TIN: Terapan Informatika Nusantara
Vol 7, No 2, July 2026, page 1134-1144
ISSN 2722-7987 (Media Online)
Website https://ejurnal.seminar-id.com/index.php/tin
DOI 10.47065/tin.v7i2.10640

Rancang Bangun Asisten Virtual untuk Layanan Bengkel Berbasis Retrieval-Augmented Generation dan Tool Calling

Patrick Ardian Yoga Purnomo[*], Dwi Budi Santoso

Fakultas Teknologi Informasi dan Industri, Program Studi Sistem Informasi, Universitas Stikubank, Semarang, Indonesia
Email: [1,*]patrickardianyogapurnomo@mhs.unisbank.ac.id, [2]dbs@edu.unisbank.ac.id
Email Penulis Korespondensi: patrickardianyogapurnomo@mhs.unisbank.ac.id

Abstrak–Perkembangan Artificial Intelligence (AI) membuka peluang baru bagi usaha kecil dan menengah (UKM) untuk meningkatkan kualitas layanan pelanggan melalui otomatisasi komunikasi. Bengkel C Maestro Semarang masih melayani pertanyaan pelanggan secara manual melalui telepon dan aplikasi pesan, sehingga respons terhadap informasi biaya servis, ketersediaan jadwal, proses reservasi, dan status pengerjaan kendaraan sering mengalami keterlambatan, bergantung pada ketersediaan petugas, serta belum dapat diberikan secara konsisten selama 24 jam. Kondisi tersebut menyebabkan proses pelayanan menjadi kurang efisien dan berpotensi menurunkan kepuasan pelanggan. Penelitian ini bertujuan merancang dan membangun asisten virtual berbasis Retrieval-Augmented Generation (RAG) dengan memanfaatkan OpenRouter Application Programming Interface (API) untuk mengatasi permasalahan tersebut. Sistem dikembangkan menggunakan arsitektur client-server berbasis TypeScript yang berjalan pada lingkungan runtime Node.js. Pendekatan RAG diimplementasikan melalui mekanisme context injection dengan mengintegrasikan knowledge base internal bengkel yang memuat informasi layanan, estimasi harga, Frequently Asked Questions (FAQ), panduan diagnosis awal, dan informasi pendukung lainnya ke dalam system prompt dari Large Language Model (LLM). Selain itu, sistem mengimplementasikan mekanisme tool calling yang memungkinkan asisten virtual menjalankan fungsi bisnis seperti reservasi servis, pengecekan status kendaraan, dan eskalasi keluhan kepada montir. Data operasional disimpan menggunakan basis data SQLite, sedangkan antarmuka pengguna disediakan melalui aplikasi percakapan berbasis web dengan dukungan Server-Sent Events (SSE) untuk menghasilkan respons secara real-time. Hasil evaluasi menunjukkan bahwa sistem mampu menghasilkan respons yang relevan berdasarkan knowledge base, mengotomatisasi proses reservasi servis, serta melakukan eskalasi terhadap permintaan yang berada di luar kemampuannya. Penelitian ini memberikan kontribusi melalui pengembangan arsitektur asisten virtual yang mengintegrasikan Retrieval-Augmented Generation berbasis context injection dengan mekanisme tool calling menggunakan OpenRouter API untuk mendukung layanan informasi sekaligus proses operasional bengkel. Dengan demikian, asisten virtual yang dikembangkan menawarkan solusi digital yang efektif, praktis, dan ekonomis dalam mendukung digitalisasi layanan pelanggan pada bengkel otomotif skala usaha mikro, kecil, dan menengah.

Kata Kunci: Asisten Virtual; *Retrieval-Augmented Generation*; *OpenRouter API*; *Large Language Model*; Bengkel Otomotif

Abstract–The development of Artificial Intelligence (AI) has created new opportunities for Small and Medium Enterprises (SMEs) to improve customer service quality through communication automation. C Maestro Workshop in Semarang still handles customer inquiries manually through telephone and messaging applications, resulting in delayed responses to service cost inquiries, service reservations, and vehicle repair status updates. These services also depend on staff availability and cannot be provided consistently on a 24-hour basis, leading to inefficient customer service and potentially reducing customer satisfaction. This study aims to design and develop a Retrieval-Augmented Generation (RAG)-based virtual assistant utilizing the OpenRouter Application Programming Interface (API) to address these issues. The system was developed using a TypeScript-based client-server architecture running on the Node.js runtime environment. The RAG approach was implemented through a context injection mechanism by integrating the workshop's internal knowledge base, which includes service information, estimated service costs, Frequently Asked Questions (FAQ), initial diagnostic guidance, and other supporting information, into the system prompt of the Large Language Model (LLM). In addition, the system implements a tool-calling mechanism that enables the virtual assistant to perform business functions such as service reservations, vehicle repair status inquiries, and complaint escalation to mechanics. Operational data are stored in an SQLite database, while the user interface is provided through a web-based chat application with Server-Sent Events (SSE) support to deliver real-time responses. The evaluation results indicate that the system is capable of generating relevant responses based on the knowledge base, automating the service reservation process, and escalating requests beyond its capabilities. This study contributes by developing a virtual assistant architecture that integrates Retrieval-Augmented Generation (RAG) based on context injection with a tool calling mechanism using the OpenRouter API to support both information services and workshop operational processes. Therefore, the developed virtual assistant offers an effective, practical, and cost-efficient digital solution for supporting the digital transformation of customer services in small and medium-sized automotive workshops.

Keywords: Virtual Assistant; Retrieval-Augmented Generation; OpenRouter API; Large Language Model; Automotive Workshop

1. PENDAHULUAN

Perkembangan teknologi digital telah mengubah cara pelaku usaha memberikan layanan kepada pelanggan. Pemanfaatan kecerdasan buatan (Artificial Intelligence/AI) kini tidak lagi menjadi milik perusahaan besar saja usaha kecil dan menengah pun mulai menerapkannya untuk mendukung efisiensi operasional sekaligus kualitas pelayanan. Di sektor jasa otomotif misalnya, pelanggan tidak lagi datang ke bengkel semata untuk servis kendaraan. Mereka juga mengharapkan informasi yang bisa diakses kapan saja melalui berbagai platform digital. Hal ini mendorong bengkel untuk memanfaatkan teknologi agar penyampaian informasi kepada pelanggan berlangsung lebih cepat, efisien, dan mudah diakses.

Bengkel C Maestro di Kota Semarang mengalami persoalan serupa. Sebagai bengkel yang melayani perawatan dan perbaikan kendaraan, C Maestro menerima banyak pertanyaan berulang dari pelanggan soal harga layanan, cara booking servis, jam operasional, sampai status pengerjaan kendaraan. Selama ini seluruh proses tersebut masih

TIN: Terapan Informatika Nusantara
Vol 7, No 2, July 2026, page 1134-1144
ISSN 2722-7987 (Media Online)
Website https://ejurnal.seminar-id.com/index.php/tin
DOI 10.47065/tin.v7i2.10640

ditangani secara manual, sehingga staf harus terus-menerus terlibat menjawab pertanyaan yang pada dasarnya sama. Akibatnya waktu kerja staf banyak tersita untuk hal yang berulang, sementara pelanggan yang menghubungi di luar jam operasional atau saat bengkel sedang sibuk terpaksa menunggu lebih lama untuk mendapat respons. Diperlukan solusi yang dapat menyampaikan informasi secara otomatis tanpa mengorbankan kualitas pelayanan.

Large Language Model (LLM) memang unggul dalam memahami bahasa alami, tetapi penggunaannya secara mandiri tetap memiliki keterbatasan. Salah satunya, model cenderung menghasilkan informasi yang tidak akurat ketika tidak memiliki konteks yang cukup mengenai suatu domain. LLM juga tidak memiliki akses langsung ke data internal organisasi—daftar layanan, harga, maupun prosedur operasional yang terus diperbarui—sehingga respons yang dihasilkan belum tentu mencerminkan kondisi sebenarnya. *Retrieval-Augmented Generation* (RAG) dikembangkan untuk mengatasi hal ini, dengan menggabungkan kemampuan generatif LLM dan informasi dari basis pengetahuan eksternal. (Lewis et al., 2020) menjelaskan bahwa pendekatan ini mampu meningkatkan kualitas respons model bahasa melalui pemanfaatan informasi tambahan dari basis pengetahuan tertentu. (Gao et al., 2024) sependapat, menyatakan bahwa integrasi mekanisme retrieval pada sistem berbasis LLM meningkatkan relevansi jawaban dibandingkan model generatif tanpa sumber pengetahuan eksternal. Temuan serupa disampaikan (Ram et al., 2023), yang menunjukkan bahwa kombinasi retrieval dan model generatif menghasilkan respons lebih kontekstual dalam berbagai skenario percakapan. Di sisi lain, (OpenAI, 2024) melaporkan bahwa GPT-4 mampu memahami instruksi kompleks, sehingga berpotensi diterapkan pada berbagai sistem pelayanan pelanggan berbasis percakapan.

Beberapa penelitian sebelumnya telah mengembangkan chatbot berbasis Large Language Model (LLM) dan Retrieval-Augmented Generation (RAG) pada berbagai domain layanan. (Lewis et al., 2020) menunjukkan bahwa RAG mampu meningkatkan akurasi jawaban dengan mengombinasikan kemampuan generatif model bahasa dan proses retrieval dari sumber pengetahuan eksternal. (Ram et al., 2023) juga membuktikan bahwa integrasi mekanisme retrieval menghasilkan respons yang lebih kontekstual dibandingkan pendekatan generatif murni. (Gao et al., 2024; Saidi Rahman et al., 2025) mengemukakan bahwa RAG merupakan pendekatan yang efektif untuk mengurangi hallucination pada LLM melalui penyediaan konteks yang relevan sebelum proses generasi jawaban.

Pada penelitian di Indonesia, (Elysia & Herianto, 2024) , (Dharma et al., 2025) (Albert & Voutama, 2025), serta (Saman et al., 2025) menunjukkan bahwa penerapan RAG mampu meningkatkan kualitas layanan chatbot melalui pemanfaatan knowledge base sehingga menghasilkan respons yang lebih relevan sesuai kebutuhan pengguna. Selain itu, perkembangan mekanisme tool calling melalui ToolLLM dan Gorilla menunjukkan bahwa LLM mampu berinteraksi dengan berbagai Application Programming Interface (API) untuk menjalankan fungsi tertentu secara dinamis (Patil et al., 2023; Qin et al., 2023).

Meskipun demikian, penelitian-penelitian tersebut umumnya masih berfokus pada peningkatan kualitas layanan informasi sehingga chatbot lebih banyak dimanfaatkan sebagai media tanya jawab. Integrasi mekanisme tool calling dengan pendekatan Retrieval-Augmented Generation (RAG) dalam satu sistem yang mampu mendukung proses bisnis secara langsung masih terbatas. Selain itu, implementasi RAG berbasis context injection menggunakan OpenRouter API pada layanan bengkel otomotif skala UMKM juga belum banyak dilaporkan dalam penelitian sebelumnya. Oleh karena itu, masih terdapat kesenjangan penelitian dalam pengembangan asisten virtual yang tidak hanya mampu memberikan informasi yang kontekstual, tetapi juga menjalankan fungsi operasional melalui satu antarmuka percakapan.

Pada sektor layanan otomotif, kebutuhan pengguna tidak hanya terbatas pada memperoleh informasi, tetapi juga melakukan berbagai aktivitas operasional seperti reservasi servis, pengecekan status kendaraan, serta penyampaian keluhan melalui satu media komunikasi. Untuk mengatasi kesenjangan tersebut, penelitian ini mengintegrasikan teknologi *Retrieval-Augmented Generation* (RAG) berbasis *context injection*, mekanisme *tool calling*, dan OpenRouter API dalam satu sistem asisten virtual untuk layanan bengkel. Integrasi tersebut memungkinkan model bahasa tidak hanya menghasilkan jawaban berdasarkan *knowledge base* internal bengkel, tetapi juga mengeksekusi fungsi bisnis melalui pemanggilan *tools* yang terhubung dengan basis data. Dengan demikian, penelitian ini berkontribusi melalui pengembangan arsitektur asisten virtual yang menggabungkan kemampuan pemahaman bahasa alami, pengambilan informasi berbasis RAG, dan otomatisasi layanan operasional seperti reservasi servis, pengecekan status kendaraan, serta eskalasi kepada mekanik dalam satu sistem yang sesuai untuk mendukung digitalisasi layanan bengkel otomotif skala UMKM.

Berdasarkan berbagai penelitian terdahulu, Retrieval-Augmented Generation (RAG) terbukti mampu meningkatkan kualitas respons Large Language Model melalui pemanfaatan basis pengetahuan eksternal, sedangkan pendekatan tool calling memungkinkan model bahasa berinteraksi dengan layanan atau aplikasi eksternal untuk menjalankan berbagai fungsi operasional (Gao et al., 2024; Qin et al., 2023; Schick et al., 2023). Meskipun demikian, sebagian besar penelitian masih membahas kedua pendekatan tersebut secara terpisah. Penelitian mengenai RAG umumnya berfokus pada peningkatan kualitas jawaban berbasis dokumen, sedangkan penelitian mengenai tool calling lebih banyak menitikberatkan pada kemampuan model dalam menggunakan API atau tool eksternal tanpa mengintegrasikannya dengan *knowledge base* pada domain tertentu.

TIN: Terapan Informatika Nusantara
Vol 7, No 2, July 2026, page 1134-1144
ISSN 2722-7987 (Media Online)
Website https://ejurnal.seminar-id.com/index.php/tin
DOI 10.47065/tin.v7i2.10640

2. METODOLOGI PENELITIAN

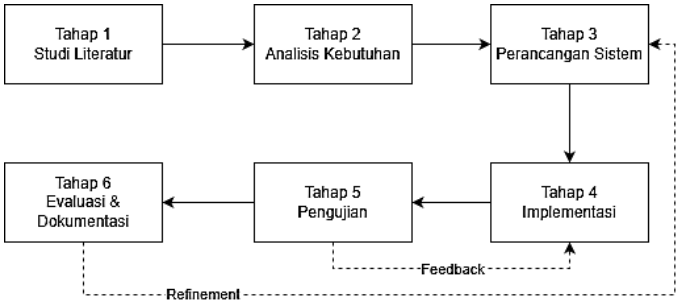
2.1 Kerangka Dasar Penelitian

Penelitian ini menggunakan metode Research and Development (R&D) untuk menghasilkan dan menguji produk perangkat lunak berupa asisten virtual berbasis *Retrieval-Augmented Generation* (RAG) yang terintegrasi dengan *tool calling* guna mendukung pelayanan informasi pada Bengkel C Maestro Semarang. Metode R&D dipilih karena penelitian tidak hanya menghasilkan produk perangkat lunak, tetapi juga mengevaluasi fungsionalitasnya agar sesuai dengan kebutuhan pengguna dan dapat diterapkan pada lingkungan nyata (Sugiyono, 2022) Produk yang dikembangkan berupa aplikasi asisten virtual berbasis web yang mampu menjawab pertanyaan pelanggan berdasarkan *knowledge base* serta menjalankan fungsi operasional bengkel melalui mekanisme *tool calling.*

Objek penelitian adalah Bengkel C Maestro Semarang sebagai studi kasus implementasi sistem. Sistem dikembangkan menggunakan arsitektur *client-server* berbasis Node.js dengan memanfaatkan OpenRouter API sebagai penyedia layanan *Large Language Model* (LLM), SQLite sebagai basis data, serta antarmuka berbasis web. Pemilihan teknologi tersebut disesuaikan dengan kebutuhan implementasi sistem yang ringan, mudah dikembangkan, dan mendukung integrasi layanan kecerdasan buatan. Pengembangan sistem dilakukan menggunakan metode Agile Software Development karena mampu mendukung proses pengembangan secara bertahap sehingga setiap fungsi dapat dievaluasi dan disempurnakan sesuai kebutuhan pengguna (Hidayah Nova et al., 2022; Pressman & Maxim, 2020)

2.2 Tahapan Penelitian

Gambar berikut menunjukkan alur penelitian yang digunakan dalam pengembangan sistem. Setiap tahapan menghasilkan keluaran yang menjadi masukan pada tahap berikutnya sehingga proses pengembangan berlangsung secara sistematis.



Gambar 1. Tahapan Penelitian

Gambar 1 menunjukkan alur tahapan penelitian yang digunakan dalam pengembangan sistem. Setiap tahapan menghasilkan keluaran yang menjadi masukan bagi tahapan berikutnya sehingga proses pengembangan berlangsung secara sistematis. Tahap ini bertujuan untuk mengumpulkan dan mempelajari berbagai referensi yang berkaitan dengan penelitian sebagai dasar dalam pengembangan sistem:

1. Tahap pertama adalah mengidentifikasi kebutuhan sistem, baik kebutuhan pengguna maupun kebutuhan fungsional dan nonfungsional yang akan menjadi acuan dalam pengembangan.
2. Tahapan selanjutnya berfokus pada penyusunan desain sistem, meliputi arsitektur, alur proses, dan komponen yang akan diimplementasikan.
3. Tahap implementasi merupakan proses pembangunan sistem berdasarkan rancangan yang telah dibuat pada tahap sebelumnya.
4. Tahap ini bertujuan untuk memastikan sistem telah berfungsi sesuai dengan kebutuhan melalui serangkaian proses pengujian.
5. Tahap terakhir dilakukan untuk mengevaluasi hasil pengembangan sistem serta mendokumentasikan seluruh proses dan hasil penelitian.

2.2.1 Studi Literatur

Tahap studi literatur dilakukan untuk mengumpulkan referensi yang berkaitan dengan Artificial Intelligence (AI), Large Language Model (LLM), Retrieval-Augmented Generation (RAG), tool calling, OpenRouter API, Node.js, serta metode pengembangan perangkat lunak. Referensi diperoleh dari jurnal ilmiah, buku, prosiding konferensi, dan dokumentasi resmi yang relevan. Hasil studi literatur digunakan sebagai dasar dalam menentukan kebutuhan sistem dan merancang solusi yang sesuai (Gao et al., 2024; Lewis et al., 2020; Russell et al., 2021; Zhao et al., 2023)

2.2.2 Analisis Kebutuhan

Analisis kebutuhan dilakukan melalui observasi proses bisnis Bengkel C Maestro Semarang untuk mengidentifikasi kebutuhan pengguna. Tahap ini menghasilkan kebutuhan fungsional berupa layanan informasi bengkel, reservasi servis, pengecekan status kendaraan, dan eskalasi kepada montir, sedangkan kebutuhan nonfungsional meliputi kemudahan penggunaan, keamanan, kecepatan respons, dan kemudahan pemeliharaan sistem (Pressman & Maxim, 2020)


TIN: Terapan Informatika Nusantara
Vol 7, No 2, July 2026, page 1134-1144
ISSN 2722-7987 (Media Online)
Website https://ejurnal.seminar-id.com/index.php/tin
DOI 10.47065/tin.v7i2.10640

2.2.3 Perancangan Sistem

Pada tahap ini disusun rancangan arsitektur *client-server*, struktur basis data SQLite, desain antarmuka pengguna, penyusunan *knowledge base* dalam format Markdown, serta rancangan integrasi OpenRouter API. Selain itu, dirancang mekanisme *context injection* untuk memberikan konteks kepada model bahasa dan mekanisme *tool calling* agar model dapat memanggil fungsi-fungsi operasional yang tersedia pada sistem (Mozilla Developer Network, 2024; OpenJS Foundation, 2024).

2.2.4 Implementasi

Rancangan yang telah dibuat kemudian diimplementasikan menggunakan TypeScript pada platform Node.js. Implementasi meliputi pembangunan antarmuka pengguna berbasis web, integrasi OpenRouter API, pengelolaan *knowledge base*, penerapan mekanisme *Retrieval-Augmented Generation* berbasis *context injection*, serta pengembangan fungsi *tool calling* yang mendukung proses reservasi, pengecekan status kendaraan, registrasi pelanggan, dan eskalasi kepada mekanik (OpenAI, 2024)

2.2.5 Pengujian Sistem

Tahap pengujian dilakukan menggunakan metode Black Box Testing untuk memastikan seluruh fungsi sistem berjalan sesuai kebutuhan. Pengujian meliputi fungsi pemberian informasi, reservasi servis, pengecekan status kendaraan, penggunaan tool calling, serta eskalasi kepada montir. Hasil pengujian kemudian dievaluasi untuk mengetahui tingkat kesesuaian sistem terhadap kebutuhan pengguna (Pressman et al., 2020; Sommerville, 2016).

2.2.6 Evaluasi & Dokumentasi

Evaluasi dilakukan terhadap hasil implementasi dan pengujian untuk menilai keberhasilan sistem dalam memenuhi tujuan penelitian. Seluruh proses pengembangan, hasil pengujian, serta temuan selama penelitian didokumentasikan sebagai bahan analisis dan rekomendasi pengembangan pada penelitian selanjutnya. Tahap ini menghasilkan kesimpulan mengenai efektivitas penerapan Retrieval-Augmented Generation dan tool calling dalam meningkatkan layanan informasi pada Bengkel C Maestro Semarang.

3. HASIL DAN PEMBAHASAN

3.1 Hasil Implementasi Sistem

Penelitian ini menghasilkan sistem asisten virtual berbasis Retrieval-Augmented Generation (RAG) yang terintegrasi dengan mekanisme tool calling untuk mendukung layanan informasi pada Bengkel C Maestro Semarang. Sistem dikembangkan menggunakan metodologi Agile Software Development melalui tahapan analisis kebutuhan, perancangan, implementasi, pengujian, dan evaluasi sebagaimana dijelaskan pada bagian metodologi penelitian. Hasil implementasi menunjukkan sistem mampu menjawab pertanyaan pelanggan berdasarkan knowledge base. Sistem juga menjalankan fungsi operasional seperti reservasi servis, pengecekan status kendaraan, registrasi pelanggan, dan eskalasi kepada mekanik melalui pemanggilan fungsi (tool calling).

Implementasi RAG memanfaatkan pendekatan context injection. Informasi dari knowledge base dimuat ke dalam system prompt sebelum dikirimkan ke Large Language Model (LLM) melalui OpenRouter API. Model bahasa memperoleh konteks yang sesuai domain Bengkel C Maestro Semarang. Hasilnya, respons yang dihasilkan lebih relevan dan risiko hallucination pada pertanyaan spesifik seputar layanan bengkel berkurang (Gao et al., 2024; Lewis et al., 2020).

Sistem juga menerapkan tool calling, yang memungkinkan model bahasa memanggil fungsi yang tersedia pada aplikasi sesuai kebutuhan percakapan. Asisten virtual dengan demikian tidak sekadar menjawab pertanyaan, melainkan juga berinteraksi langsung dengan basis data untuk menjalankan proses bisnis bengkel. Hasil implementasi selanjutnya dijelaskan melalui pembahasan mengenai gambaran umum sistem, implementasi RAG, implementasi tool calling, implementasi antarmuka pengguna, implementasi keamanan sistem, serta hasil pengujian menggunakan metode Black Box Testing.

3.1.1 Gambaran Umum Sistem

Sistem yang dikembangkan pada penelitian ini adalah asisten virtual berbasis Retrieval-Augmented Generation (RAG) untuk mendukung layanan informasi dan operasional Bengkel C Maestro Semarang. Sistem berbentuk aplikasi berbasis web sehingga dapat diakses melalui peramban (web browser) oleh pelanggan maupun administrator tanpa instalasi aplikasi tambahan. Tujuan utama pengembangan adalah meningkatkan kecepatan dan konsistensi pelayanan, dengan menyediakan informasi layanan secara otomatis, membantu proses reservasi servis, memberikan informasi status kendaraan, serta memfasilitasi konsultasi awal mengenai permasalahan kendaraan.

Arsitektur sistem menerapkan model *client-server* yang terdiri atas antarmuka pengguna (*client*), server aplikasi berbasis Node.js, basis data SQLite, *knowledge base*, dan layanan *Large Language Model* (LLM) yang diakses melalui OpenRouter API. Pendekatan ini sejalan dengan implementasi chatbot berbasis website yang mengintegrasikan RAG sebagai pendukung layanan informasi digital (Dharma et al., 2025). Antarmuka pengguna berfungsi sebagai media

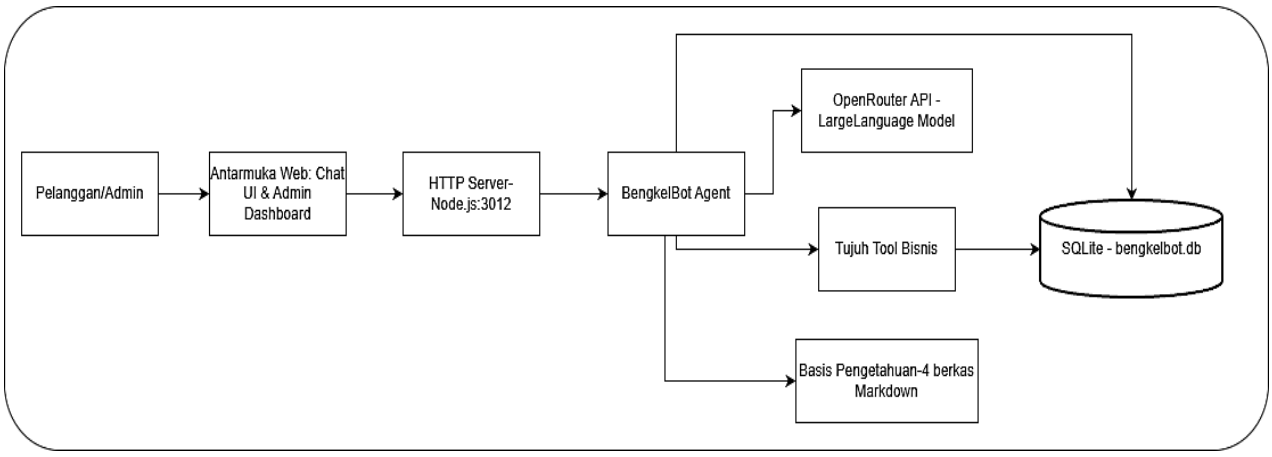
TIN: Terapan Informatika Nusantara
Vol 7, No 2, July 2026, page 1134-1144
ISSN 2722-7987 (Media Online)
Website https://ejurnal.seminar-id.com/index.php/tin
DOI 10.47065/tin.v7i2.10640

interaksi pelanggan menggunakan konsep *conversational interface*. Server aplikasi bertindak sebagai pusat orkestrasi yang memproses seluruh permintaan pengguna, mulai dari memvalidasi masukan, mengelola sesi percakapan, mengambil konteks yang relevan dari *knowledge base* melalui mekanisme *Retrieval-Augmented Generation* (RAG), hingga menentukan apakah permintaan dapat dijawab secara langsung atau memerlukan pemanggilan fungsi (*tool calling*). Arsitektur ini menerapkan pemisahan antara lapisan presentasi, logika aplikasi, dan pengelolaan data sehingga proses pengembangan, pemeliharaan, dan perluasan fungsi sistem dapat dilakukan secara lebih modular. Pendekatan tersebut sejalan dengan arsitektur sistem berbasis LLM yang mengintegrasikan mekanisme *retrieval* dan penggunaan *external tools* untuk meningkatkan kemampuan sistem dalam memberikan respons yang akurat sekaligus menjalankan fungsi operasional (Gao et al., 2024; Wang et al., 2024)

Setiap pertanyaan pengguna terlebih dahulu diproses melalui mekanisme RAG berbasis *context injection*. Server memuat informasi relevan dari *knowledge base*, lalu menyisipkannya ke dalam system prompt sebelum dikirim ke model bahasa melalui OpenRouter API. Model memperoleh konteks yang sesuai domain Bengkel C Maestro Semarang. Jawaban yang dihasilkan pun lebih relevan, dan risiko munculnya informasi yang tidak sesuai (hallucination) berkurang (Gao et al., 2024; Lewis et al., 2020)

Apabila pertanyaan pengguna memerlukan tindakan operasional seperti reservasi servis, pengecekan status kendaraan, atau eskalasi ke mekanik, sistem menjalankan mekanisme tool calling. Model bahasa memilih fungsi yang sesuai berdasarkan konteks percakapan, lalu server mengeksekusi fungsi tersebut untuk berinteraksi dengan basis data atau layanan internal. Hasil eksekusi dikembalikan ke model bahasa untuk disusun menjadi respons yang mudah dipahami pengguna. Lewat jalur ini, asisten virtual bisa langsung menjalankan proses bisnis bengkel, bukan sekadar menjawab pertanyaan, semua melalui satu antarmuka percakapan.

Hasil implementasi menunjukkan kelima komponen ini, yaitu antarmuka pengguna, *knowledge base*, OpenRouter API, basis data SQLite, dan mekanisme tool calling, berjalan sebagai satu sistem terintegrasi. Integrasi ini menjadi dasar bagi pembahasan fitur-fitur utama pada subbab berikutnya, meliputi implementasi RAG, implementasi tool calling, antarmuka pengguna, keamanan sistem, serta pengujian fungsional menggunakan metode Black Box Testing.



Gambar 1. Arsitektur sistem yang telah diimplementasikan

Berdasarkan Gambar 2, arsitektur sistem terdiri atas beberapa komponen utama yang saling terintegrasi sebagai berikut:

1. Antarmuka Pengguna (User Interface) berfungsi sebagai media interaksi antara pengguna dengan sistem melalui fitur chat dan dashboard administrasi.
2. Server Aplikasi bertugas menerima permintaan dari pengguna, mengelola alur pemrosesan, serta mengoordinasikan komunikasi dengan seluruh komponen sistem.
3. OpenRouter API (Large Language Model) digunakan sebagai penyedia layanan model bahasa untuk memahami pertanyaan pengguna dan menghasilkan respons yang sesuai.
4. Basis Pengetahuan (Knowledge Base) berisi dokumen referensi dalam format Markdown yang dimanfaatkan oleh mekanisme Retrieval-Augmented Generation (RAG) untuk memberikan informasi yang relevan.
5. Tool Calling menyediakan fungsi-fungsi bisnis yang dapat dipanggil oleh model bahasa untuk melakukan operasi tertentu sesuai kebutuhan pengguna.
6. Basis Data SQLite digunakan untuk menyimpan data aplikasi, termasuk data yang diperlukan dalam proses operasional sistem.

3.1.2 Implementasi Retrieval-Augmented Generation (RAG)

Komponen utama yang membedakan sistem ini dengan chatbot konvensional adalah penerapan metode *Retrieval-Augmented Generation* (RAG). Pada penelitian ini, implementasi RAG dilakukan menggunakan pendekatan *context injection*, yaitu seluruh dokumen pengetahuan yang relevan dimasukkan ke dalam *system prompt* sebelum model bahasa menghasilkan respons.

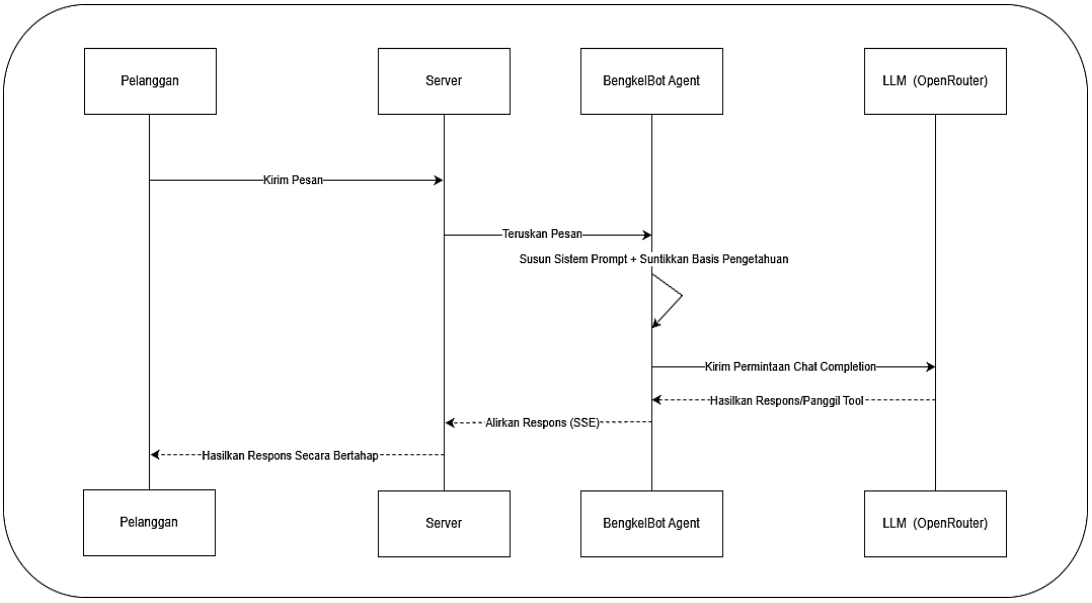
TIN: Terapan Informatika Nusantara
Vol 7, No 2, July 2026, page 1134-1144
ISSN 2722-7987 (Media Online)
Website https://ejurnal.seminar-id.com/index.php/tin
DOI 10.47065/tin.v7i2.10640

Pendekatan tersebut dipilih karena jumlah dokumen yang dimiliki Bengkel C Maestro relatif kecil sehingga tidak memerlukan penggunaan *vector database*. Selain lebih sederhana, pendekatan ini juga mempercepat proses implementasi serta mempermudah pemeliharaan data oleh administrator. Rincian *knowledge base* ditunjukkan pada Tabel 1.

Tabel 1. *Knowledge* Base Sistem

Dokumen	Isi Informasi
Faq.md	Informasi umum bengkel, jam operasional, garansi, dan FAQ pelanggan.
Services.md	Daftar layanan, estimasi biaya servis, jenis oli, serta suku cadang.
Diagnostic.md	Basis pengetahuan diagnosis awal berdasarkan gejala kendaraan.
Slang.md	Daftar bahasa sehari-hari dan istilah lokal yang digunakan pelanggan.

Berdasarkan Tabel 1, sistem menggunakan empat dokumen utama sebagai sumber konteks yaitu FAQ, daftar layanan, istilah bahasa lokal, dan panduan diagnosa awal. FAQ menjawab pertanyaan umum pelanggan mengenai layanan bengkel. Daftar layanan memuat jenis servis, estimasi biaya, dan waktu pengerjaan. Istilah bahasa lokal membantu model mengenali kosakata yang umum dipakai pelanggan dalam percakapan sehari-hari. Panduan diagnosa awal berisi gejala kerusakan kendaraan beserta rekomendasi pemeriksaan awal. Dengan pembagian ini, administrator bisa memperbarui satu dokumen tanpa menyentuh dokumen lain. Gambar 3 menunjukkan alur implementasi Retrieval-Augmented Generation (RAG) yang digunakan dalam penelitian ini.



Gambar 2. Menunjukkan alur implementasi metode Retrieval-Augmented Generation.

Berdasarkan Gambar 3 dapat diketahui bahwa sistem tidak melakukan pencarian informasi secara langsung ke internet. Seluruh jawaban dihasilkan berdasarkan dokumen internal bengkel yang telah disusun sebelumnya sehingga informasi yang diberikan tetap konsisten dan sesuai dengan kondisi operasional bengkel.

3.1.3 Implementasi Tool Calling

Selain menghasilkan jawaban dalam bentuk teks, sistem juga mampu menjalankan berbagai fungsi operasional melalui mekanisme *tool calling*. Pada mekanisme ini, model bahasa dapat menentukan kapan harus memanggil suatu fungsi (*tool*) berdasarkan maksud (*intent*) dari pertanyaan pengguna. Sebanyak tujuh *tool* berhasil diimplementasikan pada penelitian ini. Daftar fungsi tersebut ditampilkan pada Tabel 2.

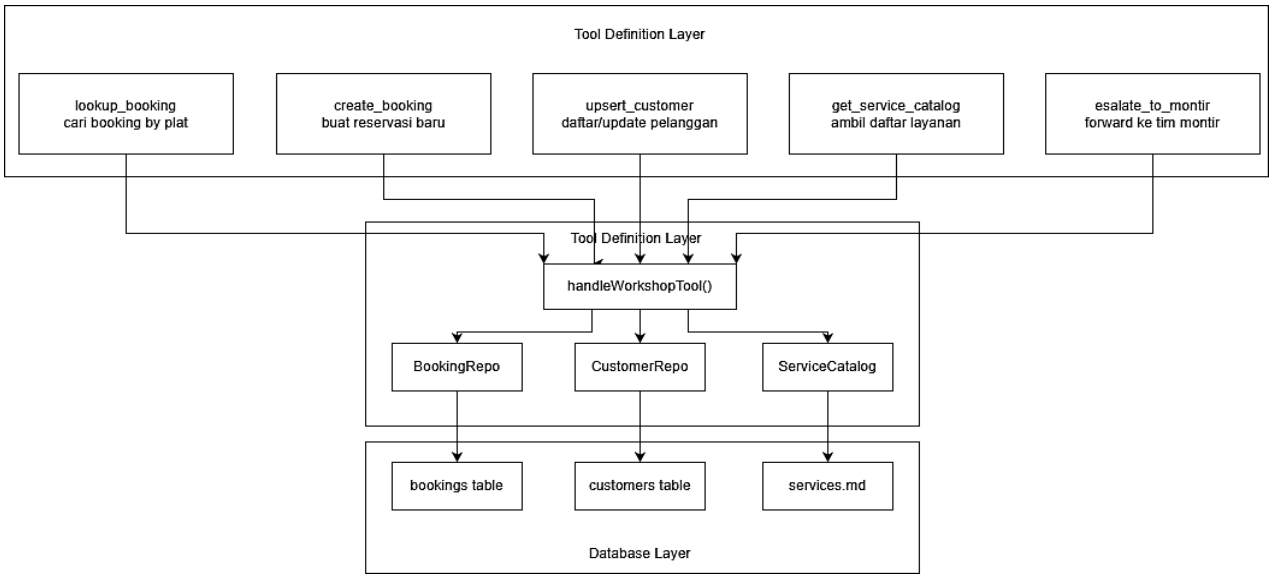
Tabel 2. Implementasi Tool Calling

No	Tool	Parameter	Return	Trigger Intent
1	*lookup_booking*	plate_number: string	Data booking (status, layanan, dll)	"Cek status B 1234 CD"
2	*create_booking*	service_type, plate_number, car_model, preferred_date	Konfirmasi booking	"Saya mau booking..."
3	*upsert_customer*	name, phone, car_model, plate_number	Data pelanggan	Sebelum create_booking
4	*update_booking_status*	booking_id, status, notes	Konfirmasi update	Montir: /status B1234CD done
5	*get_service_catalog*	(none)	Daftar 35 layanan +	"Berapa harga servis?"

TIN: Terapan Informatika Nusantara
Vol 7, No 2, July 2026, page 1134-1144
ISSN 2722-7987 (Media Online)
Website https://ejurnal.seminar-id.com/index.php/tin
DOI 10.47065/tin.v7i2.10640

No	Tool	Parameter	Return	Trigger Intent
			harga	
6	*escalate_to_montir*	customer_name, plate_number, summary	Konfirmasi eskalasi	Pelanggan marah/gejala kompleks
7	*send_whatsapp_message*	phone_number, message	Status pengiriman	Konfirmasi otomatis

Berdasarkan Tabel 2, penelitian ini mengimplementasikan tujuh fungsi utama yang mendukung proses bisnis Bengkel C Maestro Semarang. lookup_booking mencari data reservasi berdasarkan nomor polisi kendaraan; create_booking membuat reservasi servis baru dari informasi yang diberikan pengguna; upsert_customer menambahkan atau memperbarui data pelanggan sebelum reservasi diproses. Ada juga update_booking_status, yang dipakai administrator atau mekanik untuk memperbarui status pengerjaan kendaraan agar pelanggan langsung mendapat info terbaru. get_service_catalog mengambil daftar layanan beserta estimasi biaya dari *knowledge base*, escalate_to_montir meneruskan percakapan ke mekanik saat ditemukan masalah yang perlu penanganan lebih lanjut, dan send_whatsapp_message mengirim notifikasi atau konfirmasi ke pelanggan lewat WhatsApp. Implementasi arsitektur tool calling ditunjukkan pada Gambar 4.



Gambar 3. Arsitektur Implementasi Tool Calling

Mekanisme tool calling dibagi menjadi tiga lapisan: Tool Definition Layer, Tool Handler Layer, dan Database Layer. Tool Definition Layer berisi definisi seluruh fungsi yang dapat dipanggil model bahasa, seperti lookup_booking, create_booking, upsert_customer, get_service_catalog, dan escalate_to_montir. Setiap fungsi memiliki parameter berbeda sesuai kebutuhan proses bisnis, sehingga model dapat memilih fungsi yang tepat berdasarkan konteks percakapan. Setelah model menentukan fungsi yang akan dipanggil, permintaan diteruskan ke Tool Handler Layer melalui fungsi handleWorkshopTool(). Lapisan ini melakukan validasi parameter, menentukan repositori yang sesuai, dan mengoordinasikan komunikasi antara model bahasa dengan sumber data pada sistem. Seluruh logika bisnis jadi terpusat pada satu komponen, yang membuat pengelolaan fungsi lebih terstruktur dan mudah dikembangkan. Tool Handler Layer sendiri berinteraksi dengan Database Layer melalui tiga repositori: BookingRepo, CustomerRepo, dan ServiceCatalog. BookingRepo mengelola data reservasi pada tabel bookings, CustomerRepo mengelola data pelanggan pada tabel customers, dan ServiceCatalog mengambil informasi layanan dari dokumen services.md yang menjadi bagian dari *knowledge base*. Setelah data selesai diambil atau disimpan, hasil eksekusi dikembalikan ke model bahasa untuk disusun menjadi respons akhir yang ditampilkan kepada pengguna melalui antarmuka percakapan.

Hasil implementasi menunjukkan mekanisme tool calling berhasil mengintegrasikan kemampuan pemahaman bahasa alami LLM dengan proses bisnis Bengkel C Maestro Semarang. Sistem menjawab pertanyaan pelanggan berdasarkan *knowledge base* sekaligus menjalankan layanan operasional langsung, semua dalam satu antarmuka percakapan. Pelanggan cukup mengetik apa yang mereka butuhkan, tanpa harus berpindah halaman untuk reservasi, cek status kendaraan, atau mencari informasi layanan.

3.1.4 Implementasi Antarmuka Pengguna

Antarmuka pengguna (User Interface/UI) dirancang berbasis web agar pelanggan maupun administrator bisa mengaksesnya tanpa instalasi aplikasi tambahan. Perancangan mengutamakan kemudahan penggunaan (usability): tampilan sederhana, responsif, dan mudah dipahami. Lewat antarmuka ini, pengguna berinteraksi langsung dengan sistem virtual memakai bahasa alami untuk memperoleh informasi layanan, melakukan reservasi servis, mengecek status kendaraan, atau berkonsultasi awal soal masalah kendaraan.

TIN: Terapan Informatika Nusantara
Vol 7, No 2, July 2026, page 1134-1144
ISSN 2722-7987 (Media Online)
Website https://ejurnal.seminar-id.com/index.php/tin
DOI 10.47065/tin.v7i2.10640

Gambar 4. Tampilan antarmuka pelanggan

Gambar 5 menunjukkan antarmuka percakapan yang dirancang menyerupai aplikasi perpesanan, agar pengguna bisa berinteraksi secara intuitif dengan bahasa sehari-hari. Saat pengguna mengirim pertanyaan, sistem meneruskannya ke server untuk diproses lewat mekanisme Retrieval-Augmented Generation (RAG). Jika pertanyaan hanya butuh informasi, sistem menjawab berdasarkan *knowledge base*. Jika pengguna meminta tindakan seperti reservasi atau cek status kendaraan, sistem menjalankan tool calling lebih dulu sebelum menjawab. Hasilnya terasa seperti mengobrol dengan petugas layanan pelanggan, bukan mengisi formulir.

Gambar 5. Tampilan Dashboard Admin

Gambar 6 menunjukkan dashboard administrator, tempat berbagai fungsi pengelolaan data berada. Administrator bisa memperbarui isi *knowledge base*, mengelola daftar layanan, memeriksa data pelanggan, melihat data reservasi, sesi chat pelanggan dan memantau status pengerjaan kendaraan. Setiap perubahan lewat dashboard langsung dipakai sistem pada proses *context injection*, tanpa perlu melatih ulang model bahasa. Artinya, apa pun yang diketik chatbot ke pelanggan selalu mengikuti kondisi terbaru di Bengkel C Maestro Semarang.

Hasil implementasi menunjukkan kedua bagian antarmuka ini bekerja sesuai perannya masing-masing: pelanggan mendapat informasi dan layanan lewat satu percakapan, administrator mengelola data lewat dashboard terpisah. Karena antarmuka, RAG, tool calling, dan basis data saling terhubung, pelayanan yang dihasilkan lebih cepat dan konsisten dibanding proses manual, sekaligus lebih mudah dikelola dari sisi administrator.

3.1.5 Implementasi Keamanan Sistem.

Keamanan sistem diterapkan untuk melindungi data pelanggan, data reservasi, serta layanan yang terintegrasi dengan OpenRouter API. Selain menjaga kerahasiaan informasi, mekanisme keamanan ini mencegah penyalahgunaan layanan, memastikan hanya pengguna berwenang yang mengakses fitur administrasi, dan menjaga integritas data yang diproses sistem. Implementasi keamanan dilakukan pada beberapa lapisan: autentikasi administrator, validasi sesi percakapan, pembatasan akses API, hingga validasi data masukan. Tujuannya, setiap permintaan yang masuk ke sistem sudah memenuhi syarat keamanan sebelum diteruskan ke proses bisnis maupun ke layanan Large Language Model (LLM). Tabel 3 merangkum lima mekanisme yang diterapkan. Rate Limiting membatasi jumlah permintaan dalam periode tertentu, jadi kalau ada pihak yang mencoba membanjiri API dengan permintaan berulang, sistem sudah punya penahannya. HMAC Token memvalidasi sesi percakapan, sehingga server bisa memastikan keaslian tiap permintaan yang masuk sebelum memprosesnya lebih jauh.

TIN: Terapan Informatika Nusantara
Vol 7, No 2, July 2026, page 1134-1144
ISSN 2722-7987 (Media Online)
Website https://ejurnal.seminar-id.com/index.php/tin
DOI 10.47065/tin.v7i2.10640

Tabel 3. Implementasi Keamanan Sistem

Mekanisme	Tujuan
Rate Limiting	Mencegah penyalahgunaan API
HMAC Token	Validasi sesi percakapan
Session Authentication	Autentikasi administrator
Input Validation	Mencegah serangan injeksi
API Key Protection	Melindungi kredensial OpenRouter

Session Authentication memastikan hanya administrator yang sudah berhasil login yang bisa membuka dashboard dan mengelola data sistem. Input Validation bekerja di lapisan lain: setiap parameter dari pengguna maupun model bahasa diperiksa dulu sebelum dipakai untuk tool calling atau disimpan ke basis data, untuk menutup celah kesalahan format data atau serangan injeksi. Terakhir, API Key Protection menjaga kerahasiaan kredensial OpenRouter API. API Key tidak pernah disimpan atau dikirim ke sisi klien; ia hanya hidup di environment variables pada server, sehingga cuma aplikasi backend yang bisa mengaksesnya. Kalau kredensial ini sampai bocor ke sisi klien, risikonya jauh lebih besar, karena itulah pendekatan ini dipilih. Hasil implementasi menunjukkan kelima mekanisme ini saling menutup celah yang berbeda: Rate Limiting dan API Key Protection menahan penyalahgunaan dari luar, HMAC Token dan Session Authentication menjaga siapa yang boleh masuk, sedangkan Input Validation menjaga data yang sudah masuk tetap bersih sebelum menyentuh basis data atau OpenRouter API. Sistem ini berjalan secara operasional sekaligus memenuhi kebutuhan keamanan yang layak untuk asisten virtual berbasis web

3.1.6 Hasil Pengujian Black Box.

Pengujian sistem menggunakan metode *Black Box Testing* untuk mengevaluasi kesesuaian fungsi sistem dengan kebutuhan fungsional yang telah ditentukan. Metode ini memfokuskan pengujian pada masukan (input) dan keluaran (output), tanpa memperhatikan implementasi kode program di dalamnya. Setiap fungsi diuji lewat skenario yang mewakili aktivitas pengguna maupun administrator, untuk memastikan seluruh fitur berjalan sesuai spesifikasi yang dirancang (Pressman & Maxim, 2020).

Skenario pengujian mencakup autentikasi administrator, percakapan dengan chatbot, pencarian data reservasi, pembuatan reservasi servis, pengelolaan data pelanggan, pengambilan informasi layanan, eskalasi kepada mekanik, pembaruan status kendaraan, dan pengiriman notifikasi kepada pelanggan. Tiap skenario dievaluasi dengan membandingkan hasil yang diharapkan dan hasil aktual saat sistem dijalankan.

Tabel 4. Hasil Pengujian Black Box

Skenario Pengujian	Hasil yang Diharapkan	Hasil Pengujian	Status
Menanyakan harga servis	Sistem menampilkan estimasi biaya sesuai *knowledge* base	Berhasil menampilkan estimasi harga	Berhasil
Booking servis	Data booking tersimpan ke basis data	Booking berhasil dibuat	Berhasil
Cek status kendaraan	Sistem menampilkan status berdasarkan plat nomor	Status berhasil ditampilkan	Berhasil
Diagnosa awal	Sistem memberikan kemungkinan penyebab kerusakan	Diagnosa awal berhasil diberikan	Berhasil
Eskalasi ke montir	Sistem melakukan eskalasi untuk kasus kompleks	Eskalasi berhasil dilakukan	Berhasil
Bahasa lokal	Sistem memahami istilah Jawa/Semarang	Respons sesuai konteks	Berhasil
Pertanyaan di luar domain	Sistem mengarahkan percakapan kembali ke topik bengkel	Respons sesuai	Berhasil

Berdasarkan Tabel 4, seluruh skenario pengujian berhasil menghasilkan keluaran sesuai dengan yang diharapkan. Pengujian mencakup kemampuan sistem dalam memberikan informasi layanan berdasarkan *knowledge base*, melakukan reservasi servis, menampilkan status kendaraan, memberikan diagnosa awal, melakukan eskalasi kepada mekanik, memahami istilah bahasa lokal, serta menangani pertanyaan di luar domain bengkel. Seluruh skenario memperoleh status berhasil, yang menunjukkan bahwa fungsi-fungsi utama sistem telah berjalan sesuai kebutuhan pengguna.

Mekanisme keamanan turut diuji dan berjalan sesuai rancangan. Input validation menolak parameter yang tidak sesuai. Rate limiting, validasi sesi lewat HMAC Token, dan perlindungan API Key tetap menjaga keamanan komunikasi antara aplikasi dan OpenRouter API sepanjang pengujian berlangsung. Hasil pengujian ini memperlihatkan bahwa RAG, tool calling, basis data SQLite, dan OpenRouter API bisa bekerja terpadu untuk mendukung layanan informasi dan operasional Bengkel C Maestro Semarang. Chatbot menjawab pertanyaan dari *knowledge base* sekaligus menjalankan fungsi bisnis sesuai permintaan pengguna, semua lewat satu antarmuka percakapan, sejalan dengan kebutuhan fungsional yang ditetapkan pada tahap analisis kebutuhan.

Temuan penelitian ini sejalan dengan (Qin et al., 2023), yang menunjukkan bahwa Large Language Model dapat diperluas kemampuannya melalui pemanfaatan *tool calling* untuk berinteraksi dengan berbagai layanan dan API eksternal. Pada penelitian ini, integrasi *Retrieval-Augmented Generation* (RAG) dengan *tool calling* memungkinkan

TIN: Terapan Informatika Nusantara
Vol 7, No 2, July 2026, page 1134-1144
ISSN 2722-7987 (Media Online)
Website https://ejurnal.seminar-id.com/index.php/tin
DOI 10.47065/tin.v7i2.10640

chatbot tidak hanya memberikan informasi berdasarkan *knowledge base*, tetapi juga menjalankan proses operasional seperti reservasi servis, pengecekan status kendaraan, dan eskalasi kepada mekanik. Integrasi tersebut menjadikan sistem yang dikembangkan tidak hanya berfungsi sebagai penyedia informasi, tetapi juga sebagai asisten virtual yang mampu mendukung pelaksanaan layanan bengkel secara langsung (Gao et al., 2024; Qin et al., 2023).digunakan pelanggan. Hasil tersebut menunjukkan bahwa implementasi RAG berbasis context injection dan mekanisme tool calling telah memenuhi tujuan penelitian dalam mendukung layanan pelanggan Bengkel C Maestro Semarang.

3.2 Pembahasan

Hasil penelitian ini juga sejalan dengan penelitian (Elysia & Herianto, 2024; Saman et al., 2025) yang menunjukkan bahwa penerapan *Retrieval-Augmented Generation* (RAG) mampu meningkatkan kualitas layanan informasi melalui pemanfaatan *knowledge base* sebagai sumber konteks dalam menghasilkan respons yang lebih relevan. Temuan serupa juga ditunjukkan oleh (Albert & Voutama, 2025) yang mengembangkan chatbot berbasis dokumen PDF menggunakan pendekatan RAG untuk meningkatkan akurasi jawaban terhadap pertanyaan pengguna. Kedua penelitian tersebut membuktikan bahwa integrasi sumber pengetahuan eksternal dapat meningkatkan kemampuan chatbot dalam memberikan informasi yang sesuai dengan konteks. Namun, penelitian ini memberikan pengembangan lebih lanjut dengan mengintegrasikan mekanisme *tool calling* ke dalam arsitektur RAG. Integrasi tersebut memungkinkan chatbot tidak hanya memberikan informasi berdasarkan *knowledge base*, tetapi juga menjalankan fungsi operasional, seperti melakukan reservasi servis, mengecek status kendaraan, memperoleh daftar layanan, serta mengeskalasi pelanggan kepada mekanik melalui satu antarmuka percakapan. Dengan demikian, kontribusi utama penelitian ini tidak hanya terletak pada peningkatan kualitas respons melalui RAG, tetapi juga pada perluasan kemampuan asisten virtual untuk mendukung proses bisnis secara langsung sehingga lebih sesuai diterapkan pada layanan operasional bengkel skala UMKM.

4. KESIMPULAN

Penelitian ini berhasil merancang dan membangun asisten virtual layanan Bengkel C Maestro Semarang dengan mengintegrasikan Retrieval-Augmented Generation (RAG) dan *tool calling* dalam satu sistem berbasis *Large Language Model* (LLM). Integrasi tersebut membuat sistem dapat memberikan informasi dari *knowledge base* sekaligus menjalankan fungsi operasional, seperti reservasi servis, pengecekan status kendaraan, dan eskalasi kepada mekanik, melalui satu antarmuka percakapan. Hasil pengujian *Black Box Testing* menunjukkan bahwa seluruh fungsi utama sistem telah berjalan sesuai kebutuhan fungsional, meliputi antarmuka pengguna, mekanisme RAG, *tool calling*, basis data SQLite, serta integrasi dengan OpenRouter API. Penerapan *context injection* sebagai mekanisme RAG juga terbukti mampu menghasilkan respons yang relevan pada domain dengan *knowledge base* yang relatif kecil dan jarang mengalami perubahan, tanpa memerlukan *vector database* maupun *embedding-based retrieval*. Dengan demikian, pendekatan yang diusulkan efektif diterapkan pada layanan bengkel skala UMKM karena mampu menggabungkan penyampaian informasi dan pelaksanaan proses bisnis dalam satu sistem percakapan. Penelitian ini memberikan kontribusi melalui pengembangan arsitektur asisten virtual yang mengintegrasikan RAG berbasis *context injection*, mekanisme *tool calling*, dan OpenRouter API sehingga mampu mendukung layanan informasi sekaligus proses operasional bengkel dalam satu sistem. Kontribusi tersebut menunjukkan bahwa pendekatan yang diusulkan dapat menjadi alternatif implementasi asisten virtual yang praktis dan sesuai untuk mendukung digitalisasi layanan pada bengkel otomotif skala UMKM**.** Meskipun demikian, penelitian ini masih memiliki keterbatasan, yaitu penurunan efisiensi ketika ukuran *knowledge base* bertambah besar serta potensi kesalahan pemilihan fungsi pada mekanisme *tool calling* apabila maksud pengguna ambigu. Oleh karena itu, penelitian selanjutnya dapat mengembangkan sistem dengan memanfaatkan *vector database*, *embedding-based retrieval*, maupun kemampuan multimodal untuk meningkatkan akurasi respons, skalabilitas sistem, dan cakupan layanan asisten virtual.

REFERENCES

Albert, G. D., & Voutama, A. (2025). Pengembangan Chatbot Berbasis Pdf Menggunakan Local Retrieval-Augmented Generation (Rag) Dan Ollama. *Jurnal Informatika Dan Teknik Elektro Terapan*, *13*(2). https://doi.org/10.23960/jitet.v13i2.6361

Dharma, R., Raya, A., Lia, F., Cahyanti, D., & Mandiri, N. (2025). Perancangan Sistem Informasi Chatbot Retrieval Augmented Generation Berbasis Website Pada PT. Revolusi Cita Edukasi. *Journal Computer Science*, *4*(1), 15–21. https://doi.org/10.31294/m75d4782

Elysia, S., & Herianto. (2024). Chatbot Berbasis Retrieval Augmented Generation (RAG) untuk Peningkatan Layanan Informasi Sekolah. *Journal TIFDA (Technology Information and Data Analytic)*, *1*(2), 52–58. https://doi.org/10.70491/tifda.v1i2.52

Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Dai, Y., Sun, J., Wang, M., & Wang, H. (2024). *Retrieval-Augmented Generation for Large Language Models: A Survey*. http://arxiv.org/abs/2312.10997

Hidayah Nova, S., Puji Widodo, A., & Warsito, B. (2022). Analisis Metode Agile pada Pengembangan Sistem Informasi Berbasis Website: Systematic Literature Review Analysis of Agile Method on Website-Based


TIN: Terapan Informatika Nusantara
Vol 7, No 2, July 2026, page 1134-1144
ISSN 2722-7987 (Media Online)
Website https://ejurnal.seminar-id.com/index.php/tin
DOI 10.47065/tin.v7i2.10640

Information System Development: Systematic Literature Review. *Techno.COM*, *21*(1), 139–148. https://doi.org/10.33633/tc.v21i1.5659

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W., Rocktäschel, T., Riedel, S., & Kiela, D. (2020). *Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks*. http://arxiv.org/abs/2005.11401

Mozilla Developer Network. (2024). *Using Server-Sent Events*. Mozilla Developer Network. https://developer.mozilla.org/en-US/docs/Web/API/Server-sent_events

OpenAI. (2024). *OpenAI API Documentation*. OpenAI. https://platform.openai.com/docs

OpenJS Foundation. (2024). *Node.js Documentation*. https://nodejs.org/docs/latest/api/

Patil, S. G., Zhang, T., Wang, X., & Gonzalez, J. E. (2023). *Gorilla: Large Language Model Connected with Massive APIs*. http://arxiv.org/abs/2305.15334

Pressman, R. S., & Maxim, B. R. (2020). *Software Engineering: A Practitioner's Approach* (9th ed.). McGraw-Hill Education.

Qin, Y., Liang, S., Ye, Y., Zhu, K., Yan, L., Lu, Y., Lin, Y., Cong, X., Tang, X., Qian, B., Zhao, S., Hong, L., Tian, R., Xie, R., Zhou, J., Gerstein, M., Li, D., Liu, Z., & Sun, M. (2023). *ToolLLM: Facilitating Large Language Models to Master 16000+ Real-world APIs*. http://arxiv.org/abs/2307.16789

Ram, O., Levine, Y., Dalmedigos, I., Muhlgay, D., Shashua, A., Leyton-Brown, K., & Shoham, Y. (2023). *In-Context Retrieval-Augmented Language Models*. http://arxiv.org/abs/2302.00083

Russell, S. J. ., Norvig, Peter., Chang, M.-Wei., Devlin, Jacob., Dragan, Anca., Forsyth, David., Goodfellow, Ian., Malik, Jitendra., Mansinghka, Vikash., Pearl, Judea., & Wooldridge, M. J. . (2021). *Artificial intelligence : a modern approach*. Pearson.

Saidi Rahman, M., Ekawati, F., & Indra Wijaya, Y. (2025). Chatbot Ai Sebagai Media Pencarian Informasi Dengan Menggunakan Metode Large Language Models (Llm) Ai Chatbot As A Media For Searching Information With Natural Language Processing Methods. In *Jurnal Sains Komputer dan Teknologi Informasi e-issn* (Vol. 7, Number 2). https://doi.org/DOI:10.33084/jsakti.v7i2.9952

Saman, M., Ahmad, G., Alfarisy, F., Amelia, R., & Fadhliana, N. R. (2025). A RAG-Based Academic Information Chatbot Using Lightweight LLaMA and Indo-Sencence-BERT. *Indonesian Journal of Artificial Intelligence and Data Mining (IJAIDM)*, *8*(3), 687–696. https://doi.org/10.24014/ijaidm.v8i3.38150

Schick, T., Dwivedi-Yu, J., Dessì, R., Raileanu, R., Lomeli, M., Hambro, E., Zettlemoyer, L., Cancedda, N., & Scialom, T. (2023). *Toolformer: Language Models Can Teach Themselves to Use Tools*. https://doi.org/10.48550/arXiv.2302.04761

Sugiyono. (2022). *Metode Penelitian Kuantitatif, Kualitatif, dan R&D* (2nd ed.). Alfabeta.

Wang, L., Ma, C., Feng, X., Zhang, Z., Yang, H., Zhang, J., Chen, Z., Tang, J., Chen, X., Lin, Y., Zhao, W. X., Wei, Z., & Wen, J. (2024). A survey on large language model based autonomous agents. In *Frontiers of Computer Science* (Vol. 18, Number 6). Higher Education Press Limited Company. https://doi.org/10.1007/s11704-024-40231-1

Zhao, W. X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., Min, Y., Zhang, B., Zhang, J., Dong, Z., Du, Y., Yang, C., Chen, Y., Chen, Z., Jiang, J., Ren, R., Li, Y., Tang, X., Liu, Z., … Wen, J.-R. (2023). *A Survey of Large Language Models*. http://arxiv.org/abs/2303.18223