REC QUESTION PAPER GENERATOR

Abstract

Today, institutions spend many days preparing well-articulated question papers for students across various subjects. The questions themselves need to be structured so that each subject's units are given equal weight. Additionally, the questions must guarantee that students receive proper educational value. The professors have to put in a lot of time and effort to formulate, check, and turn in the question papers. This application aims to streamline the process of creating these question papers by leveraging large language models, the Retrieval Augmented Generation (RAG) technique, and an intuitive interface. The content from the PDF notes is extracted using Natural Language Processing (NLP) techniques, and it is then saved in a vector database. The retrieval augmented generation (RAG) technique and the LangChain framework are used by the large language model to generate question papers based on predefined question paper patterns and the context provided by the Portable Document Format (PDF) notes.

Keywords Question Paper Generation · PDF · Natural Language Processing · Retrieval Augmented Generation · LangChain · Large Language Models · Llama-2 · Vector Database

1. INTRODUCTION

The automation of question paper generation, compared to the manual traditional methods, has a lot of advantages in today's busy world. Creating question papers is a rather tedious process for educators, as it requires much thought so that the questions are thorough and evaluate students' knowledge and skills. Through automation, all the questions come into equal consideration to effectively assess critical thinking and problem-solving capabilities.This application uses Python, Llama2, LangChain, and Streamlit to automate the generation of question papers. Llama2, developed by Meta AI, is the backbone of generating questions in this application. LangChain, a Python framework, uses these large language models to build effective applications. Streamlit, another Python web framework, is used to build an application interface through which a user can upload notes with a single click and generate question papers.

Artificial intelligence is the core of this application in technologies like natural language processing, reasoning, perception, and learning. More specifically, natural language processing is used to read the context of the uploaded notes and generate relevant questions that make sense. The context of the note is stored in a vector database in a mathematical format. This information is used by the LLM to generate questions based on this information.The process initiates with the user uploading their notes using the drag-and-drop interface. These notes are further converted into a mathematical format and stored in the vector database. The LLM, using this contextual information, generates questions that fit the specified question paper pattern, ensuring relevance in the diversity of questions.Generation of question papers, in this way, saves time and effort of educators, ensuring quality assessment. The questions drafted challenge the students and increase their engagement with the subject matter more seriously. Moreover, the application provides the facility to change the pattern of question papers with the choice of different open-source models like LLMs.

2. LITERATURE REVIEW

This application will automate the generation of question papers, using subject notes and a given pattern of question paper. The manual generation of question papers is one of the laborious jobs that takes a lot of effort from professors. In addition to framing questions that would correctly assess the understanding of topics by the students, this needs to conform to the educational objectives in addition. There are also additional issues, such as question papers are revised several times and have to be generated in different sets.Our application automates all the above tasks, and we lessen the burden on the educators and ensure high standard quality for the generated question paper. The application allows users to upload their notes through a simple drag-and-drop user interface, and with one button click, a whole question paper is generated. This removes the need for manual work and excessive editing.Large language models, like the ChatGPT-4, can be used to achieve results similar to that but are not optimized for this task. Such general models require elaborated prompts and a lot of human intervention to produce well-structured questions. In contrast, our application is designed only for the generation of question papers, making our application a much easier and user-friendly alternative.

Flexibility is one of the main application advantages. The user can easily change the question paper pattern by changing the pattern variable in the code, thus enabling its customization for specific needs.

In addition, the application supports the use of different open-source models, giving the user a choice between using a Llama2 model by Meta AI, Mixtral models by Mistral AI, and many more. This makes sure that a user is not limited to just ChatGPT-4 but is free to choose the best model that would work for his needs.Besides saving teachers some much-needed effort, our application ensures that the question papers so generated maintain quality and relevance. To ensure that professors are more focused on lecturing and reaching out to students other than administrative roles, we automate tasks like framing of questions and revision cycles.The application is also characterized by a user-friendly interface whereby users can upload their subject notes using drag-and-drop functionality. With a single click, the question paper is generated without any need for human labor and heavy editing. Our approach is efficient and contrasts with the use of giant language models such as ChatGPT-4, which can be used but needs detailed prompts and human inputs to finally come up with questions well-structured for an examination.Our application also offers flexibility; users can easily modify the pattern of the question paper just by changing the variable pattern within the code. This ability ensures that question papers meet specific educational needs and objectives. Further, the application accepts several open-source models such as Llama2 by Meta AI, Mixtral models by Mistral AI, and so forth. This freedom in our application allows the user to choose the most relevant model for the work, thus enhancing the effectiveness and relevance of the question papers generated.Thus, our application is an important advancement in automating educational processes, where a simple and efficient solution is provided in generating question papers of good quality and customized to suit the educational context of the individual.

3. SYSTEM ARCHITECTURE

3.1 Application System Architecture

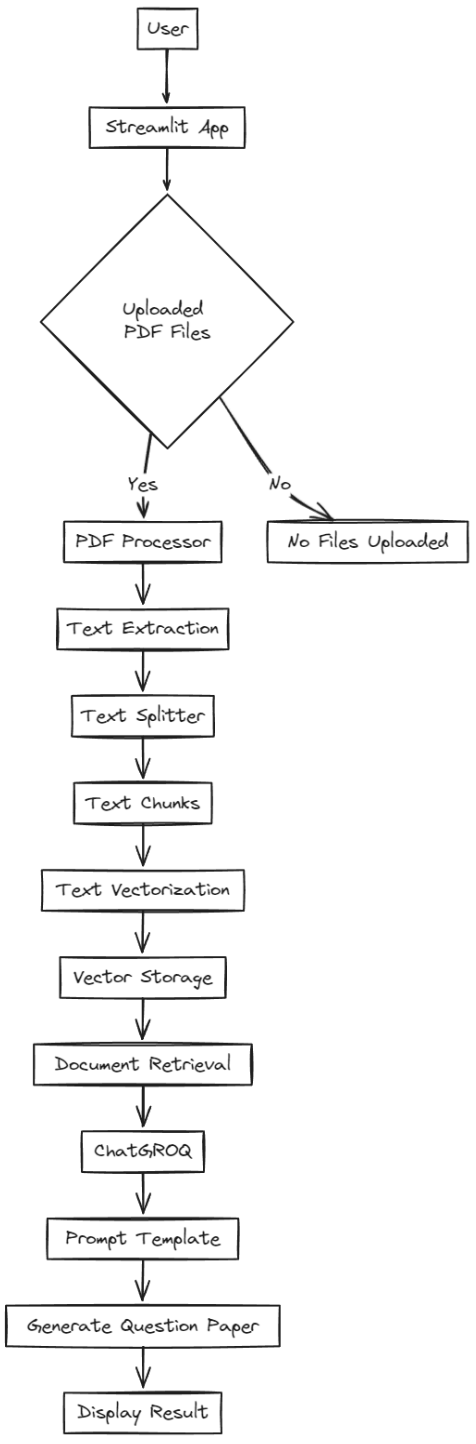


Fig 1. System Architecture

The system architecture, as seen in Fig 1, represents the process of generating question papers based on the PDF notes uploaded via a StreamLit application. Users interact with the application for file upload of PDF files containing study materials. After uploading, PDF Processor initiates the process of text extraction from each document, which will be further segmented by the Text Splitter component into manageable chunks.Further, the text chunks are taken to Text Vectorization, where they are transformed into vector representations in things like Ollama Embeddings. These vectors then go into Vector Storage, usually managed by FAISS for efficient retrieval.From here, when a user specifies the exam type, for example, CAT-1 or End Semester, through prompts on the Streamlit interface, the document retrieval module retrieves relevant text chunks based on the stored vectors. This retrieval process will be done using a retrieval chain, ensuring the system identifies and gathers all the relevant content for question paper generation.ChatGroq, through the Groq API and a prompt template specified for that exam type, uses the generated question chunks to return structured question papers.It uses the template to stipulate the format and contents that should be expected for every type of exam. The displayed question paper that has been generated shows to the user with Display Result via the Streamlit interface. The application, hence, harnesses text processing, vectorization, and retrieval techniques, integrated with an API, to make it super efficient and advanced with regard to automated generation of question papers, which supports efficiency and personalization in educational settings.

3.2 Natural Language Processing

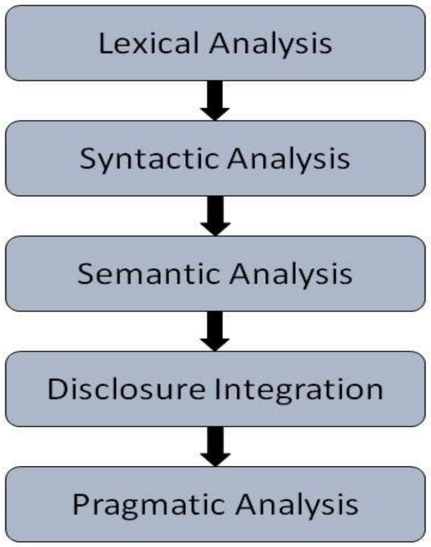


Fig 2. Steps in Natural Language Processing

Natural language processing is one of the many applications of Artificial Intelligence (AI). Using deep learning algorithms we can improve the ability of the AI to interpret and generate natural language which would be useful in this research[1]. Through Natural Language Processing(NLP) techniques, we are able to extract the contents of the notes and store it using a Vector Database. There are 5 phases in Natural Language Processing as seen in Fig 1.

3.3 Vector Database

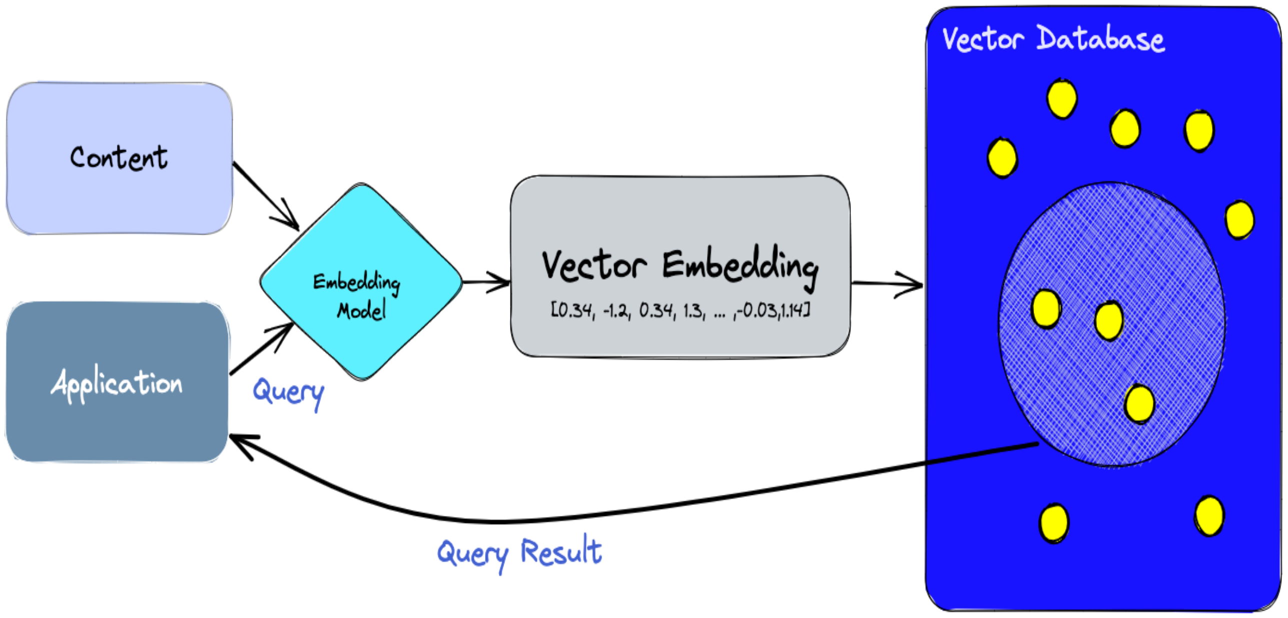


Fig 3. Role of Vector Database in Applications

A vector database indexes and stores vector embeddings for fast retrieval and similarity search.[2] One of the main advantages of vector databases is their ability to handle various data types such as text, images, audio, etc. which is essential for storing information from the notes and perform similarity search, clustering and classification. Each paragraph and sentence is transformed into a vector representation and are indexed and stored into the database.

3.4 Working of a Vector Database

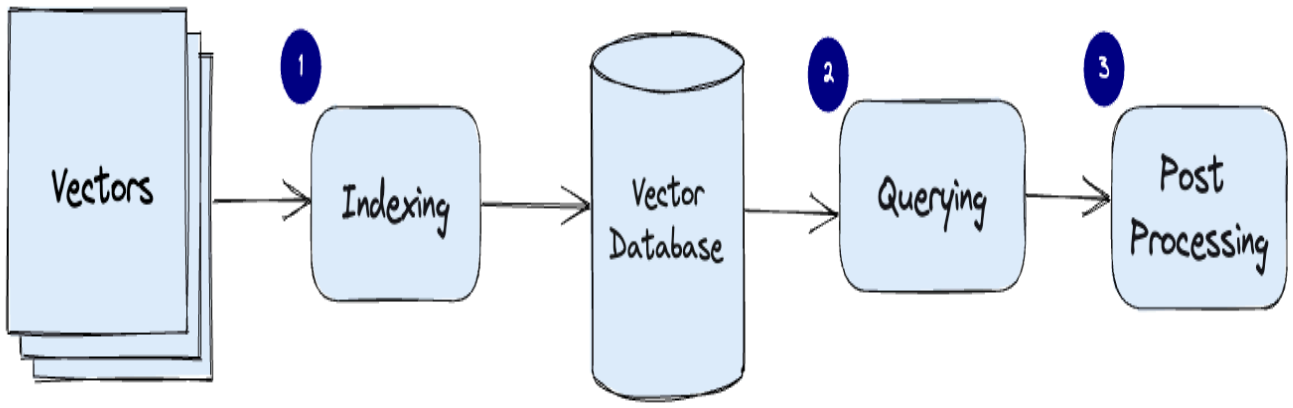


Fig 4. Working of a Vector Database

1. Indexing: The vector database indexes vectors using an algorithm such as PQ, LSH, or HNSW (more on these below). This step maps the vectors to a data structure that will enable faster searching.
2. Querying: The vector database compares the indexed query vector to the indexed vectors in the dataset to find the nearest neighbors (applying a similarity metric used by that index)
3. Post Processing: In some cases, the vector database retrieves the final nearest neighbors from the dataset and post-processes them to return the final results. This step can include re-ranking the nearest neighbors using a different similarity measure.[3]

3.4 Retrieval-Augmented Generation (RAG)

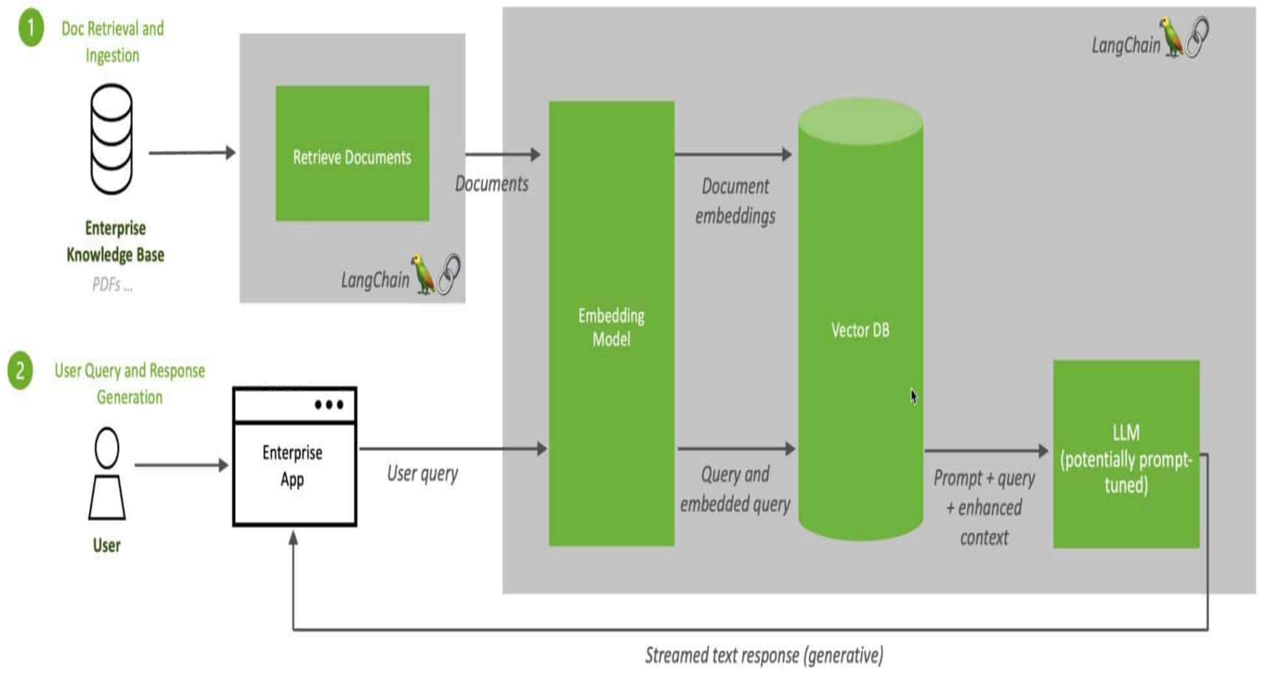

1
Doc Retrieval and Ingestion
Enterprise Knowledge Base
PDFs ...
Retrieve Documents
LangChain
Documents
Document embeddings
LangChain
Embedding Model
Vector DB
2
User Query and Response Generation
Enterprise App
User
User query
Query and embedded query
Prompt + query + enhanced context
LLM (potentially prompt-tuned)
Streamed text response (generative)

Fig 5. Retrieval-Augmented Generation Sequence Diagram

RAG is an AI framework for retrieving facts from an external knowledge base to ground large language models (LLMs) on the most accurate, up-to-date information and to give users insight into LLMs' generative process.[4] Retrieval-Augmented Generation (RAG) plays an important role in ensuring the accuracy and focus of the generated questions. While LLMs like Llama-2 are powerful tools for text generation, their knowledge is limited to the data they were trained on. The training data used might not include the exact content of the notes that are required for the questions in the question paper. Thus, RAG is used to bridge this gap by acting as a mechanism to retrieve relevant information from a knowledge base (Vector Database).

4. RESULTS

REC QPG - This is a web application designed to generate question papers based on uploaded PDF notes. The design of the application is given below.

4.1 Upload Notes

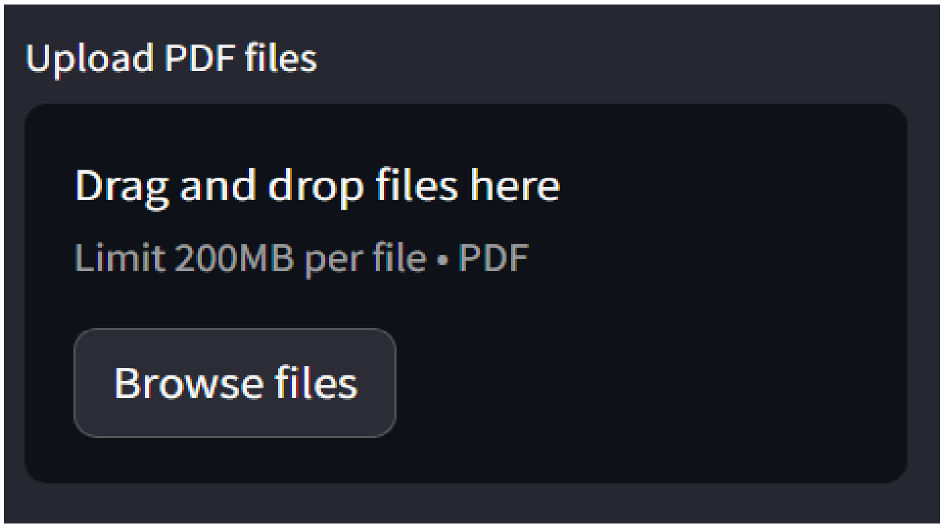
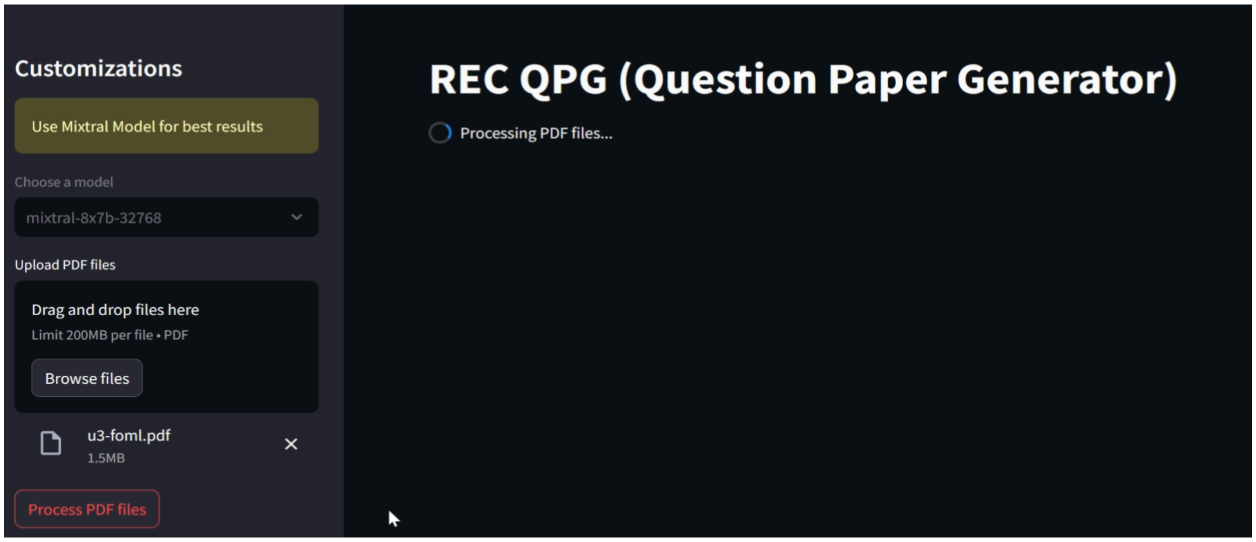


Fig 5. PDF Upload Screen

First, the user has to drag and drop the PDF files of the notes that have to be used as context for the question paper. Optionally, the user can also choose to open file explorer and select the files that are to be uploaded.

4.2 Process PDF Files


Customizations

Use Mixtral Model for best results

Choose a model: mixtral-8x7b-32768

Upload PDF files

Drag and drop files here
Limit 200MB per file • PDF

Browse files

u3-foml.pdf 1.5MB

Process PDF files

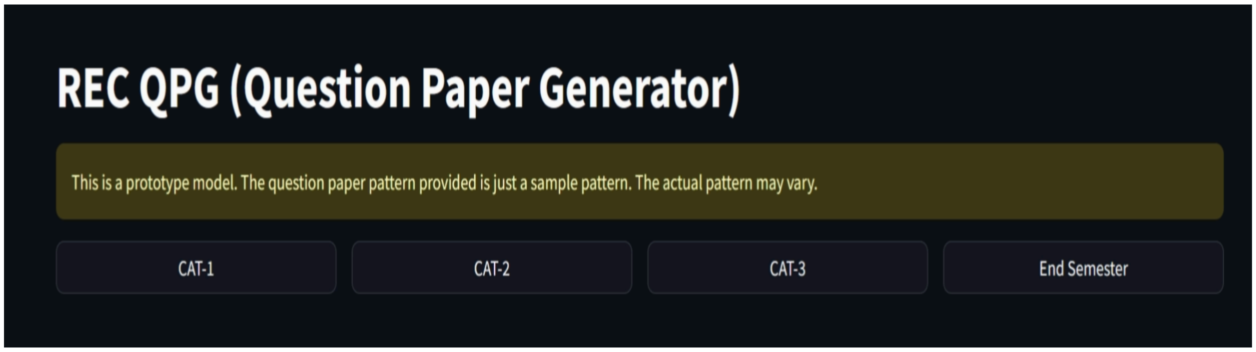
REC QPG (Question Paper Generator)

Processing PDF files...

Fig 6. Processing PDF Files

The PDF files are then processed in phases. First, the PDF is converted into plain text and the large amounts of text is split into smaller chunks. Each of these chunks represent some meaningful data. This chunk is then stored into a vector database in a mathematical representation and can be used for generation of the question papers.

4.3 Choose Question Paper Pattern

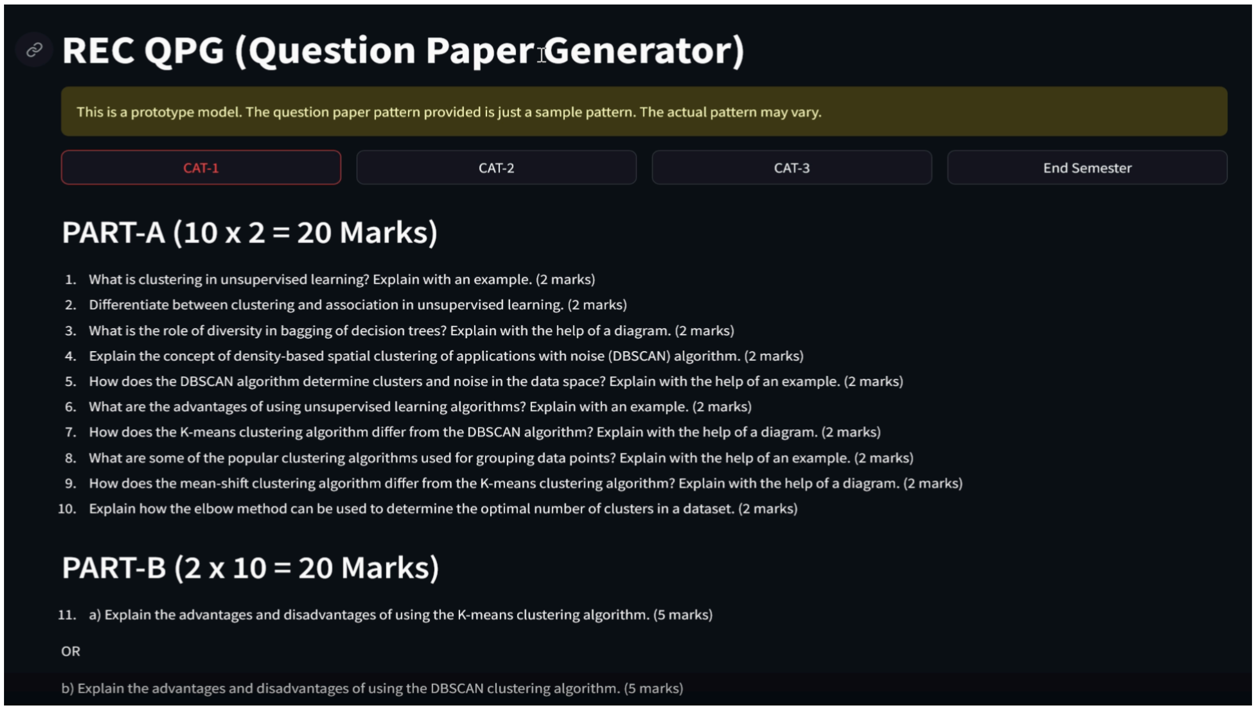

REC QPG (Question Paper Generator)

This is a prototype model. The question paper pattern provided is just a sample pattern. The actual pattern may vary.

Fig 7. Question Paper Pattern Buttons

The user has the choice of selecting from the various examinations. Depending on the user's selection a pattern for the question paper will be set. The questions that are drafted will be based strictly upon the basis of the specified question paper pattern.

4.4 Generated Question Paper


REC QPG (Question Paper Generator)

This is a prototype model. The question paper pattern provided is just a sample pattern. The actual pattern may vary.

PART-A (10 x 2 = 20 Marks)

1. What is clustering in unsupervised learning? Explain with an example. (2 marks)
2. Differentiate between clustering and association in unsupervised learning. (2 marks)
3. What is the role of diversity in bagging of decision trees? Explain with the help of a diagram. (2 marks)
4. Explain the concept of density-based spatial clustering of applications with noise (DBSCAN) algorithm. (2 marks)
5. How does the DBSCAN algorithm determine clusters and noise in the data space? Explain with the help of an example. (2 marks)
6. What are the advantages of using unsupervised learning algorithms? Explain with an example. (2 marks)
7. How does the K-means clustering algorithm differ from the DBSCAN algorithm? Explain with the help of a diagram. (2 marks)
8. What are some of the popular clustering algorithms used for grouping data points? Explain with the help of an example. (2 marks)
9. How does the mean-shift clustering algorithm differ from the K-means clustering algorithm? Explain with the help of a diagram. (2 marks)
10. Explain how the elbow method can be used to determine the optimal number of clusters in a dataset. (2 marks)

PART-B (2 x 10 = 20 Marks)

11. a) Explain the advantages and disadvantages of using the K-means clustering algorithm. (5 marks)

OR

b) Explain the advantages and disadvantages of using the DBSCAN clustering algorithm. (5 marks)

Fig 8. Final Output

Finally, the question paper is generated based on the specified pattern and the questions are based upon the uploaded notes. Each generation will give a new set of questions, so this can be used to generate multiple sets of a single paper.

5. CONCLUSION

In conclusion, this application not only accelerates the question paper generation process but also improves the quality and relevance of the papers generated. Automating the monotonous aspects of question paper creation enables professors to spend more time focusing on other aspects of teaching and engaging their students.

Furthermore, this application creates a level playing field for all assessment aspects by ensuring that all subjects and topics are well-represented in the question papers. Moreover, institutions can customize the question papers to align with their individual curriculum demands and assessment objectives. Be it based on the difficulty of questions, some topics that should be given weight, or the types of questions to include, the educators will have the freedom to adapt question papers to themes that suit their various categories.By leveraging technologies such as natural language processing, large language models, and retrieval augmented generation, the application empowers educators to create high-quality question papers efficiently, benefiting both students and institutions.Through NLP, LLM, and RAG, the application facilitates educators in generating the best question papers with high efficiency, benefiting students and the institution in various ways. With the integration of NLP, the application delves into the contents of subject notes and thus derives questions that are contextually relevant and meet the educational criteria. The application aims to create appropriate and meaningful questions that will test the comprehension and critical thinking skills of students with paramount effectiveness.Furthermore, the RAG increases precision and relevance by integrating the information retrieval capability. This approach not only makes the assessment process much more enriching but also includes the capacity for the educator to draw from a wide range of sources while formulating questions. The application saves time by automating tasks that an educator has to do, like formatting and arranging questions in a specified pattern while generating question papers. This allows professors to spend more time on interactive teaching methods and student support.Most importantly, the application tailors question papers according to specific demands from the curriculum and objectives of the assessment, thus ensuring that educators develop assessments for their specific needs and educational goals. Whether it is question difficulty, topic weighting, or picking the right type of question, the educator has the freedom to design an assessment that will be best at measuring understanding and progress.This is a leap in educational technology because it makes assessment practices easier, faster, and individualistic. This application automatically removes tedious administrative jobs from educators' tasks and enhances the quality and relevance of assessments. Ultimately improving student learning outcomes and contributing to the excellence of education at the institution.

References:

1. Alawwad, Hessa Abdulrahman, Areej Alhothali, Usman Naseem, Ali Alkhathlan, and Amani Jamal. "Enhancing Textbook Question Answering Task with Large Language Models and Retrieval Augmented Generation." *arXiv preprint arXiv:2402.05128* (2024).
2. Madhav, Dinesh, Sanskruti Nijai, Urvashi Patel, and Komal Champanerkar. "Question Generation from PDF using LangChain." In *2024 11th International Conference on Computing for Sustainable Global Development (INDIACom)*, pp. 218-222. IEEE, 2024.
3. Pinecone. "What is a Vector Database?"
4. IBM Research. "Retrieval-Augmented Generation (RAG)."
5. Brown, T. B., et al. (2020). Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33, 1877-1901.
6. Radford, A., et al. (2019). Language models are unsupervised multitask learners. *OpenAI Blog*.
7. Raffel, C., et al. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140), 1-67.
8. 8. Black, S., et al. (2022). GPT-NeoX-20B: An open-source autoregressive language model. *arXiv preprint*.
9. Lewis, P., et al. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. *Advances in Neural Information Processing Systems*, 33, 9459-9474.
10. Izacard, G., & Grave, E. (2020). Leveraging passage retrieval with generative models for open domain question answering. *arXiv preprint*.
11. Yih, W. T., et al. (2020). Pre-trained text encoders for scalable zero-shot retrieval. *arXiv preprint*.
12. LangChain Documentation. (2022). LangChain.
13. LangChain GitHub Repository. (2022).
14. Sultanum, N., et al. (2019). PDF-tExtract: a flexible framework for PDF data extraction and visualization. *Proceedings of the 12th ACM International Conference on Web Search and Data Mining*.
15. Lozano, A. C., et al. (2019). Tabula: Extracting tables from PDFs. *The Journal of Open Source*

Software, 4(36), 1246.

16. Zhong, X., et al. (2019). PubLayNet: largest dataset ever for document layout analysis. *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*.

17. Johnson, J., et al. (2019). Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3), 535-547.

18. Guo, R., et al. (2020). Accelerating large-scale inference with anisotropic vector quantization. *Proceedings of the 37th International Conference on Machine Learning*.

19. Harris, C. R., et al. (2020). Array programming with NumPy. *Nature*, 585(7825), 357-362

20. Zhu, Y., et al. (2019). Texar: A modularized, versatile, and extensible toolkit for text generation. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

21. Agarwal, R., & Mannem, P. (2021). Automatic gap-fill question generation from text books. *Proceedings of the Sixth Workshop on Innovative Use of NLP for Building Educational Applications*.

22. Ghosh, S., et al. (2020). Automated generation of multiple-choice questions for vocabulary assessment. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(01), 888-895.

23. Gikandi, J. W., et al. (2020). Online formative assessment in higher education: A review of the literature. *Computers & Education*, 57(4), 2333-2351.

24. Nicol, D. J., & Macfarlane‐Dick, D. (2021). Formative assessment and self‐regulated learning: A model and seven principles of good feedback practice. *Studies in Higher Education*, 31(2), 199-218.

25. Haladyna, T. M., & Downing, S. M. (2020). A taxonomy of multiple-choice item-writing rules. *Applied Measurement in Education*, 2(1), 37-50.

26. Miller, M. D., Linn, R. L., & Gronlund, N. E. (2021). *Measurement and assessment in teaching*. Pearson.

27. Manning, C. D., et al. (2020). The Stanford CoreNLP natural language processing toolkit. *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*.

28. Bird, S., et al. (2020). *Natural language processing with Python: analyzing text with the natural language toolkit*. O'Reilly Media, Inc

29. Bello, M., & Usman, M. (2021). An overview of document layout analysis. *Journal of Computer Science and Technology*, 31(5), 1068-1085

30. Wilkerson, M., & Roos, B. (2022). Extracting data from PDFs using open source tools. *Journal*

of Open Research Software, 2(1).

31. Schlegel, V., et al. (2020). Fast and accurate nearest neighbor search with the euclidean k-means tree. *IEEE Transactions on Knowledge and Data Engineering*, 32(1), 82-96.

32. Aumüller, M., et al. (2020). Ann-Benchmarks: A benchmarking tool for approximate nearest neighbor algorithms. *Information Systems*, 87, 101374.