ExamGenerator

Zac Castaneda
Department of Computer Science
Texas A&M University - Corpus Christi
Corpus Christi, Texas, United States of America
Ecastaneda7@islander.tamucc.edu

Ryan Grieb
Department of Computer Science
Texas A&M University - Corpus Christi
Corpus Christi, Texas, United States of America
rgrieb@islander.tamucc.edu

Christopher Corkill
Department of Computer Science
Texas A&M University - Corpus Christi
Corpus Christi, Texas, United States of America
ccorkill@islander.tamucc.edu

Nico Elwell
Department of Computer Science
Texas A&M University - Corpus Christi
Corpus Christi, Texas, United States of America
nelwell@islander.tamucc.edu

Gabriel Cavazos
Department of Computer Science
Texas A&M University - Corpus Christi
Corpus Christi, Texas, United States of America
Gcavazos3@islander.tamucc.edu

***Abstract*— This article presents ExamGenerator, an online tool driven by artificial intelligence that can create exam questions and answers automatically from document inputs including Word, PDF, and text files. ExamGenerator turns course materials into extensive tests and quizzes with multiple-choice, true/false, and multiple-selection questions by leveraging Open AI's ChatGPT API and additional libraries.**

The purpose of ExamGenerator, associated research, suggested methods, and system architecture are all described in this study. The system attempts to provide high-quality assessments in much less time than educators currently spend creating them. With automation, Exam Generator helps educators maintain consistency and order across different types of content and gives them a better variety of questions that adapts to different levels of learning. This process sets Exam Generator as a key component in educational assessment by improving the educator's workflow. ExamGenerator has the potential to have a significant impact on exam development efficiency, assessment quality uniformity, and question type generation, that provides and maintains a flexible tool for educators of all concepts.

Keywords—ChatGPT, OpenAI, Exam, NLP, Generation

I. Introduction

Today's ever changing digital education environment has made learning and evaluation tools ever-more imperative. Exam question generation using traditional methods is labor- and time-intensive, which forces educators to look for creative solutions that streamline the process. ExamGenerator emerges as a solution to these challenges, utilizing cutting-edge technologies to enhance both the efficiency and accuracy of exam creation. By automating key steps and offering customizable features, it empowers educators to focus on what truly matters—ensuring high-quality education. Furthermore, its intuitive interface makes it accessible to educators of all technical levels, providing a seamless transition from outdated methods. By tapping into vast educational content and applying sophisticated natural language processing techniques with OpenAI, ExamGenerator will be able to produce professionally made exam and quiz content. This technology also addresses an educational need that enhances assessments. Allowing exams to be tailored to different formats or learning concepts that are required by modern classroom environments.

A. Problem Statement

Creating exam questions from educational resources usually requires a lot of manual work, which can be time-consuming and prone to mistakes. Teachers often struggle with limited time, maintaining consistent question quality, and covering all the important material. As digital content grows, manually going through documents to find relevant information for assessments becomes increasingly impractical. Having learning objectives that maintain a consistent level of difficulty and adherence across exams presents an additional layer of complexity. As a result, there is a pressing need for an automated solution that can efficiently transform difficult diverse educational documents into easily read structured exam formats, while maintaining accuracy and relevance. This tool would significantly decrease the preparation time and efforts on educators, shifting the focus toward delivering content rich instruction.

B. Motivation

ExamGenerator was developed with the goal of simplifying the process of producing high quality tests and quizzes for instructors and educational establishments. ExamGenerator saves teachers time and allows them to concentrate on improving the learning environment for their students by automating the question production process. Our team has a special interest in this solution due to us being

from families in education. Our team is motivated to create a technology that relieves instructors of the load of creating exams, since we have seen firsthand the difficulties, they encounter in juggling administrative responsibilities and classroom management. ExamGenerator aims to produce an effective way to generate test questions that maintain academic concepts without requiring hours of manual energy. This tool saves time as well as generates comprehensive questions that cover key topics thoroughly. Thus, ExamGenerator hopes to motivate educators to prioritize the students goals and learning experience without being drained to contribute quality education.

II. RELATED WORK

Numerous tools leverage artificial intelligence for educational purposes, particularly in quiz and exam generation. Below are some notable platforms and their limitations in relation to ExamGenerator:

- **Quizlet**: A widely-used study tool, Quizlet enables users to create flashcards and quizzes. However, it relies heavily on user-generated input for question creation, lacking automation and efficiency in question generation.

- **Socrative**: This interactive platform facilitates real-time quizzes and polls, providing immediate feedback to users. Nonetheless, it requires manual question creation, which can limit scalability for larger educational resources and materials.

- **ProProfs Quiz Maker**: An online tool that allows for quiz creation with diverse question types and templates. However, like the aforementioned tools, it predominantly relies on manual input, failing to automate the extraction of questions from educational documents.

- **QTI-compliant Tools**: These tools adhere to the Question and Test Interoperability (QTI) standard for sharing assessment content but do not address the automation of question generation, which remains a labor-intensive task.

- **AI-Powered Educational Platforms**: While platforms like Gradescope and Respondus streamline grading and online testing, they do not focus on the automation of question generation from existing educational documents.

- **Respondus**: This popular tool is used for exam creation and secure test taking especially for online exams. It can be integrated into blackboard and canvas however like other tools it focuses on exam security more than automated test creation. Questions are usually manually entered.

In contrast, while existing tools provide different functionalities, they often rely on manual input. *ExamGenerator* uniquely automates the creation of test questions by employing advanced natural language processing techniques. By extracting relevant information from a variety of document formats (such as PDF, DOCX, and Excel) and leveraging AI technologies like OpenAI's ChatGPT API, *ExamGenerator* offers a streamlined, efficient solution that significantly reduces the time and effort educators spend on exam creation. This innovation not only enhances the quality and consistency of generated assessments but also empowers educators to focus more on teaching rather than administrative tasks.

III. PROPOSED APPROACHES

ExamGenerator is a deliberate invention that meticulously integrates cutting-edge technology, contemporary frameworks, and user-centered design concepts to produce a scalable, highly efficient exam generation automation solution. With the use of cutting-edge natural language processing, adaptable file format management, and smooth connection with educational platforms, the tool aims to provide educators with a flexible and intuitive interface. The key approaches include:

A. Natural Language Processing with APIs:

ExamGenerator aims to integrate advanced natural language processing (NLP) capabilities through the ChatGPT API, enabling the application to analyze and interpret educational documents for the extraction of relevant information to generate diverse, contextually accurate exam questions. The use of the ChatGPT API will allow ExamGenerator to handle complex educational language, ensuring that generated questions are both relevant and well-suited to various educational needs. By employing sophisticated NLP techniques, the system can produce questions that are adaptable in format and complexity, catering to different levels of understanding and learning objectives.

In addition to ChatGPT, ExamGenerator may integrate other leading NLP APIs, such as Claude and Gemini, to expand its functionality and improve question variety and depth. Claude, developed by Anthropic, is designed with a strong focus on ethical AI and interpretability, making it suitable for generating exam content that aligns with educational standards while avoiding potentially biased or inaccurate interpretations. This addition could allow ExamGenerator to benefit from Claude’s emphasis on clarity and safety, further enhancing the reliability of question generation.

Gemini, Google DeepMind's latest NLP model, could also be incorporated to enhance content diversity. Known for its ability to handle nuanced contextual understanding and multimodal capabilities, Gemini would enable ExamGenerator to generate questions that account for deeper, context-specific nuances within educational documents. With Gemini, the system could extend its question generation to include more nuanced analytical questions and provide richer feedback by interpreting complex subject matter accurately.

Using a combination of these APIs would allow ExamGenerator to dynamically select the most suitable NLP service based on task complexity or subject specificity. This multi-API approach would enhance ExamGenerator's adaptability, ensuring that it can handle a wide variety of document types and educational contexts. Additionally, this versatility enables more granular control over question formatting, complexity, and thematic alignment, contributing to a well-rounded and personalized question-generation experience.

B. Import Formats:

ExamGenerator will be designed to support various input file formats, including PDF, DOCX, and Excel. Because of this flexibility, instructors will be able to upload instructional resources in the format of their choice, increasing the tool's accessibility. Multiple file formats are included to make it easy for users to incorporate the tool into their current processes. In addition, the system will make use of libraries like pandas for Excel spreadsheets and python-docx for DOCX files in order to guarantee precise content extraction and facilitate the processing of various educational resources.

C. Export Formats:

The tool will offer multiple options for exporting generated exams, including PDF, DOCX, and plain text formats. This flexibility would allow educators to use the generated assessments across various platforms while maintaining compatibility with existing educational standards. Additionally, ExamGenerator could integrate with popular learning management systems such as Blackboard, Canvas, and Moodle, enabling users to export exams directly into these platforms. This would streamline instructor operations and lessen the need for manual imports by making the deployment of assessments and management of student contributions simpler.

ExamGenerator hopes to serve instructors and students alike by combining these strategies to offer an effective, reliable, and user-focused solution for automating the creation of exam questions.

IV. System Design

The system design for ExamGenerator is structured to leverage modern technologies and methodologies to ensure the proposed approaches are able to be implemented in a timely manner. The architecture consists of several key components:

A. Web Framework:

The application will be developed using the Quart framework, selected for its support of asynchronous actions. Quart is an asyncio-based framework that allows for non-blocking requests, enabling ExamGenerator to handle multiple user requests concurrently. This capability is crucial for providing a responsive user experience, especially when processing large documents or handling simultaneous requests from multiple educators. Quart's compatibility with the Flask ecosystem also allows for easy integration of existing Flask extensions, which can enhance functionality.

B. Containerization with Docker:

To enhance deployment, scalability, and maintainability, ExamGenerator will leverage Docker for comprehensive containerization of its core components. Docker will package the application and its dependencies into isolated containers, providing a consistent environment across development, testing, and production. This setup ensures that each component of ExamGenerator, from the web framework to the backend services, operates in a controlled, reproducible environment, reducing issues related to discrepancies between local and production setups.

Each major component of ExamGenerator will run within its own Docker container for improved modularity and resource management. The web application itself, built on the asynchronous Quart framework, will be containerized to handle user requests efficiently, with support for concurrent connections and asynchronous processing. This configuration will facilitate responsive interactions and rapid data handling, especially as user demands increase.

For document processing, the unstructured-io API will also run in a dedicated container. This allows ExamGenerator to isolate document conversion processes, ensuring the API and its dependencies are encapsulated and manageable within a separate environment. Running unstructured-io in its own container also enables resource allocation specific to document extraction, so that any processing workload, like converting PDFs or extracting text from various file types, doesn't interfere with the application's overall performance.

A third container will host the MariaDB database, which manages essential data such as user information, exam content, and historical records. By containerizing MariaDB, we gain flexibility in database management, with controlled updates, efficient scaling, and streamlined backup procedures, all of which are supported by Docker's volume capabilities.

Furthermore, Docker's orchestration tools, such as Docker Compose and Kubernetes, will allow seamless management of these individual containers. Docker Compose will facilitate local development by simplifying the setup of multi-container configurations, while Kubernetes will enable robust container orchestration for production. This setup not only supports scaling but also improves fault tolerance, making it easier to add new features and services as ExamGenerator evolves.

C. Database Management:

A MariaDB database will be employed to manage and store user data, exam questions, and historical records. Running MariaDB within a Docker container enables a consistent, isolated environment, simplifying deployment across different platforms and ensuring compatibility with other services in the application stack. Containerizing MariaDB also supports easier updates, scaling, and streamlined backup procedures, which are essential for maintaining the integrity of user data and exam records.

MariaDB was chosen for its reliability, high performance, and compatibility with MySQL, making it a robust option for relational data storage. The database schema will be meticulously designed to manage diverse data types, such as user profiles, uploaded documents, generated exams, and feedback data, to ensure both functionality and efficiency. Through proper indexing and normalization, the database will support rapid query execution, enabling quick data retrieval during exam generation and user interactions. Additionally, we will implement regular data backup and recovery strategies, further facilitated by Docker's volume management, to protect against data loss and maintain continuity in case of unexpected failures.

D. Document to Text Conversion:

To facilitate the extraction of text from various document formats, ExamGenerator will integrate the unstructured-io API, which will run in a Docker container to ensure seamless deployment and efficient resource management. By containerizing the unstructured-io API, we can encapsulate its dependencies and configuration, allowing ExamGenerator to handle PDF conversion to structured text in a controlled, consistent environment. This approach also enhances the application's scalability and compatibility across different platforms.

In addition to PDF files, ExamGenerator will support various other document types to broaden its utility for instructors. For instance, the application will leverage python-docx to handle DOCX files, enabling the extraction of both text content and essential formatting information, which can be instrumental in replicating the structure and emphasis of exam questions. To support Excel-based exams and quizzes, pandas will be utilized for reading spreadsheet data, allowing ExamGenerator to interpret tables, rows, and columns as structured data ideal for generating well-organized exam content. This modular and flexible document processing capability ensures that ExamGenerator can handle a wide array of formats, making it a versatile tool for educators aiming to streamline the exam creation process.

E. Natural Language Processing:

The application integrates the ChatGPT API to generate high-quality exam questions from educational content. Initially, content is converted to text using the Unstructured-IO API, which is then processed by a specialized prompt tailored to the specific question types requested by the user. For multiple choice, plausible distractors are generated; for free response, open-ended prompts encourage detailed answers; and for fill in the blank, key concepts are omitted strategically. This customization ensures the generated questions are consistent, relevant, and properly aligned with the educational goals.

The system's ability to convert various document formats into text allows educators to create targeted exam questions with ease. By offering customizable question types, such as multiple choice, free response, fill in the blank, or true/false, the platform empowers educators to generate diverse assessment materials that align with their specific curriculum requirements. Future enhancements may include adaptive difficulty features, allowing questions to be dynamically adjusted based on user input to better match different educational levels.

F. User Interface:

The application is designed with a simple, modern, and user-friendly interface that caters specifically to teachers, ensuring they can easily navigate and use it without difficulty. Teachers will feel empowered to seamlessly interact with the system, focusing on their educational needs without cumbersome technical hurdles.

The interface supports a drag-and-drop upload feature, allowing users to quickly and easily upload educational materials in various formats such as PDF, DOCX, and Excel. Loading bars are displayed during the upload and document processing phases, providing clear visual feedback on the status and progress of the uploaded materials. This intuitive upload process means teachers can work efficiently without needing to manually prompt or interact with a complex AI system.

The documents page is thoughtfully designed to enhance productivity. A list of processed and processing documents is displayed on the left-hand side, giving users an organized overview of their materials. When a document is selected, the generated questions appear on the right-hand side of the page, allowing for a straightforward review and edit process. This

split-screen approach ensures that users can easily manage their documents while maintaining focus on the generated content.

One key aspect of the user interface is the preview and edit functionality, which allows teachers to review generated questions and make any necessary formatting or wording changes before finalizing the exam. This helps ensure that the questions meet specific educational requirements, providing full control over the content.

Users can specify exam generation parameters, including the number of questions, preferred wording styles, and question formats (e.g., multiple-choice, true/false, or open-ended). This customization is seamlessly integrated into the interface, offering easy access to essential settings without overwhelming the user.

The interface offers flexible export options, allowing users to export generated exams in their preferred format, including PDF, DOCX, or plain text, all from a conveniently accessible section.

The application will prioritize accessibility and responsiveness, ensuring a smooth experience across different devices and screen sizes. The development team is also exploring future improvements to ensure ADA compliance, making the application accessible to users with disabilities and providing an inclusive experience for all educators.

V. Individual Progress Reports

Ryan has successfully implemented a feature that enables the seamless uploading of various document types, laying the groundwork for the core functionality of ExamGenerator. Additionally, he took on the task of embedding a basic user interface, which serves as the foundation for further refinement and user experience enhancements.

As of November 24th, 2024, Ryan has successfully implemented ExamGenerator's basic user interface (see Figures 1, 2, and 3).

As of November 24th, 2024, Nicholas has updated and documented the results of ExamGenerator's functions explaining its capabilities.

Christopher has taken the lead on the drafting and organization of the project paper, meticulously managing the content structure to ensure clarity and comprehensiveness. He is actively seeking out and incorporating feedback from teammates, emphasizing collaborative refinement and continuous improvement of the document's quality.

As of November 24th, 2024, Christopher continues to excel at facilitating work on the project paper. He ensures everybody writes their portions of the paper and provides feedback accordingly.

Zac has been instrumental in the development and aesthetic styling of the team's website, creating a visually appealing and user-friendly digital platform for presenting ExamGenerator. In addition to this, he has diligently worked to keep the project paper updated with the most recent developments. Zac is also playing a supportive role by assisting Nico with a programming task, demonstrating a versatile and team-oriented approach to project contributions.

Nico has thoroughly reviewed the project paper to identify areas for improvement, ensuring that the document reflects accuracy and relevancy. Concurrently, he has embarked on the crucial task of developing the question output functionality, which is central to the system's automated exam generation capabilities.

Gabriel has actively coordinated and arranged team meetings, fostering clear communication and collaboration among all members. He has taken on the responsibility of overseeing project deliverables, ensuring that all tasks align with the project's timeline and objectives. In addition, Gabriel has been researching innovative approaches to advance the project, demonstrating his commitment to identifying new opportunities for improvement and growth. He has also begun working on the visual design of the question output, focusing on how multiple-choice and true/false questions will be presented to end-users, prioritizing clarity and user engagement.

As of November 24th, 2024, Gabriel has excelled in leading the teams' means of communication.

The entirety of the team has made marginal progress as to our programs functionality.

VI. Results

The results from developing and testing ExamGenerator have successfully shown its capability to streamline exam creation while ensuring these assessments remain high quality. Below are some results from our testing phase.

A. Document Processing: ExamGenerator processed various formats such as PDF, DOCX and Excel PowerPoint slides. During testing the system extracted text content and kept the original formatting and structure which is essential for generating accurate exam questions.

B. Question Generation: The system generated multiple choice, true/false and fill in the blank questions meeting the needs for educators.

Multiple Choice Questions: The questions that were generated included well-constructed material that was relevant to the imported data.

True/False Questions: The true and false questions generated were accurate with logical consistency. This help ensure the format was clear and easy to read.

Open-Ended Questions: ExamGenerator strategically found key concepts to create helpful blanks that align with the main concepts of the content.

C. Performance: In terms of efficiency, ExamGenerator generated questions from a 25-page PDF file in under 30 seconds. Smaller pages were tested and completed in just a few seconds. Using docker to run each major component in separate containers helped ensure efficient performance and resource management as well.

D. User Experience: Uploading files and documents made ExamGenerator intuitive and user friendly. Users with little technical knowledge can navigate the system and generate exams easily. Having the ability to preview the generated questions before finalizing the exam gives users confidence in the tool's output.

For a better visual, please see Figures 1, 2 and 3.

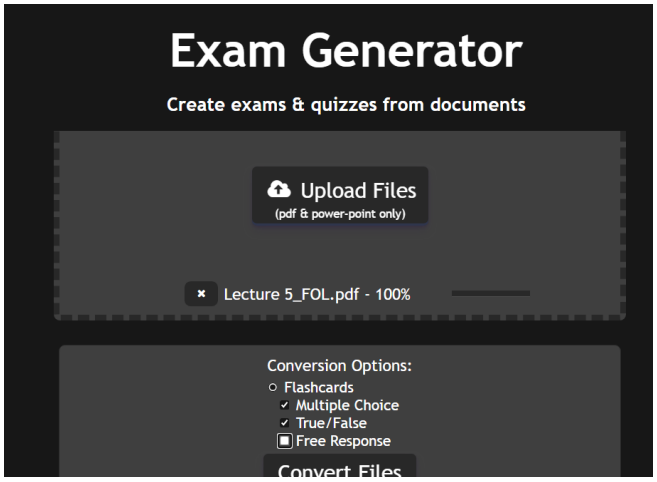

Figure 1 – The ExamGenerator home page. Users can upload files which will turn into true/false, multiple choice, and free response questions here.

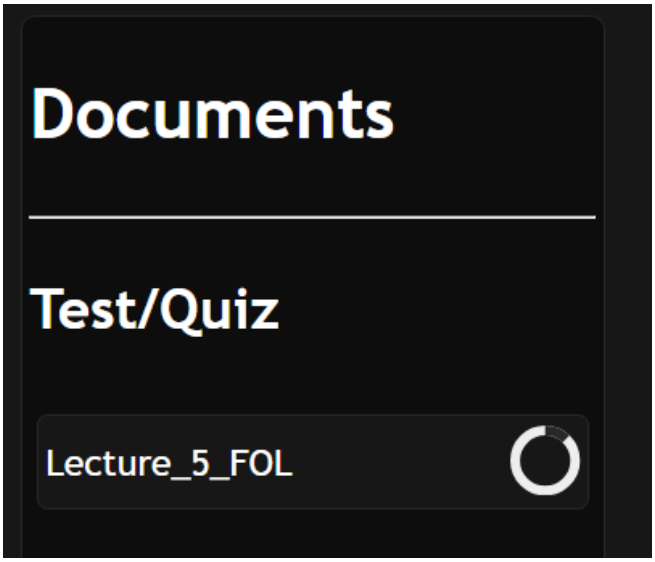

Figure 2 – ExamGenerator supports document uploading, as seen here. The user uploaded a document titled "Lecture_5_FOL".

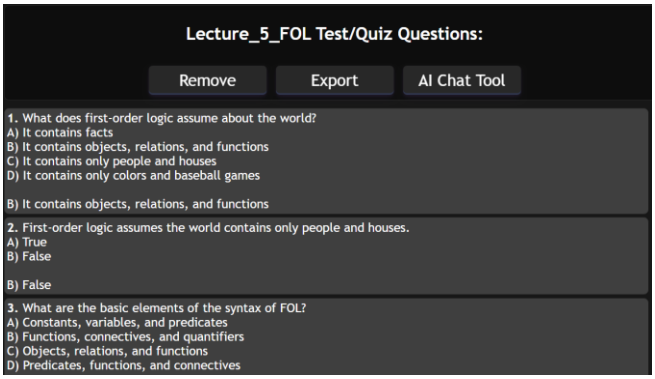

Figure 3 – An example of some of the questions ExamGenerator came up with.

VII. References

S. Bulathwela, H. Muse, and E. Yilmaz, "Scalable Educational Question Generation with Pre-trained Language Models," *arXiv.org*, May 13, 2023. https://arxiv.org/abs/2305.07871

"The implications of using AI to generate exam answers," *Turnitin.com*, Jun. 13, 2024. https://www.turnitin.com/blog/the-implications-of-using-ai-to-generate-exam-answers

B. Li, V. L. Lowell, C. Wang, and X. Li, "A Systematic Review of the First Year of Publications on ChatGPT and Language Education: Examining Research on ChatGPT's Use in Language Learning and Teaching," *Computers and education. Artificial intelligence*, pp. 100266–100266, Jul. 2024, doi: https://doi.org/10.1016/j.caeai.2024.100266.

S. Wolfram, "What Is ChatGPT Doing … and Why Does It Work?," *writings.stephenwolfram.com*, Feb. 14, 2023.

https://writings.stephenwolfram.com/2023/02/what-is-chatgpt-doing-and-why-does-it-work/

J. von Garrel and J. Mayer, “Which features of AI-based tools are important for students? A choice-based conjoint analysis,” *Computers and Education: Artificial Intelligence*, vol. 7, p. 100311, Dec. 2024, doi: https://doi.org/10.1016/j.caeai.2024.100311.

“What is ChatGPT doing...and why does it work?,” *www.youtube.com*. https://www.youtube.com/watch?v=flXrLGPY3SU.

L. Tunstall, Leandro von Werra, and T. Wolf, *Natural Language Processing with Transformers, Revised Edition*. O’Reilly Media, 2022.

D. Rothman, *Transformers for Natural Language Processing and Computer Vision*. Packt Publishing Ltd, 2024.

B. Auffarth, *Generative AI with LangChain*. Packt Publishing Ltd, 2023.

VIII. Appendix

TBA