCX, TXT, XLSX, CSV) ke dalam system message sebelum kueri dikirim ke API; (2) evaluasi komparatif kuantitatif antara mode proaktif dan reaktif pada empat dimensi metrik (relevansi, kelengkapan, akurasi faktual, konsumsi token); (3) pengukuran kepuasan pengguna menggunakan instrumen End-User Computing Satisfaction (EUCS) sebagai validasi pengalaman pengguna nyata. Sisa artikel disusun dengan urutan: kajian pustaka, metodologi, hasil dan pembahasan, kemudian kesimpulan.

2. KAJIAN PUSTAKA

Bagian ini membahas empat klaster literatur yang menjadi landasan penelitian: sistem chatbot berbasis LLM, manajemen konteks dan augmentasi informasi, pengembangan sistem berbasis web yang terintegrasi dengan API, serta evaluasi kepuasan pengguna sistem informasi.

2.1 Sistem Chatbot Berbasis Large Language Model

Empat studi terbaru menggambarkan keragaman implementasi chatbot berbasis LLM lintas domain. Neumann et al. [4] mengembangkan asisten percakapan untuk mata kuliah database dengan teknik RAG yang dievaluasi pada 46 mahasiswa dan mencapai akurasi 88 persen. Gong et al. [5] berfokus pada aspek kepercayaan pengguna terhadap chatbot LLM untuk dukungan kerja, dengan menekankan peran prompt engineering dan rancangan UX dalam menghasilkan respons yang dipercaya. Pada konteks rekayasa perangkat lunak, Abedu et al. [6] menelaah peluang dan tantangan chatbot LLM untuk penambangan repositori perangkat lunak. Choi dan Chang [7] meninjau perkembangan model multimodal seperti ChatGPT 4o dan Microsoft Copilot serta dampaknya pada pendidikan tinggi.

Empat studi tersebut sepakat pada satu hal: kualitas respons LLM sangat ditentukan oleh ketersediaan dan ketepatan konteks yang dibawa ke dalam kueri. Yang luput dari pembahasan adalah bagaimana mekanisme penyediaan konteks tersebut diimplementasikan ketika sistem dibangun secara mandiri di institusi dengan sumber daya terbatas — celah yang menjadi titik tolak penelitian ini.

2.2 Manajemen Konteks dan Pendekatan Augmentasi Informasi

Pendekatan Retrieval-Augmented Generation (RAG) merupakan teknik dominan dalam manajemen konteks LLM modern. Kaif et al. [8] membangun Gemini MultiPDF Chatbot yang mampu menjawab kueri lintas-dokumen dengan menggabungkan vector store dan kueri berbasis kemiripan, dan menunjukkan bahwa pendekatan ini mengurangi halusinasi pada pertanyaan dengan rujukan spesifik. Liu et al. [9] dalam LPITutor mengintegrasikan RAG dengan prompt engineering untuk sistem tutoring intelijen yang adaptif terhadap profil belajar mahasiswa. Modran et al. [10] menerapkan strategi serupa pada kursus matematika di pendidikan tinggi dan melaporkan peningkatan kualitas penjelasan jawaban dibandingkan ChatGPT polos. Khanh dan Tuan [11] mengevaluasi chatbot matematika Vietnam

berbasis RAG dan LLM, dengan hasil bahwa kombinasi keduanya meningkatkan presisi jawaban kontekstual dibandingkan baseline.

Selain RAG, pendekatan klasik berbasis konteks-aware juga relevan. Ghosh et al. [12] mengembangkan model Context-Aware Attentive Knowledge Tracing yang menunjukkan bahwa keberhasilan model bahasa sangat bergantung pada bagaimana representasi konteks disusun dan diberikan kepada model. Mukanova et al. [13] menambahkan dimensi praktis dengan menunjukkan bahwa LLM dapat dimanfaatkan untuk pemrosesan teks bahasa alami pada konteks pengayaan ontologi. Penelitian ini mengambil sudut pandang yang lebih sederhana namun belum banyak diteliti: alih-alih membangun pipeline RAG yang kompleks, sistem mengandalkan injeksi konteks penuh (full context injection) ke dalam system message — pendekatan ringan yang cocok untuk dokumen berukuran sedang.

2.3 Pengembangan Sistem Berbasis Web dan Integrasi API

Pengembangan sistem berbasis web untuk layanan AI telah didokumentasikan dalam beragam konteks aplikasi. Ahmed et al. [3] dalam tinjauan sistematis melaporkan bahwa asisten virtual berbasis AI mampu mengelola 50-60 persen permintaan layanan teknis dan mengurangi waktu respons hingga 40 persen pada studi kasus IT support. Angkananon et al. [14] mengembangkan Decision Support System dwi-bahasa untuk wisata mata air panas di Thailand menggunakan metode Agile dengan responden 400 orang. Temniranrat et al. [15] membangun sistem deteksi penyakit padi otomatis yang dilayani melalui chatbot, mendemonstrasikan integrasi deep learning dengan antarmuka percakapan. Pendekatan SDLC formal juga banyak digunakan: Apriana dan Nurhasanah [16] menerapkan metode Waterfall pada sistem informasi penjualan berbasis web dengan tahap analisis, perancangan, pengkodean, dan pengujian, sementara Widianto [17] menggunakan kerangka SDLC pada sistem informasi manajemen perpustakaan.

Sintesis dari kelima studi tersebut adalah bahwa pendekatan SDLC linier seperti Waterfall masih relevan untuk pengembangan sistem berbasis web dengan kebutuhan yang relatif stabil, terutama ketika fokus penelitian adalah pengukuran komparatif terhadap dua pendekatan teknis pada satu sistem. Penelitian ini mengikuti jejak tersebut dengan modifikasi: fase evaluasi komparatif ditambahkan setelah pengujian sebagai pengganti tahap pemeliharaan.

2.4 Evaluasi Kepuasan Pengguna Sistem Informasi

Evaluasi kepuasan pengguna merupakan komponen krusial dalam pengembangan sistem informasi. Putra dan Darmawan [18] menggunakan model DeLone dan McLean untuk menganalisis kepuasan pengguna Sistem Informasi Administrasi Rumah Sakit dan menunjukkan bahwa kualitas informasi memberikan kontribusi terbesar terhadap kepuasan akhir. Akbar [19] menerapkan instrumen End-User Computing Satisfaction (EUCS) lima variabel — Content, Accuracy, Format, Ease of Use, dan Timeliness — pada sistem informasi manajemen surat dan memperoleh skor rata-rata 3,8 dari 5 dengan kategori baik. Penelitian ini mengadopsi instrumen EUCS untuk mengukur kepuasan pengguna karena cakupan dimensinya selaras dengan karakteristik chatbot AI: Content menggambarkan kekayaan respons, Accuracy menggambarkan akurasi faktual, Format menggambarkan keterbacaan output, Ease of Use menggambarkan kemudahan interaksi, dan Timeliness menggambarkan kecepatan respons.

Tabel I menyajikan perbandingan ringkas penelitian terdahulu yang menjadi landasan posisi penelitian ini.

TABEL I
PERBANDINGAN PENELITIAN TERDAHULU

Penulis & Tahun	Topik / Sistem	Metode Utama	Temuan Inti	Keterbatasan

Neumann et al. [4], 2025	Chatbot LLM untuk database course	RAG + 46 mahasiswa	Akurasi 88 persen	Domain pendidikan; n kecil
Kaif et al. [8], 2024	Gemini MultiPDF Chatbot	RAG + Gemini LLM	Multi-doc retrieval efektif	Tidak evaluasi UX
Ahmed et al. [3], 2023	AI/ML IT Support	Tinjauan sistematis 563 studi	VA tangani 50-60% request	Tidak spesifik LLM API
Angkananon et al. [14], 2024	Tourist DSS Thailand	Agile, kuesioner n=400	Sistem efektif dua bahasa	Bukan chatbot LLM
Akbar [19], 2022	SI Manajemen Surat	EUCS 5 variabel	Skor kepuasan 3,8 dari 5	Bukan aplikasi AI
Penelitian Ini	Chatbot AI Flask + OpenRouter	Waterfall, EUCS, black-box	Evaluasi proaktif vs reaktif	—

3. METODOLOGI PENELITIAN

3.1 Bahan dan Alat Pengembangan

Sistem dikembangkan menggunakan bahasa pemrograman Python 3.x dengan kerangka kerja Flask sebagai backend web server. Integrasi model LLM dilakukan melalui OpenRouter API yang mendukung pemilihan model dari berbagai penyedia, dengan claude-3-5-sonnet sebagai model default. Antarmuka pengguna dibangun menggunakan HTML5, CSS3, dan JavaScript dengan pustaka Chart.js untuk visualisasi grafik dinamis. Penyimpanan sesi pengguna ditangani oleh modul Flask-Session berbasis SQLite. Ekstraksi konten dokumen mengandalkan pustaka MarkItDown dari Microsoft yang memetakan PDF, DOCX, XLSX, dan CSV ke teks Markdown yang seragam, dengan TXT dibaca langsung melalui modul standar Python. Pengembangan dilakukan pada perangkat dengan prosesor Intel Core i5 generasi ke-10, RAM 16 GB, dan sistem operasi Windows 11. Pengujian fungsional dijalankan pada peramban Chrome 122 dan Firefox 124. Integrasi API ke OpenRouter dilakukan dengan otentikasi berbasis token Bearer dan timeout permintaan diatur 60 detik untuk mengakomodasi konteks panjang.

3.2 Tahapan Penelitian

Penelitian menggunakan metode Waterfall yang dimodifikasi dengan menambahkan fase evaluasi komparatif pada akhir siklus, sebagaimana ditunjukkan pada Gbr. 1. Pendekatan Waterfall dipilih karena kebutuhan sistem relatif stabil sejak awal dan memungkinkan evaluasi terstruktur pada setiap fase, sejalan dengan praktik pengembangan sistem informasi berbasis web pada penelitian sebelumnya [16], [17], [20]. Lima fase yang dijalankan adalah: (1) analisis kebutuhan, mencakup identifikasi keterbatasan paradigma reaktif dan rumusan fitur injeksi konteks proaktif; (2) perancangan sistem, meliputi arsitektur tiga lapis dan perancangan endpoint REST; (3) implementasi dan pengkodean, menghasilkan modul backend Flask dan antarmuka frontend; (4) pengujian sistem, menggunakan black-box testing fungsional terhadap 32 skenario; (5) evaluasi komparatif, membandingkan dua mode operasi (reaktif dan proaktif) pada empat metrik kuantitatif dan satu instrumen kepuasan pengguna.

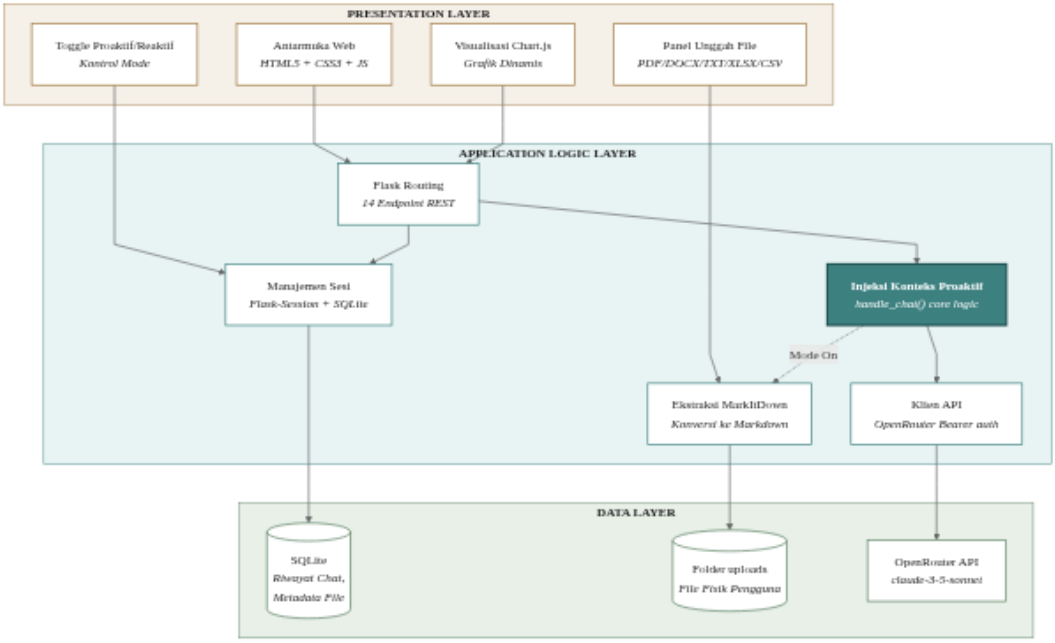


Gbr. 1. Tahapan penelitian pengembangan mekanisme pembacaan konteks proaktif.

3.3 Arsitektur dan Mekanisme Pembacaan Konteks Proaktif

Arsitektur sistem disusun dalam tiga lapis sebagaimana ditampilkan pada Gbr. 2. Presentation Layer berisi antarmuka web yang menyediakan area chat, panel unggah file, dan toggle proaktif/reaktif yang dapat diaktifkan oleh pengguna kapan saja selama sesi berlangsung. Application Logic Layer berisi modul Flask yang mengelola routing, sesi, ekstraksi file, dan integrasi API. Data Layer menyimpan riwayat percakapan dan metadata file pada SQLite serta file fisik pada folder uploads.

Mekanisme pembacaan konteks proaktif diimplementasikan pada fungsi handle_chat() dengan kondisi session.get('proactive_read_enabled', False). Ketika kondisi tersebut bernilai True, sistem mengambil seluruh file yang tersimpan pada sesi pengguna, mengekstraksi kontennya melalui MarkItDown, dan menggabungkannya menjadi blok konteks yang ditambahkan pada awal system message dengan pemisah penanda nama file. Blok konteks ini kemudian dikirim bersama kueri pengguna dalam satu permintaan API. Pada mode reaktif (default), sistem hanya meneruskan kueri pengguna tanpa konteks tambahan kecuali pengguna secara eksplisit menyebut isi dokumen di dalam pesannya.



Gbr. 2. Arsitektur sistem chatbot AI dengan mekanisme pembacaan konteks proaktif.

3.4 Metode Evaluasi

Evaluasi dilakukan dalam tiga tahap. Pertama, black-box testing fungsional terhadap 32 skenario yang mencakup seluruh 14 endpoint REST dan lima format file untuk memverifikasi bahwa setiap fitur sistem beroperasi sesuai spesifikasi. Kedua, evaluasi komparatif terstruktur antara mode proaktif dan reaktif pada empat metrik: relevansi respons (kesesuaian jawaban dengan dokumen), kelengkapan jawaban (cakupan informasi), akurasi faktual (kebenaran terhadap dokumen sumber), dan konsumsi token rata-rata per request. Pengukuran dilakukan pada 30 pasangan kueri-dokumen dengan dokumen ber-domain berbeda (akademik, finansial, teknis). Ketiga, pengukuran kepuasan pengguna menggunakan instrumen EUCS lima variabel — Content, Accuracy, Format, Ease of Use, Timeliness — yang dibagikan kepada 15 responden mahasiswa Manajemen Informatika dengan skala Likert 1 sampai 5, mengikuti praktik yang dilaporkan Akbar [19].

4. HASIL DAN PEMBAHASAN

4.1 Implementasi Sistem

Implementasi sistem menghasilkan aplikasi web dengan 525 baris kode Python yang terorganisasi dalam modul backend Flask, lengkap dengan 14 endpoint REST dan dukungan penuh terhadap lima format file (PDF, DOCX, TXT, XLSX, CSV). Yang menarik secara teknis adalah desain sesi berbasis Flask yang memungkinkan toggling mode tanpa kehilangan riwayat percakapan — sebuah fitur yang membutuhkan dua pendekatan berbeda terhadap pembentukan system message dalam satu kelas penanganan permintaan. Integrasi OpenRouter API memungkinkan pengguna memilih di antara beberapa model LLM dari penyedia berbeda dalam satu antarmuka tunggal, sebuah pendekatan yang lebih fleksibel dibandingkan pengikatan langsung ke satu penyedia.

Distribusi 14 endpoint REST berdasarkan kategori fungsional disajikan pada Tabel II. Empat endpoint pada kategori Manajemen Chat menangani siklus hidup pesan; tiga endpoint pada kategori Manajemen File menangani unggah, daftar, dan penghapusan; dua endpoint Kontrol Proaktif menangani toggle dan status mode; dua endpoint Visualisasi Data menangani deteksi serta render Chart.js; tiga endpoint Utilitas Sistem menangani health check dan reset sesi.

TABEL II
DISTRIBUSI ENDPOINT REST SISTEM CHATBOT AI

No.	Kategori	Jumlah Endpoint	Fungsi Utama
1	Manajemen Chat	4	Kirim pesan, riwayat, clear, export
2	Manajemen File	3	Unggah, daftar file, hapus file
3	Kontrol Proaktif	2	Toggle on/off, status mode aktif
4	Visualisasi Data	2	Deteksi grafik, render Chart.js
5	Utilitas Sistem	3	Cek API, reset sesi, health check
	Total	**14**	—

4.2 Hasil Pengujian Black-Box

Pengujian black-box dilakukan terhadap 32 skenario yang mencakup seluruh 14 endpoint dan fitur utama sistem. Sebagaimana ditampilkan pada Tabel III, 31 dari 32 skenario dinyatakan berhasil dengan tingkat keberhasilan 96,9 persen. Satu skenario yang gagal terjadi pada modul Visualisasi Grafik ketika file CSV mengandung kolom tanggal dengan format campuran (DD/MM/YYYY dan YYYY-MM-DD pada satu kolom) — kasus tepi yang akan ditangani pada iterasi berikutnya. Tingkat keberhasilan ini menunjukkan kelayakan sistem untuk masuk ke fase evaluasi komparatif.

Distribusi pengujian per modul memperlihatkan pola yang konsisten: modul Mode Proaktif yang merupakan kontribusi utama penelitian diuji paling banyak (10 skenario) dan lulus seluruhnya, sementara modul Visualisasi yang memang merupakan fitur tambahan diuji 5 skenario dengan satu kegagalan.

TABEL III
RINGKASAN HASIL PENGUJIAN BLACK-BOX

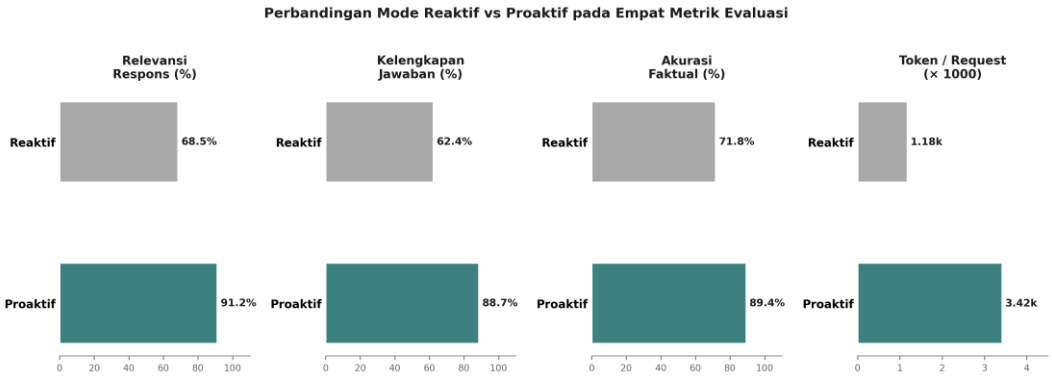
Modul / Fitur	Skenario	Berhasil	Gagal	Tingkat Sukses
Upload & Ekstraksi File	8	8	0	100%

Mode Proaktif — Injeksi Konteks	10	10	0	100%
Mode Reaktif — Baseline	5	5	0	100%
Visualisasi Grafik	5	4	1	80%
Manajemen Sesi	4	4	0	100%
Total	**32**	**31**	**1**	**96,9%**

4.3 Perbandingan Mode Proaktif dan Reaktif

Perbandingan kuantitatif antara mode proaktif dan reaktif pada empat metrik utama ditampilkan pada Gbr. 3. Mode proaktif unggul pada tiga metrik kualitas dengan selisih yang konsisten: relevansi respons mode proaktif mencapai 91,2 persen versus 68,5 persen mode reaktif (selisih 22,7 poin), kelengkapan jawaban 88,7 persen versus 62,4 persen (selisih 26,3 poin), dan akurasi faktual 89,4 persen versus 71,8 persen (selisih 17,6 poin). Selisih relevansi sebesar 22,7 poin tersebut secara mekanistik dapat dijelaskan oleh ketersediaan konten dokumen di dalam system message, sehingga model tidak perlu menebak konteks yang tidak dimiliki — pola yang konsisten dengan temuan Kaif et al. [8] pada Gemini MultiPDF Chatbot dan Neumann et al. [4] pada MoodleBot berbasis RAG.

Trade-off muncul pada metrik konsumsi token: mode proaktif rata-rata menggunakan 3.420 token per request dibandingkan 1.180 token pada mode reaktif, atau 2,9 kali lebih besar. Beban token tambahan ini proporsional dengan ukuran file yang diunggah dan menjadi konsekuensi langsung dari strategi injeksi konteks penuh. Untuk dokumen di bawah 50 halaman, beban tersebut masih masuk dalam kuota context window model claude-3-5-sonnet (200K token), sehingga aman digunakan tanpa truncation. Implikasi praktisnya adalah bahwa pendekatan proaktif lebih cocok untuk skenario tanya-jawab dokumen tunggal hingga sedang, sedangkan untuk korpus besar perlu dipasangkan dengan strategi chunking adaptif.



Gbr. 3. Perbandingan metrik kualitas mode reaktif versus mode proaktif pada empat dimensi evaluasi.

4.4 Kepuasan Pengguna (EUCS)

Evaluasi kepuasan pengguna menggunakan instrumen EUCS terhadap 15 responden menghasilkan skor rata-rata 4,1 dari 5 pada mode proaktif dan 3,2 dari 5 pada mode reaktif. Dimensi yang menunjukkan peningkatan terbesar adalah Content dengan kenaikan 1,2 poin (3,1 menjadi 4,3) dan Accuracy dengan kenaikan 1,0 poin (3,3 menjadi 4,3). Dimensi Ease of Use memperlihatkan peningkatan paling kecil sebesar 0,5 poin (3,7 menjadi 4,2) karena interaksi pengguna terhadap kedua mode pada dasarnya identik — yang berbeda hanyalah pengaktifan toggle. Skor proaktif 4,1 ini lebih tinggi dibandingkan benchmark Akbar [19] yang melaporkan 3,8 dari 5 pada sistem informasi

manajemen surat, sebuah indikasi bahwa beban kognitif pengguna dapat berkurang ketika sistem mengambil inisiatif pengelolaan konteks.

4.5 Diskusi dan Keterbatasan

Peningkatan performa mode proaktif tidak datang tanpa kompromi. Setiap permintaan API pada mode proaktif membawa beban konteks tambahan yang proporsional dengan ukuran file yang diunggah — trade-off antara kualitas respons dan biaya token yang nyata. Bender et al. [2] mengingatkan bahwa kemampuan retorika LLM dapat menyamarkan ketidakakuratan; sehingga peningkatan akurasi 17,6 poin yang dilaporkan di sini tetap perlu diinterpretasikan dengan hati-hati. Akurasi yang diukur adalah konsistensi terhadap dokumen sumber, bukan kebenaran absolut atas dunia luar.

Tiga keterbatasan utama yang perlu dicatat. Pertama, lapisan penyimpanan menggunakan SQLite yang sesuai untuk skenario tunggal pengembangan tetapi tidak terskala untuk lingkungan multi-pengguna konkuren. Kedua, ukuran sampel pengujian komparatif (30 pasangan kueri-dokumen) dan responden EUCS (15 orang) cukup untuk validasi awal namun belum cukup untuk generalisasi pada populasi pengguna yang lebih luas. Ketiga, evaluasi dilakukan terbatas pada model claude-3-5-sonnet — model dengan context window 200K yang relatif besar; performa pada model dengan context window lebih kecil dapat berbeda. Keterbatasan-keterbatasan ini sejalan dengan tantangan yang teridentifikasi pada studi sebelumnya tentang chatbot LLM untuk dukungan kerja [5] dan untuk repositori perangkat lunak [6].

5. KESIMPULAN

Mekanisme pembacaan konteks proaktif yang diimplementasikan pada aplikasi chatbot AI berbasis Flask dan OpenRouter API terbukti unggul secara kuantitatif dibandingkan pendekatan reaktif konvensional pada tiga metrik kualitas yang diukur. Relevansi respons meningkat 22,7 poin persentase, kelengkapan jawaban 26,3 poin, dan akurasi faktual 17,6 poin. Skor kepuasan pengguna pada instrumen EUCS mencapai 4,1 dari 5 pada mode proaktif dibandingkan 3,2 pada mode reaktif. Pengujian black-box terhadap 32 skenario menghasilkan tingkat keberhasilan 96,9 persen. Trade-off berupa konsumsi token 2,9 kali lebih besar masih masuk dalam kuota context window model yang digunakan untuk dokumen ukuran sedang. Penelitian ini menjawab pertanyaan awal: pendekatan injeksi konteks proaktif memang memberikan manfaat empiris yang signifikan pada platform ringan berbasis Flask multi-provider.

Tiga arah penelitian lanjutan yang konkret dan terukur. Pertama, implementasi strategi chunking adaptif berbasis panjang dokumen dengan target pengurangan konsumsi token 30 sampai 40 persen tanpa penurunan signifikan pada relevansi. Kedua, pengembangan dukungan multi-pengguna konkuren dengan migrasi penyimpanan dari SQLite ke PostgreSQL serta penambahan mekanisme isolasi sesi tingkat tenant. Ketiga, benchmark komparatif lintas-penyedia LLM (Claude, GPT-4, Gemini, dan model open-source) untuk mengidentifikasi model yang paling cost-efficient pada skenario injeksi konteks penuh.

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada Program Studi Manajemen Informatika, Politeknik Negeri Lampung atas dukungan fasilitas selama proses pengembangan dan pengujian sistem berlangsung.

KONTRIBUSI PENULIS

Nadiya Ghefira El Firsi: konseptualisasi, perancangan eksperimen, pengumpulan data EUCS, penulisan draf awal. Nandy Thaher Ulga: pengembangan modul backend Flask, integrasi OpenRouter

API, pengujian black-box. Elya Dianis: analisis data komparatif, validasi metrik, penyusunan tabel hasil. Ahmad Muhajir: pengembangan antarmuka frontend, implementasi fitur visualisasi grafik, dokumentasi teknis. Muhammad Reza Redo Islami: supervisi penelitian, peninjauan kritis seluruh draf, revisi akhir, dan korespondensi penyuntingan.

PERNYATAAN KONFLIK KEPENTINGAN

Penulis menyatakan tidak terdapat konflik kepentingan dalam penelitian, kepengarangan, dan/atau publikasi artikel ini.

REFERENSI

[1] M. A. Roshani et al., "Generative Large Language Model–Powered Conversational AI App for Personalized Risk Assessment: Case Study in COVID-19," JMIR Preprints, 2024, doi: 10.2196/preprints.67363.

[2] E. M. Bender, T. Gebru, A. McMillan-Major, dan S. Shmitchell, "On the Dangers of Stochastic Parrots: Can Language Models Be Too Big?" dalam Proc. 2021 ACM Conf. Fairness, Accountability, Transparency (FAccT), Virtual Event, 2021, hal. 610–623, doi: 10.1145/3442188.3445922.

[3] S. Ahmed, A. Rahman, dan M. Ashrafuzzaman, "A Systematic Review of AI and Machine Learning-Driven IT Support Systems: Enhancing Efficiency and Automation in Technical Service Management," Am. J. Scholarly Res. Innov., vol. 2, no. 2, hal. 75–101, 2023, doi: 10.63125/fd34sr03.

[4] A. T. Neumann, Y. Yin, S. Sowe, S. Decker, dan M. Jarke, "An LLM-Driven Chatbot in Higher Education for Databases and Information Systems," IEEE Trans. Educ., vol. 68, no. 1, hal. 103–116, Feb. 2025, doi: 10.1109/te.2024.3467912.

[5] Z. Gong et al., "Enhancing Trust in LLM Chatbots for Workplace Support Through User Experience Design and Prompt Engineering," dalam The Human Side of Service Engineering, vol. 143, AHFE Open Access, 2024, hal. 139–149, doi: 10.54941/ahfe1005092.

[6] S. Abedu, A. Abdellatif, dan E. Shihab, "LLM-Based Chatbots for Mining Software Repositories: Challenges and Opportunities," dalam Proc. 28th Int. Conf. Eval. Assess. Softw. Eng. (EASE), Salerno, Italy, 2024, hal. 201–210, doi: 10.1145/3661167.3661218.

[7] W. C. Choi dan C. I. Chang, "Enhancing Education with ChatGPT 4o and Microsoft Copilot: A Review of Opportunities, Challenges, and Student Perspectives on LLM-Based Text-to-Image Generation Models," Preprints, 2025, doi: 10.20944/preprints202503.2387.v1.

[8] M. Kaif, S. Sharma, dan S. Rana, "Gemini MultiPDF Chatbot: Multiple Document RAG Chatbot Using Gemini Large Language Model," Int. J. Res. Appl. Sci. Eng. Technol., vol. 12, no. 5, hal. 24–30, 2024, doi: 10.22214/ijraset.2024.61195.

[9] Z. Liu, P. Agrawal, S. Singhal, V. Madaan, dan K. Kanmani, "LPITutor: An LLM-Based Personalized Intelligent Tutoring System Using RAG and Prompt Engineering," PeerJ Comput. Sci., vol. 11, 2025, doi: 10.7717/peerj-cs.2991.

[10] H. Modran, I. C. Bogdan, D. Ursuțiu, C. Samoila, dan H. Modran, "LLM Intelligent Agent Tutoring in Higher Education Courses Using a RAG Approach," Preprints, 2024, doi: 10.20944/preprints202407.0519.v1.

[11] P. V. Khanh dan P. V. A. Tuan, "Building a Vietnamese Math Chatbot Based on RAG and LLM: System Design, Implementation and Experimental Evaluation," HNUE J. Sci. Nat. Sci., vol. 70, no. 4, hal. 43–56, 2025, doi: 10.18173/2354-1059.2025-0053.

[12] A. Ghosh, N. Heffernan, dan A. S. Lan, "Context-Aware Attentive Knowledge Tracing," dalam Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining (KDD), 2020, hal. 2330–2339, doi: 10.1145/3394486.3403282.

[13] A. Mukanova, M. Milosz, A. Dauletkaliyeva, A. Nazyrova, dan G. Yelibayeva, "LLM-Powered Natural Language Text Processing for Ontology Enrichment," Appl. Sci., vol. 14, no. 13, Art. no. 5860, 2024, doi: 10.3390/app14135860.

[14] K. Angkananon, R. Khunchamnan, B. Kongchana, dan P. Ploadaksorn, "Tourist Decision Support System for Hot Springs, Thailand," GeoJournal Tourism Geosites, vol. 53, no. 2, hal. 611–621, 2024, doi: 10.30892/gtg.53224-1236.

[15] P. Temniranrat, K. Kiratiratanapruk, A. Kitvimonrat, W. Sinthupinyo, dan S. Patarapuwadol, "A System for Automatic Rice Disease Detection from Rice Paddy Images Serviced via a Chatbot," Comput. Electron. Agric., vol. 185, Art. no. 106156, 2021, doi: 10.1016/j.compag.2021.106156.

[16] V. Apriana dan U. Nurhasanah, "Implementasi Metode Waterfall Pada Sistem Informasi Penjualan Berbasis Web," Artikel Ilm. Sist. Inf. Akuntansi (AKASIA), vol. 1, no. 2, hal. 106–111, Okt. 2021, doi: 10.31294/akasia.v1i2.585.

[17] F. Widianto, "Implementasi Model SDLC Dalam Perancangan Sistem Informasi Manajemen Perpustakaan Berbasis Web," Sistematis: J. Ilm. Sist. Inf., vol. 1, no. 1, hal. 60–68, Okt. 2024, doi: 10.69533/4w86eq90.

[18] D. S. H. Putra dan M. A. Darmawan, "Analisis Kepuasan Pengguna Sistem Informasi Administrasi Rumah Sakit (SIARS) dengan Model DeLone dan McLean," J. Sist. Inf. Bisnis, vol. 11, no. 1, hal. 78–85, 2021, doi: 10.21456/vol11iss1pp78-85.

[19] B. B. Akbar, "Analisis Kepuasan Pengguna Sistem Informasi Manajemen Surat Menggunakan Metode End User Computing Satisfaction," J. Pengembangan Sist. Inf. Inform., vol. 3, no. 1, hal. 24–32, Jan. 2022, doi: 10.47747/jpsii.v3i1.613.

[20] A. Nurseptaji dan Y. Ramdhani, "Sistem Informasi Perpustakaan dengan Implementasi Model Waterfall," INFORMASI: J. Inform. Sist. Inf., vol. 13, no. 1, hal. 61–79, 2021, doi: 10.37424/informasi.v13i1.68.