TA-Lite: LLM Powered AI Virtual Academic Assistant

Aashi Sharma[1] **Harsh Gupta**[1] **Pranay Bandaru**[1] **Rakshit Rao**[1]
[1] **Indiana University Bloomington**

aassharm@iu.edu **hagupta@iu.edu** **pbandaru@iu.edu** **rarao@iu.edu**

Abstract

As students, we observed that while tools like ChatGPT offer convenience, they often discourage curiosity and critical thinking by providing direct answers without fostering deeper understanding. This reliance on AI as a shortcut undermines meaningful learning. In large academic settings, students may hesitate to seek clarification, while instructors face repetitive queries during peak academic periods. To address this gap, we present TA-Lite, a modular, instructor-aligned academic assistant powered by large language models (LLMs). TA Lite supports scalable and guided learning by combining Retrieval-Augmented Generation (RAG), reranking, and modular agents for summarization, note-making, and exam prep. Unlike generic chatbots, it delivers contextual hints and conceptual prompts rather than full answers. Instructors can configure the assistant's tone, define hint levels, and control content grounding through a dedicated portal. TA Lite also provides source attribution and analytics to ensure transparency and pedagogical alignment. Our results show that carefully constrained LLMs can enhance education while preserving academic integrity and supporting reflective, personalized learning.

1 Introduction

In today's higher education landscape, students and instructors face growing challenges in communication, engagement, and academic support. With increasingly large class sizes, diverse learning needs, and tight academic schedules, students often struggle to receive the timely and personalized guidance they require. Many hesitate to raise questions in class due to fear of embarrassment or a lack of confidence. Others feel overwhelmed during assignment and exam periods, unsure where to begin or whom to ask. On the other side, instructors and teaching assistants are often inundated with repetitive queries, unable to provide individualized help to every student-especially during peak times.

This communication bottleneck leads to deeper issues: unresolved doubts, reduced conceptual clarity, and heightened academic stress. While digital tools have improved content delivery, few solutions have effectively addressed the student–instructor interaction gap - particularly in a scalable and pedagogically grounded way.

TA-Lite (Teaching Assistant Lite) was developed to bridge this critical gap using the power of large language models (LLMs). Unlike generic AI chatbots, TA-Lite is purpose-built for academic environments. It enables instructors to define how the assistant behaves - controlling tone, level of detail, hinting strategies, and, most importantly, the source of truth. By using retrieval-augmented generation (RAG), TA-Lite grounds its responses in instructor-provided materials such as lecture slides, transcripts, and curated readings. This ensures that all responses stay aligned with course-specific pedagogy and content.

TA-Lite is not intended to replace human teaching but to enhance it. It offers students a judgment-free space to explore doubts, revisit course content, and engage with material at their own pace. At the same time, it reduces redundant interactions for faculty, allowing them to focus on higher-order teaching tasks and provide targeted support where it's needed most.

This report explores the motivation, design, and implementation of TA-Lite, and demonstrates how responsible use of LLMs can meaningfully improve learning experiences-making academic support more accessible, context-aware, and scalable across diverse classroom settings.

1.1 Contribution

This work introduces TA-Lite, a dual-portal AI assistant designed to enhance academic experience for both students and instructors through curriculum-aligned, transparent, and ethical AI integration. Our key contributions are summarized and expanded below:

1.1.1 Dual-Portal Architecture for Student and Instructor Collaboration

We present a novel system architecture featuring two distinct yet interconnected portals - one for students and one for instructors. The student portal offers 24/7 academic support by answering questions, generating summaries, and providing conceptual guidance tailored to course material. The instructor portal allows educators to configure the assistant's tone, depth of response, allowed data sources, and degree of hinting, thereby ensuring that AI support aligns with pedagogical goals and ethical boundaries. This dual-access framework ensures transparency, trust, and adaptability in AI-assisted learning.

1.1.2 Development of Modular Academic Agents

TA-Lite is composed of a suite of modular agents, each designed to fulfill a specific academic support function. These include:

- Summarization Agent for condensing lecture slides and transcripts into student-friendly formats.
- Exam Prep Agent for generating practice questions, flashcards, and key concept reviews.
- Evaluation Agent for helping instructors analyze student queries, identify learning gaps, and refine content delivery.
- Live Q&A Agent for interactive, context-aware question-answering rooted in the instructor's materials.

These agents work independently or together, depending on the task, making the system extensible and adaptable across disciplines and classroom formats.

1.1.3 Enhanced Retrieval and Prompt Engineering for Quality Assurance

To ensure answer quality, TA-Lite employs a hybrid retrieval pipeline that leverages both semantic search and keyword-based filtering across course artifacts (slides, transcripts, notes). Retrieved documents are reranked based on contextual relevance before being passed into customized prompts. These prompts are dynamically structured using prompt engineering strategies-such as chain-of-thought reasoning and instructor-defined scaffolding-to promote more accurate, pedagogically appropriate responses. This approach improves both the precision and educational value of AI-generated content.

1.1.4 Ethical and Transparent Design with Instructor Oversight

Recognizing the sensitivity of deploying AI in educational settings, we place ethical considerations at the core of our design. TA-Lite ensures:

- Transparency: All AI-generated responses clearly indicate when and where source material was used.
- Instructor Control: Teachers can configure the assistant's behavior to restrict direct answers, enforce hint-only mode, or limit the assistant to specific content sections.
- Auditability: All student queries and assistant responses are logged (with consent) for review, allowing educators to monitor engagement and identify misconceptions.

These features collectively promote academic integrity, reduce over-reliance on AI, and foster a more responsible use of educational technology.

2 Related Work

The rise of large language models (LLMs) such as ChatGPT, Bard, and Claude has significantly changed how students seek academic assistance. These tools are frequently used for quick

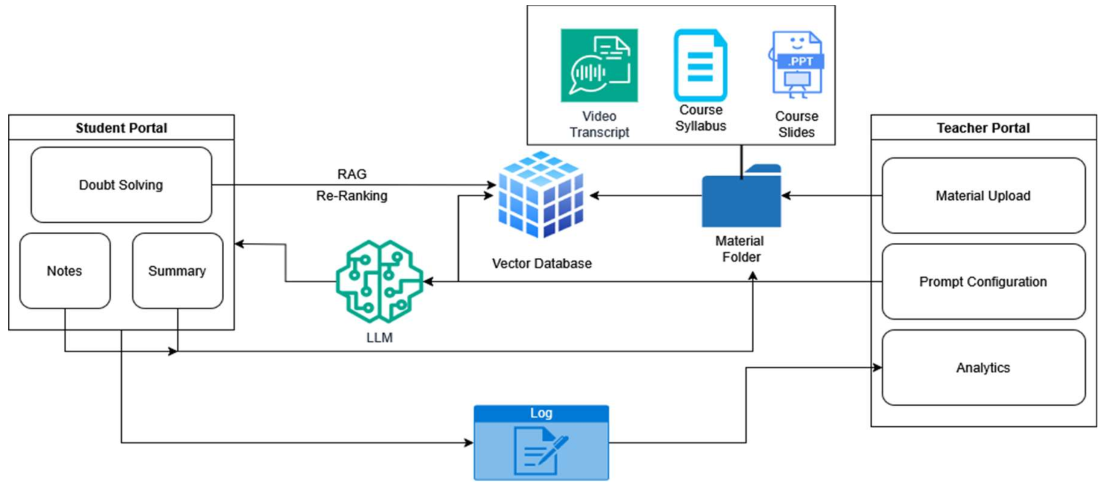


Figure 1: System Architecture

explanations, summarization, and conceptual guidance. Several educational institutions have explored embedding LLMs into Learning Management Systems (LMS) to automate routine support and enhance student engagement.

Coginiti.ai (2023), developed at the University of Sydney, is an example of an LLM-powered teaching assistant that supports course-specific tutoring and has shown promise in automating reflective questioning strategies. However, most general-purpose tools still lack fine-grained instructor control or document-grounded responses.

Kumar et al. (2022) introduced TutOR, a CS tutoring agent that provided hints grounded in instructor-approved materials. Their system demonstrated improvements in learning outcomes through retrieval-guided, non-directive feedback. Zhang et al. (2023) proposed CourseGPT, an LLM-based assistant trained on course-specific notes and syllabi. They emphasized tone customization, scaffolded prompting, and demonstrated that domain grounding significantly reduced hallucinations.

Shi et al. (2022) applied RAG in **EduChat**, retrieving lecture and textbook content to support factual, context-sensitive responses. Gao et al. (2023) further refined this approach using hierarchical chunking and reranking, improving the pedagogical alignment of outputs. Lewis et al. (2020) proposed the original RAG framework combining dense retrieval with generation, laying the foundation for most document-aware assistants.

Research in intelligent tutoring systems (ITS) supports strategies that guide learners rather than simply giving answers. VanLehn (2006) categorized tutoring strategies into step-based, hint-driven, and error remediation models. Roll et al. (2021) showed that metacognitive prompts from AI agents enhance retention. Holstein et al. (2019) advocated for transparent and collaborative AI systems that augment rather than replace educators.

Together, these works validate TA Lite's focus on hint-based interaction, retrieval-aware grounding, and instructor-aligned customization.

3 Data & Method

3.1 System Overview

As shown in Fig. 1, TA-Lite is a modular educational assistant designed to bridge the gap between student learning needs and instructor support. It consists of two interfaces: a Teacher Portal for uploading course materials, configuring how the system behaves and receiving analytics of what topic and questions are being asked, and a Student Portal for querying, summarizing, and note-making. Each function is powered by a dedicated agent that either retrieves information from course documents or works directly with full-context input. The system integrates

(a) (b)

Figure 2: Difference in number of chunks after cleaning the data.

Figure 2(a) shows the number of chunks being generated before cleaning the data; Figure 2(b) shows the number of chunks after cleaning the data. Roughly 24% reduction of chunk size was observed,

Retrieval-Augmented Generation (RAG), prompt templating, and document analytics to ensure pedagogically aligned, personalized assistance.

3.2 Data Sources

To ensure that TA-Lite provides relevant, accurate, and pedagogically aligned responses, its knowledge base is built exclusively from instructor-supplied academic materials. This guarantees that the assistant operates within the bounds of course-specific content, avoiding hallucinated or off-topic outputs. The data sources currently supported in TA-Lite include:

- Lecture slides (PDF format): These typically contain high-density visual summaries of concepts, diagrams, and key points emphasized during lectures.
- Class transcripts: These are full-text representations of what was spoken during class sessions, usually derived from recorded lectures or live captioning.
- Supplementary readings: These may include assigned academic articles, textbook excerpts, or transcripts of external multimedia resources (e.g., YouTube lectures, podcasts).

3.3 Preprocessing

The data required careful preprocessing before they can be used effectively for retrieval-augmented generation. TA-Lite employs a multi-stage preprocessing pipeline to transform raw instructional material into structured, searchable knowledge chunks.

Cleaning and Normalization: Raw educational content often contains formatting artifacts and noise that can interfere with semantic retrieval. Our preprocessing begins with a cleaning stage, which applies rule-based transformations and lightweight NLP techniques to standardize content. Key steps include:

- **Timestamp and filler removal**: Eliminates clutter like "[00:05:23]" or "uh", "you know", which are common in transcripts.
- **Speaker labels**: Removes attributions like "Prof. Smith:", which are useful in conversation logs but irrelevant for content understanding.
- **Extraneous metadata**: Strips slide numbers, watermarks, and unrelated file headers/footers that may be embedded in PDFs or OCR-generated text.

This stage ensures that only pedagogically relevant content remains for downstream processing. Using this we were able to reduce the data by ~24% while still having the most important points included (Fig 2).

3.4 RAG Module

Once cleaned, documents are split into semantically coherent chunks suitable for retrieval. TA-Lite applies different chunking strategies based on the content type. Slide decks are chunked into smaller, title-plus-body units. Since slides are typically concise and bullet-point driven, each chunk may represent a single slide or logical group of slides. Transcripts are segmented into longer sequences of tokens (e.g., 300–500 tokens per chunk) using a sliding window technique with overlap. This helps preserve context across sentences and maintains continuity of thought.

To improve the relevance and quality of retrieved content, we used a hybrid retrieval approach that considered both lecture transcripts and presentation slides. Slides tend to contain concise, well-structured summaries of key concepts, making them ideal for retrieving high-level definitions or topic outlines. However, they often lack deeper context or elaboration. In contrast, transcripts contain richer, more detailed explanations, including examples, clarifications, and spontaneous insights from live lectures. By retrieving chunks from both sources, we ensured that students received responses that were both conceptually precise and contextually rich.

Despite this dual-source strategy, initial retrieval often returned semantically weak or loosely related chunks due to noise in the transcript or generic phrases in slides. To address this, we incorporated a cross-encoder-based reranking step using the BAAI/bge-reranker-base model. This allowed us to re-order retrieved chunks based on their true semantic relevance to the query, improving the final answer quality. Reranking was especially beneficial in disambiguating vague questions and reducing the risk of irrelevant or overly verbose responses, making the overall system more robust and pedagogically aligned.

All embedded documents are stored in a FAISS vector index. For doubt resolution, we use an RAG-style retriever that performs:

- Semantic search (top-5 candidates from each document type).
- Cross-encoder-based reranking using BAAI/bge-reranker-base to prioritize semantically closer chunks.
- Hybrid merging of top-3 transcript chunks and top-3 slide chunks for balanced response quality.

For summarization and note generation, we bypass retrieval and instead use the full document (or a transcript section) directly as input context.

3.5 Prompting Strategy and Agent Design

Teachers can configure the base prompt template used by the system. This includes:

- Hint level (1–3): Controls how much guidance is given.
- Prompt style: Allows teachers to emphasize reasoning, conceptual framing, or Socratic questioning.

Prompts are dynamically assembled with formatting cues (e.g., Markdown, bold instructions) to optimize LLM compliance. Different tasks are handled by modular agents:

- Teacher Preference Agent: Adjusts prompt behavior.
- Summarizer Agent: Condenses transcripts into summaries.
- Note Agent: Generates structured notes from course content.
- Evaluation Agent: Monitors and logs query trends.
- Exam Prep Agent: Surfaces topic-relevant material near exam dates.

3.6 Implementation Details

TA Lite is implemented using a modular architecture built on top of LangChain. The core language model is Mistral (7B), served locally via Ollama, chosen for its strong reasoning capabilities with manageable resource usage. Embeddings are generated using all-MiniLM-L6-v2 from HuggingFace, selected for its balance of speed and semantic accuracy. For reranking retrieved documents, we use BAAI/bge-reranker-base, a cross-encoder model known for its high performance on sentence relevance tasks. Documents are chunked using LangChain’s RecursiveCharacterTextSplitter, with chunk size and overlap customized for each content type. Metadata is stored with each chunk to facilitate filtered retrieval and result attribution. The final vector index is stored locally using FAISS, and model responses are delivered through LangChain's ChatOllama interface. All configuration data (e.g., prompts, hint levels, exam dates) is stored in structured JSON files defined via a Streamlit-based teacher portal. The student interface also uses Streamlit, designed for querying, note generation, and summarization.

The system is entirely self-contained, allowing for local deployment without reliance on third-party APIs or cloud infrastructure, which supports transparency and scalability in academic settings.

4 Results

We evaluated TA Lite through self-testing and informal peer feedback from students enrolled in graduate-level computer science courses. The evaluation focused on the usability, effectiveness, and perceived helpfulness of the system's three primary features: doubt resolution, summarization, and note generation. While formal quantitative metrics were not used due to time constraints and the open-ended nature of outputs, we collected subjective impressions through focused hands-on trials and direct feedback conversations.

4.1 Doubt Solving with Guided Hints

Both team members and peer reviewers reported that the hint-based responses were helpful and encouraged them to think more critically about the question rather than simply receiving the answer. The layered prompting system (hint levels) was seen as particularly effective in guiding students without revealing final answers. Participants appreciated the hybrid retrieval strategy, which pulled concise facts from slides and deeper context from transcripts.

4.2 Summarization and Note Generation

Testers found the Summarizer Agent useful for condensing lecture transcripts into digestible summaries. These were especially appreciated by students who had missed class sessions or were reviewing before exams. While summaries occasionally skipped illustrative examples, they captured the main points effectively.

The Note Agent produced well-structured outlines derived from course material, which users found helpful for revision and study sheet creation. These outputs were seen as time-saving and useful for organizing content at a glance.

4.3 Transparency and Source Attribution

One of the most positively received features was the inclusion of source metadata. Users could see exactly which document and chunk the response was derived from. This transparency increased user trust and allowed for easy verification when revisiting class materials.

4.4 Instructor Controls & Prompt Customization

The teacher configuration panel was internally tested to verify that different settings—such as hint level, response tone, and exam context—consistently produced distinct and context-appropriate prompt templates. These variations ensured that the generated responses aligned with the instructor's pedagogical intent, reinforcing the system's goal of providing personalized, educator-guided assistance.

Overall, feedback received described TA Lite as "reliable," "class-aware," and "good addition along with ChatGPT," underscoring its value as a targeted academic support tool.

5 Discussion & Conclusion

TA Lite demonstrates how LLMs can be tailored for educational use by combining course-specific retrieval, instructor-guided prompting, and modular agents. The system promotes guided learning by offering hints instead of direct answers, encouraging critical thinking and deeper engagement.

5.1 Evaluation Reflections

Evaluation emerged as one of the most challenging aspects of the project. Since the assistant generates open-ended, non-deterministic responses, traditional accuracy metrics were insufficient. We instead relied on manual validation and peer feedback to assess output quality. Although we would like to conduct formal testing on a larger set of testers, but with the small group of testers with domain expertise provided input on whether the hints were helpful, the summaries accurate, and the tone appropriate for learning.

Some key observations from users included:

- Doubt Solver: Seen as useful and encouraging of thought. "I liked how it didn't just answer everything but gave a nudge in the right direction."

- Summarizer: Generally accurate and concise, with suggestions for supporting different summary lengths. "The summaries are good for getting the gist of long transcripts."

- Note Maker: Helpful for study, but "could include more points" for longer documents.
- Transparency: Users appreciated seeing the source of retrieved content and used it to verify answer credibility. One student noted, "A few chunks seemed unnecessary, but the final answer was still good."

These insights helped confirm that TA Lite supports its intended goal: to assist, not replace, the learning process.

5.2 Challenges and Limitations

While TA-Lite demonstrates strong performance within the scope of a specific classroom setting, several limitations currently restrict its broader applicability and robustness.

5.2.1 Course-Specific Design

The system is trained and evaluated only on one course, limiting its ability to generalize across subjects with different formats or instructional styles. Adapting it to other domains will require additional data and tuning.

5.2.2 Gaps in Technical and Creative Support

TA-Lite cannot assist with coding tasks or creative assignments like essays or design work. Such tasks require precise syntax or subjective reasoning, which the current system does not handle well.

5.2.3 Verbosity Issues

Responses can be overly long or dense, especially for straightforward questions. A mechanism to ensure concise and focused outputs is still needed.

5.2.4 Lack of Safety Measures

There are currently no safeguards to prevent misuse, such as students prompting uncensored or off-topic content. Content filtering and instruction enforcement are essential next steps.

Addressing these challenges will be crucial for making TA-Lite a scalable, safe, and truly general-purpose teaching assistant for academic settings.

5.3 Conclusion

By allowing professors to guide how LLMs respond and grounding the system in curated materials, TA Lite delivers a thoughtful approach to AI in education. Through its dual-portal design, retrieval-enhanced responses, and agent-based architecture, the system bridges a critical gap between scale and personalization in academic support.

Future work will focus on improving evaluation methods, expanding subject coverage, and enhancing output adaptability (e.g., summary length, note depth). With continued refinement, TA Lite can evolve into a reliable, extensible AI partner for education.

6 Ethical Consideration

Throughout the development of TA Lite, our team strictly adhered to recognized ethical standards to ensure the system was designed, deployed, and utilized responsibly within academic environments. All users interact with the assistant under the same configuration, ensuring fairness and equal learning opportunities for every student.

6.1 Data Privacy and Confidentiality

Only instructor-authorized course materials such as lecture slides and transcripts were used. No personal student data, user credentials, or sensitive information were accessed, stored, or processed at any point during development.

6.2 Academic Integrity and Anti-Misuse Safeguards

TA Lite was intentionally designed to prevent academic dishonesty. Instead of giving direct answers, it provides guided hints and learning prompts to encourage independent problem-solving and critical thinking, thereby upholding the academic code of conduct.

6.3 Transparency and Verifiability

Each AI-generated response is accompanied by a clear citation of the source document and section from which the information was derived. This ensures accountability, fosters trust, and allows users to cross-verify every output.

Acknowledgments

We thank Instructor Jisun An and AI faculty Fan Huang for their guidance and feedback throughout the development of this project. Their support helped shape our understanding of AI in educational applications. We also thank our peers for their constructive input during the evaluation phase. Additionally, we acknowledge the open-source tools and libraries including Streamlit, FAISS, OpenAI APIs, Ollama, and LangChain that supported our system implementation.

Authorship statement

All team members contributed equally and collaboratively throughout the development of TA Lite. Each member was deeply involved in every phase of the project, including ideation, system design, external content integration, prompt engineering, API implementation, evaluation, and report writing.

Aashi Sharma: Contributed to the development and integration of the External Resource Agent, participated in API implementation, prompt engineering, and collaborated on system evaluation and report preparation.

Harsh Gupta: Worked on building and integrating the RAG system including the creation of vector databases and data pre-processing and contributed to API workflows , and was actively involved in testing, refinement, and co-authoring the project documentation.

Pranay Bandaru: Participated in constructing the Teacher preference Agent and the live Q&A Agent, assisted in backend API integration and prompt configuration, and contributed to evaluation processes.

Rakshit Rao: Took part in implementing and integrating the Summarizer Agent, Note Agent, supported the web app development using streamlit, and helped with the orchestration of the analytics section of the application.

References

Gao, Rui and Xiangyang Sun. 2023. Improving Academic QA with Hierarchical Chunking and Cross-Encoder Reranking. *arXiv preprint arXiv:2408.11875*. https://arxiv.org/pdf/2408.11875

Holstein, Kenneth, Vincent Aleven, Nikol Rummel, and Bruce McLaren. 2019. Designing AI systems that augment and respect teachers. *AI Magazine*. https://onlinelibrary.wiley.com/doi/full/10.1002/aaai.12058

Kumar, Anirudh, Reva Singh, and Prashant Agrawal. 2022. TutOR: A Curriculum-Aligned, Hint-Driven CS Tutoring System. In *Proceedings of the 2022 ACM Conference on Learning at Scale*. https://dl.acm.org/doi/pdf/10.1145/3502718.3524811

Lewis, Patrick, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, et al. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. *arXiv preprint arXiv:2005.11401*. https://arxiv.org/abs/2005.11401

Roll, Ido and Bruce McLaren. 2021. Metacognitive Prompts Improve Engagement in LLM-based Dialogue Systems. *British Journal of Educational Technology*. https://bera-journals.onlinelibrary.wiley.com/doi/10.1111/bjet.13413

Shi, Jiawei, Hongyuan Tang, and Feng Zhao. 2022. EduChat: A RAG-based Academic Assistant for Factual Query Answering. *arXiv preprint arXiv:2308.02773*. https://arxiv.org/pdf/2308.02773

University of Sydney. 2023. Coginiti.ai: LLM-powered academic assistant for higher education. https://www.sydney.edu.au

VanLehn, Kurt. 2006. The behavior of tutoring systems: Does it matter? *International Journal of Artificial Intelligence in Education*, 16(3): 227–265. https://link.springer.com/article/10.1007/s40593-021-00239-1

Zhang, Tianyi, Shiqi Li, and Xiaoyu Xu. 2023. CourseGPT: Enhancing LLMs with Course-Aware Content and Instructor-Controlled Prompting. *arXiv preprint arXiv:2405.04781*. https://arxiv.org/abs/2405.04781

Mistral AI. 2023. Mistral 7B: A dense, high-performance open-weight model. https://mistral.ai/news/announcing-mistral-7b/

LangChain. 2023. LangChain: Framework for developing LLM-powered applications. https://github.com/langchain-ai/langchain

Ollama. 2023. Ollama: Run open LLMs locally. https://ollama.com/

Facebook AI Research. 2017. FAISS: A library for efficient similarity search and clustering of dense vectors. https://faiss.ai/

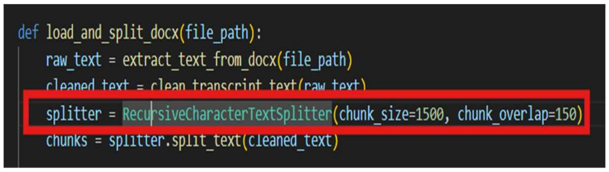
HuggingFace. 2020. Transformers: State-of-the-art Natural Language Processing for Pytorch and TensorFlow 2.0. https://huggingface.co/docs/transformers

Liu, Jingjing et al. 2023. BGE: Bootstrapped General Embeddings for Transfer Learning. https://huggingface.co/BAAI/bge-reranker-base

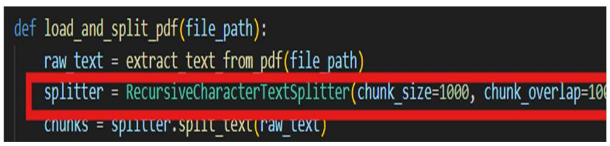
A Link for TA-Lite code and data

Codebase: https://github.iu.edu/hagupta/TA-Lite
Data:https://github.iu.edu/hagupta/TA-Lite/tree/main/teacher_data

B Document based chunking



Chunk size set for documents



Chunk size for slides(pdf)

C UI for TA-Lite

Figure 3 Shows the UI for Student Portal.
Figure 4 shows the UI for Teacher Portal.

D Configuring the LLM Prompt

Figure 5 shows the prompt generated based on Figure 4 settings.

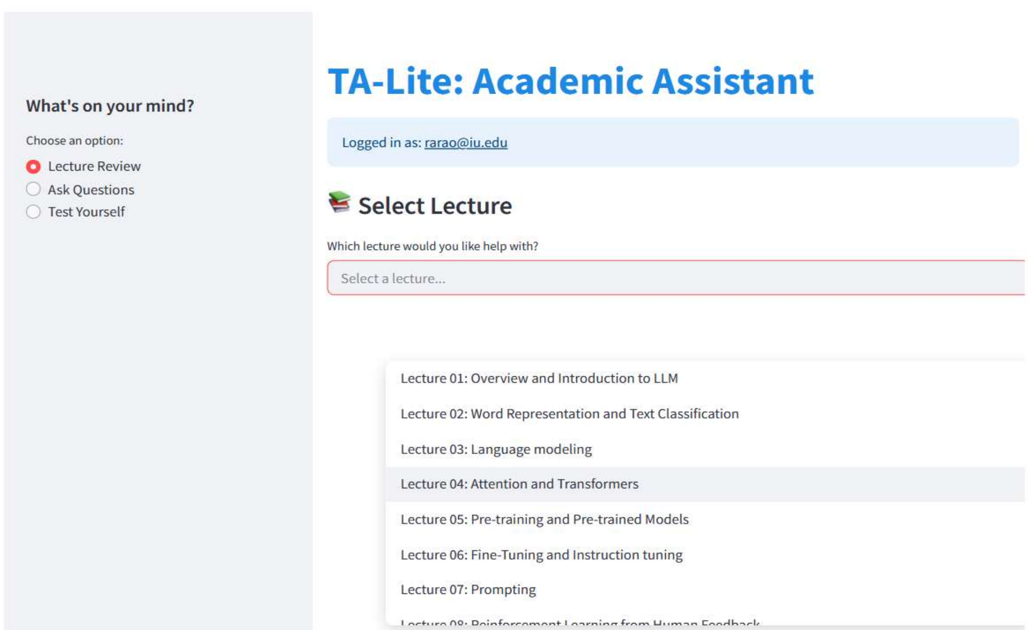


Figure 3 Student Portal

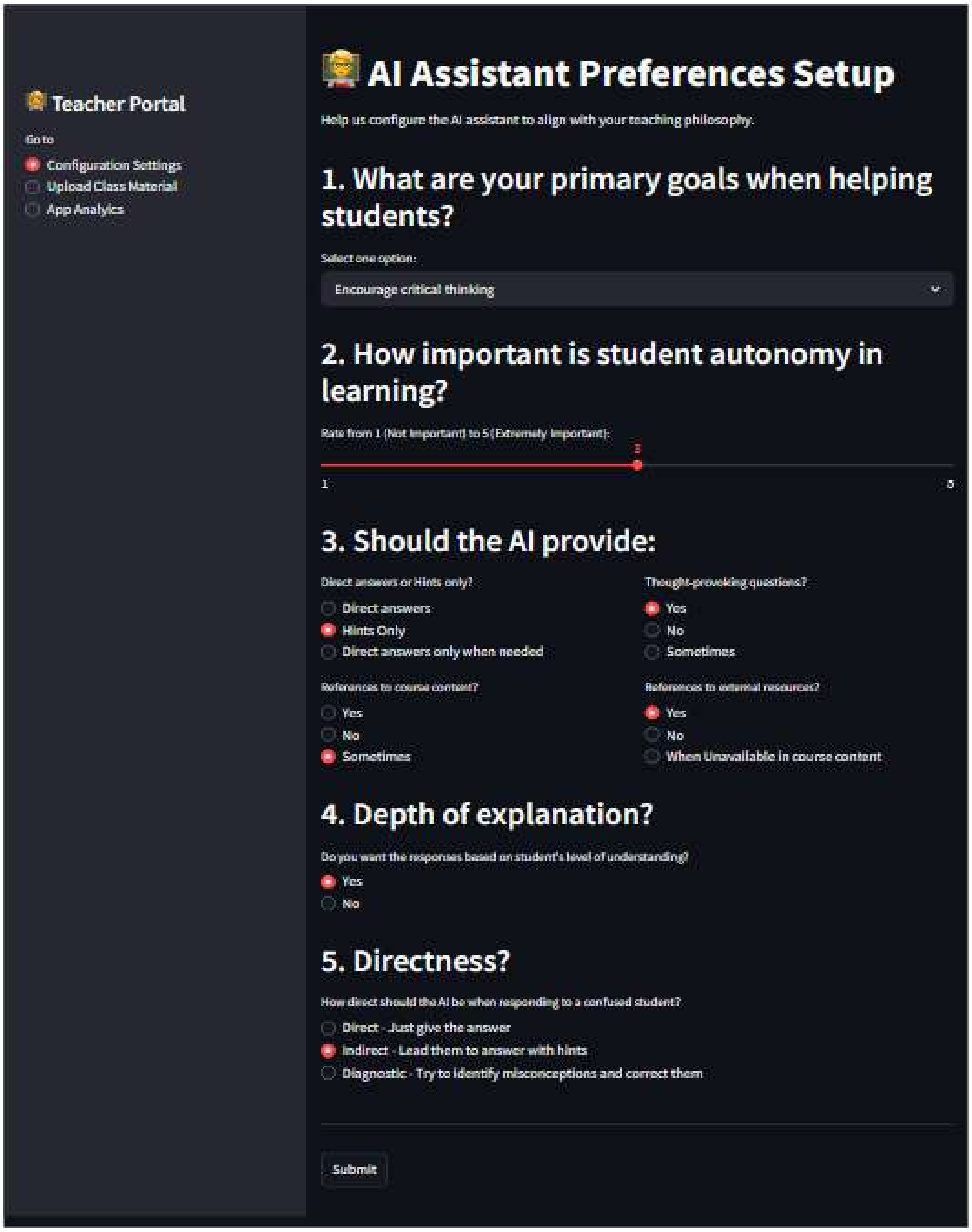


Figure 4 Teacher Portal

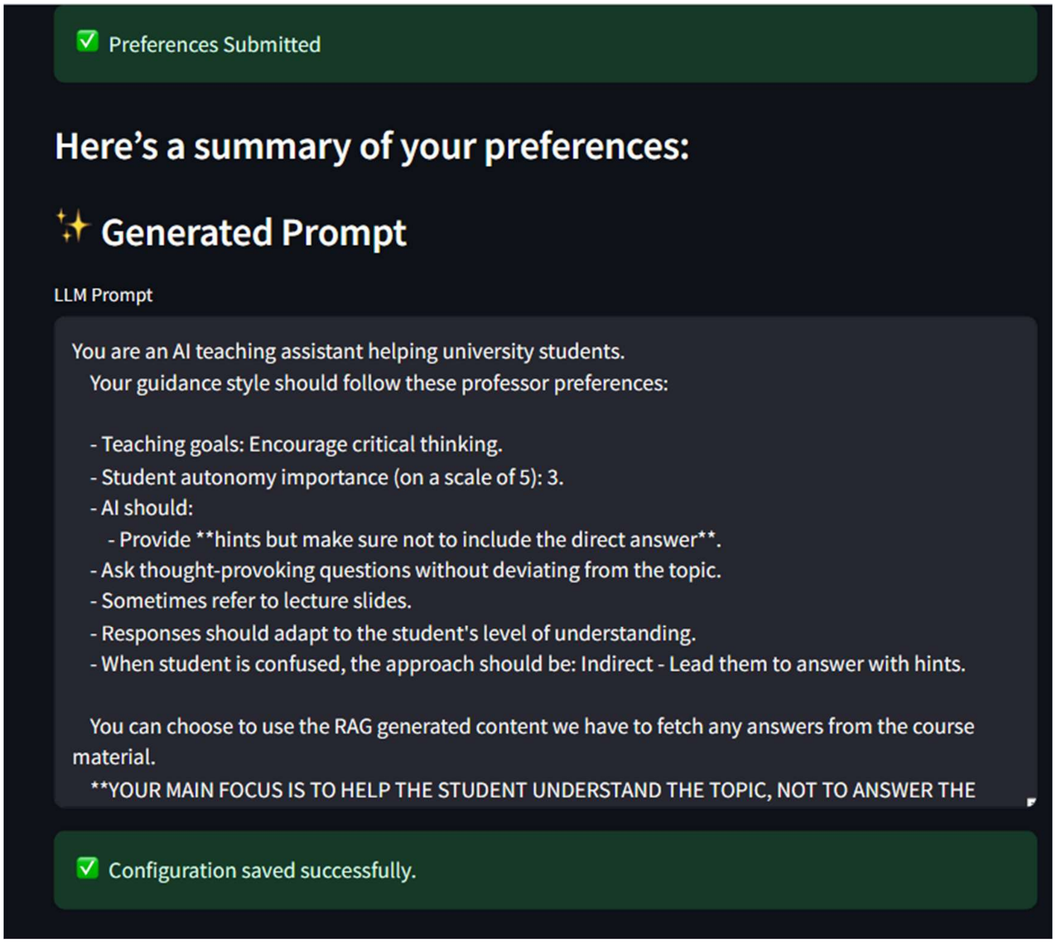


Figure 5 Prompt Generated based on Figure 4 settings

689