Department of Computer Science and Engineering
CS19643 – FOML- Foundations of Machine Learning

REC Question Paper Generator

SAJAY PRAKASH K - 210701222
PRIYADHARSHINI - 210701197

Problem Statement

Institutions invest significant time and resources in preparing meticulously crafted question papers across diverse subjects to ensure equal representation of units and uphold educational standards. However, this process often entails laborious manual effort from professors, including formulating, reviewing, and finalizing question papers. To streamline this process and enhance efficiency, an innovative application is proposed. Leveraging advanced technologies such as Natural Language Processing (NLP), Retrieval Augmented Generation (RAG), and the LangChain framework, this application aims to revolutionize question paper creation.

Motivation

- Efficiency: Reduce the time and effort required by professors to create well-structured question papers.
- Consistency: Ensure consistent quality and structure across different question papers.
- Fairness: Guarantee balanced coverage of all subject units in the questions.
- Scalability: Easily scale the process to handle multiple subjects and large volumes of content.
- Customization: Allow for customization of question types and difficulty levels.
- Error Reduction: Minimize human errors in question formulation and distribution.
- Accessibility: Provide a user-friendly interface for educators to interact with the system.

Abstract

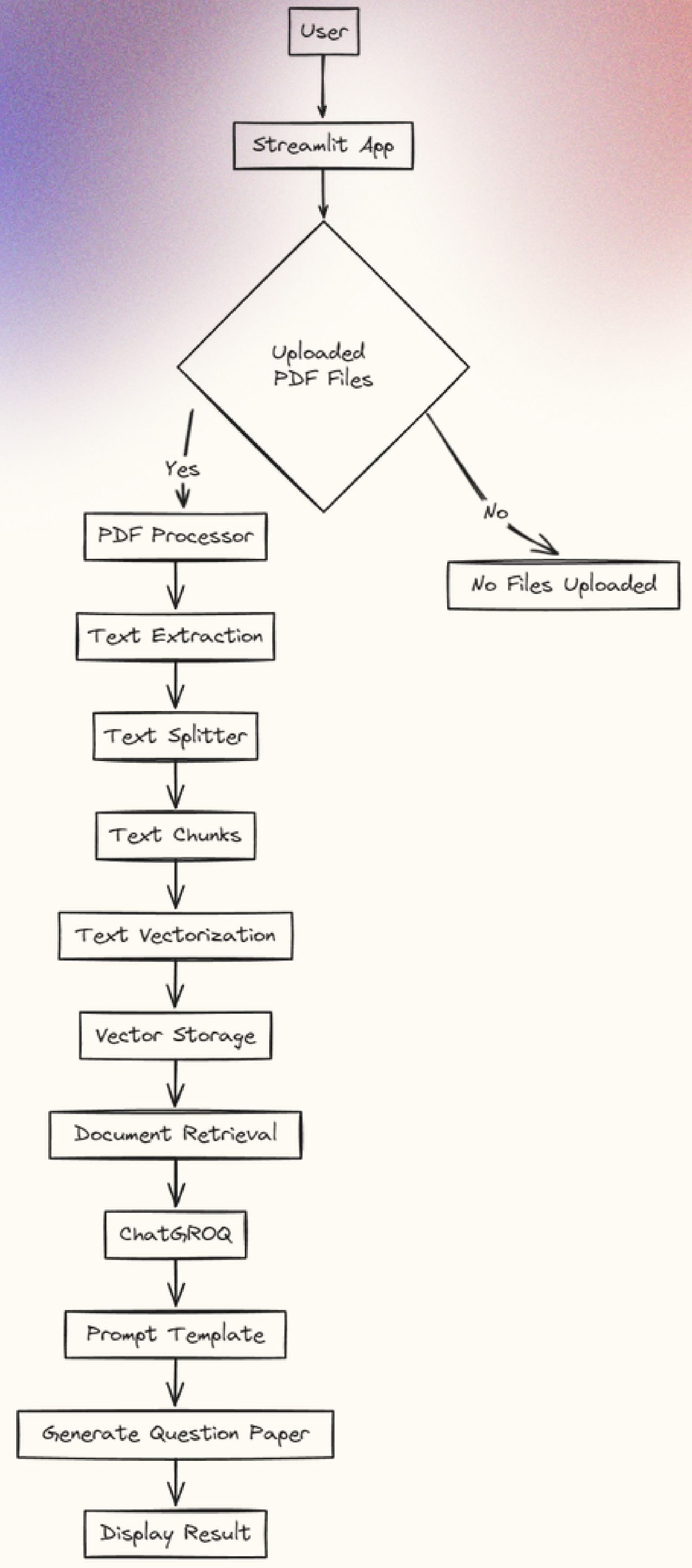
Today, institutions spend many days preparing well-articulated question papers for students across various subjects. The questions themselves need to be structured so that each subject's units are given equal weight. Additionally, the questions must guarantee that students receive proper educational 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.

System Architecture

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.



List of Modules

User Interface Module

The User Interface Module is the primary point of interaction for users within the Streamlit application. It includes components for uploading PDF files, selecting the type of exam (e.g., CAT-1, CAT-2, End Semester), and displaying the final generated question paper. Users upload their notes through a file uploader, make selections via dropdown menus, and view the generated output directly in the app, providing a seamless and intuitive experience.

PDF Processing Module

The PDF Processing Module handles the initial steps of converting user-uploaded PDF files into text. It uses libraries like pdfplumber to extract text content from each page of the PDFs. This module ensures that the text is properly prepared for further processing by managing file reading and text extraction, forming the foundation for subsequent text analysis and question paper generation.

List of Modules

Text Processing Module

The Text Processing Module takes the extracted text and prepares it for analysis by splitting it into smaller, manageable chunks using a Text Splitter. These chunks are then converted into vector representations through Text Vectorization, using embeddings such as OllamaEmbeddings. This transformation from text to vectors enables efficient storage and retrieval, facilitating the subsequent steps in the question paper generation process.

Storage and Retrieval Module

The Storage and Retrieval Module focuses on storing the vectorized text chunks and retrieving relevant segments based on user queries. It employs FAISS (Facebook AI Similarity Search) for fast and efficient similarity searches, ensuring that the system can quickly access and retrieve the most pertinent text segments. This module is critical for providing accurate and contextually relevant information during the question paper generation.

List of Modules

LLM Integration Module

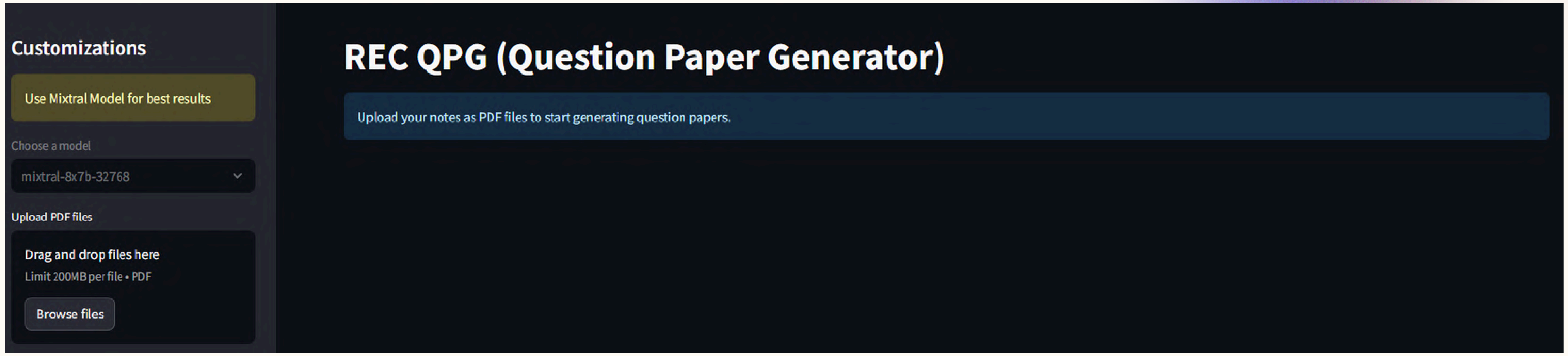
The LLM Integration Module integrates with the Groq API to leverage language models for generating question papers. Using the retrieved text segments and a predefined Prompt Template, this module generates coherent and structured questions that adhere to the specified exam format. It ensures that the output is aligned with the user's requirements, producing high-quality question papers based on the input notes.

Processing Workflow Module

The Processing Workflow Module orchestrates the entire sequence of operations from text extraction to question paper generation. It manages the retrieval chain and document chain processes, ensuring smooth transitions and data flow between stages. This module ensures that each step, from handling raw text to producing the final question paper, is executed efficiently and accurately, resulting in a cohesive workflow.

Implementation/Results of Module

Landing Page



Customizations

Use Mixtral Model for best results

Choose a model

mixtral-8x7b-32768

Upload PDF files

Drag and drop files here

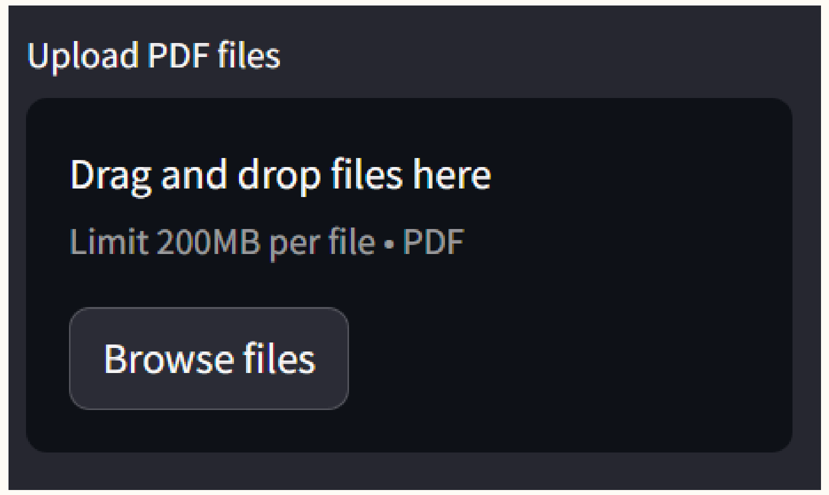
Limit 200MB per file • PDF

Browse files

REC QPG (Question Paper Generator)

Upload your notes as PDF files to start generating question papers.

File Upload



Upload PDF files

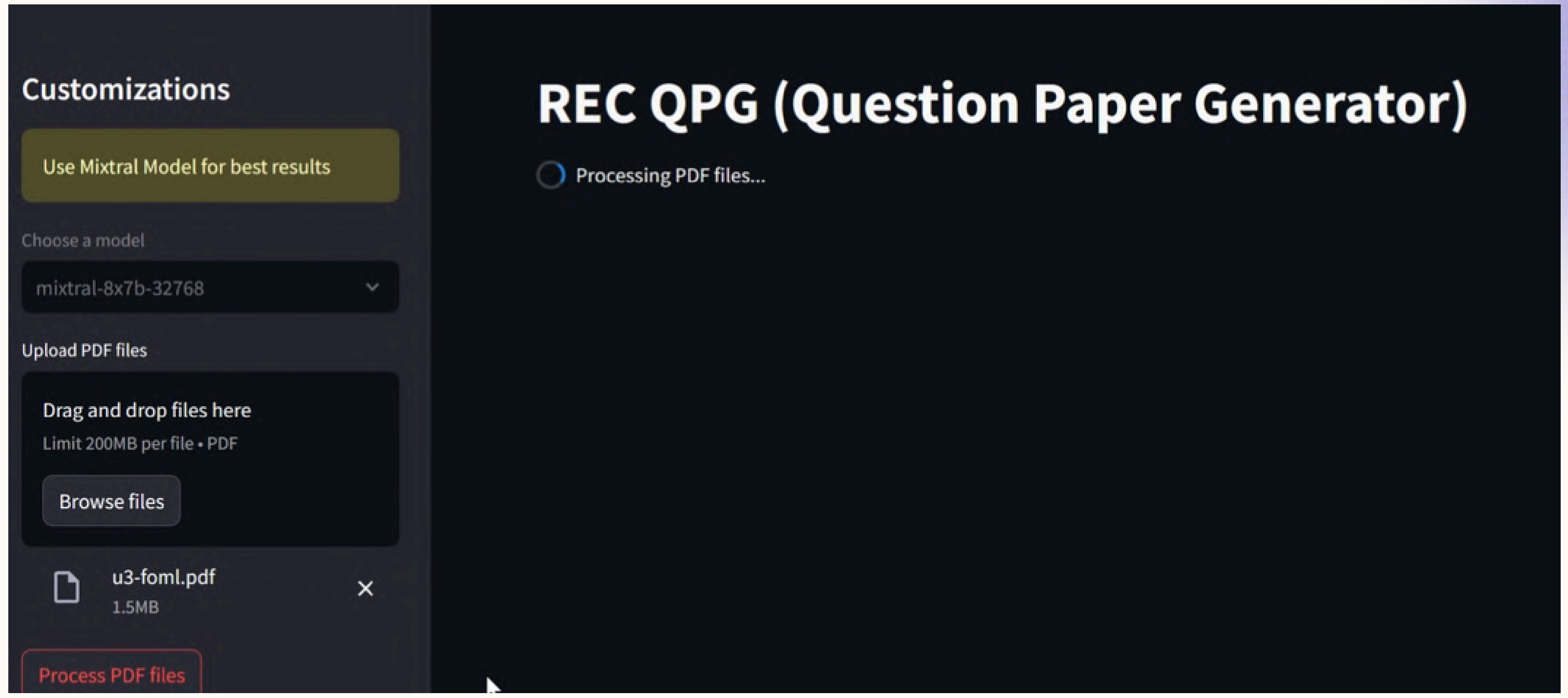
Drag and drop files here

Limit 200MB per file • PDF

Browse files

Implementation/Results of Module

Processing PDF


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

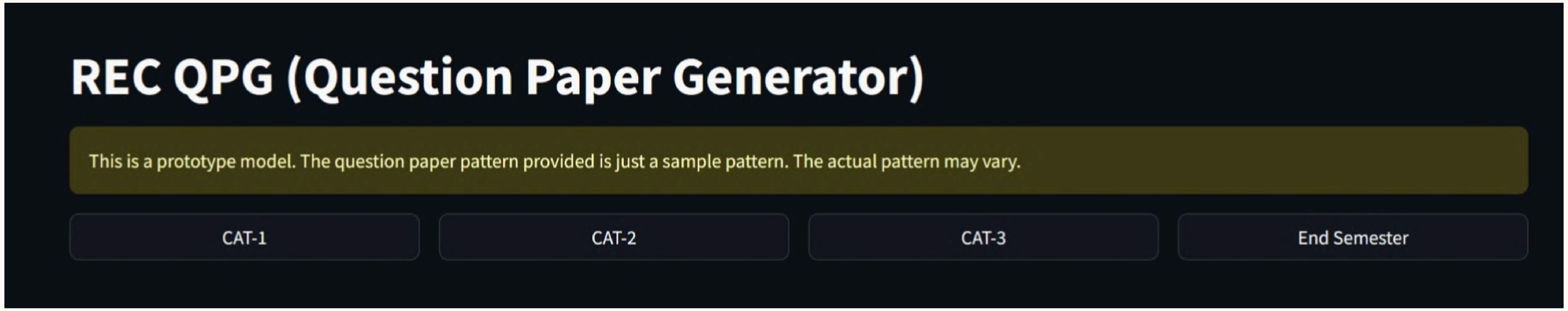
u3-foml.pdf 1.5MB

Process PDF files

REC QPG (Question Paper Generator)

Processing PDF files...

Choosing Question Paper Pattern

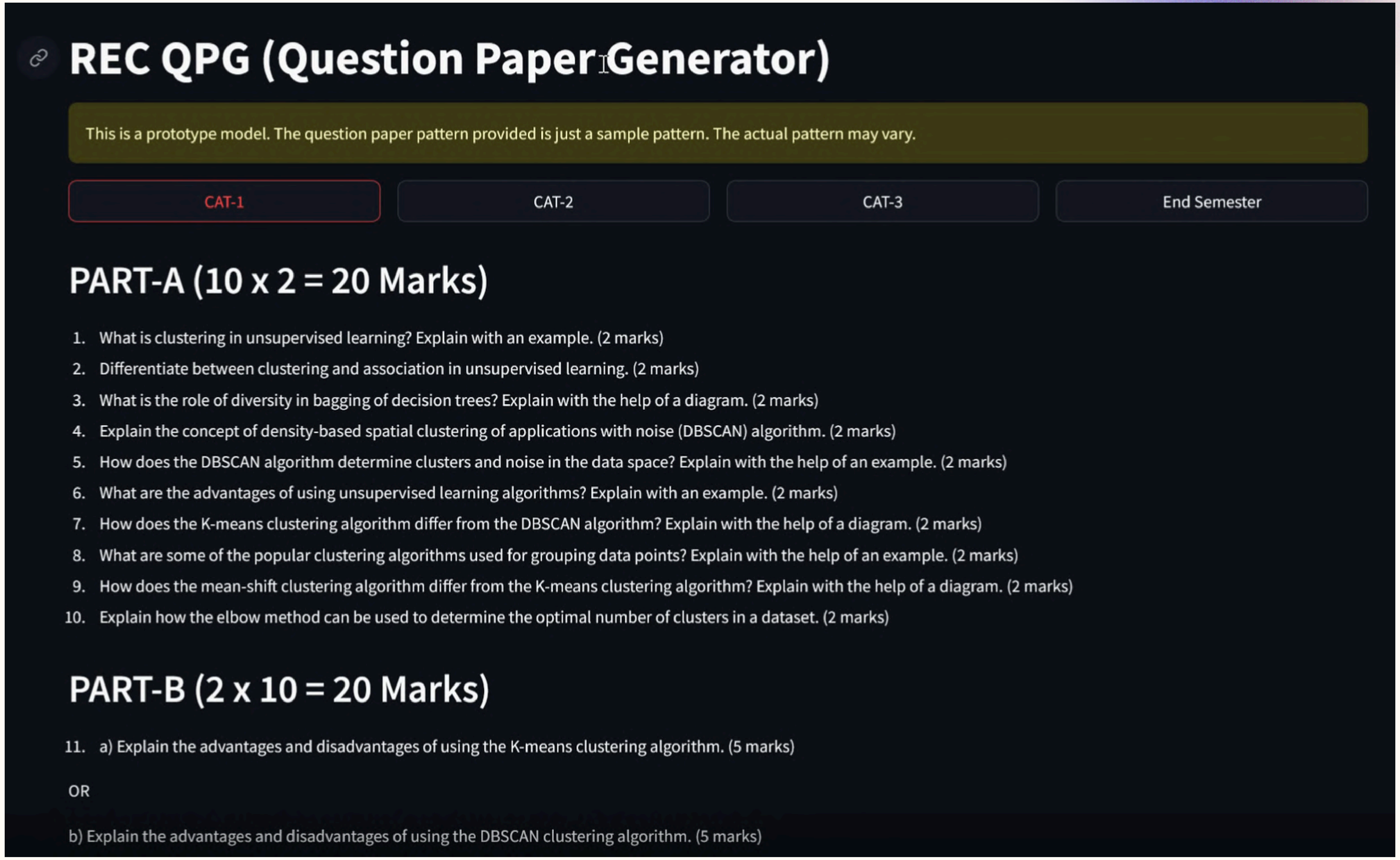

REC QPG (Question Paper Generator)

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

CAT-1 | CAT-2 | CAT-3 | End Semester

Implementation/Results of Module

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)

References

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).
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.
Pinecone. "What is a Vector Database?"
IBM Research. "Retrieval-Augmented Generation (RAG)."
Brown, T. B., et al. (2020). Language models are few-shot learners. Advances in Neural Information Processing Systems, 33, 1877-1901.
Radford, A., et al. (2019). Language models are unsupervised multitask learners. OpenAI Blog.
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. Black, S., et al. (2022). GPT-NeoX-20B: An open-source autoregressive language model. arXiv preprint.
Lewis, P., et al. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. Advances in Neural Information Processing Systems, 33, 9459-9474.
Izacard, G., & Grave, E. (2020). Leveraging passage retrieval with generative models for open domain question answering. arXiv preprint.
Yih, W. T., et al. (2020). Pre-trained text encoders for scalable zero-shot retrieval. arXiv preprint.
LangChain Documentation. (2022). LangChain.
LangChain GitHub Repository. (2022).
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.
Lozano, A. C., et al. (2019). Tabula: Extracting tables from PDFs. The Journal of Open Source Software, 4(36), 1246.

References

Zhong, X., et al. (2019). PubLayNet: largest dataset ever for document layout analysis. Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops.
Johnson, J., et al. (2019). Billion-scale similarity search with GPUs. IEEE Transactions on Big Data, 7(3), 535-547.
Guo, R., et al. (2020). Accelerating large-scale inference with anisotropic vector quantization. Proceedings of the 37th International Conference on Machine Learning.
Harris, C. R., et al. (2020). Array programming with NumPy. Nature, 585(7825), 357-362
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.
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.
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.
Gikandi, J. W., et al. (2020). Online formative assessment in higher education: A review of the literature. Computers & Education, 57(4), 2333-2351.
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.
Haladyna, T. M., & Downing, S. M. (2020). A taxonomy of multiple-choice item-writing rules. Applied Measurement in Education, 2(1), 37-50.
Miller, M. D., Linn, R. L., & Gronlund, N. E. (2021). Measurement and assessment in teaching. Pearson
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.
Bird, S., et al. (2020). Natural language processing with Python: analyzing text with the natural language toolkit. O'Reilly Media, Inc
Bello, M., & Usman, M. (2021). An overview of document layout analysis. Journal of Computer Science and Technology, 31(5), 1068-1085
Wilkerson, M., & Roos, B. (2022). Extracting data from PDFs using open source tools. Journal of Open Research Software, 2(1).

Thank You