

ITEBIS PGRI Dewantara Academic Service Chatbot Based on IndoBERT and Retrieval-Augmented Generation (RAG)

Syaiful Imron^{1*}, Arbiati Faizah², Sugianto³

^{1,2,3}Systems and Technology Information, Faculty of Information Technology,
Institute of Technology and Business PGRI Dewantara Jombang, Indonesia

Abstract. Academic administrative services must stay responsive beyond regular hours, yet repetitive student inquiries add to staff workload. Retrieval-Augmented Generation (RAG) with a fine-tuned language model offers a way to automate accurate, round the clock responses in Indonesian.

Purpose: This study develops a prototype academic-service chatbot for ITEBIS PGRI Dewantara Jombang, integrating a fine-tuned IndoBERT model with RAG architecture to address limited-service hours and repetitive handling of routine student inquiries.

Methods/Study design/approach: The system used 2,517 SQuAD-format question-answer pairs, split into 72% training (1,812), 8% validation (201), and 20% test (504). IndoBERT (indobenchmark/indobert-base-p1) was fine-tuned as an extractive reader for five epochs (batch size 8, learning rate 3×10^{-5}) with early stopping, then converted to ONNX format and INT8-quantized for CPU-based local deployment. The prototype separates a Laravel web interface from a FastAPI backend running the RAG pipeline, retrieving passages from a MySQL knowledge base via FAISS search (top-k = 5, min score 0.30) and catching frequent query-answer pairs in Redis to cut redundant database access and inference latency. Evaluation used black-box testing across five scenarios.

Result/Findings: On the 504-pair test set, the model achieved an Exact Match of 71.32% and an F1-Score of 86.63%. All five black-box scenarios passed, including the system's ability to give database-grounded responses within the tested scenarios rather than fabricated answers, though retrieval for categories with fewer passages (e.g., finance) was more sensitive to question phrasing

Novelty/Originality/Value: The novelty lies not in any single element but in combining four components: a fine-tuned IndoBERT reader for Indonesian, an academic-domain RAG architecture, a fully functional end-to-end web prototype rather than model evaluation alone, and an ONNX INT8-quantized reader paired with a Redis caching layer that jointly reduce inference latency and database load a combination not jointly addressed in prior IndoBERT or RAG-based chatbot studies.

Keywords: Chatbot, IndoBERT, Retrieval-Augmented Generation, Natural Language Processing, Academic Services

Received August 2026 / **Revised** September 2026 / **Accepted** September 2026

This work is licensed under a [Creative Commons Attribution 4.0 International License](https://creativecommons.org/licenses/by/4.0/).



INTRODUCTION

The rapid development of information technology in the industry 4.0 era requires educational institutions, including ITEBIS PGRI Dewantara Jombang, to continuously innovate in providing the best service to students. One crucial aspect of higher education management is the quality of administrative and academic services [1]. In practice, however, academic services at ITEBIS PGRI Dewantara Jombang still face several operational challenges. First, service hours are limited to regular working hours, while students' requests for information often arise outside those hours [2]. Second, administrative staff struggle to handle many questions simultaneously, resulting in a repetitive workload from answering similar questions from students [3]. This has the potential to create human resource management inefficiencies within the campus environment.

To solve this issue, a chatbot is needed as a virtual assistant based on artificial intelligence capable of automatically simulating human conversation [4]. This technology enables fast, accurate information services to be available 24 hours a day without operational time constraints [5]. Chatbot technology based on artificial intelligence through a Natural Language Processing (NLP) approach is a highly relevant

*Corresponding author.

Email addresses: imron@itebisdewantara.ac.id (Imron)

DOI: 10.15294/rji.v4i2.62938

solution. Language models based on the transformer architecture, particularly Bidirectional Encoder Representations from Transformers (BERT) and its Indonesian-language variant (IndoBERT), have become the foundation of various Indonesian natural language understanding applications, ranging from text classification to extractive question answering [6], [7], [8].

A systematic review of prior literature reveals three main groupings that help define the current study's position. Early applications [9], [10], [11] largely relied on closed-book IndoBERT setups for paraphrase and intent identification without incorporating external retrieval systems, which ultimately force continuous model retraining whenever information changes. A different subset of researchers [12], [13] successfully introduced RAG architectures into corporate and aviation sectors, though they entirely bypassed both the Indonesian language and the academic service domain. While a third wave of research [14], [15], [16] did explore RAG-based chatbots for educational institutions, their assessments were predominantly restricted to localized answer quality, they missed the opportunity to present a fully integrated web prototype capable of end-to-end user testing, and similarly lacked a dedicated focus on Indonesian language processing.

Bridging a notable gap in current literature, this project focuses on developing a prototype of an academic service chatbot a fine-tuned IndoBERT model with an extractive RAG architecture, explicitly engineered to handle and reply to interactions in the Indonesian language. What sets this study apart is its unique approach of combining four specific aspects at once. First, it deploys a fine-tuned IndoBERT model to act as an extractive reader that is highly optimized for understanding Indonesian text. Second, it links this model to a RAG architecture that's pulled directly from a customized academic database specific to the institution. Third, it involves creating a complete web prototype, proving that's model initially built in Google Collab can be successfully deployed in a local environment with a user-friendly interface. Finally, it incorporates real-world black-box testing to verify system functionality. By introducing this technology, the institution can automate its academic inquiries, which is expected to boost day-to-day campus efficiency, upgrade service quality, and ensure long-term student satisfaction.

METHODS

This research was carried out through four main stages as illustrated in Figure 1. The research stages include collecting data from official academic domain documents, fine-tuning the IndoBERT model in the Google Colab environment, building a web-based user interface (UI) connected to an AI backend, and conducting functional testing (blackbox testing) on the integrated prototype.

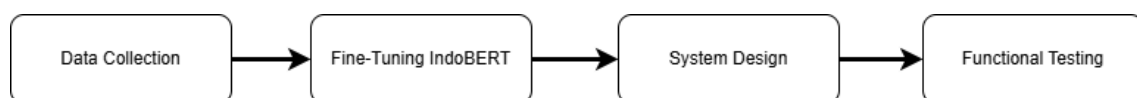


Figure 1. Research Framework

Data Collection

The reliability of a question-answering model depends heavily on the quality and relevance of the data corpus used to train it [17]. Therefore, this study uses 2,517 data observations structured in the SQuAD (Stanford Question Answering Dataset) format, which maps the tripartite relationship among the reference text (context), the user query (question), and the target extraction span (answer span). Before entering the modeling phase, the dataset underwent a series of preprocessing steps that included text sanitization, JSON structure integrity validation, and automatic index-position correction (auto-fix) of the `answer_start` attribute. This calibration step is crucial to ensure that the target text span is fully aligned with the original text, in order to prevent anomalies during parameter weight updates [17]. Furthermore, to satisfy rigorous empirical evaluation standards, the observations were partitioned proportionally and randomly at an 80:20 ratio. A total of 80% of the corpus (2,013 pairs) was allocated as the training data to provide optimal computational balance [18], while the remaining 20% (504 pairs) was isolated as the test data. This isolation of the test data is essential for simulating real-world scenarios by presenting unseen data, so that the system's generalization capability can be objectively measured while avoiding potential overfitting [19].

To facilitate the early stopping mechanism, the training data was internally partitioned using a 90:10 ratio. This specific split produced an active training set of 1,812 pairs (72% of the entire corpus) alongside a validation data consisting of 201 pairs (8%). The final configuration of the research dataset was distributed

into three distinct segments, 72% for training (1,812 pairs), 8% for validation (201 pairs), and 20% for testing (504 pairs). The validation data was exclusively designated to track training loss and trigger the early stopping condition. In contrast, the test data was rigorously quarantined from all training activities. It was deployed only once at the conclusion of the study to compute the final Exact Match and F1-Score metrics detailed in the Results and Discussion section.

Based on the exploratory analysis visualized in Figure 2, this dataset has dense text dimensionality: the average length of the reference text and the answer reach around 221 characters, while the question query averages 37 characters. The subject classification distribution also shows asymmetric stratification, dominated by the “*institusi*” entity (940 observations) and “*akademik*” entity (706 observations), followed by the “*administrasi*” (343), “*wisuda*” (222), “*prodi*” (128), “*kemahasiswaan*” (108), and “*keuangan*” (70).

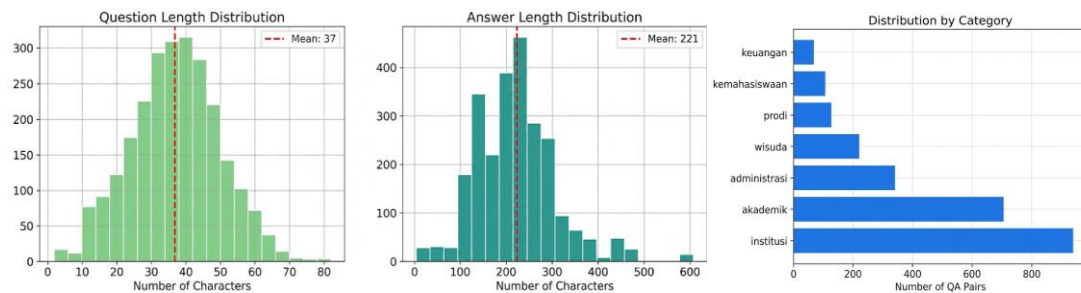


Figure 2. Dataset characteristics and distribution

Fine-Tuning IndoBERT

A crucial step in this study was performing fine-tuning on the IndoBERT (pre-trained model) language model (indobenchmark/indobert-base-p1). This process began with tokenization using the tokenizer built into the model, configured using a sliding window scheme (stride=128) to accommodate context text that exceeds the maximum token length. At this stage, the model was adapted using the academic-domain dataset so that it could understand questions and precisely extract the answer text span (answer span). The entire training process was carried out in the Google Colaboratory environment using an NVIDIA Tesla T4 GPU hardware accelerator. Technically, the fine-tuning implementation relied on the Hugging Face Transformers library along with the AdamW optimizer [20].

The hyperparameter configuration applied included a training duration of 5 epochs, a batch size of 8, a learning rate of 3×10^{-5} , and the use of mixed precision (FP16) to improve memory computation efficiency. These parameter values represent a standard proven to maintain training stability without damaging the original weights of the pre-trained model [21]. To prevent overfitting during the iteration process, an early stopping mechanism with a patience of 2 was also applied. Furthermore, the performance of the extractive question answering model was comprehensively evaluated using two standard measurement metrics. The first metric is Exact Match (EM), which calculates the proportion of predictions that exactly match the reference answer. The second metric is F1-Score, which evaluates the token overlap ratio between the predicted and reference answers, thereby providing a more tolerant assessment of surface-level text variation [22].

To ensure seamless interaction on resource-constrained hardware, the fine-tuned IndoBERT model is not deployed directly in its native PyTorch format. Instead, the model is initially transformed into the Open Neural Network Exchange (ONNX) format. Following this, dynamic quantization is applied to achieve INT8 precision by leveraging the ONNX Runtime library. This transformation significantly shrinks the model's footprint while simultaneously boosting inference speed on standard central processing units (CPUs), completely bypassing the need for dedicated GPU accelerators. Such an approach perfectly aligns with the objective of deploying the prototype locally, where high-end computational resources might be absent. Finally, this INT8-quantized ONNX model is loaded by the FastAPI backend to function as the reader module within the RAG architecture.

System Design

The prototype system architecture was developed using the separation of concerns paradigm, in which the user interface and the artificial-intelligence logic run in isolated yet interconnected environments. Technically, the prototype was built by integrating two main pillars through a RESTful API communication channel using the JSON data-exchange format, as illustrated in detail in Figure 3. At the presentation layer, the framework Laravel plays a crucial role as the face of the application (web interface) that facilitates direct interaction with users, in this case students. On the other hand, the inference layer, or AI backend, is specifically driven by FastAPI. This Python-based framework carries out high-level computational tasks, namely executing the Retrieval-Augmented Generation (RAG) mechanism combined with answer extraction by the fine-tuning IndoBERT model. This separation is consistent with best practices in intelligent-system software engineering, which recommend decoupling heavy NLP computation from interface presentation services [23], [24].

The operational workflow cycle was designed to run smoothly and responsively. The interaction begins when the user enters a question query through the chat page on the web interface. The request is then transmitted over the HTTP protocol as an API request to the endpoint FastAPI. Once the instruction is received, RAG immediately executes the retrieval stage, namely scanning and retrieving the text segment (passage) with the highest semantic relevance from the Knowledge Database stored in a MySQL database. To optimize the semantic similarity search during the retrieval stage, the system integrates Facebook AI Similarity Search (FAISS) utilizing the faiss-cpu package [13]. Operationally, raw text documents stored within the MySQL database are first converted into vector embeddings before being physically compiled into an index file named `faiss_index.bin` [13]. To guarantee that only highly relevant information is extracted, the vector searching algorithm is constrained to return a maximum of five results (`top_k = 5`) that meet a strict minimum confidence score threshold of 0.30 [12]. Utilizing this mathematical configuration minimizes the latency associated with calculating inter-document distances. Consequently, the system preserves a high degree of contextual accuracy prior to feeding the extracted passages into the IndoBERT reader module. Once the specific context is found, the IndoBERT model takes over as the reader to read the entire text and isolate the most precise word span as the answer. The final output is then repackaged as a JSON response and distributed to the Laravel interface to be displayed to the user.

Beyond the model-level optimizations, the prototype incorporates a Redis based in memory caching layer strategically positioned between the FastAPI backend and the MySQL Knowledge Database. Upon successfully processing an inquiry, the corresponding query retrieval pair is temporarily cached in Redis using a predefined time to live (TTL) mechanism. Consequently, subsequent identical or highly similar requests can be fulfilled directly from the cache, completely bypassing the need for redundant database searches and model inferences. This caching strategy is deliberately designed to minimize average response times for frequently asked questions such as admission prerequisites or academic schedules while simultaneously alleviating repetitive read operations on the central database during high-concurrency user access.

At this development stage, all component operations along with functional testing were carried out exclusively in the local development environment (localhost). This architectural decision was made to focus the evaluation on validating the system design, the smoothness of inter-module (API) communication, and systematic technical verification of the IndoBERT-RAG flow. Consequently, resilience testing (stress testing) and latency performance on a large-scale production server environment were not yet accommodated and are instead planned as a research escalation for the subsequent implementation phase. Shifting from cloud-based modeling to a local end-to-end deployment requires a careful reconfiguration of the software infrastructure. To ensure the FastAPI backend handles AI inference independently, several core libraries (fastAPI, uvicorn, transformers, and the torch framework) must be pre-installed within the Python environment. Beyond just the software dependencies, every model artifact produced during the Google Colab training sessions needs to be completely moved into the local working directory. This includes the `pytorch_model.bin` parameter file, tokenization configurations, the FAISS index, and the document mapping stored in `passages.json`. Physically migrating these assets is a mandatory technical step. Doing so guarantees that the web frontend can seamlessly access the AI features without relying on any experimental cloud runtimes.

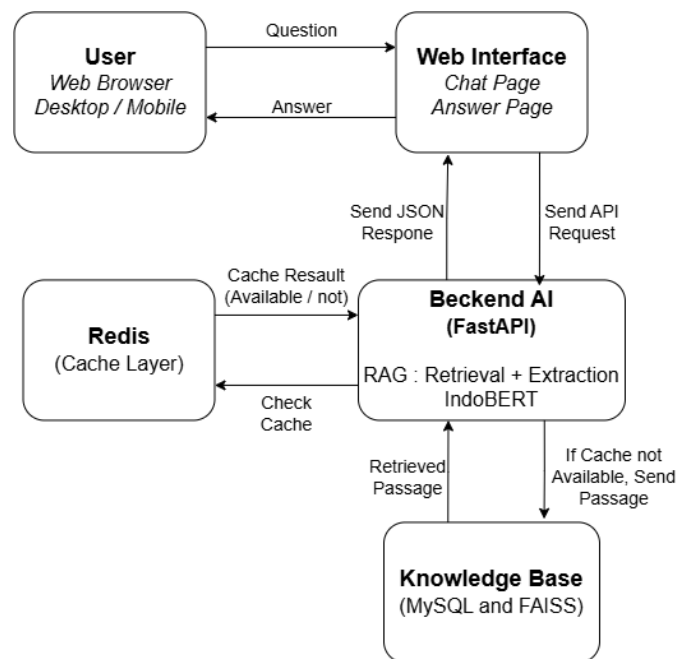


Figure 3. System design

Functional Testing

The final methodological step involves functional evaluation, specifically through black-box testing of the integrated prototype. This assessment aims to verify that the entire user interaction pipeline functions according to the initial design specifications. The evaluated flow encompasses the user's initial access to the web interface, the submission of their inquiries, and the backend's subsequent delivery of fact-based responses. Fundamentally, this testing approach ignores internal source code or structural mechanics; instead, it measures software performance strictly from the end-user's viewpoint. To achieve this, we applied various operational scenarios, evaluating how the system handles standard queries, help-prompt triggers, out-of-scope questions that exceed the core knowledge base, and mobile UI responsiveness. Empirical support for applying this framework in conversational AI is well-documented. For instance, research by [25] demonstrates that black-box testing effectively validates both the functional mechanics and interaction pathways of AI chatbots, ensuring they process natural language inputs accurately and consistently.

RESULT AND DISCUSSION

The Results and Discussion section presents the experimental and implementation findings obtained through the fine-tuning and evaluation process of the IndoBERT language model (indobenchmark/indobert-base-p1) connected to the Retrieval-Augmented Generation (RAG) framework. The analysis was carried out comprehensively covering the model's computational performance during training, the extraction accuracy on the testing dataset, the architectural structure of the knowledge database, and the functional testing of user interaction on the prototype interface. The overall research results are visualized in the form of a training metrics chart (learning curve), a confusion matrix, an user interface, and a blackbox testing table, to provide a complete picture of the chatbot's ability to identify queries and extract academic service information instantly and precisely.

Model Performance Evaluation

Loss decreased sharply at the beginning of training and converged within under 100 steps, while validation loss was recorded near zero by the end of the first epoch, as shown in Figure 4. This rapid convergence pattern is consistent with the characteristics of fine-tuning a medium-scale pre-trained model on a domain-specific dataset containing thousands of pairs.

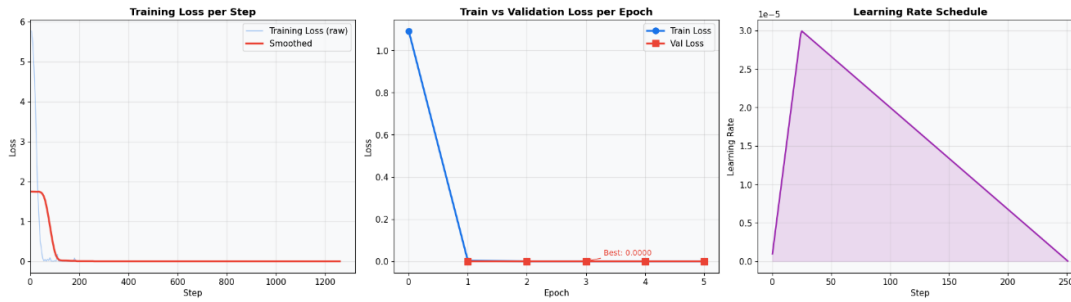


Figure 4. Training Metrics

Evaluation on 504 test dataset pairs yielded an Exact Match of 71.32% and an F1-Score of 86.63%, as shown in Figure 5. The 15.31-point gap between F1-Score and Exact Match indicates that the model quite often produces answers that are substantively correct but not word-for-word identical to the reference answer, a pattern that is typical for Indonesian-language extractive question answering. Overall, the model's accuracy on both metrics is considered adequate for the prototype stage, although room for improvement remains, particularly for categories with a smaller number of passages.

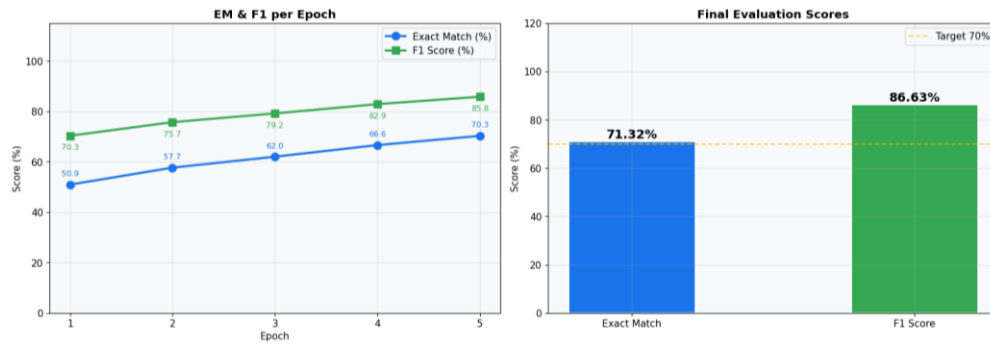


Figure 5. Exact Match dan F1-Score model IndoBERT

To examine the system's classification and retrieval performance in more detail, Figure 6 presents the confusion matrix from cross-testing on the test dataset. The dense main-diagonal pattern in the matrix confirms a high rate of prediction success (true positives), especially in dominant clusters such as “Institution” (154 correct predictions) and “Academic” (118 correct predictions). On the other hand, this visualization also reveals model vulnerability in areas of semantic overlap with limited data. For example, there is marginal misclassification in which queries that should belong to the “Institution” cluster were incorrectly mapped to the “Student Affairs” cluster in 9 cases. The empirical reality of this matrix reinforces the conclusion that enriching the number of query variations for long-tail intent categories is essential to reduce classification bias and sharpen the accuracy of the retrieval mechanism in the future.

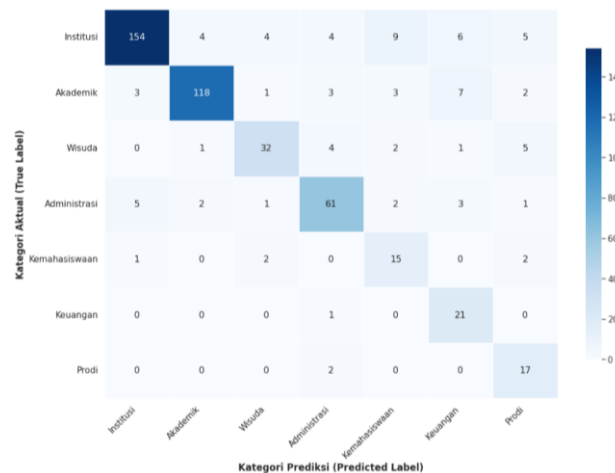


Figure 6. Confusion Matrix

Knowledge Database

Knowledge database plays a crucial role as the primary knowledge foundation underpinning the Retrieval-Augmented Generation (RAG) architecture during the inference process. A visual representation of this knowledge-base management interface can be seen directly in Figure 7, which shows a comprehensive administrative dashboard. In this module, each academic document is encapsulated along with its systematically structured metadata attributes, including category classification (such as BAAK), specific title assignment, and a preview of the descriptive text content (context preview). This modular data organization was deliberately designed to optimize the efficiency of the search (retrieval) algorithm, so that the system can quickly and precisely calculate and isolate the documents with the highest semantic similarity to the user's query.

#	Kategori	Judul	Context (preview)	Sumber	Status	Aksi
1	BAAK	Pengajuan SKL	Selamat atas kelulusannya! Untuk mengajukan SKL (Surat Keterangan Lulus) se...	Dataset ITEBIS - pengajuan_skl	Aktif	[Edit] [Delete]
2	BAAK	Pengambilan Ijazah Transkrip	Ijazah dan transkrip nilai asli dapat diambil setelah proses wisuda selesai...	Dataset ITEBIS - pengambilan_ijazah_transkrip	Aktif	[Edit] [Delete]
3	BAAK	Pengambilan SKL	Setelah selesai diproses, fisik SKL dapat diambil langsung di bagian BAU (B...	Dataset ITEBIS - pengambilan_skl	Aktif	[Edit] [Delete]
4	BAAK	Pengambilan Toga	Atribut wisuda seperti toga, topi, dan samir dapat diambil di bagian perlen...	Dataset ITEBIS - pengambilan_toga	Aktif	[Edit] [Delete]
5	BAAK	Pengertian Ktm	Kartu Tanda Mahasiswa (KTM) adalah identitas resmi mahasiswa ITEBIS PGRI De...	Dataset ITEBIS - pengertian_ktm	Aktif	[Edit] [Delete]
6	BAAK	Pengertian Mahasiswa Pindahan	Mahasiswa Pindahan adalah mahasiswa yang masuk untuk melanjutkan studi di I...	Dataset ITEBIS - pengertian_mahasiswa_pindahan	Aktif	[Edit] [Delete]
7	BAAK	Pengertian Mahasiswa	Mahasiswa Transfer adalah mahasiswa yang masuk untuk...	Dataset ITEBIS - pengertian_mahasiswa_transfer	Aktif	[Edit] [Delete]

Figure 7. Knowledge Database

The presence of this knowledge base management interface also provides empirical confirmation that the RAG component of the prototype chatbot has been physically implemented, going beyond a mere conceptual design on paper. The existence of a storage table structure that includes activation status and data-modification actions demonstrates that the knowledge database is dynamic and responsive to change. This ensures that system administrators have full control to monitor, update, and delete passages, keeping them consistently relevant to updated campus regulations. This technical validation clearly confirms that the application prototype has been equipped with a functional, robust database infrastructure that is highly capable of supporting real academic information extraction needs.

User Interface Implementation

The user-interface implementation of the prototype was designed to be interactive and responsive, providing a functional dialogue space that adapts optimally to both desktop and mobile layouts. Through this conversation window, students can ask various questions about campus services and receive instant feedback (real-time) thanks to solid integration with the AI server based on FastAPI. The smooth back-and-forth data transmission between the web interface and the main processing engine validates that the system architecture is precisely connected and able to operate very well in the local testing environment (localhost). The advantage of the Retrieval-Augmented Generation (RAG) framework in reducing the risk of information hallucination is clearly visualized when users simulate a question-and-answer session with the system. Rather than composing fictitious answers from its internal parameters, the IndoBERT model responds to specific user questions, such as graduation-registration procedures or merit-scholarship qualifications, by extracting factual text spans directly from the literature stored in the knowledge database. This system transparency is further reinforced by technical indicators below the answer text, such as the extraction confidence percentage and inference-latency timestamp, indicating that within the evaluated scenarios, the chatbot is highly capable of delivering database-grounded responses anchored strictly to the provided knowledge database.

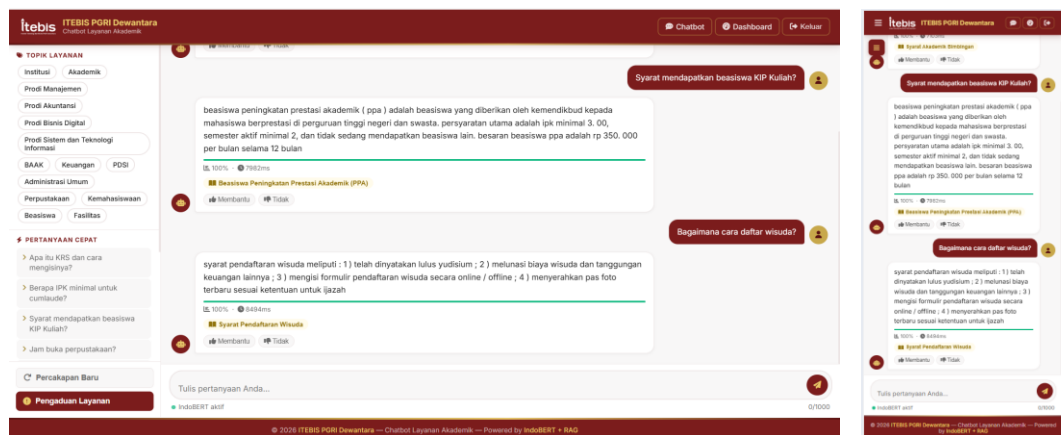


Figure 8. User Interface desktop and mobile mode

Functional Testing

Functional testing was carried out to ensure that the entire interaction flow of the prototype, from interface access and question submission to answer display, runs according to the design. Table 1 summarizes the test scenarios and their results. Referring to the test results in Table 1, all five defined operational scenarios executed successfully and met the design expectations. In the basic interaction tests (Scenarios 1, 2, and 5), the system proved capable of extracting specific regulations (such as the cumlaude honors requirement) directly from the academic database, providing adaptive responses to helper phrases, and maintaining stable, responsive layout when accessed via mobile devices. Additionally, Scenario 3 demonstrated the RAG framework's reliability in preventing misinformation. For questions beyond the campus knowledge scope, the system consistently avoided hallucination and returned a standardized default response.

The test results show that all five scenarios ran as expected. Scenario 4 in Table 1 shows that the quality of retrieval in categories with a limited number of passages (for example, finance) is more sensitive to variations in question wording compared to categories with more data (institution and academic), so adding passage variation per category is an effective improvement step at the prototype stage. This finding aligns with Figure 2, where finance and student affairs have the fewest passages.

Table 1. Chatbot Functional Testing

Testing Scenario	Expected Outcome	Empirical Observation	Status
A user inquires about the minimum GPA required to achieve cum laude honors.	The virtual assistant retrieves the specific GPA criteria directly from the core knowledge base.	The system accurately outputs the exact passage associated with the academic category.	Pass
The user interacts using predefined prompt suggestions or help commands.	The system generates a relevant response aligned with the selected topic.	The delivered text perfectly reflects the passage linked to the chosen prompt category.	Pass
A query is submitted regarding a topic completely unrelated to the campus database (e.g., asking about retail product prices).	The bot resorts to a standard fallback mechanism rather than hallucinating a fabricated answer.	A default message stating that the requested information cannot be found is displayed to the user.	Pass
The user requests instructions for tuition fee payments, a topic known to have minimal source passages available.	The system is expected to provide the correct payment guidelines extracted from the database.	Initially, the bot extracted a slightly mismatched text. However, after expanding the dataset with passage variations, the semantic retrieval functioned perfectly.	Pass (following adjustment)
Evaluating the graphical interface accessibility across different smartphone screens.	The chatbot environment remains highly responsive and fully operational on mobile browsers.	Layout elements dynamically adapt to smaller viewport dimensions while preserving all interactive features and readability.	Pass

CONCLUSION

Prototype academic service chatbot based on IndoBERT and RAG was successfully built and run locally. The integration between the language model (backend FastAPI) and the web interface (Laravel) worked well through a RESTful API. Crucially, within the tested scenarios, the system demonstrated the ability to provide database-grounded responses. However, it must be acknowledged that this performance claim may

not inherently extend beyond the restricted scope of the current functional evaluations. Technical evaluation of the model yielded an Exact Match of 71.32% and an F1-Score of 86.63% on the test set, which is considered adequate for the prototype stage. Furthermore, functional testing (black-box testing) confirmed that every scenario, from interface access to the retrieval of factual answers, operated according to the design. A primary contribution of this study lies in documenting the end-to-end design and construction of a prototype that integrates a fine-tuned NLP model into a fully testable web interface, a practical implementation rarely detailed in existing IndoBERT studies, which typically conclude at the model evaluation phase. A further technical contribution provided by this work is the deployment of the reader model as an INT8 quantified ONNX architecture, integrated alongside a Redis cache layer. Concurrently, these strategic additions are designed to suppress inference delays and alleviate database retrieval burdens at the prototype stage. Because the prototype still runs in a local environment and has not been publicly deployed, further research is recommended to carry out deployment (deployment) to a production server and more comprehensive software-quality testing, for example using the ISO/IEC 25010 standard, which has been widely adopted for evaluating AI-based systems. Enriching the variety of passages in categories with limited data (finance, student affairs) is recommended to improve the quality of retrieval identified during functional testing. Developing the architecture toward a generative LLM model is also recommended to improve the chatbot's language flexibility, supported by evaluation using dedicated RAG metrics such as RAGAS.

ACKNOWLEDGMENT

This research was supported by the Ministry of Higher Education, Science, and Technology (Kemendikisaintek) through “Penelitian Dosen Pemula” Scheme. The authors would like to thank ITEBIS PGRI Dewantara Jombang for the facilities and support provided throughout the research process.

REFERENCES

- [1] K. F. Hew, W. Huang, J. Du, and C. Jia, “Using chatbots to support student goal setting and social presence in fully online activities: learner engagement and perceptions,” *J Comput High Educ*, vol. 35, no. 1, pp. 40–68, Apr. 2023, doi: 10.1007/s12528-022-09338-x.
- [2] M. Kremantzis, A. Chondrogianni, and A. Essien, “Evaluating the impact of AI chatbots on student support and engagement in UK higher education,” *Journal of Further and Higher Education*, pp. 1–23, Jul. 2026, doi: 10.1080/0309877X.2026.2693084.
- [3] I. Engeness, M. Nohr, and T. Fosslund, “Investigating AI Chatbots’ Role in Online Learning and Digital Agency Development,” *Education Sciences*, vol. 15, no. 6, p. 674, May 2025, doi: 10.3390/educsci15060674.
- [4] G. Caldaroni, S. Jaf, and K. McGarry, “A Literature Survey of Recent Advances in Chatbots,” *Information*, vol. 13, no. 1, p. 41, Jan. 2022, doi: 10.3390/info13010041.
- [5] N. S. Amarnath and R. Nagarajan, “An Intelligent Retrieval Augmented Generation Chatbot for Contextually-Aware Conversations to Guide High School Students,” in *2024 4th International Conference on Sustainable Expert Systems (ICESES)*, Kaski, Nepal: IEEE, Oct. 2024, pp. 1393–1398. doi: 10.1109/ICESES63445.2024.10762977.
- [6] T. I. Ramadhan, A. Supriatman, and T. R. Kurniawan, “Evaluasi dan Implementasi Indobert Question Answering (QA) pada Domain Spesifik Menggunakan Mean Reciprocal Rank,” *Jurnal Algoritma*, vol. 21, no. 1, pp. 180–188, May 2024, doi: 10.33364/algoritma/v.21-1.1542.
- [7] Anugerah Simanjuntak et al., “Research and Analysis of IndoBERT Hyperparameter Tuning in Fake News Detection,” *Jurnal Nasional Teknik Elektro dan Teknologi Informasi*, vol. 13, no. 1, pp. 60–67, Feb. 2024, doi: 10.22146/jnteti.v13i1.8532.
- [8] R. I. Perwira, V. A. Permadi, D. I. Purnamasari, and R. P. Agusdin, “Domain-Specific Fine-Tuning of IndoBERT for Aspect-Based Sentiment Analysis in Indonesian Travel User-Generated Content,” *J. Inf. Syst. Eng. Bus. Intell.*, vol. 11, no. 1, pp. 30–40, Mar. 2025, doi: 10.20473/jisebi.11.1.30-40.
- [9] B. V. Kartika, M. J. Alfredo, and G. P. Kusuma, “Fine-Tuned IndoBERT Based Model and Data Augmentation for Indonesian Language Paraphrase Identification,” *RIA*, vol. 37, no. 3, pp. 733–743, Jun. 2023, doi: 10.18280/ria.370322.
- [10] R. Santosa, A. B. Nusantara, and S. Imron, “Comparative Analysis of SVM and IndoBERT for Intent Classification in Indonesian Overtime Chatbots,” *JSCE*, vol. 6, no. 3, pp. 258–270, Aug. 2025, doi: 10.61628/jsce.v6i3.2058.
- [11] D. Suhartono, M. R. N. Majiid, and R. Fredyan, “Towards automatic question generation using pre-trained model in academic field for Bahasa Indonesia,” *Educ Inf Technol*, vol. 29, no. 16, pp. 21295–21330, Nov. 2024, doi: 10.1007/s10639-024-12717-9.
- [12] T. Zheng, S. Shen, and C. Zeng, “A Retrieval-Augmented Generation Method for Question Answering on Airworthiness Regulations,” *Electronics*, vol. 14, no. 16, p. 3314, Aug. 2025, doi: 10.3390/electronics14163314.
- [13] E. Karakurt and A. Akbulut, “Retrieval-Augmented Generation (RAG) and Large Language Models (LLMs) for Enterprise Knowledge Management and Document Automation: A Systematic Literature Review,” *Applied Sciences*, vol. 16, no. 1, p. 368, Dec. 2025, doi: 10.3390/app16010368.
- [14] K. Olawore, M. McTear, and Y. Bi, “Development and Evaluation of a University Chatbot Using Deep Learning: A RAG-Based Approach,” in *Chatbots and Human-Centered AI*, A. Følstad et al., Eds. Cham: Springer Nature Switzerland, 2025, pp. 96–111, doi: 10.1007/978-3-031-88045-2_7.
- [15] L. Danuarta, V. C. Mawardi, and V. Lee, “Retrieval-Augmented Generation (RAG) Large Language Model For Educational Chatbot,” in *2024 Ninth International Conference on Informatics and Computing (ICIC)*, Medan, Indonesia, 2024, pp. 1–6, doi: 10.1109/ICIC64337.2024.10957676.
- [16] N. S. Amarnath and R. Nagarajan, “An Intelligent Retrieval Augmented Generation Chatbot for Contextually-Aware Conversations to Guide High School Students,” in *Proc. 2024 4th Int. Conf. Sustainable Expert Syst. (ICESES)*, Kaski, Nepal, 2024, pp. 1393–1398, doi: 10.1109/ICESES63445.2024.10762977.

- [17] M. Reuter et al., "Towards Reliable Retrieval in RAG Systems for Large Legal Datasets," in Proceedings of the Natural Legal Language Processing Workshop 2025, Suzhou, China: Association for Computational Linguistics, Nov. 2025, pp. 17–30, doi: 10.18653/v1/2025.nllp-1.3.
- [18] Y. A. Prasetyo, E. Utami, and A. Yaqin, "Pengaruh Komposisi Split Data Terhadap Performa Akurasi Analisis Sentimen Algoritma Naïve Bayes dan SVM," JEECOM, vol. 6, no. 2, pp. 382–390, Oct. 2024, doi: 10.33650/jeeecom.v6i2.9188.
- [19] C. A. Indranie, A. Ramadhan, J. Raharjo, R. R. N. Ikhsan, M. F. Rohadi, and D. I. Anandaputra, "Impact of Train-Test Splitting Ratios on Machine Learning Models for Daily Electricity Load Time Series Forecasting," in 2025 2nd International Conference on Information System and Information Technology (ICISIT), Yogyakarta, Indonesia, 2025, pp. 1–6, doi: 10.1109/ICISIT66233.2025.11402981.
- [20] R. Alruwaithi and S. AlHumoud, "SHIFA: SBERT-Based Healthcare Information Focused Arabic Question Answering," IEEE Access, vol. 13, pp. 86344–86355, 2025, doi: 10.1109/ACCESS.2025.3570637.
- [21] Z. Han, C. Gao, J. Liu, J. Zhang, and S. Q. Zhang, "Parameter-Efficient Fine-Tuning for Large Models: A Comprehensive Survey," Sep. 16, 2024, arXiv: arXiv:2403.14608. doi: 10.48550/arXiv.2403.14608.
- [22] W. Suwarningsih, R. A. Pramata, F. Y. Rahadika, and M. H. A. Purnomo, "RoBERTa: language modelling in building Indonesian question-answering systems," TELKOMNIKA, vol. 20, no. 6, p. 1248, Dec. 2022, doi: 10.12928/telkomnika.v20i6.24248.
- [23] S. Park, A. Kim, S. Lee, H. Lee, C. Kamyod, and C. G. Kim, "Design of REST API Client for Conversational Agent using Large Language Model with Open API System," in 2024 IEEE/ACIS 22nd International Conference on Software Engineering Research, Management and Applications (SERA), Honolulu, HI, USA: IEEE, May 2024, pp. 55–58. doi: 10.1109/SERA61261.2024.10685639.
- [24] Y. Song et al., "RestGPT: Connecting Large Language Models with Real-World RESTful APIs," Aug. 27, 2023, arXiv: arXiv:2306.06624. doi: 10.48550/arXiv.2306.06624.
- [25] A. Rahmatulloh, A. Ginanjar, I. Darmawan, N. Kurniati, and E. Haerani, "Chatbot for Diagnosis of Pregnancy Disorders using Artificial Intelligence Markup Language (AIML)," JOIV: International Journal on Informatics Visualization, vol. 7, no. 1, pp. 77–83, 2023, doi: 10.30630/joiv.7.1.1595.