Academia-GPT: Smart Tutoring and Question Generation

Yassin Khudari, Jainam Shah, Nikhila Rao Lakku, Daniel DeCosmo, Sandra Ann Mathew,
Ngirimana Stephanie De Vasantha Nyiramurehe

November 27th, 2023

I. Purpose

Artificial Intelligence (AI) in education is utilized to enhance the learning and teaching experiences for both students and instructors. These technologies have helped students and educators across the world in promoting inclusivity in education. They not only have the potential to make education more accessible to diverse learners, but they also make the learning process more interactive and enjoyable.

Our project aims at mobilizing this exciting development in AI to build a learning tool for the Jupyteach platform through Large Language Models, utilizing Retrieval Augmented Generation (RAG). There are mainly two modules associated with this platform, the AI Tutor and Question Generator.

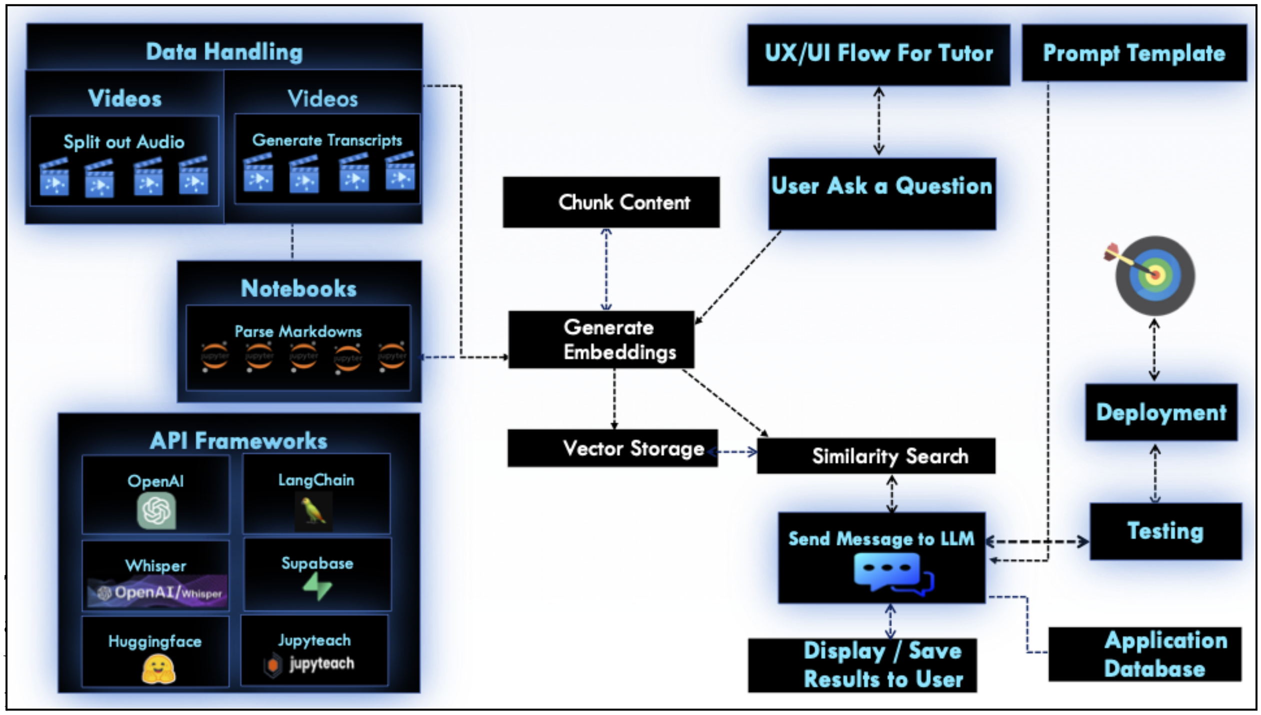
AI Tutor: The purpose of the AI Tutor is to help give students a personalized learning experience of specific topics through interactive methods. It also aims at engaging users in a dynamic and responsive learning environment in acquiring new skills and gaining knowledge.

Question Generator: The purpose of the question generator is to assist the instructors in creating questions for educational and assessment purposes.

II. High-Level Architecture

The below figure signifies the conceptual understanding of the steps involved in building the project. Key aspects include:

- Identification of the major components or modules.
- Descriptions of the interaction and dependencies between various modules.
- Assists in determining potential risks and challenges early in the project lifecycle, allowing for proactive mitigation strategies.
- Serves as a foundation for more detailed design and implementation activities.
- Supports scalability and flexibility to accommodate changes or expansion in the future.



- A user-friendly interface, which is a web application in our case, that provides students with access to lessons and tutors with generating teaching material.
- A server and an interface maintaining the requests ingested by users and a record of each user's usage data such as topics explored and learning history of the created content providing a personalized learning experience for the user.
- An adaptive learning engine that is designed based on RAG. The module takes the vectorized data and is responsible for taking user prompts, extracting key concepts, and identifying the intent behind the user's input.
- A knowledge base is usually a database that stores the vectorized data. It holds domain-specific data that the AI tutor or question generator can draw from according to the user's learning style. The knowledge base is constantly updated with new information through new content to maintain the content relevant and accurate.
- A student module that maintains a representation of the user's current knowledge skills and learning style for a personalized learning experience and recommends learning materials accordingly.
- A pedagogical module determining relevant teaching strategies and difficulty level of questions to users. This module should be designed in a highly adaptive way to cater to different dialects and informal language.
- An answer generation module that retrieves relevant information from the knowledge base and formulates a summarized and informative response. This module capability lies in its ability to generate different types of answers such as explanations, examples, and demonstrations.

Team's Preparedness: Adherence to the following guidelines will ensure the uninterrupted delivery and functioning of the project.

- Understanding Langchain, OpenAI, embedding, prompt engineering, and databases for vector storage.

Selected Model: Agile was the model selected for this project. Choosing an apt project development model holds significant implications for the success of the project. Some of the key aspects that highlight the importance of selecting an appropriate project development model are:

- It provides a structural framework and roadmap by outlining the sequence of activities, tasks, and milestones.
- Influences how resources, time, budget, and personnel are managed and allocated throughout the project.
- Helps in the assessment of risk identification and mitigation.
- Offers a degree of flexibility and adaptability by accommodating changes in crucial dynamic requirements like technology and scope.
- Impacts how closely the final model aligns with customer expectations by determining how quality assurance and testing are integrated with the development process.

The implementation of an AI Tutor and Question Generator necessitates a project development approach that is both iterative and incremental, prioritizing adaptability, collaboration, and input from customers. This method entails dividing the project into smaller, manageable segments known as iterations. Given the project's requirements, we have opted to employ the Agile model, which aligns well with the need for flexibility and continuous feedback, allowing for responsive adjustments throughout the development process.

Implementation: The implementation phase in product development signifies the stage where the actual development of the product takes place. It emphasizes on a regular demonstration and review phase that includes the delivery of functionalities and features of the project in short and frequent cycles.

III. Project Goals

The advent of Artificial Intelligence has provided an opportunity to transform education in order to unlock the potential of students as well as nurture their individual understanding capabilities. AI in education can provide various teaching approaches including collaborative learning environments, independent study, as well as several other methods.

Our endeavor was to provide them with this plethora of opportunities to empower themselves with education. To achieve this goal, our team focused on the following objectives:

- Reinforced Learning Experience: The teaching tools should adapt to a student's unique style of grasping and understanding concepts by ensuring reliable, challenging, and engaging techniques.
- Effective Knowledge Acquisition: The data retrieval should be up-to-date, encompassing information from all resources and enabling the tools to provide comprehensive answers and practice questions. It should also be able to draw connections from the knowledge base to make meaningful inferences and give insightful explanations.
- Personalized Feedback: The AI tutor should be able to take real-time feedback from students, thus providing them with error-free and concise content.

  The Question Generator must have the capability to produce questions at the desired level of difficulty and relevant to the topic specified by the instructor. Furthermore, tailored feedback and suggestions on additional resources could also be a good feature.

- Scalability and Accessibility: The system should be able to handle a high volume of users and content while also accommodating the expansion of the user base. Features such as cross-platform compatibility and accessibility should also be integrated for the students with diverse abilities to access the learning platform through various devices such as computers, tablets, and computers.

Functional Requirements

Common in Tutoring and Question Generator:

- Ability to store chat history and generated questions in a database.
- Allow user feedback from the students for the tutor and from the instructors for the question generator.
- Ability to provide references to where the information is being pulled from.
- Ability to report any potential issues to both the user and the system modeler.

AI Tutor :

- Ability to explain concepts in a specified tone.
- Ability to generate code examples for content when applicable.

Question Generator:

- Ability to generate many different types of questions such as multiple choice, fill in the blank, short response, etc.

- Ability for instructors to edit generated questions.
- Ability to create/give answers alongside the generated question for easy grading.
- Ability for instructors to set/specify question difficulty when generating questions.

Non-Functional Requirements

Common in Tutor and Question Generator:

- Keep response time and resource utilization consistent to ensure reliability as well as high performance.
- Both the AI Tutor and the Question Generator should be able to handle unexpected increases in workload.
- Adherence to data protection measures to prevent unauthorized access as well as data breaches and misuse.

Process Constraints

- Agile-Scrum Methodology: Following the Agile-Scrum methodology, which includes sprint planning, sprint reviews, and retrospectives. Sprint Duration: Sprint durations were typically 1 week for each sprint, during which specific features or tasks were completed.

Design Constraints

- Integration: Proper integration of the AI tutor and question generator into the Jupyteach LMS is necessary for allowing users and instructors to use it effectively.
- Language and Technology Stack: This project is going to be developed using Python as the primary programming language. Considering the language dependency, it is the only language that could be integrated with the OpenAI API, Langchain, and Supabase, and the final product will be deployed on JupyterHub.

Question Generator

For the question generator, the output needs to be in the format of a JSON object and therefore the Pedantic Output parser model was utilized.

IV. Project Plan

Phase	**Tasks**
Planning	● Defining project tasks and scope – Identifying target users, defining timelines and planning sprints. ● Gathering relevant data – Lecture videos and notebooks for AI Tutor and Question Generator. ● Designing knowledge representation – Choosing a working platform as well as an appropriate database storage.
Design	● Designing a RAG module and integrating it with a LLM model that works well with the data. ● Making frameworks for both the AI Tutor module and the Question Generator module.

Development	● Integrating all modules with the adaptive learning module. ● Developing a UI and interaction design. ● Establishing an appropriate interaction with the LLM by sending relevant prompts and queries. ● Developing an API Server.
Testing	● Checking if relevant chunks are extracted for both the Tutor and the Question Generator. ● Checking if the LLM responses are intune with the prompts.
Deployment	● Deploying to target platform - UI interface. ● Providing a maintenance plan.
Maintenance	● Updating the knowledge bases. ● Refining the prompts.

High- Level Steps:

- Project Planning and Gathering Requirements: The project kicked off with the team meetings and client meetings to understand the requirements provided by the JupyteachLMS team. After the requirements were finalized, a detailed project plan was laid out to accomplish the project goals.

- Data Preparation: The data that the Jupyteach team provided include the videos and Jupyter notebooks for the courses. For the data preparation, the audio from the video files were extracted and then transcribed using the Whisper tiny model to an .srt file format. The jupyter notebooks were also parsed to markdown files for the steps involved further down the road.

- Data Transformation: The first step of data transformation involves the chunking of the prepared data. Chunking is the step where large text is divided into small manageable parts so that the content is more relevant when running a similarity search. This chunked data is then embedded as vectors for easier processing of the text data.

- Vector Storage: The embeddings are stored to the vector database named Supabase. The database includes details such as the unique ID of the vectors, document ID, collection name, and the metadata.

- Retrieval Augmented Generation: This technique is implemented to improve the accuracy of the generative AI model. In this step, retrievals of relevant documents are done by the usage of system prompts designed for AI tutor and question generator. Once the relevant information has been obtained, the accurate answers are displayed.

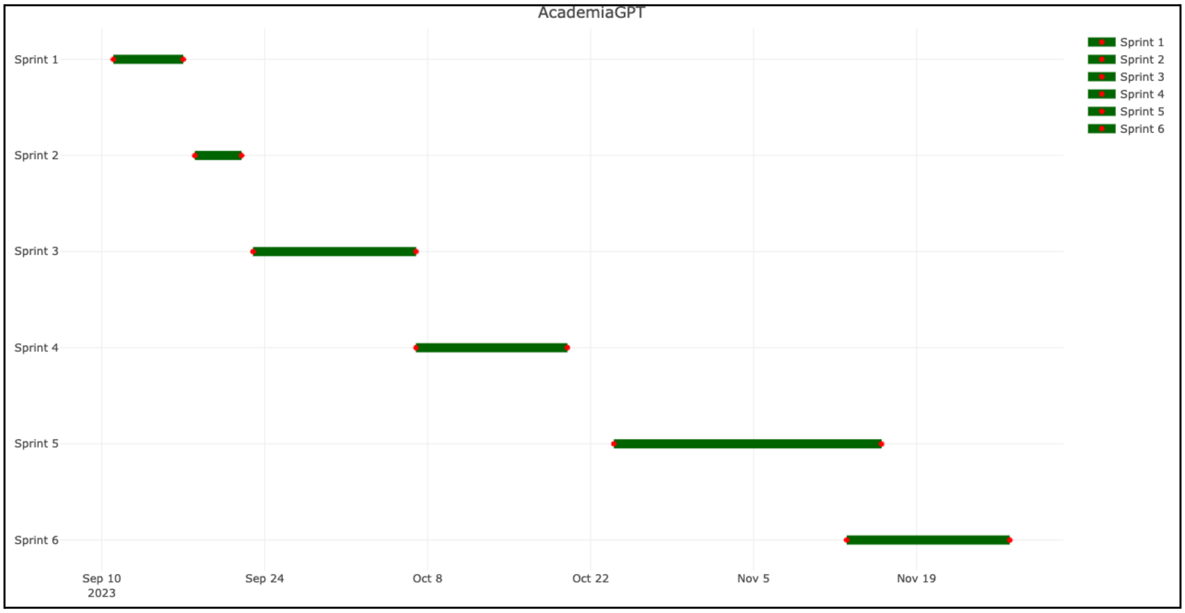
- Prompt Engineering: In this process, the main goal is to figure out what text to feed the language model to get it to take on the behavior in the way we want it to behave in terms of the generated context, style, and tone.

- Testing and Refining Prompts: Once the prompts are designed, the following step is to test them to find out the outstanding prompt. The outstanding prompt is refined so that the outputs meet the desired tone, style, and generated context.

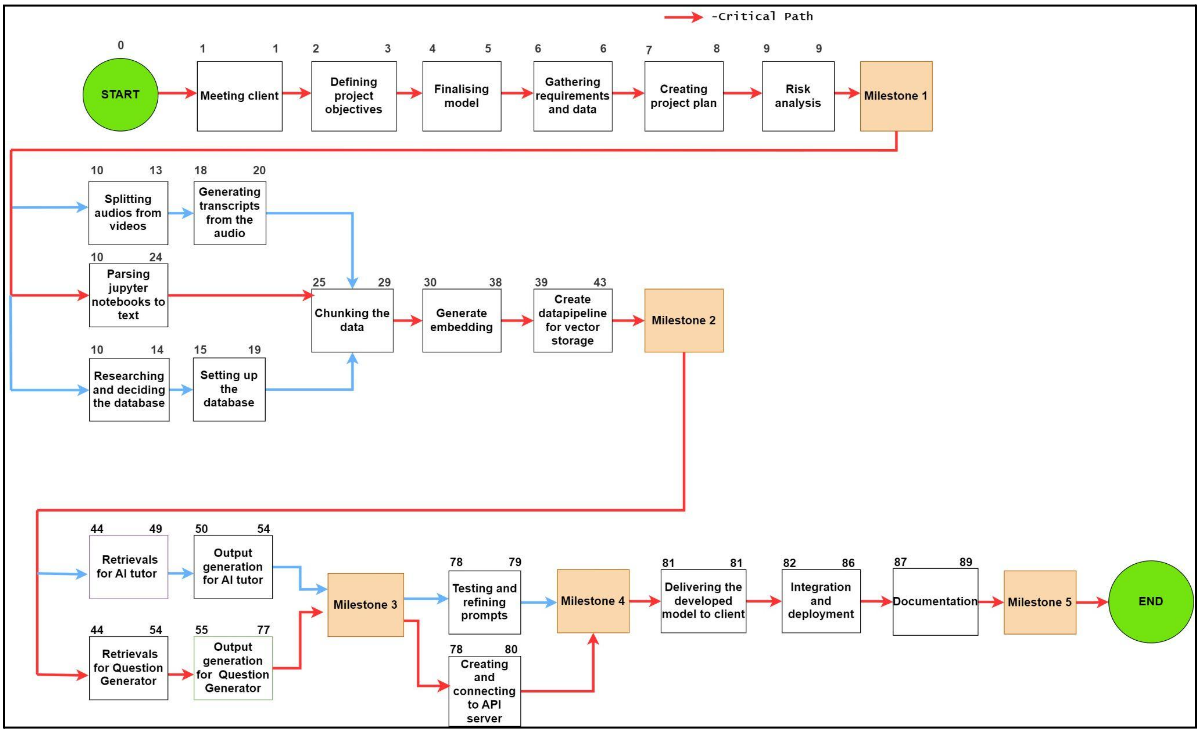
- Creating API Server: The server sets up an endpoint to interact with AI Tutor. When users ask questions, the server will send the queries to the AI Tutor and the names of the users as parameters in the form of a request. The server passes these details to a singular call function and it returns the AI Tutor's response to the user.

- Testing UI: The last step is to test the user interface. We want to make sure that the AI tutor function is able to allow the student to ask a question and it outputs relevant explanations and the question generator function is able to allow the instructor to generate test/quiz questions and relevant answers.

A. Gantt Chart



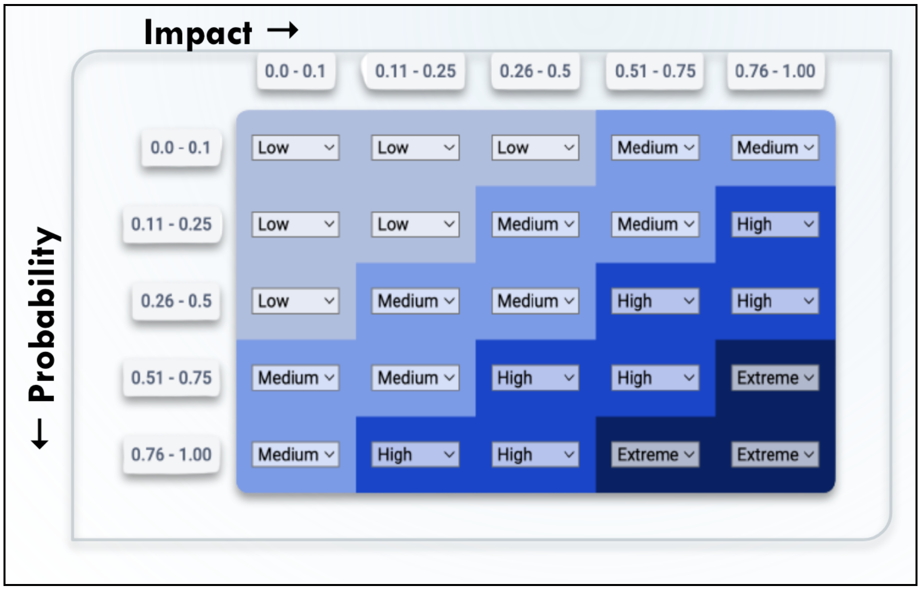
B. Critical Path


-Critical Path
0
START
1 1
Meeting client
2 3
Defining project objectives
4 5
Finalising model
6 6
Gathering requirements and data
7 8
Creating project plan
9 9
Risk analysis
Milestone 1
10 13
Splitting audios from videos
18 20
Generating transcripts from the audio
10 24
Parsing jupyter notebooks to text
25 29
Chunking the data
30 38
Generate embedding
39 43
Create datapipeline for vector storage
Milestone 2
10 14
Researching and deciding the database
15 19
Setting up the database
44 49
Retrievals for AI tutor
50 54
Output generation for AI tutor
44 54
Retrievals for Question Generator
55 77
Output generation for Question Generator
Milestone 3
78 79
Testing and refining prompts
78 80
Creating and connecting to API server
Milestone 4
81 81
Delivering the developed model to client
82 86
Integration and deployment
87 89
Documentation
Milestone 5
END

V. Risk Management

Throughout the course of our project, there are several risks that we considered and were exposed to. Some technical risks we considered were risks associated with research, integration bugs, and performance issues. Due to our project needing to be implemented with the Jupyteach online LMS, we wanted to be aware of potential integration bugs and performance issues we could run into during deployment. Some management risks that we considered were effort estimation and cost, specifically cost related to the OpenAI API that we utilized in this project. Due to the relative newness of AI technologies, we wanted to be aware of the possibility of underpredicting our effort estimation on certain tasks that could end up taking much longer than originally anticipated. One organizational risk we considered was resource acquisition. Some external risks we considered were risks associated with user trust and academic integrity concerns. Due to this project being implemented in an online LMS environment, there is always the possibility of running into situations regarding user trust and academic integrity and we wanted to stay aware of these things throughout the course of the project.

Even though we considered several risks throughout the course of this project, we did not anticipate a high probability of running into any of them. They were still risks we needed to be aware of due to the impact they would have had on the project if we did indeed run into them. With the combined efforts of all the team members as well as our amazing project mentor, we were able to avoid all of the potential risks we originally anticipated for our project.



Impact →

↓ Probability	0.0 - 0.1	0.11 - 0.25	0.26 - 0.5	0.51 - 0.75	0.76 - 1.00
0.0 - 0.1	Low	Low	Low	Medium	Medium
0.11 - 0.25	Low	Low	Medium	Medium	High
0.26 - 0.5	Low	Medium	Medium	High	High
0.51 - 0.75	Medium	Medium	High	High	Extreme
0.76 - 1.00	Medium	High	High	Extreme	Extreme

Risk	**Probability**	**Impact**	**Exposure**
Research	0.1	0.6	Medium
Integration Bugs	0.15	0.75	Medium
Performance Issues	0.05	0.6	Medium
Effort Estimation	0.1	0.8	Medium
Cost (Related to OpenAI API)	0.15	0.5	Medium
Resource Acquisition	0.05	0.8	Medium
User Trust	0.05	0.5	Low
Academic Integrity Concerns	0.1	0.4	Low

VI. Minimum Viable Product (MVP)

AcademiaGPT aims to provide automated, personalized explanations to student's questions in order to reduce confusion, get clear understanding, and improve learning outcomes. The Minimum Viable Product (MVP) will focus on core natural language processing capabilities for a single subject area. Students will be able to ask text-based questions which will be ingested, analyzed, and used to formulate helpful explanations. Explanations will emphasize written responses over other formats to establish viability.

User Stories

- As a student, I can ask conceptual questions to receive automatically generated explanations covering foundational topics in the course.
- As an instructor, I can use the question generator to design different types of questions to test students' understanding in quizzes and exams.

The initial MVP will rely primarily on text-based explanations, but it will focus on providing the information in simple and understandable language (layman's terms). Additional formats include:

1. Code examples: Displaying code snippets to illustrate concepts may boost student comprehension, especially for technical/programming material.
2. Diagrams: Visual learners often prefer diagrams depicting relationships between concepts.
3. Concept Workflow: Mapping out logic flows around decision points could aid in understanding complex procedures

Researching explanation formats will require incorporating expandable AI tutor capabilities over time.

The personality and tone expressed by the AI Tutor through language can impact perception and utility. Testing variations like formal vs informal, strict vs friendly, and serious vs conversational will shed light on our preferences. Surfacing explanations to focus groups and gathering direct feedback through usage, interviews, and surveys will guide through the iterative development process.

While starting narrow, analyzing the logs of questions asked, explanations generated, feedback received and trouble areas encountered will inform expansion to new topics. As the knowledge base grows, monitoring metrics like explanation, quality, and question similarity matches will be essential to ensuring adequate and reliable coverage. This can facilitate continually adjusting the system to handle more question complexity and variety.

VII. Final Contemplation

There were multiple difficulties that we faced when we developed our MVP due to the design constraints in the Question Generator where it was mandatory to use the Pydantic Output model for parsing the output from the LLM. This restricted the prompt engineering steps for the question generator.

To conclude, we can make our project more meaningful for the Question Generator by making it more interactive and conversational. It could also improve by increasing the difficulty of the questions that are asked by the user. For the AI tutor, there was some hallucination while retrieving outputs and generating prompts.

The next steps for this project should be creating a UI interface for the Question Generator, maintaining a robust data protection strategy, ensuring user privacy, and also refining the AI tutor based on real-world usage .

In this course, we have learned various software development methodologies that apply to different projects whether it is a startup, business development, or research. Since we picked the business development option, we got the opportunity to work with state-of-the-art tools and evolving technologies like Generative AI, Prompt Engineering, and Retrieval Augmented Generation.