International Journal of Informatics and Computation (IJICOM)
Vol. 7, No. 2, 2025
ISSN: 2685-8711, E-ISSN: 2714-5263 DOI: 10.35842/ijicom

Developing Rule-Based and AI Hybrid Chatbot for Academic Information Services

Ahmad Fauzan Zaky[1], Chanifah Indah Ratnasari[2*]

Abstract

Providing fast and accurate academic information remains a challenge in higher education, particularly when student questions are expressed informally or differ from predefined formats. Rule-based chatbots are commonly used for this purpose but often fail to recognize paraphrased or misspelled inputs, while fully generative chatbots require longer processing time and may produce unreliable responses. To address these limitations, this study developed a dual-mode academic chatbot that combined a rule-based response mechanism with a hybrid reasoning approach incorporating limited generative processing. The system was designed and refined through three iterative prototyping stages, focusing on interface usability, academic knowledge expansion, and reasoning control. After the final iteration, system performance was evaluated using ten student-generated queries that reflected real academic information needs. The evaluation showed that the rule-based mode consistently produced very fast and stable responses but achieved lower accuracy when handling non-standard inputs. The hybrid mode achieved higher response accuracy by better interpreting varied user expressions, although it required substantially longer response time. Overall, the results demonstrated that a controlled hybrid approach improved chatbot robustness while revealing clear trade-offs between response accuracy and computational efficiency.

Keywords:
Academic chatbot, Hybrid reasoning, Rule-based chatbot, Generative chatbot, Large Language Model

1. Introduction

The increasing role of technology in higher education has transformed how information is distributed and accessed by students. Traditionally, academic announcements and course schedules were displayed on bulletin boards or disseminated through manual communication. Today, such information is increasingly delivered through digital platforms such as institutional websites or social media channels. Despite this transition, students still face difficulties in obtaining academic information efficiently, as these platforms require manual searching, lack personalization, and do not support interactive assistance for student-specific inquiries. Similar limitations have also been widely reported in conventional administrative information systems across various domains, where manual or semi-digital processes remain prone to human error, slow processing, and inefficient data management, potentially leading to information inconsistency and data loss [1]. These broader challenges indicate that they are insufficient to support timely and user-centered academic information access.

To address these limitations, conversational agents such as chatbots have increasingly been adopted as alternative information delivery mechanisms. A chatbot is an automated

Corresponding Author: Chanifah Indah Ratnasari (chanifah.indah@uii.ac.id)
1 Ahmad Fauzan Zaky, Universitas Islam Indonesia, ahmad.zaky@students.uii.ac.id
2 Chanifah Indah Ratnasari, Universitas Islam Indonesia, chanifah.indah@uii.ac.id

conversational system capable of responding to user queries using natural language text [2]. In academic contexts, chatbots have been shown to effectively automate responses to frequently asked questions, including course schedules, academic regulations, registration procedures, and supervision requirements [3], [4]. Previous studies have also reported that conversational systems can enhance user engagement by enabling more natural, responsive, and human-like interaction compared to static web-based information systems [5]. These findings suggest that chatbots offer a promising approach for improving accessibility and responsiveness in academic information services.

Nevertheless, many academic chatbots reported in the literature are implemented using rule-based approaches that rely on predefined keywords and fixed response templates [6]. While this approach provides high response consistency and reliability for structured and predictable queries, it performs poorly when handling paraphrased, informal, or linguistically varied inputs commonly used by students. To overcome these limitations, recent studies have explored hybrid chatbot architectures that combine deterministic rule-based logic with generative reasoning based on Large Language Models (LLMs) [7], [8]. Although hybrid approaches offer greater linguistic flexibility, they also introduce challenges related to response controllability, latency, and the risk of generating inaccurate or overly verbose responses. These trade-offs highlight the need for a carefully designed hybrid strategy that balances reliability and flexibility in domain-specific academic settings.

This study proposes a dual-mode academic information chatbot for the Informatics Study Program at Universitas Islam Indonesia (UII), consisting of a purely rule-based mode and a hybrid mode that integrates rule-based logic with constrained generative reasoning. The novelty of this work is articulated as follows. First, the proposed system explicitly separates rule-based and hybrid reasoning modes within a single chatbot, enabling a controlled and transparent comparison between deterministic and generative approaches—an aspect that is often implicit or overlooked in previous studies. Second, unlike many existing hybrid chatbots that rely heavily on generative models, the hybrid mode in this study employs generative reasoning only as a fallback mechanism when rule-based matching fails, thereby improving robustness while limiting hallucination risks. Third, the chatbot is developed using institution-specific academic knowledge rather than generic datasets, ensuring practical relevance and real-world applicability. Finally, this study aims to examine the practical implications of deploying a dual-mode chatbot in an academic environment, with particular attention to the design considerations involved in integrating deterministic and generative reasoning for academic information services.

2. Related Works

The implementation of chatbots in academic information systems has been widely explored as institutions seek to improve accessibility and responsiveness to student inquiries. Ranoliya et al. [9] developed an interactive chatbot for university-related FAQs using Artificial Intelligence Markup Language (AIML), enabling students to obtain campus information through text-based dialogue. However, the system relied entirely on predefined rules and templates, which limited its ability to interpret unstructured, paraphrased, or linguistically varied inputs. Consequently, mismatches between user intent and chatbot responses frequently occurred, illustrating the inherent rigidity of traditional rule-based approaches. Comparable challenges have also been reported in other academic service domains. For instance, a Dialogflow-based chatbot developed for university library FAQ services experienced difficulties when handling mixed-language input, abbreviations, and previously unseen vocabulary, resulting in rigid or inaccurate responses [10]. Although implemented in a different academic context, these findings reinforce the recurring limitation of deterministic chatbot architectures when confronted with the diversity of natural language expressions commonly used by students.

To address the rigidity of purely rule-based chatbot architectures, subsequent studies have explored the integration of artificial intelligence and natural language processing to enable more flexible and interactive academic information services. Ula et al. [3] developed an AI-based chatbot framework designed to enhance academic information services by converting conventional web-based academic portals into interactive, conversational platforms. Their system integrated natural language processing to handle common student queries regarding study programs, schedules, and administrative services. The study demonstrated that AI chatbots can improve information accessibility and efficiency for students by automating routine queries, reducing the dependency on manual staff responses. However, the authors noted limitations in contextual comprehension and response accuracy for complex or ambiguous questions, indicating the need for better intent handling and domain-specific understanding.

Labadze et al. [4] conducted a systematic review on the adoption of AI-based chatbots in higher education, analyzing 67 studies across various domains such as student guidance, administrative information, and virtual tutoring. Their findings revealed that chatbots significantly enhance accessibility and responsiveness in academic communication, particularly for repetitive or procedural queries. However, they also noted challenges in ensuring response accuracy, maintaining contextual consistency, and addressing ethical and privacy concerns. These insights reinforce the importance of designing domain-specific academic chatbots with a balanced integration between rule-based reliability and generative flexibility, as implemented in this study.

Recent research has further emphasized the convergence between rule-based reasoning and large language models (LLMs). Ji et al. [11] highlighted that while generative chatbots can produce contextually rich and natural responses, they are prone to *hallucinations*—generating plausible but inaccurate information. Halvonik and Kapusta [8] demonstrated that combining rule-based precision with generative reasoning improves both reliability and conversational fluency, particularly in domain-specific contexts. Adamopoulou and Moussiades [6] also noted that hybrid chatbots can bridge the gap between deterministic logic and human-like dialogue, giving the best balance between accuracy and flexibility. This approach has been increasingly applied in educational technology, where structured factual content coexists with open-ended student interactions [12].

From these studies, it is evident that although chatbots in educational settings have advanced considerably, most systems still depend either on rigid rule-based logic or fully generative models with hallucination risks. This underscores the necessity of a hybrid academic chatbot that integrates rule-based reliability with the contextual fluency of LLMs, which forms the core contribution of the present study.

3. Proposed Method

This section explains the methodological framework used to design and develop the academic information chatbot for the Informatics Department at Universitas Islam Indonesia (UII). The study applied the prototyping development model, allowing iterative refinement of system functionality, architectural design, and interaction behavior through continuous stakeholder evaluation.

3.1 Prototyping Development Model

The development of the chatbot system follows the prototyping model described by Pressman [13], which emphasizes early system visualization, rapid construction, and repeated refinement based on stakeholder feedback. This approach is particularly appropriate for chatbot development, where conversational flow, response accuracy, and interface usability require iterative validation through real user interaction. Previous studies on interactive system development have demonstrated that the prototyping model enables the rapid delivery of functional systems that can be continuously refined to better align with user needs and system objectives [14]. Furthermore, its iterative structure supports continuous refinement of system behavior based on stakeholder feedback prior to final deployment [15], [16]. In this research, the prototyping model guided the overall development workflow, starting from initial requirement exploration to multiple cycles of construction and refinement, as illustrated in Fig. 1.

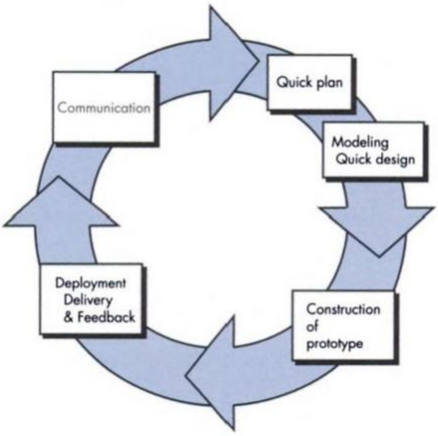


Fig. 1. Prototyping development model [13]

The process begins with the communication phase, where system requirements were gathered through interviews with the study program secretary and academic staff of the Informatics Department and observations of frequently asked student questions. This step identified key information categories—including course-related inquiries, specialization pathways, key-in procedures, and general academic questions—and confirmed the need for two chatbot modes (rule-based and hybrid), a structured knowledge base, and an interface resembling contemporary messaging applications.

The researcher then proceeded to the quick plan and quick design phase. In this stage, the conceptual system architecture was developed, covering the interaction between the PHP-based frontend, Flask middleware, Python chatbot engine, and the text-based knowledge base. Wireframes and interface sketches were created to visualize the initial layout and messaging flow. These early visuals allowed stakeholders to verify the planned functionalities and interaction patterns prior to implementation.

Prototype construction proceeded through multiple iterations, with each version incorporating additional features, improved reasoning mechanisms, interface refinements, and expanded knowledge content. Following prototype construction, the system moved into the deployment, delivery, and feedback stage, where academic staff and student testers interacted with the chatbot across all available modes. Their insights addressed issues such as ambiguous responses, slow generative performance, inadequate keyword coverage, and interface clarity. This stage provided the foundation for subsequent refinements.

The **refinement cycle** focused on addressing identified issues, improving input processing, optimizing keyword detection, adjusting interface elements, and minimizing hallucinations in the hybrid mode. Repeated evaluation ensured that each revision aligned with user expectations, functional requirements, and system performance goals.

Through these iterative cycles, informed directly by user interaction and feedback, the development process produced a stable and functional chatbot prototype suitable for academic information services. The prototyping model ensured a user-centered workflow and systematic refinement leading into the evaluation presented in later sections.

3.2 System Architecture Overview

The chatbot system is implemented using a decoupled multi-layer architecture in which the PHP-based frontend communicates with a Python engine through a Flask API. This separation ensures that the computational workload—particularly for the hybrid mode that relies on a generative model—does not overload the web server environment.

User queries entered in the PHP interface are initially processed as plain text. Before sending the message to the backend, the PHP layer converts the text into JSON, since Flask operates through JSON-based request–response exchanges. Upon receiving the JSON payload, Flask extracts the user message and forwards it to the appropriate reasoning module—either the rule-based component or the hybrid component combining deterministic matching with the Mistral-7B-Instruct-v0.2.Q4_K_M model.

After the chatbot engine produces an output, Flask packages the response into JSON and returns it to the PHP layer. The frontend then renders the message as plain text within the chat interface. This architectural choice was informed by earlier testing, where attempts to call Python scripts directly from the XAMPP environment resulted in significant memory accumulation and frequent crashes when executing the generative model. Flask provides a stable middleware layer that isolates Python processes from the PHP environment and prevents performance bottlenecks.

This system architecture is illustrated in Fig. 2. The chatbot engine implements two distinct reasoning modes: (1) a rule-based mode for fast keyword-driven responses and (2) a hybrid mode that combines rule-based matching with generative AI. Both modes access the same text-based knowledge base containing curated academic information.

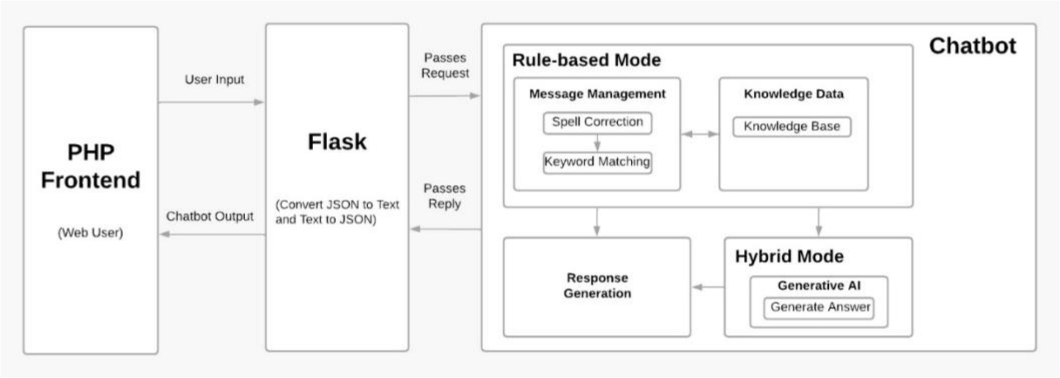


Fig. 2. System architecture of the rule-based and hybrid chatbot

3.3 Chatbot Components

The chatbot consists of several core components that work together to process user messages, retrieve relevant information, and generate responses. The components include the rule-based reasoning mechanism, the hybrid reasoning module, and the knowledge base.

A. Rule-Based Reasoning

The rule-based component matches user queries against predefined keywords extracted from the knowledge base. Before matching, input text undergoes spell correction using the difflib module, which implements the Ratcliff/Obershelp similarity algorithm. This algorithm was selected due to its availability in standard Python libraries and its effectiveness in handling minor variations or typographical errors, as demonstrated by Zidni and Iskandar [17]. If a match is found, the system responds instantly with the corresponding predefined answer. This mode is optimized for speed and used for structured, frequently asked questions.

B. Hybrid Reasoning

The hybrid mode extends rule-based reasoning by incorporating a fallback generative mechanism. If no keyword match is detected, the system invokes the *Mistral-7B-Instruct-v0.2.Q4_K_M* model, an open-source large language model distributed under the Apache 2.0 license [18], [19]. The model was accessed via its GGUF quantized distribution on Hugging Face [18], which provides efficient inference for local execution. Integrating the generative component enhances flexibility in handling paraphrased, semi-structured, or contextual queries while reducing hallucinations commonly associated with purely generative chatbots [11].

C. Knowledge Base Development

The knowledge base is stored in a text file containing keyword–response pairs. Its content was iteratively expanded based on real student questions gathered from administrative staff, interviews, and multiple rounds of user testing. As of the latest iteration, the knowledge base contains entries across the following categories: course and academic requirements, final year paths, key-in mechanisms, and general (not included in other categories). A sample of the knowledge base content is shown in Fig. 3.

```
keywords: Kalender akademik FTI UII
response:
Kalender akademik FTI dapat dilihat di: https://fit.uii.ac.id/blog/2022/07/05/kalender-akademik/
---selesai---

keywords: melihat daftar mata kuliah yang sudah saya ambil jadwal kuliah
response:
Dapat dilihat di isian gateway: Gateway > RAS > Isian.

Juga dapat dilihat di jadwal kuliah: Gateway > Akademik > Jadwal Kuliah.

---selesai---

keywords: paket matakuliah
response:
Kumpulan matakulliah untuk semester tertentu.
---selesai---
```

Fig. 3. A sample of the knowledge base content

3.4 Iterative Development Process

The system underwent three major prototyping iterations, each addressing limitations identified in earlier versions.

Iteration 1: Initial Rule-Based Prototype

The first prototype implemented only the rule-based mode with a minimal knowledge base. The interface was deliberately simple, focusing on validating core functionalities: keyword matching, message parsing, Flask–PHP communication, and basic rule-based response generation. Testing at this stage ensured the correctness of API communication and the stability of text processing.

Iteration 2: Generative Mode Integration and Interface Enhancement

In the second iteration, the generative mode was introduced to address limitations in handling paraphrased or unstructured queries. The interface was redesigned into a more modern chat layout with improved readability, message bubbles, and clearer separation of user and chatbot messages. Additional knowledge entries were incorporated based on interviews with the study program secretary and observations during early testing sessions.

Iteration 3: Hybrid Mode, Usability Refinements, and Knowledge Expansion

The third iteration introduced the hybrid reasoning mechanism, where rule-based matching is attempted first, followed by a generative fallback only when needed. This significantly improved accuracy while reducing unnecessary generative calls.

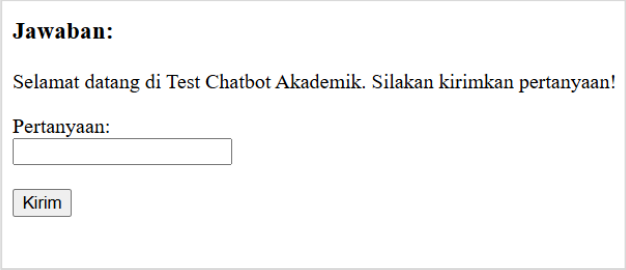
4. Experimental Setup

This section describes the experimental configuration and procedures used to evaluate the performance of the proposed dual-mode academic chatbot. The evaluation focused on response accuracy, response time, and system behavior when processing real student queries that reflect common academic information needs in the Informatics Study Program of Universitas Islam Indonesia (UII).

The experimental environment consisted of a Windows 11 (64-bit) workstation equipped with an AMD Ryzen 5 7535HS processor (six cores and twelve threads, 3.3 GHz) and 16 GB RAM. The frontend interface was developed using PHP version 8.2.12, while the backend engine—including the rule-based component and the hybrid reasoning module—was implemented in Python 3.11.11. Communication between the PHP frontend and the Python engine was facilitated through the Flask framework (version 3.1.2), which served as a lightweight middleware for handling JSON-based requests and responses. The hybrid mode utilized the Mistral-7B-Instruct-v0.2.Q4_K_M model, executed locally in its GGUF-quantized form and distributed under the Apache 2.0 license [18], [19], enabling more efficient inference within the available hardware constraints.

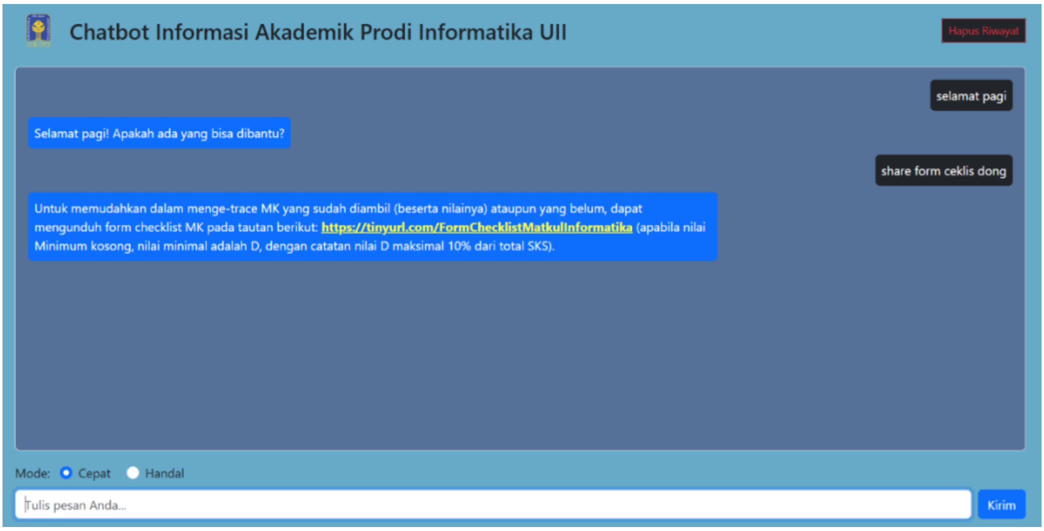
The first prototype focused on validating the core functionality of the rule-based engine and its communication through the API. The interface was intentionally minimal, consisting of a single text field and submission button (Fig. 4(a)). This version enabled evaluators to test core functionalities, including JSON message transmission, response display, and basic keyword matching.

Feedback from this phase indicated that the interface resembled a standard HTML form rather than a conversational chat environment. Testers also noted that typed messages disappeared after submission, making it difficult to track previous responses. These findings informed improvements in Iteration 2, which implemented a chat-style layout with message bubbles and clearer separation of user and chatbot messages (Fig. 4(b)). In Iteration 3, usability enhancements continued, including visual polishing, consistent bubble alignment, and the replacement of technical labels ("rule-based mode," "hybrid mode") with more intuitive names ("Mode Cepat" and "Mode Handal"), as recommended by evaluators (Fig. 4(c)). However, despite these improvements, the Iteration 3 interface still requires further refinement to achieve a cleaner and more polished design, indicating that additional UI enhancement remains necessary for future development.



(a)

(b)



(c)

Fig. 4. Interface of chatbot: (a) iteration 1, (b) iteration 2, (c) iteration 3

User feedback collected throughout the three prototyping iterations highlighted several issues that guided subsequent refinements, including response mismatches, limited keyword coverage, and usability concerns in the early interface. These inputs informed the expansion of the knowledge base, enhancements in interface design, and the gradual transition from a purely rule-based implementation to the hybrid reasoning mechanism. More detailed findings from these iterative evaluations are presented in Section 5.

During the evaluation, four users—two Informatics students and two academic staff members—were asked to freely input queries into the chatbot without any predefined script. These ten queries reflected natural user behavior, including spelling errors, paraphrased expressions, informal phrasing, incomplete questions, and domain-specific terminology, making them suitable for assessing real-world performance. Each query was submitted sequentially to both the rule-based and hybrid modes under the same hardware and software conditions, and each execution was repeated ten times to ensure consistent measurement of processing time. The system's responses were manually examined and validated by the evaluators and researcher, who classified them as correct when the output aligned with the expected academic service information or provided contextually appropriate explanations, and incorrect when it was inaccurate, irrelevant, or incomplete. Responses were independently reviewed by the researcher and academic staff to reduce subjective interpretation. For a subset of queries, tests were rerun when feasible to check consistency and identify runtime variability. Table 1 presents the complete set of queries generated by users during the evaluation process.

Table 1. Summary of user query evaluation for rule-based and hybrid chatbots

No	User Input	Rule-Based Result	Hybrid Result
1	Hello, good morning	Correct	Correct
2	Provide information on courses for the final semester	Incorrect	Correct
3	Provide information on courses for the final semester E.G	Incorrect	Correct
4	Tutorials, key-in	Correct	Correct
5	Share the academic calendar	Correct	Correct
6	Lecturer contact details. Where can I find these?	Correct	Incorrect
7	Schedule	Correct	Incorrect
8	Thesis requirements	Incorrect	Incorrect
9	When does the key-in period begin?	Incorrect	Correct
10	KKN	Correct	Correct

To illustrate the qualitative differences between the two reasoning modes, four representative samples were selected from the full evaluation dataset. These samples were chosen to demonstrate the range of possible outcomes observed during testing, including cases where both modes produced correct responses, cases where only one mode performed correctly, and cases where both modes failed to deliver accurate information. The selected examples are shown in Table 2.

Table 2. Representative examples of chatbot responses in rule-based and hybrid modes

No	User Input	Rule-Based Output	Rule-Based Result	Hybrid Output	Hybrid Result
1	share academic calendar	The FTI academic calendar can be viewed at: https://fit.uii.ac.id/blog/2022/07/05/kalender-akademik/	Correct	I will provide you with a reference to the FTI academic calendar, which can be read in detail at the following link: https://fit.uii.ac.id/blog/2022/07/05/kalender-akademik/. Note: Please ensure that the provided link is correct and active.	Correct
2	schedule	It can be viewed in the gateway input: Gateway > RAS > Input. Also available in the class schedule: Gateway > Academic > Class Schedule.	Correct	Sorry, I do not know the class schedule based on the currently available data.	Incorrect
3	When is the key-in period	CART ≠ KEY-IN. During key-in, courses that are still in the cart are not considered keyed-in. Click the “Key-in” button.	Incorrect	Sorry, I do not know based on the available data. The start time for course key-in in UIIRAS is not provided in the given information. Please refer to the video tutorial for more details.	Correct
4	Thesis requirements	The attendance requirement per class is 75% attendance.	Incorrect	To meet the thesis requirements, students must attend 75% of the thesis course semester.	Incorrect

5. Result and Analysis

User testing across three development iterations demonstrated a clear progression in usability, response accuracy, and conversational fluency. In Iteration 1, users frequently encountered response mismatches caused by rigid keyword matching and the limited scope of the early knowledge base; the interface also appeared non-conversational and lacked message persistence, causing confusion during interaction. These limitations were addressed in Iteration 2 through a substantial expansion of the knowledge base to cover commonly used paraphrases, the introduction of the generative chatbot, and a redesigned interface that improved readability and interaction flow. By Iteration 3, the system incorporated a hybrid reasoning module that combined rule-based and generative mechanisms within a single interface. The hybrid mode enabled generative fallback only when rule-based matching failed, reducing hallucinations and improving performance on informal or partially structured queries. Interface refinements implemented during this stage also made the chatbot resemble familiar messaging platforms, resulting in higher clarity, consistency, and overall usability as reflected in user feedback.

The representative examples in Table 2 provide a qualitative view of how each reasoning mode behaves across different types of user input and highlight their complementary strengths and limitations. Cases where both modes delivered correct responses indicate that the curated knowledge base already covers common academic queries adequately. Instances in which the hybrid mode succeeded while the rule-based mode failed demonstrate its strength in handling paraphrased, informal, or misspelled inputs beyond strict keyword patterns. Conversely, situations where the rule-based mode produced accurate responses but the hybrid mode did not show that generative fallback may become overly cautious or misinterpret limited textual cues. These patterns align with broader qualitative observations recorded during testing, where the rule-based mode consistently produced stable, predictable outputs, while the hybrid mode exhibited greater linguistic adaptability but occasionally generated verbose or slightly inaccurate responses—typical characteristics of LLM-based generative systems. Collectively, these findings demonstrate that the hybrid model improves robustness for diverse user inputs, although its effectiveness still depends on the completeness of the curated knowledge base and the inherent variability of generative reasoning. These qualitative patterns show how the two modes complement each other and clarify the situations in which their respective limitations emerge during real interactions.

The accuracy results are summarized in Table 3 and show clear performance differences between the two modes. The rule-based chatbot achieved 60% accuracy, with most errors occurring when queries involved paraphrasing, misspellings, or information not represented in the knowledge base—conditions that deterministic keyword matching inherently struggles with, as also noted in prior studies [6]. In contrast, the hybrid chatbot achieved 70% accuracy, benefiting from the generative fallback that allowed it to interpret semantically similar or loosely structured queries. However, two responses exhibited hallucination-like characteristics, consistent with the documented limitations of LLM-based systems [11]. These quantitative findings reinforce that while hybrid reasoning improves accuracy, particularly for flexible or unstructured inputs, the generative component still presents inherent reliability challenges.

Table 3. Accuracy comparison between rule-based and hybrid chatbot modes

Mode	Accuracy
Rule-based	60%
Hybrid	70%

To further explain the accuracy differences observed in Table 3, it is important to examine how each reasoning approach processes user input. The rule-based chatbot relies on exact or near-exact keyword matching against a curated knowledge base, which ensures highly consistent and reliable responses for well-structured academic queries. However, this deterministic mechanism becomes a major limitation when users employ paraphrased expressions, informal language, or spelling variations, leading to incorrect or unanswered responses.

In contrast, the hybrid chatbot integrates a generative component that enables semantic interpretation beyond surface-level keyword matching. This capability significantly improves the system's ability to handle loosely structured, ambiguous, or linguistically varied queries, which directly contributes to its higher accuracy score. Nevertheless, the inclusion of generative reasoning introduces a new source of error, as evidenced by a small number of hallucination-like responses. These results indicate that the accuracy improvement in the hybrid approach is primarily driven by enhanced linguistic flexibility, while the remaining errors stem from the inherent uncertainty of generative language models. This trade-off clearly explains why the hybrid mode achieves higher accuracy overall while still exhibiting occasional reliability issues.

In addition to accuracy, response time was evaluated by measuring the duration between the moment a JSON request was submitted by the frontend and the moment the JSON response was returned by Flask. Table 4 presents the detailed results across ten repeated runs per mode. The rule-based chatbot demonstrated consistently fast performance, with average response times of approximately 0.025 seconds. This speed reflects the lightweight nature of keyword-matching operations and the minimal computational overhead involved. Conversely, the hybrid chatbot required substantially longer processing times, averaging 25.268 seconds. This delay is attributable to the model loading, tokenization, and text generation steps required by the Mistral-7B-Instruct model, which aligns with the latency characteristics reported in research on LLM-based conversational systems [8].

Table 4. Response time comparison between rule-based and hybrid chatbot modes

Test No.	Rule-Based Chatbot (s)	Hybrid Chatbot (s)
1	0.03	21.08
2	0.02	11.17
3	0.03	11.55
4	0.03	22.37
5	0.03	21.57
6	0.02	22.06
7	0.02	28.36
8	0.02	28.85
9	0.02	29.74
10	0.03	28.21
Average	**0.025**	**25.268**
Standard Deviation	**0.005**	**6.758**

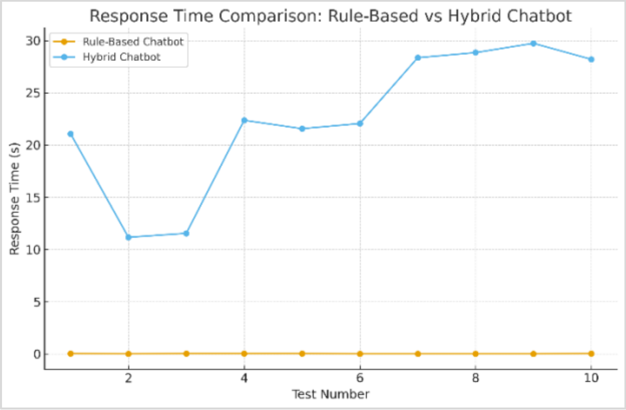


Fig. 5. Response time comparison between rule-based and hybrid chatbot modes

Fig. 5 illustrates the response time differences between the rule-based and hybrid chatbot modes across ten test queries. The rule-based chatbot consistently shows extremely low latency, with values clustered around 0.02–0.03 seconds. In contrast, the hybrid chatbot exhibits significantly higher processing time, ranging from approximately 11 to 29 seconds, reflecting the overhead of generative model processing.

The prototyping iterations also influenced system behavior. During early testing, the generative model exhibited slower performance and produced longer, less focused responses. Subsequent refinements—including updated labeling ("Mode Cepat" for rule-based mode and "Mode Handal" for hybrid mode), interface adjustments, and expanded knowledge-base coverage—improved overall usability and response relevance. These refinements contributed to a more balanced interaction experience, with the hybrid mode becoming more controlled and the rule-based mode achieving broader keyword coverage.

Overall, the experimental results confirm that the two modes serve complementary purposes. The rule-based mode is highly suitable for structured academic FAQs requiring rapid response times, while the hybrid mode is more effective for handling varied, ambiguous, or paraphrased student questions, despite its significantly higher latency. These findings align with recent literature indicating that hybrid chatbots provide a practical balance between deterministic precision and generative flexibility in domain-specific applications [6], [8].

6. Conclusion

This study addressed the challenge of providing an academic information service capable of handling both structured and loosely formulated student queries through the development of a dual-mode chatbot system. The proposed approach consists of two reasoning modes: a rule-based chatbot designed for fast and deterministic responses and a hybrid chatbot that integrates rule-based logic with generative AI to improve linguistic adaptability. The experimental results confirm findings from existing literature, showing that rule-based systems excel in speed and consistency, while hybrid or generative approaches offer stronger linguistic flexibility at the cost of higher computational latency. In this study, the rule-based mode proved highly effective for structured academic queries such as schedules, procedures, and frequently asked questions due to its predictable behavior and extremely low response time. In contrast, the hybrid mode achieved higher overall accuracy when handling paraphrased, informal, or ambiguous queries by leveraging generative reasoning, albeit with increased response time and occasional hallucination-like outputs.

These findings indicate that neither approach alone is sufficient to address the full spectrum of student information needs. Instead, maintaining both rule-based and hybrid chatbot modes within a single academic information system enables complementary strengths to be utilized depending on query complexity. The main contribution of this research lies in demonstrating how a lightweight hybrid architecture, combined with a curated rule-based knowledge base, can enhance robustness and adaptability without fully replacing deterministic methods. Looking forward, future work will focus on improving the completeness and maintainability of the academic knowledge base, optimizing generative inference time to reduce latency, and expanding evaluation to larger and more diverse user groups. Further research may also explore lightweight language models, confidence-aware response strategies, and adaptive mode-switching mechanisms to better balance accuracy, reliability, and computational efficiency in real-world academic deployments.

References

[1] B. R. Gumilang, A. Nugroho, and A. T. Zy, “Development of a Web-based Administration System for The Bantu Sodara Community Using the Waterfall Method,” *International Journal of Informatics and Computation (IJICOM)*, vol. 6, no. 2, 2024, doi: 10.35842/ijicom.v6i2.95.

[2] G. Ilieva, T. Yankova, S. Klisarova-Belcheva, A. Dimitrov, M. Bratkov, and D. Angelov, “Effects of Generative Chatbots in Higher Education,” *Information*, vol. 14, no. 9, p. 492, Sep. 2023, doi: 10.3390/info14090492.

[3] M. Ula, R. Hardi, and I. Hipiny, “An Improved Structure for Academic Information Services through AI Chatbots,” *Journal of Engineering Science and Technology Review*, vol. 16, no. 5, pp. 164–173, 2023, doi: 10.25103/jestr.165.20.

[4] L. Labadze, M. Grigolia, and L. Machaidze, “Role of AI chatbots in education: systematic literature review,” Dec. 01, 2023, *Springer Science and Business Media Deutschland GmbH*. doi: 10.1186/s41239-023-00426-1.

[5] M. Skjuve, A. Følstad, K. I. Fostervold, and P. B. Brandtzaeg, “My Chatbot Companion - a Study of Human-Chatbot Relationships,” *International Journal of Human Computer Studies*, vol. 149, May 2021, doi: 10.1016/j.ijhcs.2021.102601.

[6] E. Adamopoulou and L. Moussiades, “Chatbots: History, technology, and applications,” *Machine Learning with Applications*, vol. 2, pp. 1–18, Dec. 2020, doi: 10.1016/j.mlwa.2020.100006.

[7] S. A. Thorat and V. D. Jadhav, “A Review on Implementation Issues of Rule-based Chatbot Systems.” [Online]. Available: https://ssrn.com/abstract=3567047

[8] D. Halvonik and J. Kapusta, “Large Language Models and Rule-Based Approaches in Domain-Specific Communication,” *IEEE Access*, vol. 12, pp. 107046–107058, 2024, doi: 10.1109/ACCESS.2024.3436902.

[9] B. R. Ranoliya, N. Raghuwanshi, and S. Singh, "Chatbot for university related FAQs," in *2017 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2017*, Institute of Electrical and Electronics Engineers Inc., Nov. 2017, pp. 1525–1530. doi: 10.1109/ICACCI.2017.8126057.

[10] S. Prananta Barus and E. Surijati, "Chatbot with Dialogflow for FAQ Services in Matana University Library," *International Journal of Informatics and Computation (IJICOM)*, vol. 3, no. 2, 2021, doi: 10.35842/ijicom.v3i2.43.

[11] Z. Ji *et al.*, "Survey of Hallucination in Natural Language Generation," Dec. 31, 2023, *Association for Computing Machinery*. doi: 10.1145/3571730.

[12] T. E. Putri and G. Ramadhan, "Penerapan Chatbot sebagai Alat Pembelajaran untuk Pengembangan Pendidikan Karakter," 2024.

[13] R. S. Pressman, *Software Engineering: A Practitioner's Approach*. McGraw-Hill, 2005.

[14] M. I. Abigail and K. Haryono, "Building an Effective Quran Learning System with Prototyping Model," *International Journal of Informatics and Computation (IJICOM)*, vol. 7, no. 2, p. 2025, doi: 10.35842/ijicom.v7i2.138.

[15] N. Rachma and I. Muhlas, "Comparison Of Waterfall And Prototyping Models In Research And Development (R&D) Methods For Android-Based Learning Application Design," *Jurnal Inovatif : Inovasi Teknologi Informasi dan Informatika*, vol. 5, no. 1, pp. 36–39, 2022, doi: 10.32832/inovatif.

[16] B. Camburn *et al.*, "Design prototyping methods: State of the art in strategies, techniques, and guidelines," *Design Science*, vol. 3, 2017, doi: 10.1017/dsj.2017.10.

[17] P. Zidni and A. Iskandar, "Perbandingan Algoritma Ratcliff/Obershelp dan Levenshtein Distance dalam Chatbot AIML untuk Universitas XYZ."

[18] TheBloke, "Mistral-7B-Instruct-v0.2-GGUF," Hugging Face. [Online]. Available: https://huggingface.co/TheBloke/Mistral-7B-Instruct-v0.2-GGUF

[19] Apache Software Foundation, "Apache License, Version 2.0." [Online]. Available: https://www.apache.org/licenses/LICENSE-2.0