Indonesian Journal of Electrical Engineering and Computer Science
Vol. 42, No. 1, April 2026, pp. 205∼214
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v42.i1.pp205-214

Complexity aware cascade architecture for improving user satisfaction in conversational AI

Constantinus Satrio[1], Devi Fitrianah[2]
[1]Computer Science Department, BINUS Graduate Program – Master of Computer Science,
Bina Nusantara University, Jakarta, Indonesia
[2]Computer Science Department, Bina Nusantara University, Jakarta, Indonesia

Article Info

Article history:

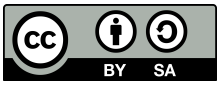
Received Aug 21, 2025
Revised Jan 4, 2026
Accepted Mar 4, 2026

Keywords:

Chatbot architecture
Complexity aware cascade
Conversational AI
Retrieval-augmented generation
Service quality
Task completion rate
User satisfaction

ABSTRACT

Conventional task-oriented chatbots frequently suffer from task incompletions and low user satisfaction when handling complex queries. This research introduces the complexity aware cascade, an adaptive architecture that improves user service quality by dynamically matching query complexity with the appropriate computational response. The system uses confidence and relevance scores to intelligently route requests through a sequence of a natural language understanding (NLU) model, a retrieval-augmented generation (RAG) pipeline, or a large language model (LLM). The tiered architecture was evaluated via a randomized controlled trial (RCT) with 150 participants, measuring task success and user satisfaction. The full cascade achieved a 90% journey completion rate, representing a 92.3% improvement over baseline system and substantial gains in SERVQUAL-based service quality scores. The experiment was conducted in a domain-specific knowledge base (essential oils) with a convenience sample that does not represent the global population, and no real-time deployment or long-term cost analysis was performed. Accordingly, the findings should be interpreted as evidence of effectiveness in a limited setting rather than as directly scalable to all domains. Even with these limitations, this study provides arigorously tested blueprint for developing more robust and user-centric conversational AI systems.



Corresponding Author:

Constantinus Satrio
Computer Science Department, BINUS Graduate Program – Master of Computer Science
Bina Nusantara University
Jakarta, Indonesia
Email: constantinus.satrio@binus.ac.id

1. INTRODUCTION

Conversational AI, particularly chatbots, has become a critical component of modern customer service systems [1]-[3]. While effective at handling routine queries, conventional chatbots exhibit a significant weakness: their inability to gracefully manage conversations outside their pre-defined training scope [4], [5]. This brittleness leads to high rates of conversational failure, user frustration, and ultimately, a negative impact on customer satisfaction [6]. Existing literature of multi-stage systems, such as the static, keyword-based clarification mechanism from Lautraite *et al.* [7], often lack dynamic, automated strategies. Frameworks like PromptChainer [8], for instance, show the potential of chaining LLMs but require manual programming. This reveals a research gap for an intelligent architecture that automatically manages conversational complexity to improve user outcomes [9], [10].

Prior studies on customer-service chatbots have focused primarily on how interaction quality and interface design driver user satisfaction, often treating the underlying system architectures as a ”black box”. Research by Adam *et al.* [1] and Hsu and Lin [11] demonstrates how perceived service quality, social presence, and conversational quality influence user compliance and loyalty, while other work examines the roles of anthropomorphism and personality in user management [3], [5], [6], [10], [12]-[15]. However, these studies typically implement conversational engines as single stage systems, leaving architectural choices such as multi-stage routing, retrieval augmentation, and LLM integration largely unevaluated as primary determinants of user experience. Systematic reviews further highlight a concentration on technical performance or heterogeneous satisfaction scales, with limited attention to how specific architectural configurations affect perceived service quality. This highlights a lack of controlled experiments that jointly analyze task success, service quality perceptions, and system latency across alternative system designs [2], [4], [9], [16].

Architecturally, the proposed complexity aware cascade (CAC) builds on prior multi-stage conversational systems and retrieval-augmented language models. While existing frameworks demonstrate that decomposing complex queries into stages improves answer quality and tools like PromptChainer enable complex applications through prompt-chaining [7], [8], these approaches often rely on fixed routing logic or manual configuration. Similarly, models such as RAG and REALM show how significant performance gains in knowledge-intensive tasks, yet they are typically evaluated using technical benchmarks rather than human-centered service quality [17], [18]. This study addresses these gaps by operationalizing a CAC that integrates RASA, RAG, and LLM into a dynamic pipeline, linking objective journey completion rates (JCR) and response times to SERVQUAL-based user satisfaction. Our results indicate that routing medium and high complexity queries through deeper architectural stages improves both JCR and perceived service quality compared to single-stage baselines. The LLM stage primarily enhances user experience rather than providing statistically significant gains in task completion.

In this context, user experience extends beyond task completion, as prior research demonstrates that perceived service quality encompassing reliability, responsiveness, assurance, and empathy is a vital predictor of satisfaction, trust, and continued use. These findings align with the broader service-quality literature where SERVQUAL-inspired scales are used to quantify technology-mediated services and explain outcomes such as loyalty and compliance [6]. Motivated by this body of work, the present study adopts a SERVQUAL-based instrument to evaluate the chatbot as a service interface, measuring success by how well it manages expectations, uncertainty, and trust alongside task success. This paper contributes by proposing the Complexity Aware Cascade architecture for adaptive query routing based on complexity and safety constraints, implementing an end-to-end prototype using RASA, a grounded RAG pipeline, and a safety-conscious LLM as a cost-aware ”gatekeeper”, and conducting a randomized controlled trial with 150 participants to compare three architectures (NLU-only, NLU+RAG, and CAC with LLM). Finally, we analyze the trade-offs between task completion, user satisfaction, safety, and computational cost, demonstrating that while the LLM stage primarily enhances user experience rather than raw task success, it provides a validated blueprint for building more effective, trust-based conversational AI [12]-[19].

2. PROPOSED COMPLEXITY AWARE CASCADE ARCHITECTURE

The CAC architecture is designed to manage conversational workflows by matching query complexity with the appropriate level of computational power [15]. The CAC consists of three sequential stages, orchestrated by a dynamic routing logic.

2.1. Architecture stages

Stage 1: Specialized NLU (RASA). The first stage acts as a fast, computationally inexpensive gatekeeper. It utilizes the RASA framework to handle most in-scope, task-oriented user intents and is optimized for speed and efficiency on known conversational paths [20].

Stage 2: Retrieval-augmented generation (RAG). If the RASA model's NLU confidence is below a threshold (e.g., < 0.75), the system triggers a RAG pipeline [17], [18]. The RAG process involves several key steps, starting with an initial semantic retrieval from a vector database, followed by TF-IDF relevance filtering to re-rank candidates based on keyword relevance and ensure context precision. A cross-encoder then performs a joint, deep analysis of query-document pairs to provide a deterministic relevance score that is combined with the initial NLU confidence to determine escalation to the LLM or rejection of out-of-scope queries for safety. To maintain operational efficiency, principal component analysis (PCA) is utilized for context compression to

prevent window overflow in the final generation step, and 4-bit quantization is employed to significantly reduce memory footprint and accelerate inference without sacrificing performance.

Stage 3: LLM-based response generation. The curated context from the RAG pipeline is passed via the LangChain framework to a large-scale generative model (e.g., ChatGPT) [21], [22]. This final stage synthesizes the information to generate a coherent, contextually grounded response. This tiered management ensures that the most computationally expensive resource—the generative LLM—is used only when necessary and is always supplied with a high-quality, optimized context [23].

2.2. Implementation details

The system, built with Python [24] and Flask [25], uses a blocked randomization scheme. Participants were randomly assigned to one of the three conditions RASA_ONLY, RASA_RAG, RASA_RAG_LLM. This ensure equal group sizes (n=50 per group). Stage switching is triggered when RASA's confidence is below 0.75 or on a fallback. The RAG pipeline uses nomic-ai/nomic-embed-text-v1.5 [26] for embeddings, cross-encoder/ms-marco-MiniLM-L-6-v2 [27] as cross encoder, and a quantized unsloth/Phi-3-mini-4k-instruct-bnb-4bit for generation [28]. The final LLM stage uses OpenAI's gpt-4o-mini, incorporating chat history for context. All interactions are logged in a PostgreSQL database with the pgvector extension for analysis. The system escalates from RASA to RAG if confidence is below 0.75 or on a fallback. The final LLM prompt combines retrieved context with chat history for a more coherent response.

Handovers are governed by real-time quantitative metrics [29]. Escalation from Stage 1 to 2 is triggered by a low NLU confidence score. Decision to escalate from Stage 2 to 3 is determined by cross-encoder relevance score, which provides deterministic judgment for the retrieved documents' relevance to the user's query [30]. This second metric allows the system to intelligently decide if the retrieved information is sufficient or if the full power of the generative model is required. The system architecture (Figure 1) first routes all queries to Stage 1 for RASA NLU assessment. A confidence score acts as the initial gate: high-confidence queries are answered directly, while low-confidence ones escalate to the Stage 2 RAG pipeline for context retrieval and re-ranking. A second gate then uses the cross-encoder's relevance score to judge the retrieved context. If the context is highly relevant, the local quantized LLM responds. Only when both NLU confidence and context relevance are low does the query escalate to Stage 3, where the gpt-4o-mini model handles the most complex queries using the curated context and conversation history.

3. METHOD

To validate the proposed architecture, we employed a randomized controlled trial (RCT) with a focus on user-centric outcomes [31].

3.1. Experimental design

To systematically evaluate each component's contribution, the experiment was designed as an ablation study. Participants were randomly assigned to one of three system configurations: (1) Condition A (single-stage): A weak baseline chatbot using only the RASA framework; (2) Condition B (two-stage): An intermediate system where RASA falls back to the RAG pipeline for unresolved queries, then presenting retrieved context directly to the user; (3) Condition C (multi-stage): The complete three stage architecture, including the final generative LLM. This design allows for a direct comparison of the performance gains from the RAG retrieval stage (A vs. B) and the subsequent generative synthesis stage (B vs. C) [32].

3.2. Performance metrics

We evaluated the system's impact on user experience with two primary metrics: (1) Journey Completion Rate (JCR): A measure of task success, where categorical ratings were converted to numerical scores "Ya, semua tugas selesai dengan baik." = 100 (complete success), "Hanya beberapa tugas yang selesai" = 40 (partially successful), Tidak, saya mengalami banyak masalah = 20 (mostly unsuccessful) [16]; (2) User Satisfaction (SERVQUAL): Measured using the SERVQUAL framework. The survey comprised 16 items rated on a 5-point Likert scale (1 = Strongly Disagree to 5 = Strongly Agree), grouped into five dimensions: Tangibles (3 items), Reliability (4 items), Responsiveness (3 items), Assurance (3 items), and Empathy (3 items). A composite score for each dimension was calculated for each participant by averaging the scores of its corresponding items [11].

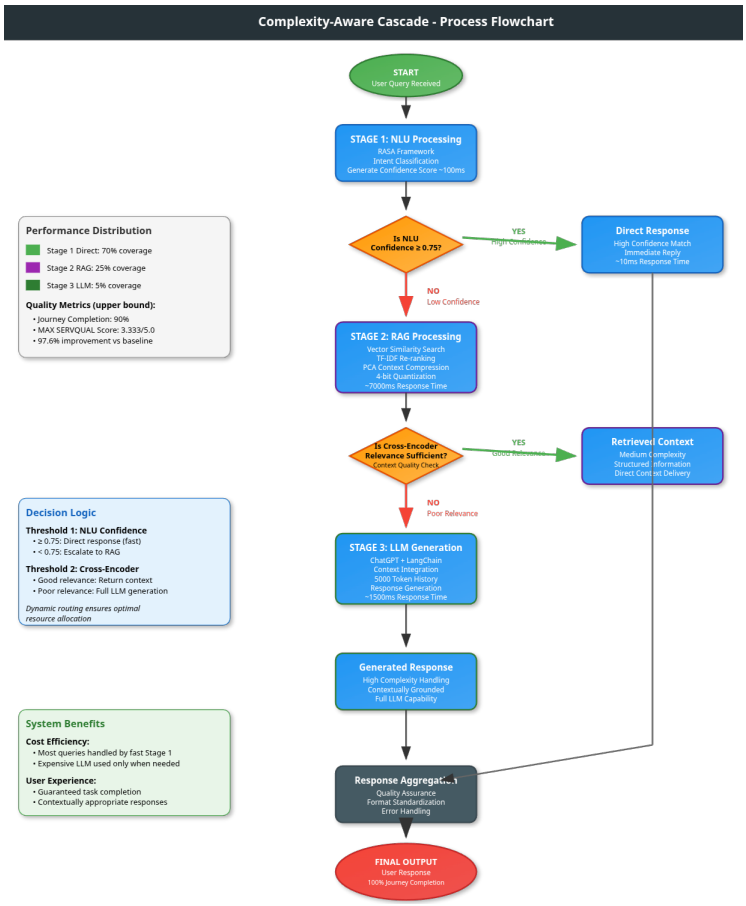


Figure 1. System flowchart of complexity aware cascade architecture

3.3. Participant recruitment

Participants (N=150) were recruited via advertisements posted on a variety of online platforms, including university student forums, general social media groups (e.g., Facebook, Instagram), and community messaging boards (e.g., WhatsApp) to gather a diverse sample, encompassing a wide spectrum of professional backgrounds and levels of technical familiarity. Gender distribution is 60.97% male and 39.3% female. Participants were randomly assigned to one of the three conditions (n=50 per group). All chat logs and survey responses were systematically recorded.

3.4. Participant procedure and scenarios

A standardized procedure was implemented through a rdedicated web interface. Participants were first briefed and then guided through five distinct task scenarios of varying complexity: (1) Simple query: A straightforward fact-based question; (2) Medium complexity Requiring multi-step reasoning within a known domain; (3) Complex query: A question requiring synthesis of information from multiple topics; (4) Out-of-context query: A question designed to be beyond the base chatbot's training data; (5) Creative query: A task requiring the generation of original text or ideas.

For each scenario, participants were provided with example prompts and a "copy to clipboard" function to standardize input phrasing across conditions. After interacting with the chatbot for all scenarios, participants completed the JCR and SERVQUAL questionnaire.

3.5. Data analysis

Due to the ordinal nature of Likert scale data and the significant negative skew and non-normal distribution identified in the SERVQUAL responses (particularly in Condition A), non-parametric statistical tests were employed. Group comparisons: The Kruskal-Wallis H test, a non-parametric alternative to one-way ANOVA, was used to determine if there were statistically significant differences in the median JCR and

SERVQUAL dimension scores across the three conditions. If a significant difference was found, Dunn's post-hoc test with Bonferroni correction was applied for pairwise comparisons. Correlation analysis: The Spearman's rank-order correlation coefficient (ρ) was used to assess the monotonic relationship between the ordinal condition variable (A=1, B=2, C=3) and the user outcome metrics (JCR, SERVQUAL scores). All analyses were conducted using Python (v3.12.3) with appropriate libraries (details can be seen on GitHub), and a p-value of < 0.05 was considered statistically significant.

4. RESULTS AND DISCUSSION

Analysis of the experimental data indicates a statistically significant, positive correlation between the implementation of the complexity aware cascade architecture and the measured outcomes of task success and user satisfaction.

4.1. Task success (JCR)

The distribution of JCR scores across the three architectures is summarized in Table 1. The table reports mean JCR, as well as the proportion of participants who experienced complete versus failed journeys under each configuration, allowing a direct comparison of how architectural complexity impacts task success. The analysis of mean JCR revealed a significant improvement in task success as architectural complexity increased. The mean JCR score increased from 46.8% for the Single-Stage system to 77.2% for the two-stage system, and 90% mean JCR for the multi-stage architecture.

Table 1. Mean journey completion rate by chatbot architecture

Architecture	Mean JCR (%)	Success rate (100%)	Failure rate (<50%)
Single-stage	46.8	24.0	76.0
Two-stage	77.2	68.0	32.0
Multi-stage	90	86.0	14.0

4.2. User satisfaction (SERVQUAL analysis)

The improvements in task success translated directly to a higher perception of service quality as shown in Table 2 and Figure 2. The median satisfaction scores improved progressively from a baseline of 1.133 for the single-stage system to 2.417 for the multi-stage architecture. This substantial uplift demonstrates that system-level choices serve as a direct lever for enhancing user trust and perceptions of reliability. Furthermore, the multi-stage configuration achieved a more consistent user experience, demonstrated by a tighter interquartile range (IQR = 0.616) compared to the two-stage system (IQR = 0.842), suggesting that the final generative stage provides a more stable and predictable service interface.

Table 2. SERVQUAL scores by chatbot architecture

Architecture	N	Median	IQR
Single-stage	50	1.133	0.250
Two-stage	50	1.842	0.842
Multi-stage	50	2.417	0.616

4.3. Statistical findings

To formally test whether these observed differences were statistically significant, we applied non-parametric analyses suited to the ordinal and skewed nature of the data. Table 3 summarizes the main statistical findings, including the Kruskal–Wallis tests for overall architectural effects, the Spearman rank correlations between architectural complexity and the outcome metrics, and the post-hoc comparison between the RAG-only and RAG+LLM conditions.

Kruskal–Wallis tests confirmed that architectural complexity had a significant effect on both user satisfaction (SERVQUAL, H=85.39, p<0.001) and task completion (JCR, H=37.70, p<0.001). Spearman's ρ further showed strong correlation between complexity and satisfaction (ρ=0.748) and a moderate correlation with task success (ρ=0.490). Post-hoc analysis indicated that adding the LLM stage did not significantly improve JCR over the RAG system alone (p=0.107), suggesting that its main contribution lies in enhancing the user experience rather than task success. Finally, the weak correlation between SERVQUAL and JCR (ρ=0.297) underscores that satisfaction and success, while related, are distinct outcomes.

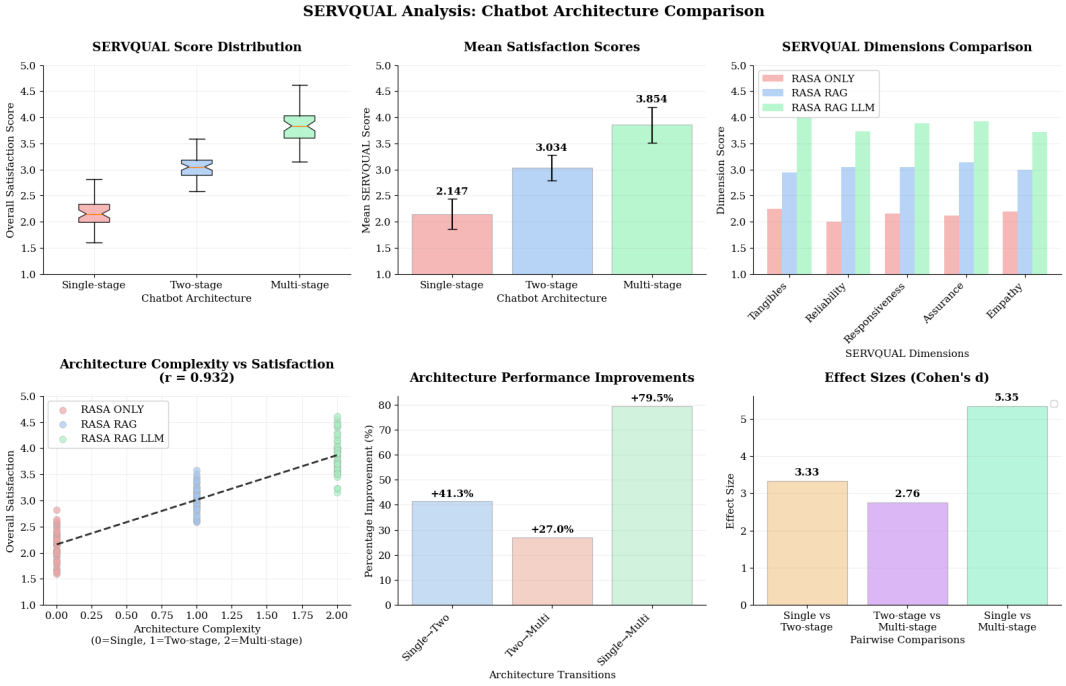


Figure 2. Box plots showing the distribution of overall SERVQUAL scores for the single-stage, two-stage, and multi-stage architectures

Table 3. Key statistical findings along with statistical test type

Key Finding	Statistical Test	Result	p-value
Overall architectural impact	Kruskal-Wallis H	SERVQUAL:H=85.39, JCR:H=37.70	< 0.001
Complexity drives satisfaction	Spearman's ρ	$\rho = 0.748$ (Very Strong)	< 0.001
Complexity moderately improves task success	Spearman's ρ	$\rho = 0.490$ (Moderate)	< 0.001
LLM enhances experience, not just success	Post-hoc Dunn's test (RAG vs RAG+LLM)	JCR Comparison: p = 0.107	0.1070
Task Success $\neq$ User Satisfaction	Spearman's ρ (SERVQUAL$\leftrightarrow$JCR)	$\rho = 0.297$ (Weak)	< 0.001

4.4. Discussion

The study's findings reveal a substantial improvement in task success correlated with architectural complexity. From a quantitative perspective, the gains achieved by the CAC are large relative to those reported in prior chatbot research. Adam *et al.* [1] experimentally examined how interaction-design manipulations (foot-in-the-door requests and anthropomorphic design cues) affect user compliance with a customer-service chatbot. In their study, compliance rose from 63% in the control condition to 77% with only foot-in-the-door cues, 84% with only anthropomorphic cues, and 95% when both were combined, an absolute gain of 32 percentage points and a relative improvement of around 51% over the baseline [1]. In our randomized controlled trial, JCR increased from 46.8% in the single-stage baseline to 77.2% in the two-stage architecture and 90.0% in the full CAC. This corresponds to an absolute gain of 43.2 percentage points and a relative improvement of 92.3% in task completion. Whereas Adam *et al.* [1] attribute their effects to surface-level interaction design for a single request, our results show that system-level architectural changes can yield improvements of comparable or greater magnitude in end-to-end journey completion across a multi-turn, information-seeking domain.

A similar pattern emerges when comparing our findings with studies that focus on customer satisfaction and loyalty rather than task completion. Hsu and Lin [11] modelled satisfaction and loyalty toward customer-service chatbots using an extended e-service quality framework, showing that AI chatbot service recovery quality and conversational quality significantly predict satisfaction, and that core service quality and satisfaction predict loyalty. Their structural model reports strong paths from chatbot quality to satisfaction and loyalty (standardized coefficients exceeding 0.5) and explains a large proportion of variance in loyalty (R^2 above

0.60), but it does not include an explicit task-completion or journey-completion metric. In contrast, our experiment links architectural configuration, objective JCR, and SERVQUAL scores within a single RCT: the full CAC configuration simultaneously improves task completion by 92.3% relative to the baseline and produces a substantial uplift in SERVQUAL-based perceived service quality. In other words, our results complement Hsu and Lin's construct-level findings by showing that a concrete architectural design choice—complexity-aware cascading—can be treated as a lever for jointly improving service quality and completion outcomes, not only for shifting latent satisfaction and loyalty scores.

Prior experimental work on chatbot user experience also tends to report effects primarily at the perceptual level. Haugeland *et al.* [3], for example, conducted a randomized experiment (n = 35) comparing topic-led versus task-led conversations and button-based versus free-text interaction in a customer-service chatbot. They found that topic-led conversations significantly increased perceived anthropomorphism and hedonic quality, while button-based interaction improved pragmatic and hedonic quality, all measured on seven-point UX scales. However, their study does not report changes in objective task success or completion rates, and the sample size is relatively small compared to our RCT with 150 participants. Our findings extend this line of work by showing that architectural complexity—operationalized as a dual-threshold cascade over RASA, RAG, and LLM stages—not only affects user perceptions (SERVQUAL) but also leads to large, quantifiable gains in end-to-end journey completion.

Systematic reviews of user-experience assessment with conversational agents remain relatively small in scope and highlight a fragmented evaluation landscape. For example, Tubin *et al.* [16] systematically searched four major HCI-related databases (ACM, IEEE, Springer, and Scopus) and initially identified 482 papers, but only 27 studies met their inclusion criteria for reporting how user experience with conversational agents was assessed. Their analysis shows a wide dispersion of methods, heavy reliance on self-created post-test questionnaires, and very limited use of validated UX instruments or combined pre-, during-, and post-use assessments [16]. In parallel, Gamboa-Cruzado *et al.* [2] review customer-service chatbots and document a fragmented landscape of application domains, technologies, and evaluation practices, while more recent reviews of customer-service chatbot experience similarly synthesize dozens of empirical studies without identifying many that jointly analyze service-quality perceptions and hard performance metrics such as completion or resolution rates [9], [16]. Against this backdrop, the present study adds a relatively rare data point: a controlled, four-arm RCT that explicitly varies the architecture (single stage vs. naïve stack vs. three-stage CAC vs. four-stage CAC with LLM), uses a structured SERVQUAL-based instrument, and reports both JCR and SERVQUAL outcomes. The size of the observed differences—over 40 percentage points in JCR between baseline and full CAC—indicates that architectural choices can have an impact on user outcomes comparable to or larger than those reported for interaction-design manipulations and service-quality perceptions alone.

During the experiment, users occasionally entered off-domain queries to the chatbot's designed knowledge base of essential oils (e.g., geopolitical topics unrelated to essential oils). In these instances, the cross-encoder's relevance scoring acted as a safety system. By returning a near-zero confidence score (e.g., 0.000047), it correctly identified the query as out-of-scope. Consequently, the chatbot refused to generate a speculative or potentially incorrect answer, instead informing the user of its limitations. While this is the desired behaviour for a trustworthy AI, each refusal was logged as an incomplete journey, thereby preventing the JCR from reaching a perfect score.

This intentional refusal to answer irrelevant queries explains the ceiling on the JCR and the SERVQUAL scores. It represents a deliberate trade-off: sacrificing a perfect task completion metric to ensure high Reliability and Assurance. Users perceive the system as more trustworthy not only because it provides correct answers but also because it is honest about what it does not know. This confirms that task success and user satisfaction are not strictly correlated, underscoring the importance of reliability and transparency as independent drivers of user experience. This underscores a key design principle: optimizing for user-facing outcomes of safety and reliability is more valuable than striving for a flawless but potentially misleading completion rate.

4.5. Limitations

This study has several important limitations that must be considered when interpreting the results. First, the experiment was conducted in a single, domain-specific knowledge base (essential oils) using scripted scenarios. As a result, the findings may not be generalize to high-stakes domains like finance or healthcare. Second, participants recruitment via university forums and social media resulted in a convenience sample that is relatively homogeneous and does not represent the global population. Most critically, the research lacked long-

term exploration, as no real-time production deployment trials or large-scale computational cost studies were performed to measure long-term user retention or infrastructure costs. While the methodological contribution lies in the architectural design rather than a new model class, the findings provide a validated blueprint for practical application. Future work should evaluate the architecture across multiple domains with more diverse and representative samples under field conditions that capture long-term behavior and operational constraints. Furthermore, it is recommended that practitioners adopt the CAC as an incremental layer, utilizing the dual-threshold routing to reserve expensive LLM calls for complex cases while letting the NLU or RAG tiers handle routine queries to balance satisfaction and cost in large-scale customer service environments.

5. CONCLUSION

This research introduced and empirically validated the CAC, a system architecture for managing conversational AI workflows. Through a RCT with 150 participants, we demonstrated that this multi-stage fallback system significantly outperforms simpler architectures in its ability to complete user tasks and improve user-perceived service quality. Ultimately, this research provides a validated blueprint for building more effective conversational AI, demonstrating that true user-centric design involves a careful balance between task completion and the system's ability to earn user trust by safely managing its own limitations. The principles demonstrated here can inform future research into developing safety-grounded, complexity-aware conversational AI for practical deployment. For practitioners, these findings suggest that a complexity-aware cascade can be adopted as an incremental layer on top of existing task-oriented chatbots. In large-scale customer-service environments, the dual-threshold routing logic can be used to reserve LLM calls for genuinely complex or ambiguous cases, while letting a fast NLU or RAG tier handle routine queries, thereby balancing user satisfaction and infrastructure cost. The architecture can also be integrated with existing ticketing systems and knowledge bases by treating each stage as a modular service that exposes standardized interfaces for logging, escalation, and monitoring. Future deployments should instrument both task-level metrics (e.g., resolution rates, hand-off to human agents) and service-quality indicators (e.g., SERVQUAL dimensions) to continuously tune thresholds and escalation policies. Extending this evaluation to multiple domains and long-running production systems is a natural next step to validate how well the CAC scales beyond the controlled experimental setting reported here.

ACKNOWLEDGMENTS

The authors acknowledge the use of AI tools to assist in the preparation of this manuscript. Assistance was provided for code generation and debugging (Anthropic's Claude) and for editorial support, including language polishing and ensuring conciseness (Google's Gemini). The authors retain full responsibility for the final content, all interpretations, and the integrity of the work.

FUNDING INFORMATION

This work was supported and funded by the Binus Graduate Program, Bina Nusantara University.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Constantinus Satrio	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	
Devi Fitrianah		✓			✓	✓	✓			✓	✓	✓		

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal Analysis
I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting
Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject Administration
Fu : **Fu**nding Acquisition

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

DATA AVAILABILITY

The anonymized summary dataset and the computational notebook that support the findings of this study are openly available in https://doi.org/10.5281/zenodo.18839338. Additional data, including the detailed anonymized chat logs and participant-level assignments, are not publicly available to protect participant privacy but can be obtained from the corresponding author upon reasonable request.

REFERENCES

[1] M. Adam, M. Wessel, and A. Benlian, "Ai-based chatbots in customer service and their effects on user compliance," *Electronic Markets*, vol. 31, no. 2, pp. 427–445, Jun. 2021.

[2] J. Gamboa-Cruzado *et al.*, "Chatbots for customer service: A comprehensive systematic literature review," *Journal of Theoretical and Applied Information Technology*, vol. 15, p. 19, 2022, accessed: Aug. 19, 2025. [Online]. Available: http://www.jatit.org/volumes/Vol100No19/16Vol100No19.pdf

[3] I. K. F. Haugeland, A. Følstad, C. Taylor, and C. Alexander, "Understanding the user experience of customer service chatbots: An experimental study of chatbot interaction design," *International Journal of Human-Computer Studies*, vol. 161, p. 102788, May 2022.

[4] M. A. Kuhail, N. Alturki, S. Alramlawi, and K. Alhejori, "Interacting with educational chatbots: A systematic review," *Education and Information Technologies*, vol. 28, no. 1, pp. 973–1018, Jan. 2023.

[5] X. Xing, M. Song, Y. Duan, and J. Mou, "Effects of different service failure types and recovery strategies on the consumer response mechanism of chatbots," *Technology in Society*, vol. 70, p. 102049, Aug. 2022.

[6] E. Svikhnushina, A. Placinta, and P. Pu, "User expectations of conversational chatbots based on online reviews," in *Proceedings of the 2021 ACM Designing Interactive Systems Conference*, Jun. 2021, pp. 1481–1491.

[7] H. Lautraite, N. Naji, L. Marceau, M. Queudot, and E. Charton, "Multi-stage clarification in conversational ai: The case of question-answering dialogue systems," Oct. 2021, accessed: Aug. 19, 2025. [Online]. Available: https://arxiv.org/pdf/2110.15235

[8] T. Wu *et al.*, "Promptchainer: Chaining large language model prompts through visual programming," in *Proceedings of the CHI Conference on Human Factors in Computing Systems*, Apr. 2022.

[9] J. Deriu *et al.*, "Survey on evaluation methods for dialogue systems," *Artificial Intelligence Review*, vol. 54, no. 1, pp. 755–810, Jan. 2021.

[10] G. R. S. Silva and E. D. Canedo, "Towards user-centric guidelines for chatbot conversational design," *International Journal of Human–Computer Interaction*, vol. 40, no. 2, pp. 98–120, 2024.

[11] C. L. Hsu and J. C. C. Lin, "Understanding the user satisfaction and loyalty of customer service chatbots," *Journal of Retailing and Consumer Services*, vol. 71, p. 103211, Mar. 2023.

[12] L. M. de Cosmo, L. Piper, and A. Di Vittorio, "The role of attitude toward chatbots and privacy concern on the relationship between attitude toward mobile advertising and behavioral intent to use chatbots," *Italian Journal of Marketing*, vol. 2021, no. 1–2, pp. 83–102, Jun. 2021.

[13] J. Moilanen, A. Visuri, S. A. Suryanarayana, A. Alorwu, K. Yatani, and S. Hosio, "Measuring the effect of mental health chatbot personality on user engagement," in *ACM International Conference Proceeding Series*, Nov. 2022, pp. 138–150.

[14] V. Ta *et al.*, "User experiences of social support from companion chatbots in everyday contexts: Thematic analysis," *Journal of Medical Internet Research*, vol. 22, no. 3, p. e16235, Mar. 2020.

[15] J. Chen, F. Guo, Z. Ren, M. Li, and J. Ham, "Effects of anthropomorphic design cues of chatbots on users' perception and visual behaviors," *International Journal of Human–Computer Interaction*, vol. 40, no. 14, pp. 3636–3654, 2024.

[16] C. Tubin, J. P. Mazuco Rodriguez, and A. C. B. de Marchi, "User experience with conversational agent: a systematic review of assessment methods," *Behaviour & Information Technology*, vol. 41, no. 16, pp. 3519–3529, Dec. 2022.

[17] P. Lewis *et al.*, "Retrieval-augmented generation for knowledge-intensive nlp tasks," in *Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 9459–9474, accessed: Aug. 19, 2025. [Online]. Available: https://github.com/huggingface/transformers/blob/master/

[18] K. Guu, K. Lee, Z. Tung, P. Pasupat, and M.-W. Chang, "Retrieval augmented language model pre-training," in *Proceedings of the 37th International Conference on Machine Learning*, 2020, accessed: Aug. 19, 2025. [Online]. Available: https://proceedings.mlr.press/v119/guu20a.html

[19] C. Prentice and M. Nguyen, "Engaging and retaining customers with ai and employee service," *Journal of Retailing and Consumer Services*, vol. 56, p. 102186, Sep. 2020.

[20] %BIBentryALTinterwordspacing R. Technologies, "Rasa: Open source language understanding and dialogue management," Dec. 2017, accessed: Aug. 19, 2025. [Online]. Available: https://arxiv.org/pdf/1712.05181

[21] "langchain-ai/langchain: Build context-aware reasoning applications," accessed: Aug. 19, 2025. [Online]. Available: https://github.com/langchain-ai/langchain

[22] "Chatgpt," accessed: Aug. 21, 2025. [Online]. Available: https://chatgpt.com/

[23] T. B. Brown *et al.*, "Language models are few-shot learners," in *Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 1877–1901, accessed: Aug. 19, 2025. [Online]. Available: https://commoncrawl.org/the-data/

[24] "Welcome to python.org," accessed: Aug. 21, 2025. [Online]. Available: https://www.python.org/

[25] "pallets/flask: The python micro framework for building web applications," accessed: Aug. 21, 2025. [Online]. Available: https://github.com/pallets/flask

[26] "nomic-ai/nomic-embed-text-v1.5," accessed: Aug. 21, 2025. [Online]. Available: https://huggingface.co/nomic-ai/nomic-embed-text-v1.5
[27] "cross-encoder/ms-marco-minilm-l6-v2," accessed: Aug. 21, 2025. [Online]. Available: https://huggingface.co/cross-encoder/ms-marco-MiniLM-L6-v2
[28] "unsloth/phi-3-mini-4k-instruct-bnb-4bit," accessed: Aug. 21, 2025. [Online]. Available: https://huggingface.co/unsloth/Phi-3-mini-4k-instruct-bnb-4bit
[29] S. Roller *et al.*, "Recipes for building an open-domain chatbot," in *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics*, Apr. 2021, pp. 300–325.
[30] S. Humeau, K. Shuster, M.-A. Lachaux, and J. Weston, "Poly-encoders: Transformer architectures and pre-training strategies for fast and accurate multi-sentence scoring," Apr. 2019, accessed: Aug. 19, 2025. [Online]. Available: https://arxiv.org/pdf/1905.01969
[31] C. Schillings, D. Meissner, B. Erb, D. Schultchen, E. Bendig, and O. Pollatos, "A chatbot-based intervention with elme to improve stress and health-related parameters in a stressed sample," *Frontiers in Digital Health*, vol. 5, p. 1046202, Mar. 2023.
[32] I. Hameed *et al.*, "Based-xai: Breaking ablation studies down for explainable artificial intelligence," Jul. 2022, accessed: Aug. 19, 2025. [Online]. Available: https://arxiv.org/pdf/2207.05566

BIOGRAPHIES OF AUTHORS

Constantinus Satrio is a graduate student in the Master of Computer Science program at Bina Nusantara University. He holds bachelor's degrees in computer applications and computer science. Prior to his postgraduate studies, he established a robust career in the tech industry, holding roles as a software engineer, senior data engineer, and database administrator. His current research focuses on the practical application of Artificial Intelligence, with a particular specialization in building efficient and scalable Retrieval Augmented Generation (RAG) systems. His work aims to bridge the gap between large scale industrial systems and cutting-edge AI research, leveraging his industry background to inform academic inquiry. As the lead author for this study, he was the primary architect of the Complexity Aware Cascade system, responsible for its conceptual design, end-to-end software implementation, and the execution and analysis of the experimental results. He can be contacted at email: constantinus.satrio@binus.ac.id.

Devi Fitrianah is an Associate Professor at the Master of Computer Science Department at Bina Nusantara University. She received her bachelor's degree in computer science from Bina Nusantara University, followed by a master's degree in information technology and a Ph.D. in Computer Science from Universitas Indonesia in 2008 and 2015, respectively. In 2014, she was awarded a sandwich program placement at Michigan State University, USA. Her research interests include Data Mining, Machine Learning, Artificial Intelligence, and Applied Remote Sensing. For this study, Dr. Devi served as the principal investigator and research supervisor. She guided the conceptualization of the cascade architecture, provided critical insights into the methodology, and oversaw the final analysis and validation of the findings. She can be contacted at email: devi.fitrianah@binus.ac.id.