ΠΑΝΕΠΙΣΤΗΜΙΟ
ΠΕΛΟΠΟΝΝΗΣΟΥ
University of the Peloponnese

M.Sc. Computer Science

Postgraduate Thesis

Automated test generation and marking using Local LLMs

Ioannis Papachristou

Student ID: **2022202302015**

Supervisor: Grigorios Dimitroulakos, Laboratory – Teaching Staff.

Supervisor: Konstantinos Vasilakis, Teaching – Research Staff.

TRIPOLI

February 2025

Table of contents

Table of figures

Abstract

This case study presents an innovative exam creation and grading system powered by advanced Natural Language Processing (NLP) and Llama 3.1. The system generates clear, grammatically accurate questions in English and Greek from both short text and long documents. It supports diverse question formats across various difficulty levels, ensuring semantically distinct content while minimizing redundancy. Grading utilizes a semantic similarity model to accurately evaluate essay and open-ended responses, offering partial credit and reducing bias from phrasing or syntax based on Named Entity Recognition (NER). A key advantage is its ability to run locally on ordinary personal computers without requiring specialized AI systems. The system also provides feedback on graded responses. Evaluations using metrics such as ROUGE, BLEU, diversity scores, and cosine similarity demonstrate its effectiveness, outperforming state-of-the-art models like BERT and T5 for educational assessment tasks.

Keywords: Exam Creation System, Natural Language Processing (NLP), Automated Grading, Llama 3.1, Question-Answer Generation, Retrieval Augmented Generation

Introduction

The rapid evolution of Artificial Intelligence (AI), particularly of large language models (LLMs), has demonstrated significant potential in automating tasks and offering a wide range of capabilities that enhance both teaching and learning experiences. LLMs are now enabling teacher assisted learning, where AI complements human instruction by alleviating routine tasks, allowing educators to concentrate more on the interactive and creative elements of their work [1]. For students, these models provide access to better content, minimizing human error and ensuring fairer assessments while additionally, they have the potential to bridge the language gaps between teachers and students, delivering higher-quality educational content and improving the overall learning process.[2]

One area where this can be fully appreciated is the examination process. Until now, teachers and evaluators have had to rely on creating questions and providing answers based on materials they have studied extensively, often from large documents (e.g., entire books). The conventional approach to creating examination papers in academic settings is often a manual process, which is not only time consuming but also repetitive. This method increases the likelihood of inefficiencies, such as potential bias or manipulation, often compromising the integrity and fairness of the process. [3] Furthermore designing effective assessments is a longstanding challenge in higher education, highlighted by numerous quality assurance reviews and often suffering from limited diversity of assessment types [4].

In this thesis, we present a novel Natural Language Processing (NLP) system designed to analyze extensive contexts, generate a diverse array of question types with corresponding correct answers, deliver accurate translations across multiple languages, and provide a grading mechanism based on sentence similarity. The system integrates advanced contextual analysis with translation and automated evaluation, offering a comprehensive solution for generating and grading educational content.

Section 1 covers related work, laying the foundation for the study. Section 2 outlines the motivation, followed by system design in Section 3. Section 4 details system implementation, including context input (4.1), language detection (4.2), translation (4.3), question-answer generation (4.4), grading (4.5), and scalability (4.6).

Section 5 discusses the web application, focusing on the framework (5.1) and user interface (5.2). Section 6 covers experimentation, including BERT and T5 (6.1.1), OpenAI models (6.1.2), Deepseek-R1 (6.1.3), and Llama models (6.1.4). It also explores performance improvements such as quantization, retrieval-augmented generation (6.1.5), input randomization (6.1.6), and prompt engineering (6.1.7), followed by testing (6.1.8).

Translation methods are covered in Section 6.2, including Opus-mt models (6.2.1) and fine-tuning Llama 3.1 (6.2.2), followed by answer validation (6.3). Sections 7–9 discuss future work, conclusions, and references.

1. Related Work

Many researchers have focused on the distinct tasks required for generating exam content using Natural Language Processing (NLP) techniques individually. To better understand this process, the problem can be categorized into three key areas.

The first crucial point in creating an exam paper is question-answer generation. A major area of study in question generation is the application of deep learning neural networks. Significant contributions have been made in this area and while some approaches focus on generating questions by extracting information from text documents [5, 6], others emphasize paraphrasing existing content into question formats [7], demonstrating different techniques for automatic question generation based on source material. These contributions have laid the groundwork for automating the question creation process in educational assessments with the emergence of large language models (LLMs) in recent years leading to significant advancements in question answering tasks. NLP systems like BERT, T5 and GPT have played a crucial role in achieving impressive performance by generating questions and retrieving correct answers from large contexts. [8, 9, 10, 11].

To create a comprehensive examination system, the next critical task involves student answer assessment and the subsequent grading based on their responses. This process, simpler compared to question generation, has also gathered significant attention. The primary methods explored for this task include grammar analysis and semantic evaluation, both aimed at assigning a similarity score between the student's response and the correct answer given by the system. By leveraging these techniques, the system can effectively measure how well a student's answer aligns with the expected solution, enabling accurate and consistent grading across different question types. [12]

A critical challenge encountered was the development of an adequate translation system for the Greek language. Unfortunately, many mainstream model implementations lack robust support for Greek and other underrepresented languages, which can result in inaccurate outputs during both question generation and answer evaluation [13, 14]. After consultation with experts in the field, the most effective solution identified was to translate the provided context from Greek to English before feeding it into the large language model (LLM).

There are also papers published supporting research on building comprehensive exam generation systems, though they often rely on different NLP frameworks and lack the integration of the latest Llama, OpenAI or DeepSeek models [30].

In this work, we integrate and extend existing methodologies into a comprehensive system that leverages diverse techniques for QA generation, student grading, assessment explanation, and translation. This unified pipeline delivers a robust, accurate, scalable, and flexible solution for exam creation and automated evaluation. Building on prior research, our end-to-end approach effectively addresses a wide range of educational needs.

2. Motivation

Prior experience in student assessment, signifies that it is evident the process of exam generation can be time consuming and complex, often requiring a balance between subjective interpretation and objective evaluation. Traditional exam creation and grading processes demand considerable time and expertise, often placing an undue burden on teachers and staff, especially when large student populations are involved. Moreover, the creation of diverse, contextually appropriate questions that accurately assess a range of competencies is an inherently complex task, requiring both subject matter expertise and pedagogical experience.

These challenges can be effectively addressed by implementing advanced systems that harness the power of Artificial Intelligence (AI) and Natural Language Processing (NLP), offering a transformative solution by automating the generation of high-quality exam questions and enabling streamlined grading processes. Through AI and NLP technologies, we can develop tools that not only generate diverse and contextually relevant questions but also provide automated, scalable assessment methods, reducing the necessity for direct human intervention in both the creation and evaluation phases.

In recent years, particularly following the COVID-19 pandemic [22], the emergence of remote examinations has significantly altered the methods of student assessment. This shift presents a new challenge for examiners, who are now tasked with developing questions that are both diverse and dynamic in nature. The objective is to reduce the risk of academic dishonesty among students and therefore, exam question design must adapt, incorporating mechanisms to maintain academic integrity while offering a robust and fair assessment of student knowledge; something that can be greatly achieved by using contexts that are large and are analyzed thoroughly.

The implications of these advancements are profound. In fast-paced educational environments, the need for efficient, scalable, and high-quality assessment tools has never been more evident. As student numbers rise and remote learning becomes the norm, educators must balance academic rigor with practical constraints. AI-driven systems ease this drawback by automating repetitive tasks, allowing teachers to focus on meaningful interactions.

Furthermore, in today’s increasingly globalized and diverse classrooms, there are additional complexities to consider. The modern educational landscape, especially in university and academic institutions, often involves multilingual and multicultural settings, where language barriers can impede both teaching and learning. By integrating NLP models that support multilingual capabilities, AI-driven systems can help bridge these communication gaps, ensuring that exam questions and assessments are clear, fair, and accessible to all students, regardless of their native language or cultural background.

3. System Design

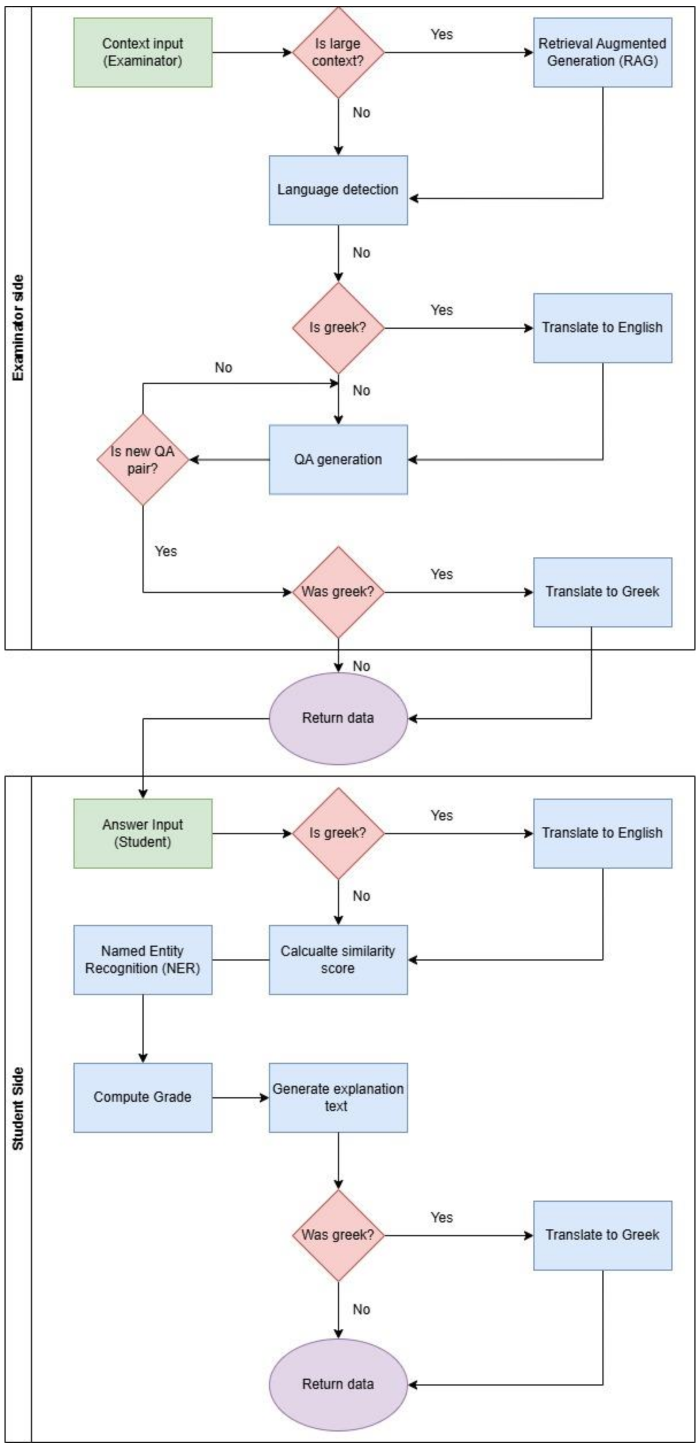


Figure 1: System Structure Chart

The system processes inputs to generate questions, allocate scores, and provide corresponding answers, handling a wide range of contexs from short texts spanning a few hundred characters to full-length documents. For extensive contexts, Retrieval Augmented Generation (RAG) is employed to condense the input into a manageable size, enabling efficient processing and memory management.

To ensure compatibility, it identifies the input language and checks its support within Llama 3.1. Officially supported languages include English, German, French, Italian, Portuguese, Hindi, Spanish, and Thai [16], allowing the system to proceed directly to question generation when these languages are detected. For Greek, which is not officially supported but essential for the Greek educational system, the input is translated into English using two fine-tuned Llama 3.1 model adding further compatibility. After processing, the system translates the output back into Greek, ensuring the end users receive results in their preferred language. This bidirectional translation process maintains both accuracy and compatibility with the model's language support capabilities.

Once the text is processed or translated, it is passed to Llama 3.1 along with well-engineered prompt instructions. The model generates a set of questions and corresponding answers, formatted in a predefined JSON structure for standardization and ease of use and after each question generation a similarity checking step takes place to avoid duplicates or overly similar semantically questions.

When student answers are submitted, the system identifies the response language and if the response is in a supported language, the grading process proceeds directly. For Greek responses, the answers are first translated into a supported language, typically English, before grading begins. The grading process involves several steps: cosine similarity calculation, which computes the similarity between the student's answers and the correct answers generated by the Llama-based QA system; Named Entity Recognition (NER), which identifies critical entities in the student’s response and compares them with the expected answers to avoid terminology biases; and explanation generation, where Llama 3.1 provides a detailed justification for the assessment by incorporating the context, the question, the student’s response, and the correct answer.

Based on the similarity score and detail score computed by the corresponding NER results, the system then assigns a grade based on an algebraic formula. For Greek users, the final grade and explanation are translated back into Greek before being returned, ensuring a seamless experience for end users while maintaining the accuracy of the assessment process.

4. System Implementation

To harness the full potential of natural language processing, it is important to assemble a comprehensive and meticulously configured software environment on the local machine. Such an environment enables the execution of complex computational tasks integral to artificial intelligence applications, particularly those involving large-scale language models.

A critical component of this configuration is the hardware setup. Modern NVIDIA GPUs that support CUDA (Compute Unified Device Architecture) are essential. CUDA provides a parallel computing platform and programming model that significantly accelerates the performance of deep learning tasks by leveraging the massively parallel processing capabilities of GPUs. In contrast, relying solely on CPU processing can result in unacceptably slow execution times, especially when handling the computationally intensive operations characteristic of state-of-the-art NLP models.

Equally important is the software stack. The PyTorch library is a foundational element in this ecosystem, offering dynamic computational graphs and a user-friendly interface that has made it a preferred choice for both researchers and developers. It is crucial to ensure that the installed version of PyTorch is fully compatible with the CUDA version present on the system. This compatibility is vital for maximizing the performance gains provided by GPU acceleration and for preventing potential runtime conflicts that can arise from certain version mismatches.

Python is the programming language of choice due to its simplicity, readability, and the vast ecosystem of libraries that support data science and machine learning workflows. The language's flexibility and the availability of numerous specialized packages make it ideal for development and experimentation in AI research. Among these packages, the Transformers library from Hugging Face stands out for its robust collection of pre-trained NLP models and tools that simplify the deployment of complex language processing pipelines.

Integrating the Transformers library into an environment requires obtaining an account with Hugging Face and securing an access token. This process ensures that access to each model is controlled and that users comply with licensing agreements and usage policies, serving as a credential that authorizes the retrieval of models from the Hugging Face Hub, streamlining the integration process and enabling quick updates and model management.

4.1 Context Input

In the exam question-answer generation system, the main input provided is text, which can range from a few tens or hundreds of characters of plain text to entire documents. When the input consists of shorter contexts, the system automatically inserts the text into a predefined prompt, which is then passed to the Llama-3.1-8B model for question generation. While the

Llama 3.1 model is designed to process up to 128,000 tokens, performance declines significantly when large contexts are provided.

Empirical testing revealed that when the system received a substantial amount of text, the processing time increased drastically, often resulting in memory outages, while for smaller contexts, the system typically generated responses within five minutes. To address this, a solution was implemented that splits the input context based on its size, along with a better quantization configuration setting of 8 bits more suitable to a high-end personal computer. When the input exceeded a certain threshold of characters, the system required the insertion of a text file (such as a book) and a Retrieval Augmented Generation (RAG) technique was applied. This approach enabled users to provide specific keywords, which were semantically searched within the inserted document to further specify the context from which the questions should be generated. RAG would extract relevant, smaller paragraphs that matched these terms and merge them into a single context that is then fed into Llama 3.1, significantly reducing the context size and allowing the system to operate efficiently. By implementing RAG, the performance issues were resolved, ensuring timely responses and preventing CUDA memory errors.

4.2 Language detection

The Llama model, like most widely used large language models, tends to struggle with underrepresented unsupported languages, a limitation that has been widely observed in literature [18]. By default, the model offers robust support for languages such as English, German, French, Italian, Portuguese, Hindi, Spanish, and Thai. However, to ensure effective operation with input data, it is crucial to first identify the language of the provided text and as result this step helps determine whether the language is supported by the model.

To achieve this, Natural Language Processing (NLP) models specialized in language detection are employed to accurately identify the language in which a given passage is written, allowing for an informed decision on whether the language falls within the model's supported set. In this system, the "xlm-roberta-base-language-detection" model [15] was used for this purpose, ensuring precise language identification and improving overall system performance.

4.3 Translation

The translation model enhances the existing capabilities of Llama-3.1-8B by extending support to the Greek language, a functionality previously unavailable in its default configuration. By enabling the translation of Greek text to English, the system ensures compatibility with contexts presented in languages that were not natively supported, broadening its applicability to a wider range of educational systems.

To achieve this, two distinct fine-tuned models were employed: Johnnypjp/Llama-3.1-8b-english-greek-translation-task [23] and Johnnypjp/Llama-3.1-8b-greek-translation-task [24]. These models are specifically designed for bidirectional translation between English and

Greek. The former facilitates translations from English to Greek, while the latter handles translations from Greek to English, an approach that ensures high-quality translations in both directions, by utilizing the needed specialized datasets needed for fine tuning each model.

The translation process is activated only when the language detection model confirms with high confidence that the input text is in Greek. It is applied at two distinct stages: initially, when the input context is provided (Greek to English), and subsequently, after the response generated by the QA model is processed (English to Greek). This two-step process ensures that the system accurately interprets the original context while delivering responses in the intended language.

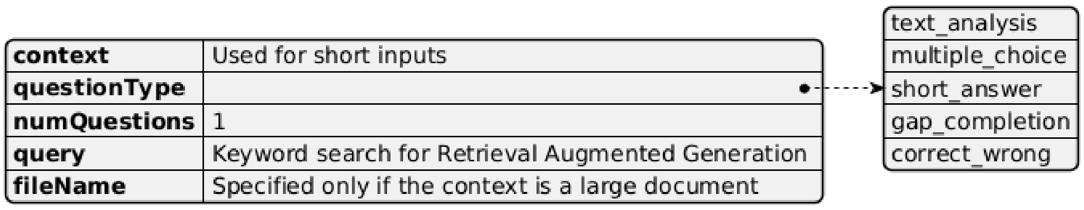
The same translation framework is integrated into the assessment system. Students' answers are processed in the language used for the exam, ensuring consistency in evaluation and enabling the system to support linguistically diverse educational scenarios effectively.

4.4 Question-Answer Generation

In recent years, large language models (LLMs) have excelled in generating question-answer (QA) pairs (25), offering precise and relevant questions along with accurate answers based on provided text. Llama 3.1 supports a context length up to 128,000 tokens, enabling the processing of even extensive texts, such as short books.

In the system, the exam question generation functionality is exposed via a Python Flask API that receives input in the form of a structured JSON, which allows it to interpret the context, generate exam questions and respond with a suitable output. The JSON structure is a critical aspect of this process, as it ensures that the model can accurately receive the necessary input data and generate relevant questions in response.

The required JSON format is carefully designed to accommodate various parameters that dictate the nature of the questions to be generated. The request incorporates configurable fields for defining the input context and it further allows specifying the desired number of questions. When a file is provided, it enables a Retrieval Augmented Generation (RAG) keyword query to focus the question generation process on specific topics while additionally, offering detailed question settings, including type, difficulty level, and assigned individual score, provided for generating diverse and tailored exam questions, ensuring adaptability to various educational needs while maintaining precision and diversity in the generated questions.



context	Used for short inputs
questionType	
numQuestions	1
query	Keyword search for Retrieval Augmented Generation
fileName	Specified only if the context is a large document

text_analysis
multiple_choice
short_answer
gap_completion
correct_wrong

Figure 2: QA API request JSON format

For accurate processing of the given context, specific guidelines play a crucial role, allowing the system to return responses that meet particular requirements. In this instance, the design focuses on ensuring that responses are delivered in a predefined format, maximizing precision so that the output is structured as a valid JSON object containing the fields (as depicted in figure 3) for the generated question, its type, and the corresponding correct answer.

Additionally, the system is designed to handle a variety of question types (essay, multiple choice, short answer, gap completion, true/false) and difficulties (easy, medium, hard), providing the freedom to assign separate marks to each question while maintaining a formal and structured format.

Through extensive analysis of the input and carefully designed instructions, the system generates question-answer (QA) pairs while minimizing repetitive topics by computing a similarity score for each question, which is then compared against a list of previously generated questions to identify and eliminate potential duplicates, thereby ensuring the creation of unique and diverse question sets.

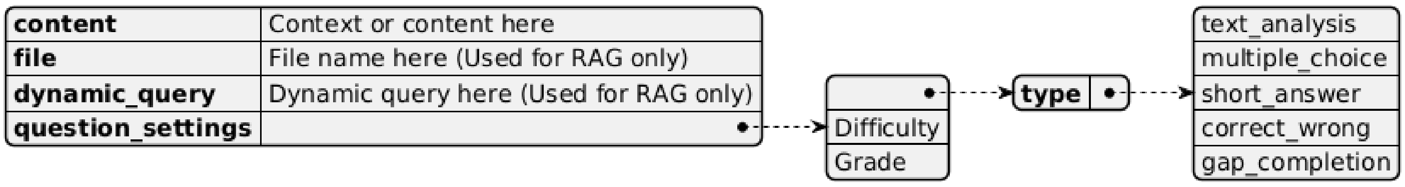


Figure 3:QA API response JSON format

To accurately calculate the cumulative relative mark, depending on the number of questions i given, the following formula was used:

$$g_i = \frac{G}{N}$$

Where:

- *N = Total number of questions*
- *G = Maximum total grade*
- *g_i = Grade for each question i*

4.5 Grading

The grading system employs semantic similarity models, “sentence-transformers/all-mpnet-base-v2” [17] and “FacebookAI/xlm-roberta-large-finetuned-conll03-english” [33] to evaluate student responses. It compares each student’s answer to the correct answer (generated by the Llama model) using a cosine similarity measure that ranges from –1 to 1, with higher values indicating greater semantic resemblance.

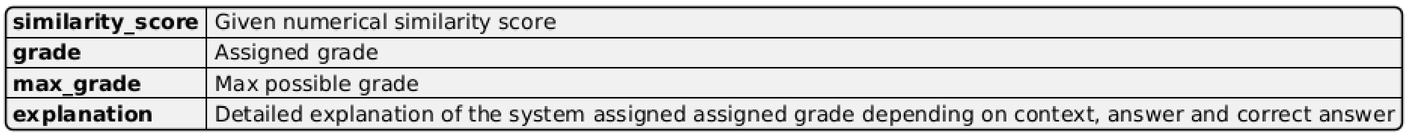
To determine a grade, the system first calculates a base grade by mapping the similarity score onto predefined thresholds. For instance, a similarity score above 0.8 results in full credit (i.e., the maximum grade), while lower scores receive proportionally reduced credit. In addition to this base grade, the system incorporates a detail matching score that reflects how well specific details in the student response align with the correct answer. This detail score is blended with the base grade using a configurable weight (defaulting to 0.3), ensuring that both overall semantic similarity and attention to detail contribute to the final grade. The resulting value is then capped at the maximum possible grade.

$$G(s, M, d, w) = \begin{cases} M \times [(1 - w) + (w \times d)], & if\ s > 0.8 \\ M \times \left[\left(0.8 + 0.2 \times (s - 0.7)\right) \times \left((1 - w) + (w \times d)\right)\right], & if\ 0.7 < s \leq 0.8 \\ M \times \left[\left(0.5 + 0.3 \times (s - 0.5)\right) \times \left((1 - w) + (w \times d)\right)\right], & if\ 0.5 < s \leq 0.7 \\ M \times \left[\left(0.2 + 0.3 \times (s - 0.5)\right) \times \left((1 - w) + (w \times d)\right)\right], & if\ 0.3 < s \leq 0.5 \\ M \times s \times [(1 - w) + (w \times d)], & if\ s \leq 0.3 \end{cases}$$

Where:

- *G is the grade*
- *s is the similarity score (between 0 and 1),*
- *M is the maximum grade.*
- *d is the detail score*
- *w is the detail weight*

The API once again returns the grading parameters in a predefined JSON format. All the needed information for assessment are included in this JSON and are available for printing directly in any application later on.



similarity_score	Given numerical similarity score
grade	Assigned grade
max_grade	Max possible grade
explanation	Detailed explanation of the system assigned assigned grade depending on context, answer and correct answer

Figure 4: Grading API response

4.6 Scalability

To address scalability requirements and accommodate the rapid evolution of artificial intelligence models, the system has been architected with a modular design. This architecture not only facilitates the easy modification of each processing step but also allows for the seamless integration of additional steps as needed. Such flexibility is critical for modern AI systems, where the optimal model selected today may be superseded by a more effective alternative tomorrow.

To achieve this design, the system is decomposed into discrete functions that execute sequentially, ensuring that each function initiates synchronously only after the successful completion of its predecessor. This structure enhances both maintainability and adaptability, allowing for efficient updates or replacements of individual components without necessitating a complete system overhaul.

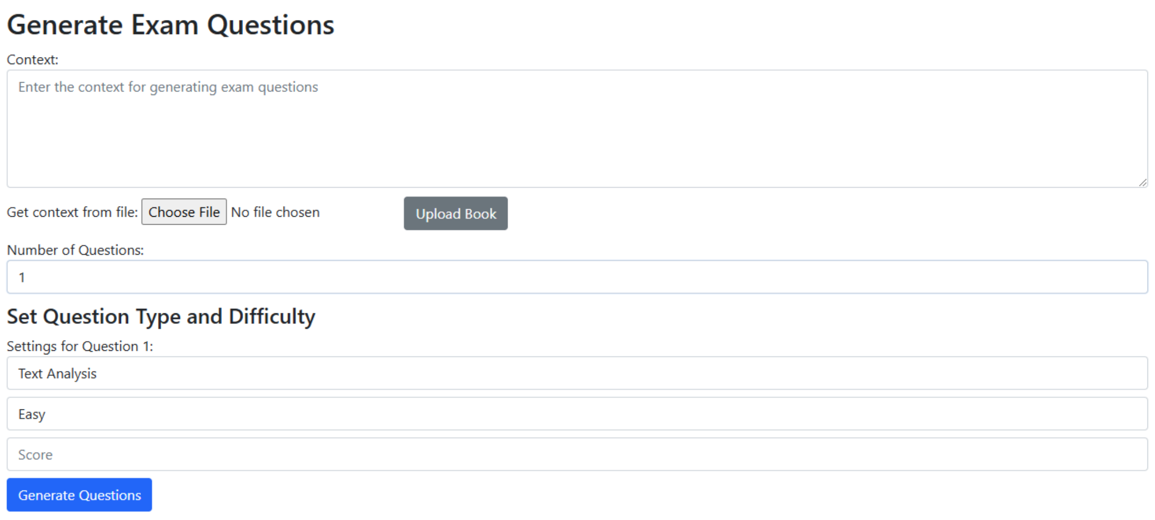
A key factor in achieving scalability was the integration of the Hugging Face Hub Transformers libraries. These libraries offer direct download and plug-and-play capabilities, permitting model changes by simply updating the model identifier. This functionality was rigorously tested during the experimentation phase, enabling the identification of the most capable model for each processing stage and facilitating rapid and seamless transitions between different models.

5. Web application

To facilitate and further test the system, a web application was developed using the latest .NET Core 8 MVC framework. This application serves as the User Interface (UI) for managing interactions between the user and the Flask API, handling both requests and responses. The system's architecture utilizes models, controllers, and services to process data and assist communication between the front and back end components.

5.1 Teacher interface

The UI includes a form that acts as a teacher's interface, providing the input exam context, either as plain text or by uploading a document, the ability to select the question types and specify the number of questions to be generated as well as the individual settings for each question (type, difficulty, grade).


Generate Exam Questions

Context:

Enter the context for generating exam questions

Get context from file:

Number of Questions: 1

Set Question Type and Difficulty

Settings for Question 1:

Text Analysis

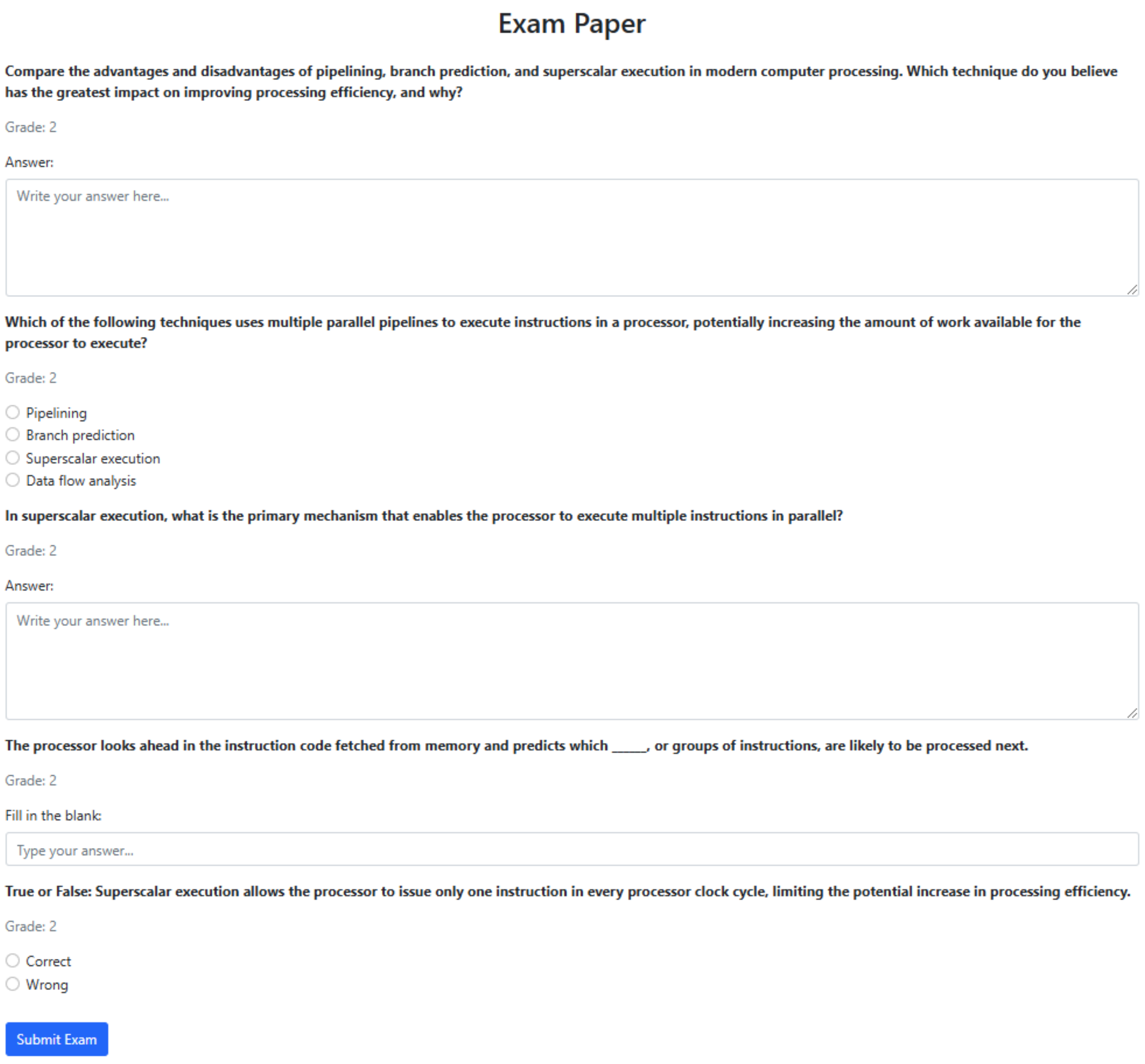
Easy

Score

Figure 5: Exam generation

5.2 Student interface

Upon submitting this information by clicking the "Generate" button, the system creates exam questions which are then displayed within the UI, and depending on their type (e.g., essay, multiple-choice), appropriate HTML elements such as text boxes or radio buttons are dynamically rendered to create the exam paper. There are also basic instructions on how to answer, the grade of each individual questions and an error system, which is used to display appropriate messages in cases where failures might happen. The form is also validated via .NetCore.



Exam Paper

Compare the advantages and disadvantages of pipelining, branch prediction, and superscalar execution in modern computer processing. Which technique do you believe has the greatest impact on improving processing efficiency, and why?

Grade: 2

Answer:

Write your answer here...

Which of the following techniques uses multiple parallel pipelines to execute instructions in a processor, potentially increasing the amount of work available for the processor to execute?

Grade: 2

- Pipelining
- Branch prediction
- Superscalar execution
- Data flow analysis

In superscalar execution, what is the primary mechanism that enables the processor to execute multiple instructions in parallel?

Grade: 2

Answer:

Write your answer here...

The processor looks ahead in the instruction code fetched from memory and predicts which _____, or groups of instructions, are likely to be processed next.

Grade: 2

Fill in the blank:

Type your answer...

True or False: Superscalar execution allows the processor to issue only one instruction in every processor clock cycle, limiting the potential increase in processing efficiency.

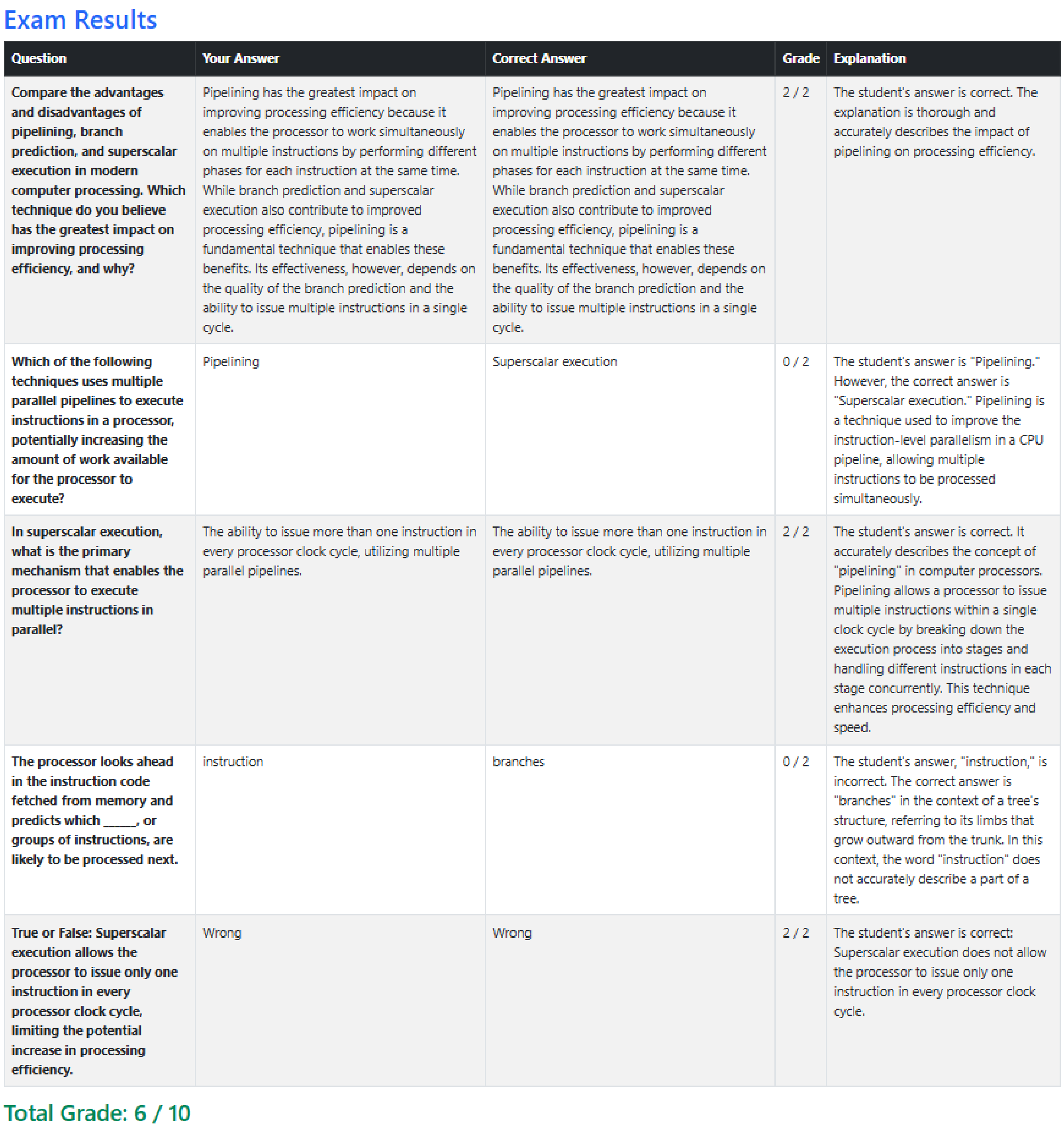
Grade: 2

- Correct
- Wrong

Figure 6: Exam Paper

5.3 Printing the results

Students complete the exam by submitting their answers. After submission, the system automatically processes the responses, evaluates them, and calculates a final grade based on individual scores. It also provides an explanation for each answer and its corresponding grade within the given context.



Exam Results

Question	Your Answer	Correct Answer	Grade	Explanation
Compare the advantages and disadvantages of pipelining, branch prediction, and superscalar execution in modern computer processing. Which technique do you believe has the greatest impact on improving processing efficiency, and why?	Pipelining has the greatest impact on improving processing efficiency because it enables the processor to work simultaneously on multiple instructions by performing different phases for each instruction at the same time. While branch prediction and superscalar execution also contribute to improved processing efficiency, pipelining is a fundamental technique that enables these benefits. Its effectiveness, however, depends on the quality of the branch prediction and the ability to issue multiple instructions in a single cycle.	Pipelining has the greatest impact on improving processing efficiency because it enables the processor to work simultaneously on multiple instructions by performing different phases for each instruction at the same time. While branch prediction and superscalar execution also contribute to improved processing efficiency, pipelining is a fundamental technique that enables these benefits. Its effectiveness, however, depends on the quality of the branch prediction and the ability to issue multiple instructions in a single cycle.	2 / 2	The student's answer is correct. The explanation is thorough and accurately describes the impact of pipelining on processing efficiency.
Which of the following techniques uses multiple parallel pipelines to execute instructions in a processor, potentially increasing the amount of work available for the processor to execute?	Pipelining	Superscalar execution	0 / 2	The student's answer is "Pipelining." However, the correct answer is "Superscalar execution." Pipelining is a technique used to improve the instruction-level parallelism in a CPU pipeline, allowing multiple instructions to be processed simultaneously.
In superscalar execution, what is the primary mechanism that enables the processor to execute multiple instructions in parallel?	The ability to issue more than one instruction in every processor clock cycle, utilizing multiple parallel pipelines.	The ability to issue more than one instruction in every processor clock cycle, utilizing multiple parallel pipelines.	2 / 2	The student's answer is correct. It accurately describes the concept of "pipelining" in computer processors. Pipelining allows a processor to issue multiple instructions within a single clock cycle by breaking down the execution process into stages and handling different instructions in each stage concurrently. This technique enhances processing efficiency and speed.
The processor looks ahead in the instruction code fetched from memory and predicts which _____, or groups of instructions, are likely to be processed next.	instruction	branches	0 / 2	The student's answer, "instruction," is incorrect. The correct answer is "branches" in the context of a tree's structure, referring to its limbs that grow outward from the trunk. In this context, the word "instruction" does not accurately describe a part of a tree.
True or False: Superscalar execution allows the processor to issue only one instruction in every processor clock cycle, limiting the potential increase in processing efficiency.	Wrong	Wrong	2 / 2	The student's answer is correct: Superscalar execution does not allow the processor to issue only one instruction in every processor clock cycle.

Total Grade: 6 / 10

Figure 7: Exam results

6. Experimentation

To achieve the optimal solution, each component of the system underwent extensive and rigorous testing with a variety of techniques and models to identify the most effective approach. Furthermore, metrics were systematically calculated for each part, to justify the selection of the best solution combining mathematical data with empirical analysis, ensuring that both theoretical evidence and practical performance were considered in determining the optimal method. Areas of experimentation followed the pattern of the system structure and were separated into QA generation, language detection and translation, as well as student assessment and grading techniques.

6.1 Question – Answer systems

To explore and determine the optimal solution for the question-answer generation subsystem, various NLP models were tested, each offering distinct strengths and limitations. The primary focus of the experimentation was centered on evaluating performance, cost-efficiency, capacity for handling large contexts, language support and adaptability. These axes were critical in assessing how each model contributed to the goal of generating high quality exam questions while balancing computational efficiency and scalability.

6.1.1 BERT and T5

In testing, BERT [39] demonstrated strong performance in providing accurate answers to specific tasks, with its transformer-based architecture excelling at capturing intricate linguistic patterns, making it well-suited for question-answering tasks. However, a notable limitation of BERT was its restricted context length, capped at 512 tokens, significantly limiting its practicality in applications requiring extensive input processing, such as analyzing large texts or generating questions from broader contexts.

The T5 (Text-to-Text Transfer Transformer) [19] model exhibited similar impressive capabilities in language understanding and generation, with its unified text-to-text framework enabling seamless handling of various NLP tasks and marking a significant advancement in transfer learning. Pre-trained on vast datasets and fine-tuned for downstream tasks, T5 delivers robust performance across diverse applications, including question-answer generation, where the fine-tuned model variant "mrm8488/t5-base-finetuned-question-generation-ap" [20], trained on the SQUAD v1.1 dataset [21], effectively produced high-quality questions and answers. Despite these strengths, T5, much like BERT, was hindered by a token limit of 512, which significantly restricted its ability to process large bodies of text.

The shared limitation between BERT and T5 had significant implications for the system's requirements, as both models struggled to handle extensive contexts—a critical necessity for generating exam questions from large datasets. Even with enhancements like Retrieval Augmented Generation (RAG), the token limit of these models remained a bottleneck and while they excelled in their respective strengths, their inability to effectively manage large inputs ultimately undermined their viability for this application.

Context splitting

A potential solution to address the limited context length offered by the T5 and BERT models was to segment the context into smaller token batches and sequentially input them into the model. While this approach theoretically mitigated the context length constraint, it proved ineffective for larger contexts, as the model tended to become overloaded. Additionally, not all segments contained semantically rich or informative content sufficient for generating meaningful questions, resulting in an inefficient distribution of the input context.

6.1.2 Open-AI models

Open-AI offers a great variety of models suited to different needs, while providing a high level of accuracy, excellent language support and ease of use. Unfortunately, the lack of local system support and the expensive costs associated with their APIs, make them unaffordable for researchers and institutions that are not willing to pay for a dedicated exam generation system.

GPT 4

GPT- 4 has a context length of 32,000 tokens, while the turbo version offers a significant increase in context length making the model capable of processing context lengths up to 128,000 tokens [36, 37]. The model also excels in performance across a wide range of general tasks and is well suited for tasks where high level of inference or creativity is required.

Even though the turbo version is more than capable of being used instead of Llama for this part of the system, the primary limitation with OpenAI's GPT models stems from their substantial size, which exceeds the capacity of ordinary personal computers. Consequently, these models are not open source and only accessible through OpenAI’s API, a paid service that presents challenges for applications involving substantial amounts of contextual text or numerous requests. Retrieval Augmented Generation (RAG) can help by reducing the required context length and potentially enabling the inclusion of extensive material, but this approach remains costly and demands careful risk assessment to determine its viability for large scale systems. Furthermore, GPT-4 models have limited flexibility in fine-tuning, as they are provided in a fully pre-trained state and are not readily adaptable by third-party developers for custom tuning.

The results from the ROUGE metric calculations reveal the following insights about the generated questions. For ROUGE-1, the average precision was calculated to be 4.43%, reflecting the degree of overlap in unigrams between the generated and reference text. The average recall was substantially higher at 66.73%, indicating a strong ability to capture relevant unigrams from the reference text. The F-measure for ROUGE-1, which balances precision and recall, was relatively low at 8.4%, suggesting room for improvement in both precision and recall achieving a better balance.

For ROUGE-L, which considers the longest common subsequences, the precision averaged 3.82%, while the recall stood at 57.12%. This signifies a moderate ability to maintain the sequence of reference text. The F-measure was 7.16%, highlighting similar trends as in ROUGE-1. Finally, the semantic similarity between the generated and reference text averaged 79.95%, indicating a reasonably high conceptual alignment, despite the low ROUGE scores.

O1

GPT O1 distinguishes itself through an optimized architecture that emphasizes enhanced contextual understanding and an extended memory capacity, allowing it to manage longer and more complex interactions effectively. Its design prioritizes efficient multi-step reasoning and precise content generation, ensuring that even intricate queries are addressed with clarity and depth.

Just like its predecessor, O1 is constrained by prohibitively high operational costs that make it unsuitable for low-cost, small-scale applications such as exam creation systems. The testing and implementation of its pipeline alone requires a comprehensive cost analysis to ensure budget compatibility, while the model's demand for advanced, specialized hardware and significant maintenance expenses further complicate its deployment in resource-limited environments. Additionally, the architectural complexity of O1 necessitates specialized expertise and results in prolonged development cycles, and when scaling the system, additional computational and infrastructural investments become necessary. The reliance on high-quality, extensive datasets introduces further financial and logistical challenges, compounding the overall cost and complexity of the system for small local applications.

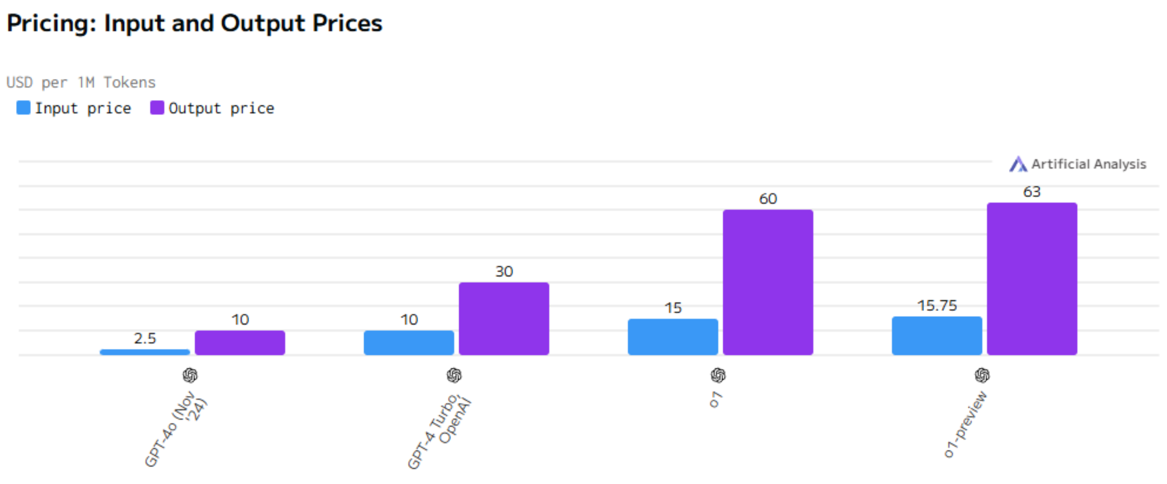


Figure 8: Pricing costs of individual OpenAI models

6.1.3 Deepseek-R1

DeepSeek has introduced its first-generation reasoning models, DeepSeek-R1-Zero and DeepSeek-R1, utilizing reinforcement learning to achieve advanced reasoning capabilities. Notably, DeepSeek-R1 integrates cold-start data to enhance performance and address inherent limitations, ultimately matching or surpassing state-of-the-art benchmarks in reasoning, mathematics, and coding tasks. [40]

Empirical and mathematical evaluations demonstrate that the model delivers performance comparable to OpenAI's O1 model at merely one-eighth of the cost, albeit with a slightly reduced token processing speed. The model features a context length of 128,000 tokens, akin to LLaMA 3.1, enabling efficient processing of extensive textual data such as large books. However, the base model requires substantial computational resources, rendering it impractical for standard personal computers. Consequently, an 8-billion-parameter distilled variant derived from LLaMA 3.1 was tested as a more accessible alternative. [41]

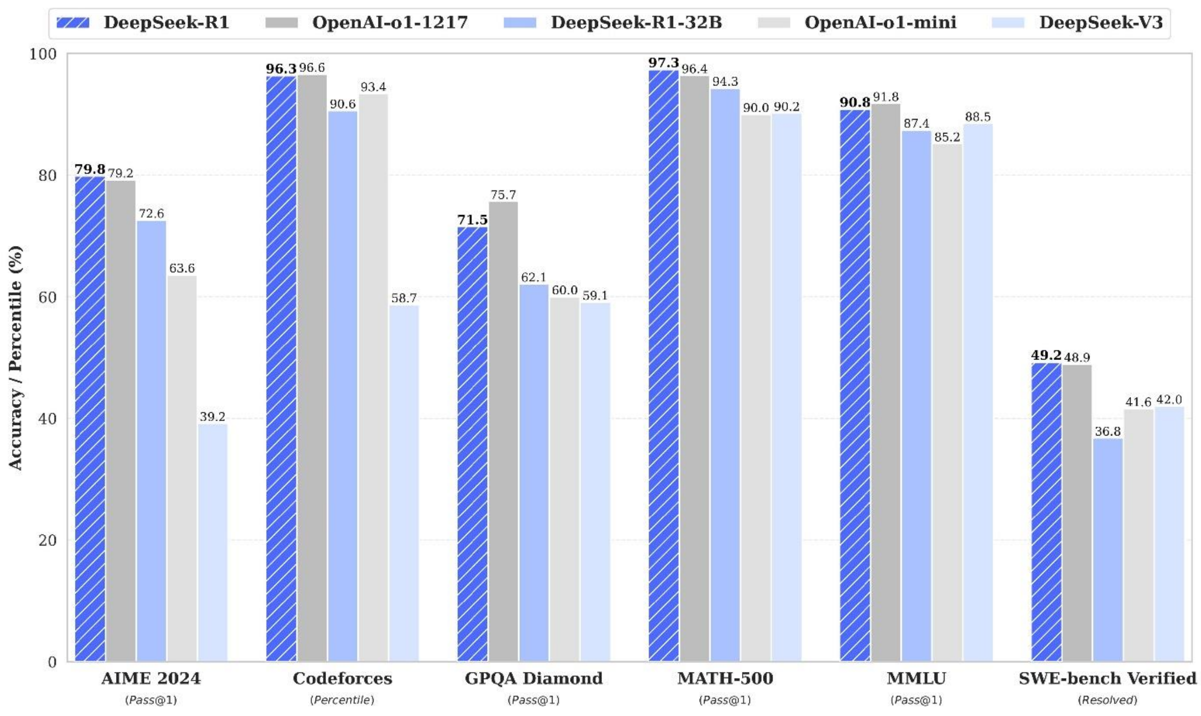


Figure 9: DeepSeek-R1 model performance comparison

DeepSeek-R1-Distill-Llama-8B

DeepSeek R1 distilled from Llama 8B [42] is a great alternative for using the DeepSeek model in a smaller scale, able to run on local computers with consumer-grade hardware. Distillation involves developing smaller, more efficient models from larger ones, maintaining most of their reasoning ability while lowering computational requirements. DeepSeek managed to produce a series of distilled, leveraging smaller parameter size Qwen and Llama architecture.

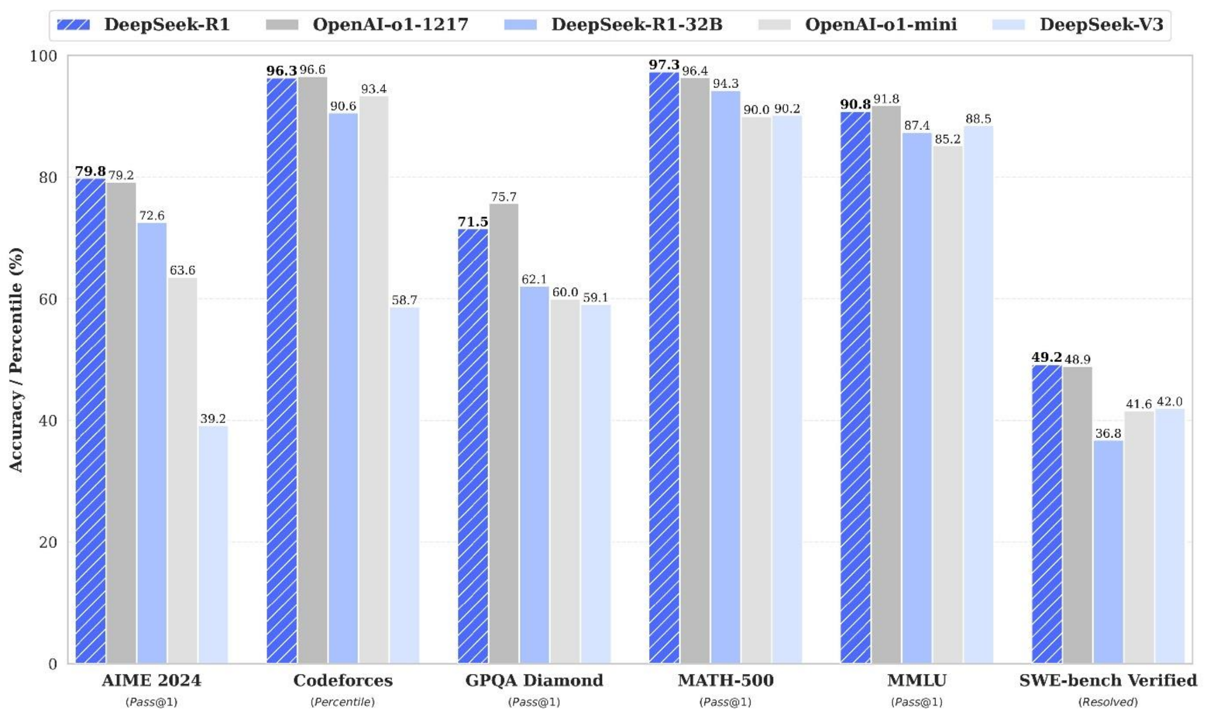


Figure 10: DeepSeek-R1-Distill-Llama-8B model performance comparison

The primary version, DeepSeek-R1-Distill-Llama-8B, retains a context window of 128,000 tokens while delivering performance comparable to that of Llama 3.1 8B. Comprehensive evaluations of various quantization techniques have demonstrated that the 8-bit integer configuration yields optimal performance on high-end personal computer systems.

Empirical testing further indicates that, with an appropriately structured prompt, DeepSeek is capable of accurately interpreting input queries and generating improved responses. The model's ability to provide reasoning and explanatory details during its inference process has proven particularly valuable. This feature aids in diagnosing misunderstandings inherent to the task, thereby facilitating more effective prompt fine-tuning.

Additionally, DeepSeek exhibits efficient resource management, notably consuming less VRAM when processing shorter context inputs, while maintaining performance on par with Llama 3.1 for longer contexts. This efficiency not only benefits the processing of small plain text contexts but also contributes to the overall scalability of the system, allowing the use of heavier models running parallel with DeepSeek.

Finally, the system's capacity to comprehend previously generated content reduces the occurrence of duplicate questions, thereby streamlining the generation process and enhancing overall performance.

Despite these benefits, a major drawback that rendered the model impractical is its limited language support, with only English and Chinese having sufficient results. This limitation greatly reduces its practicality regarding multilingual education environments

6.1.4 Llama

Meta's Llama models provide a great open-source alternative to proprietary large language models such as OpenAI's GPT and DeepSeek series. The Llama family of models, notably designed local implementation, offers extensive contextual windows which cater effectively to systems that process substantial and complex input.

The open-source nature of Llama models ensures they are not only freely accessible but also highly adaptable for scientific and non-commercial applications, removing licensing constraints that often accompany commercial models. By using Meta's Llama, this system leverages both the flexibility and scalability required for dynamic exam generation on a high-end personal computer.

A variety of publicly available Llama model versions offer distinct capabilities, necessitating a systematic evaluation to identify the one best suited for optimal exam question generation.

Llama *2-7B*

The Llama 2-7B model was not particularly suitable for this application simply by the fact it had a relatively insufficient sequence length of 4096 tokens [34]. Furthermore, empirical testing of the model as well as performance metrics showed that it struggled to always provide the desired results.

The Llama-2-7B model required five iterative revisions to sufficiently understand the context and generate questions that aligned with the provided material. However, the generated questions exhibited relatively low precision, as reflected in a ROUGE-1 precision of 14.97% and a ROUGE-L precision of 20.75%, while the recall was considerably higher at 53.57%. This suggests that the model extracted large portions of the text verbatim rather than rephrasing or generating novel content. Additionally, the cosine similarity score of 50.62% indicates only moderate semantic alignment with the original context. These results highlight that while the model could identify relevant terms, the lack of specificity and depth in question formulation limited its effectiveness for exam generation.

Llama 3-8B

Llama 3 [35] comes with a sequence length of 8,192 tokens and even though again limited in its length, the performance of the model was evaluated to assess the quality differences with the Llama 2-7b version.

The Llama-3-8B model demonstrated improved question generation capabilities compared to Llama-2-7B. Its ROUGE-1 precision of 51.50% and recall of 35.87% indicate that it effectively captures a substantial portion of relevant terms while maintaining moderate precision. The ROUGE-L precision (25.51%) and recall (16.61%) suggest that while the generated questions align structurally with the context, precision decreases for longer sequences. Additionally, the cosine similarity score of 68.86% reflects stronger semantic alignment, signifying that the model produces more coherent and contextually relevant questions. These results highlight

Llama-3-8B’s enhanced ability to generate well-formed and meaningful exam questions compared to its predecessor.

Llama 3.1-8B

During system development, the release of Llama 3.1 marked a significant advancement over the previous Llama 3 model, particularly through its extended context length, now reaching 128,000 tokens. This increase enables Llama 3.1 to effectively manage extensive texts required for exam question generation. Additionally, performance assessments indicate notable enhancements in both semantic and contextual understanding over its predecessor. There is also the benefit of using 8 billion parameters, which means that with the utilization of the correct quantization depending on the system, the model can run on high-end personal computers.

In terms of semantic alignment, Llama-3.1-8B demonstrates a notable improvement, achieving a cosine similarity of 78.57%, surpassing Llama-3-8B’s 68.86%. This increase suggests that Llama-3.1-8B more effectively captures the underlying meaning within a given context. Additionally, ROUGE-1 precision (48.01%) and recall (56.15%) show an increase over Llama-3-8B, indicating stronger term retrieval while maintaining contextual accuracy, while at the same time, the ROUGE-L precision (30.68%) and recall (35.88%) further reinforce this improvement in structural alignment. These results highlight that Llama-3.1-8B not only captures nuanced meaning more effectively but also enhances text alignment, making it particularly well-suited for exam question generation where semantic validity and accurate paraphrasing are crucial.

Llama 3.2–3B

Meta introduced a new iteration of its Llama model series, the Llama 3.2, engineered to provide a smaller, faster solution while maintaining both high context length capabilities and robust performance. Compared to its predecessor, Llama 3.1, the new model achieves a notable improvement in size and memory allocation, reflecting its balance between semantic accuracy and computational efficiency.

Performance metrics indicate that Llama-3.2-3B achieved a ROUGE-1 precision of 49.03% and recall of 50.17%, demonstrating a balanced ability to retrieve relevant terms while maintaining contextual coherence. Compared to Llama-3.1-8B, which had 48.01% precision and 51.76% recall, Llama-3.2-3B exhibits a slight precision advantage but a minor recall trade-off. Additionally, ROUGE-L precision (23.70%) and recall (24.25%) reflect a moderate structural alignment, though slightly lower than Llama-3.1-8B’s 30.68% precision and 35.88% recall.

In terms of semantic alignment, Llama-3.2-3B achieves a cosine similarity score of 71.81%, lower than Llama-3.1-8B’s 78.57%, suggesting that while Llama-3.2-3B maintains solid structural accuracy, it may not capture semantic depth as effectively. This positions Llama-3.2-3B as a model optimized for generating well-structured questions with a balance between precision and recall, while Llama-3.1-8B excels in deeper contextual understanding, making it potentially more suitable for exam question generation where nuanced meaning is critical.

Given Llama 3.2’s optimization for smaller systems and not providing the best precission, this model was not selected as the primary model for the exam question-answer generation system, as the main focus was to provide the best quality of generative text.

Final Verdict

Considering all these factors, Llama 3.1 was selected as the most suitable model and even though it may lack in certain areas compared to DeepSeek R1, as proven by the calculated metrics and empirical testing, the broader language support offered by the model made it a more practical choice for generating the questions and answers.

Table 1: Model performance metrics

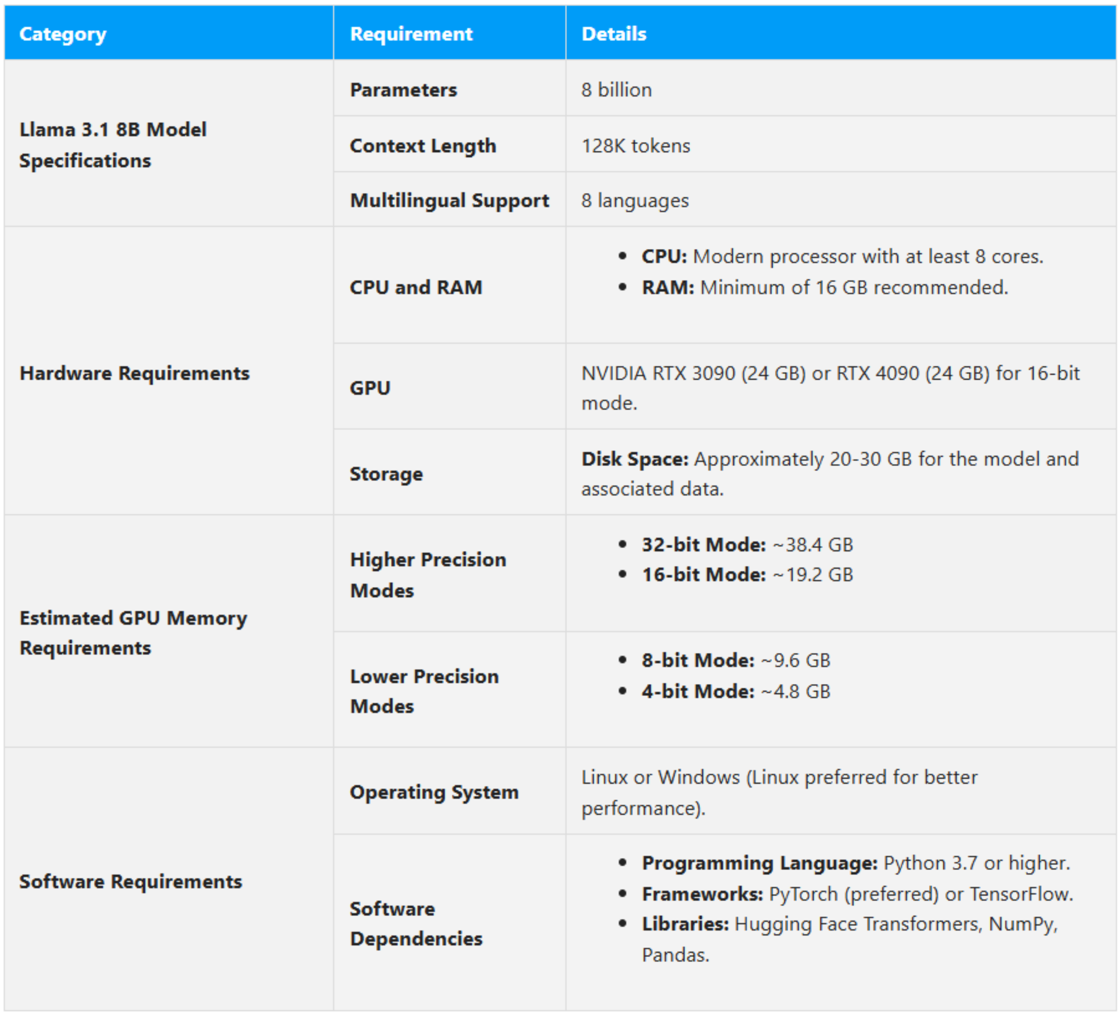
Model	ROUGE-1 Precision	ROUGE-1 Recall	ROUGE-1 F-measure	ROUGE-L Precision	ROUGE-L Recall	ROUGE-L F-measure	Cosine Similarity (%)
Llama 2-7B	14.97%	53.57%	9.09%	20.75%	10.22%	15.36%	50.62%
Llama-3-8B	51.50%	35.87%	40.00%	25.51%	16.61%	20.12%	68.86%
Llama-3.1-8B	48.01%	56.15%	51.76%	30.68%	35.88%	33.08%	78.57%
Llama-3.2-3B	49.03%	50.17%	49.59%	23.70%	24.25%	23.97%	71.81%
GPT-4o	4.4%	66.73%	8.4%	3.82%	57.12%	7.16%	79.95%
DeepSeek-R1-Distill-Llama-8B	34.29%	79.40%	47.9%	15.93%	36.88%	22.24%	82.26%

Performance improvements

While the system utilizing the Llama-3.1-8B model achieved satisfactory performance in the default setting specified by Llama, a significant issue persisted in the extensive time required for generating relevant exam questions and answers. For short contexts, the average processing time was approximately five minutes, while for longer contexts, the time increased exponentially, often resulting in CUDA memory limitations.

Quantization

To further optimize processing efficiency without compromising quality, various quantization techniques were assessed. Initial trials with 4-bit quantization (Q4) substantially reduced generation time, as this adjustment proved to reduce the processing time for shorter and larger contexts with a compromise in output reliability and prompt understanding. To further increase generation precision, the 8-bit quantization mode (Q8) was tested with optimal results, increasing the quality of exam QA pairs with a minimum increase on processing time. Unfortunately, the 16GB of VRAM provided by a single Nvidia RTX 4080 super setup, was not sufficient to enough to allow the increase of quantization mode. The processing of tokens per second by the system was very slow for Llama-3.1-8B in 16-bit mode (FP16), because the model was constantly being offloaded to the CPU to counter the shortage of GPU memory and therefore the time required for analyzing the context and generating the QA pairs was increased drastically and often causing CUDA memory outages. As result, 8-bit quantization was proved to be the optimal solution, considering also the system requirements as calculated by Meta. [38]



Category	Requirement	Details
Llama 3.1 8B Model Specifications	**Parameters**	8 billion
	Context Length	128K tokens
	Multilingual Support	8 languages
Hardware Requirements	**CPU and RAM**	• **CPU:** Modern processor with at least 8 cores. • **RAM:** Minimum of 16 GB recommended.
	GPU	NVIDIA RTX 3090 (24 GB) or RTX 4090 (24 GB) for 16-bit mode.
	Storage	**Disk Space:** Approximately 20-30 GB for the model and associated data.
Estimated GPU Memory Requirements	**Higher Precision Modes**	• **32-bit Mode:** ~38.4 GB • **16-bit Mode:** ~19.2 GB
	Lower Precision Modes	• **8-bit Mode:** ~9.6 GB • **4-bit Mode:** ~4.8 GB
Software Requirements	**Operating System**	Linux or Windows (Linux preferred for better performance).
	Software Dependencies	• **Programming Language:** Python 3.7 or higher. • **Frameworks:** PyTorch (preferred) or TensorFlow. • **Libraries:** Hugging Face Transformers, NumPy, Pandas.

Figure 11: Llama 3.1 8B system specification and requirements

Table 2: Quantization performance

Bit Configuration	Tokens Generated	Time Taken (seconds)	Tokens per Second	Notes
4-bit	84	4.09	20.52	Successful execution
8-bit	103	20.11	5.12	Successful execution, better precision, more time to complete
16-bit	N/A	N/A	N/A	Ran out of GPU VRAM

6.1.5 Retrieval Augmented Generation

To address the challenges of extensive context and to further optimize memory management, Retrieval-Augmented Generation (RAG) was implemented to condense the context by selectively prioritizing relevant content, becoming available when input contexts exceed 2,500 characters and allowing the system to manage larger contexts more effectively by focusing the model's context window on essential information. Minor preprocessing was also added to remove whitespaces found on different structures of the book (e.g. paragraphs, different chapters etc.).

6.1.6 Randomizing the input

A significant issue encountered during the development of the QA generation system was the repetitive generation of identical questions by Llama 3.1 from the provided context. This problem arose due to the system's reliance on a narrow selection of top tokens extracted from the context, which limited the diversity of the input data and consequently, set the model to generate questions with similar meaning in every iteration. As these repeated questions were flagged and rejected by the semantic similarity model, the system entered an infinite loop, continuously producing and declining the same questions. This redundancy highlighted a critical flaw in the context processing approach, where insufficient variation in the input constrained the model's ability to generate diverse and meaningful outputs.

To address this issue, additional parameters were introduced during the generation phase to enhance Llama 3.1's creativity when producing input. Parameters such as temperature and Top-K were fine-tuned to balance diversity and accuracy, as excessively high values led to over-generalization by the model.

6.1.7 Prompt Engineering

Prompt engineering serves as the foundational framework for ensuring the efficacy, precision, and pedagogical value of the automated question-generation system. Meticulously crafted prompts carefully designed through trial and error are essential for producing contextually relevant, instructionally sound, and technically robust outputs.

Firstly, each prompt template explicitly defines task requirements to ensure alignment with pedagogical goals. For example, for essay questions the prompt mandates analysis of cause-effect relationships or theoretical implications, steering the system away from superficial inquiries. Multiple-choice templates enforce the creation of plausible distractors rooted in common errors, ensuring diagnostic validity, while coding exercises require escaped code formatting and success criteria to balance technical rigor with usability. By embedding granular directives, such as avoiding duplicates or including context-specific code snippets prompts eliminate ambiguity and constrain outputs to the desired scope.

Parameters such as Cutting Knowledge Date and Today Date ensure temporal relevance, critical in domains like programming where outdated practices may mislead learners and produce deprecated results. Difficulty levels tailor questions to target audiences, while duplication checks guarantee a diverse question bank, enhancing assessment reliability, further ensuring the system remains responsive to evolving educational needs and content updates.

Strict JSON response formats are succeeded only by implementing the correct prompt instructions and standardizing outputs for integration into downstream applications, achieved by providing the system with examples of structured JSON objects, ensuring uniformity and interoperability with downstream tasks and reduced post-processing overhead.

Furthermore, the modular prompt design supports extensibility and is the core of the system's ability to handle contexts from different topics. Coding exercise templates, for example, could be adapted for multiple programming languages by modifying the structure and comparative analysis essay prompts could be repurposed for humanities or scientific contexts by updating the context variable, future-proofing the system against emerging educational demands and enabling multidisciplinary exam generation.

6.1.8 Testing

At the conclusion of the system's development cycle, an extensive and meticulously planned testing phase was initiated. During this period, the program was rigorously evaluated by a diverse group of users drawn from various professional backgrounds. These evaluators, each representing distinct areas of expertise, were tasked with thoroughly testing the system's functionalities and providing comprehensive feedback on its performance, usability, and overall effectiveness by completing a questionnaire. The overarching goal of this phase was to ensure that the system met user expectations, both by the teacher and student sides.

The results of this evaluation were overwhelmingly positive, with overall user satisfaction being reported as very high. A notable strength of the system was its robust ability to generate contextually relevant questions, demonstrated through the consistent production of questions that were not only appropriate to the context but also varied across multiple question types. The system's design allowed it to seamlessly transition between different forms of questioning, ensuring that a wide spectrum of assessment scenarios could be accommodated from standard multiple-choice queries to more intricate open-ended formats.

In addition to its question generation capabilities, the system demonstrated a high degree of precision in identifying correct answers. Evaluators observed that, in most cases, the system accurately pinpointed the correct responses, thereby reinforcing its utility as a reliable assessment tool. Furthermore, the system provided detailed feedback that enabled users to gain a deeper understanding of their assessments. This feedback mechanism not only clarified the rationale behind each evaluation but also served as a valuable learning aid for users, thereby enhancing the overall educational value of the system.

Another noteworthy feature of the system was its method of partial grading. In instances where responses were partially correct, the system was able to allocate partial credit. This nuanced approach to grading was particularly beneficial in contexts and open-ended question types where binary right-or-wrong assessments might have been overly simplistic.

Users also highlighted the practical benefits of the system in terms of time efficiency. Many reported that the system could significantly reduce the time they spent on routine exam building tasks, a feature that they found to be especially valuable in their daily professional activities. Considering these benefits, a substantial number of users expressed their intention to integrate the system into their regular workflows and were enthusiastic about recommending it to colleagues across various fields.

Despite these strengths, users identified certain areas where improvements were necessary. One specific issue that emerged during testing was the occurrence of repetitive content in the generated question-answer pairs. These instances of redundancy, although infrequent, were significant enough to warrant attention. Subsequent investigations revealed that the unexpected repetitions were linked to the functioning of the Llama component within the system and limitations of context produced by RAG. This discovery prompted a series of targeted interventions aimed at resolving the issue, ensuring that future iterations would maintain a higher standard of content diversity and quality.

During the evaluation, an additional limitation was identified: the system initially offered a limited variety of question types, particularly those focusing on the practical applications of contextual information. To address this shortfall, a new category of questions specifically designed to encompass coding exercises was introduced. This new question type was complemented by enhancements to the existing prompts, which were enriched with detailed instructions aimed at fostering practical application skills and promoting critical thinking. As a result of these targeted improvements, the system now possesses the capability to detect topics related to programming and to generate corresponding coding exercises. Simultaneously, the traditional question types have been refined to deliver more practical outcomes, such as computing mathematical or programming problems and results in multiple-choice formats and thus providing users with a more robust and application-oriented assessment tool.

6.2 Translation

Translating the provided context into English or Greek posed a unique challenge. Specifically, certain terms within the context, often domain-specific or technical, were not intended to be translated and maintaining the integrity of such terms in their original language was crucial for ensuring both linguistic and semantic accuracy.

6.2.1 Opus-mt-en-el and Opus-mt-grk-en

The initial model assessed for this translation task was Helsinki-NLP/opus-mt-en-el [26]. Empirical evaluations demonstrated that, while the model effectively translated content into Greek in a manner that was comprehensible to native speakers, its performance revealed a critical shortcoming. Specifically, although the English translations generated by the model were sufficiently accurate for subsequent processing by Llama, a persistent issue was observed: the model frequently translated technical terms and domain-specific terminology that should have remained in English, thereby altering the original context. This inconsistency in terminology translation reduced both the accuracy and utility of the translated output for downstream tasks requiring precise technical language.

Quantitative evaluation using the BLEU metric yielded a score of 64%, reflecting a moderate alignment with reference translations and while this metric indicates a reasonable degree of fidelity in translation, the inconsistencies in preserving technical terms underscore the need for further model fine-tuning or the incorporation of specialized post-processing steps, further complicating the system structure.

A subsequent evaluation was conducted using opus-mt-grk-en [28], the Greek-to-English counterpart of the initial model. Although this model exhibited better overall performance compared to opus-mt-en-el, it still encountered challenges with terminology translation. The BLEU score for this model was 61%, indicating moderate alignment between its outputs and reference sentences and while it generally preserved the structure and meaning of the source text, inconsistencies in translating key technical terms were again evident. Such errors compromise the precision and clarity required for technical translations, particularly in specialized domains that exist in higher education.

6.2.2 Fine tuning Llama 3.1

To address this issue, the idea of using Helsinki-NLP/opus-mt-en-el and opus-mt-grk-en was scraped and the new solution involved fine-tuning Llama 3.1 to enhance its ability to manage Greek-to-English and English-to-Greek translations while adhering to a specific guideline that scientific terminology should not be translated. This fine-tuning process required a robust and well labeled dataset to enable supervised training, ensuring the system could accurately comprehend and translate the given text. Llama was once again chosen due to its ability to process large contexts and to limit the number of diverse models used in the system. The sub-dataset selected for this purpose was Helsinki-NLP/europarl/el-en [27], which comprises 1.29 million conversations from the European Parliament, offering high-quality translations between Greek and English.

The training process posed a distinct challenge. Training the Llama 3.1-8B model proved difficult on standard computing systems (AMD Ryzen 5 7600x, Nvidia RTX 4080 super, 32gb DDR5 RAM) and even on Google Colab's specialized environment (T4 GPU, high RAM configuration). Both setups frequently encountered CUDA memory limitations, resulting in repeated interruptions and an inability to complete the training process. To overcome this obstacle, the Alpaca Unsloth fine-tuning utility was employed [29]. This tool provided a streamlined and resource efficient approach to training, enabling the process to complete successfully and consequently, two specialized translation models for Greek-to-English and English-to-Greek were developed [23, 24].

Empirical evaluations of the fine-tuned models yielded promising results. Both systems effectively provided accurate translations for both languages while preserving most of the context-specific terminology in its original form.

In this analysis, two distinct BLEU scores were obtained: 84% for the English to Greek system and 67% for the Greek to English. The score of 84% indicates a high degree of overlap between the predicted and reference translations, suggesting that the model closely approximates human translation for this specific instance. On the other hand, the score of 67%, while lower, still demonstrates an acceptable level of alignment between the prediction and references.

Table 3: Blue scores of individual models

Model	Blue Scores	Notes
Helsinki-NLP/opus-mt-en-el	64%	Demonstrated moderate translation accuracy but frequently mistranslated domain-specific terminology, reducing its suitability for technical tasks.

Helsinki-NLP/ opus-mt-grk-en	61%	Exhibited terminology inconsistencies, compromising technical precision.
Fine-tuned Llama 3.1 (English-Greek)	84%	Achieved high translation accuracy, effectively preserving scientific and domain-specific terminology while closely approximating human translation quality.
Fine-tuned Llama 3.1 (Greek – English)	67%	Delivered acceptable translation accuracy, preserving context and terminology, with minor mistakes.

6.3 Answer validation

In the answer validation subsystem, the primary objective is to measure the semantic divergence between a model-provided correct answer and the corresponding response given by a student. This process is essential to accurately evaluate the student's understanding relative to the expected answer. To achieve this goal, two cosine similarity models from the sentence transformer family were tested sentence-transformers/all-MiniLM-L6-v2 [31] and sentence-transformers/all-mpnet-base-v2. By leveraging these models, the system quantifies semantic alignment, offering an effective means for comparative assessment in student evaluation and grading.

In overall performance evaluations across diverse tasks, metrics have demonstrated that sentence-transformers/all-mpnet-base-v2 [32] exhibits superior accuracy and reliability in delivering cosine similarity measures, compared to other models. Empirical analysis corroborates this conclusion, with all-mpnet-base-v2 showing a notably refined capacity to differentiate between responses with ambiguous or nuanced meanings. This model consistently produced mid-range cosine similarity values for such ambiguous cases, accurately reflecting the intended grading scale, while maintaining distinct high and low similarity scores for the most and least appropriate answers. In contrast, while all-MiniLM-L6-v2 was reliable in assigning clear high or low values, it exhibited limitations in handling intermediate cases, as it sometimes assigned cosine similarity values that diverged from the nuanced evaluation a human assessor might provide, resulting in less precise alignment with human grading standards for responses with subtle semantic distinctions.

To enhance the system's capability to evaluate student responses with greater precision, an additional processing step was incorporated. This involves utilizing a Named Entity Recognition (NER) model ''FacebookAI/xlm-roberta-large-finetuned-conll03-english'' [33] to extract key entities and details from the text. By isolating and analyzing these elements, the system assigns individual scores to specific details within the response, thereby ensuring that the grading process remains highly context-aware and tailored to the nuances of the provided answers.

7. Future Work

The future advancements in local Large Language Models (LLMs) hold transformative potential for the field of education, particularly in the domain of exam generation. A key challenge identified during this research was the implementation of a model capable of operating efficiently on standard personal computers while simultaneously offering a large context window and maintaining high-quality performance in specific natural language processing tasks.

Recent developments in the field underscore the promise of achieving these goals. For instance, Meta’s ongoing enhancements to the Llama series, such as the anticipated release of Llama 3.3 with 70 billion parameters, aim to provide performance comparable to larger models like Llama 3.1 with 405 billion parameters. These innovations demonstrate how reducing parameter sizes while maintaining high performance can make LLMs more accessible to researchers and developers operating with limited computational resources. Future breakthroughs are likely, as major technology companies continue to focus on optimizing model sizes without compromising quality.

Moreover, advancements in consumer-grade hardware, such as Nvidia's RTX 5000 series, particularly the RTX 5090 with 32 GB of VRAM, are poised to significantly enhance the ability to run larger models or maximize quantization configurations for smaller models, such as Llama 3.1 with 8 billion parameters while providing an affordable card able to fit in a high-end personal computer. As hardware evolves, GPUs with greater memory capacities will likely align with the increasing demands of modern AI technologies, enabling more efficient processing of computationally intensive tasks. Nvidia has already demonstrated a clear focus on this trajectory, underscoring its commitment to supporting the next generation of AI applications.

In addition to hardware improvements, ongoing efforts to develop more effective optimization techniques for existing models are noteworthy. Innovations such as Floating Point 8-bit (FP8) quantization are particularly promising, as for instance, the application of FP8 quantization to Llama 3.1 has shown potential to enhance the model’s performance to a level comparable to its 16-bit counterpart, effectively balancing efficiency with accuracy.

Lastly, focusing specifically on educational applications, the creation and utilization of specialized datasets offer another avenue for improving the capabilities of LLMs. By fine-tuning models with datasets tailored to educational contexts, researchers can achieve significant enhancements in generating high-quality question-answer pairs, calculating semantic similarity scores, and performing Named Entity Recognition. Such advancements could revolutionize automated exam generation and grading, providing scalable and efficient solutions for educators worldwide.

8. Conclusion

In conclusion, this research introduces an innovative approach to the development of an exam generation and grading system by leveraging state-of-the-art local Large Language Models (LLMs). The study emphasizes critical aspects, including context size, the quality and diversity of Question–Answer pairs generated by different systems, and the computational burden imposed on hardware resources.

A significant part of this work involved employing advanced prompt engineering techniques to refine model outputs, ensuring the generation of accurate, diverse, and pedagogically aligned questions. Additionally, Retrieval-Augmented Generation (RAG) was integrated to address the challenge of processing large contexts. By semantically searching for relevant keywords and extracting focused paragraphs from larger datasets, RAG allowed the system to feed optimized inputs into the LLM, improving both performance and accuracy while reducing computational overhead.

The research also expanded the language capabilities of local LLMs by fine-tuning the Llama model to perform English–Greek and Greek–English translation tasks effectively, further extending its applicability in multilingual educational settings.

To ensure optimal performance, various models were empirically tested and evaluated using established metrics, allowing for the identification of the most efficient and effective solutions. Throughout this process, the main challenges were systematically identified, analyzed, and reported, including issues related to duplicate questions, large context handling, and computational efficiency.

Furthermore, a novel formula was developed to assess students' exam performance by integrating similarity and detail scores, utilizing advanced Natural Language Processing (NLP) techniques. This contribution enhances the grading process, ensuring a robust and equitable assessment system while demonstrating the transformative potential of LLMs in educational applications.

9. References

[1] Shen Wang, Tianlong Xu, Hang Li, Chaoli Zhang, Joleen Liang, Jiliang Tang, Philip S. Yu, and Qingsong Wen. 2024. *Large Language Models for Education: A Survey and Outlook*. arXiv, March 2024.

https://arxiv.org/abs/2403.18105

[2] Jeon, J., Lee, S. Large language models in education: A focus on the complementary relationship between human teachers and ChatGPT. *Educ Inf Technol* 28, 15873–15892 (2023).

https://doi.org/10.1007/s10639-023-11834-1

[3] Kiran, Fenil & Gopal, Hital & Dalvi, Ashwini. (2017). Automatic Question Paper Generator System. International Journal of Computer Applications. 166. 42-47. 10.5120/ijca2017914138.

https://doi.org/10.5120/ijca2017914138

[4] French, S., Dickerson, A. & Mulder, R.A. A review of the benefits and drawbacks of high-stakes final examinations in higher education. *High Educ* 88, 893–918 (2024). https://doi.org/10.1007/s10734-023-01148-z

[5] Yu Chen, Lingfei Wu, and Mohammed J. Zaki. 2020. Reinforcement Learning Based Graph-to-Sequence Model for Natural Question Generation. In *Proceedings of the International Conference on Learning Representations (ICLR) 2020*. https://arxiv.org/abs/1908.04942

[6] Liuyin Wang, Zihan Xu, Zibo Lin, Haitao Zheng, and Ying Shen. 2020. Answer-driven Deep Question Generation based on Reinforcement Learning. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5159–5170, Barcelona, Spain (Online). International Committee on Computational Linguistics.

https://aclanthology.org/2020.coling-main.452/

[7] Michael Heilman and Noah A. Smith. 2010. Good Question! Statistical Ranking for Question Generation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 609–617, Los Angeles, California. Association for Computational Linguistics.

https://aclanthology.org/N10-1086/

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. arXiv, May 2019. https://arxiv.org/abs/1810.04805

[9] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning

with a unified text-to-text transformer. J. Mach. Learn. Res. 21, 1, Article 140 (January 2020).

https://arxiv.org/abs/1910.10683

[10] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. *Language Models are Few-Shot Learners*. arXiv, May 2020.

https://arxiv.org/abs/2005.14165

[11] William Gantt, Lelia Glass, and Aaron Steven White. 2022. Decomposing and Recomposing Event Structure. *Transactions of the Association for Computational Linguistics*, 10:17–34.

https://aclanthology.org/2022.tacl-1.2/

[12] V. Suresh, R. Agasthiya, J. Ajay, A. A. Gold and D. Chandru, "AI based Automated Essay Grading System using NLP," 2023 7th International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, 2023, pp. 547-552, doi: 10.1109/ICICCS56967.2023.10142822.

https://ieeexplore.ieee.org/document/10142822

[13] Katikapalli Subramanyam Kalyan. 2023. *A Survey of GPT-3 Family Large Language Models Including ChatGPT and GPT-4*. arXiv, October 2023.

https://arxiv.org/abs/2310.12321

[14] Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How Multilingual is Multilingual BERT?. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy. Association for Computational Linguistics.

https://aclanthology.org/P19-1493/

[15] Luca Papariello. 2021. xlm-roberta-base-language-detection. Hugging Face. Retrieved January 6, 2025 from

https://huggingface.co/papluca/xlm-roberta-base-language-detection.

[16] Meta AI. 2023. Llama-3.1-8B. Hugging Face. Retrieved January 6, 2025 from https://huggingface.co/meta-llama/Llama-3.1-8B

[17] Sentence-Transformers. 2020. all-mpnet-base-v2. Hugging Face. Retrieved January 6, 2025 from

https://huggingface.co/sentence-transformers/all-mpnet-base-v2

[18] Jun Zhao, Zhihao Zhang, Luhui Gao, Qi Zhang, Tao Gui, and Xuanjing Huang. 2024. *LLaMA Beyond English: An Empirical Study on Language Capability Transfer*. arXiv, January 2024.

https://arxiv.org/abs/2401.01055

[19] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. *Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer*. *Journal of Machine Learning Research* 21, 140 (2020), 1–67.

https://arxiv.org/abs/1910.10683

[20] Manuel Romero. 2020. t5-base-finetuned-question-generation-ap. Hugging Face. Retrieved January 6, 2025 from

https://huggingface.co/mrm8488/t5-base-finetuned-question-generation-ap

[21] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. *SQuAD: 100,000+ Questions for Machine Comprehension of Text*. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP 2016)*, 2383–2392. https://doi.org/10.18653/v1/D16-1264

[22] Sandvik, L. V., Svendsen, B., Strømme, A., Smith, K., Aasmundstad Sommervold, O., & Aarønes Angvik, S. (2022). Assessment during COVID-19: Students and Teachers in Limbo When the Classroom Disappeared. *Educational Assessment*, *28*(1), 11–26. https://doi.org/10.1080/10627197.2022.2122953

[23] Johnnypjp. 2024. Llama-3.1-8b-english-greek-translation-task. Hugging Face. Retrieved January 6, 2025 from

https://huggingface.co/Johnnypjp/Llama-3.1-8b-english-greek-translation-task

[24] Johnnypjp. 2024. Llama-3.1-8b-greek-translation-task. Hugging Face. Retrieved January 6, 2025 from

https://huggingface.co/Johnnypjp/Llama-3.1-8b-greek-translation-task

[25] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. *Language Models are Few-Shot Learners*. In *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, 1877–1901.

https://arxiv.org/abs/2005.14165

[26] Helsinki-NLP. 2020. opus-mt-en-el. Hugging Face. Retrieved January 6, 2025 from https://huggingface.co/Helsinki-NLP/opus-mt-en-el

[27] Helsinki-NLP. Europarl Dataset. Retrieved January 7, 2025, from https://huggingface.co/datasets/Helsinki-NLP/europarl

[28] Helsinki-NLP. 2020. opus-mt-grk-en. Hugging Face. Retrieved January 6, 2025 from https://huggingface.co/Helsinki-NLP/opus-mt-grk-en

[29] unslothai. 2023. unsloth: An open-source library for language understanding. GitHub repository. Retrieved January 6, 2025 from

https://github.com/unslothai/unsloth

[30] Hadzhikoleva, S.; Rachovski, T.; Ivanov, I.; Hadzhikolev, E.; Dimitrov, G. Automated Test Creation Using Large Language Models: A Practical Application. *Appl. Sci.* 2024, *14*, 9125. https://doi.org/10.3390/app14199125

[31] Sentence-Transformers. 2020. all-MiniLM-L6-v2. Hugging Face. Retrieved January 6, 2025 from

https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

[32] Sentence-Transformers. 2020. all-mpnet-base-v2. Hugging Face. Retrieved January 6, 2025 from

https://huggingface.co/sentence-transformers/all-mpnet-base-v2

[33] FacebookAI. 2020. xlm-roberta-large-finetuned-conll03-english. Hugging Face. Retrieved January 6, 2025 from

https://huggingface.co/FacebookAI/xlm-roberta-large-finetuned-conll03-english

[34] Meta. 2023. Llama-2-7b. Hugging Face. Retrieved January 6, 2025 from https://huggingface.co/meta-llama/Llama-2-7b

[35] Meta. 2024. Meta-Llama-3-8B. Hugging Face. Retrieved January 6, 2025 from https://huggingface.co/meta-llama/Meta-Llama-3-8B

[36] OpenAI. 2025. GPT-4.0 and More Tools to ChatGPT Free. OpenAI. Retrieved January 6, 2025 from

https://openai.com/index/gpt-4o-and-more-tools-to-chatgpt-free/

[37] OpenAI. 2025. GPT-4 Turbo and GPT-4. OpenAI Platform Documentation. Retrieved January 6, 2025 from

https://platform.openai.com/docs/models#gpt-4-turbo-and-gpt-4

[38] Meta LLaMA AI Model. 2025. Requirements for LLaMA 3.1 8B. Retrieved January 13, 2025 from

https://llamaimodel.com/requirements/#8B

[39] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2019), 4171–4186. https://doi.org/10.18653/v1/N19-1423

[40] DeepSeek-AI et al. 2025. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. arXiv, January 2025.

https://arxiv.org/html/2501.12948v1

[41] DeepSeek AI. 2025. DeepSeek-R1. Hugging Face. Retrieved February 1, 2025, from https://huggingface.co/deepseek-ai/DeepSeek-R1

[42] DeepSeek AI. 2025. DeepSeek-R1-Distill-Llama-8B. Hugging Face. Retrieved February 1, 2025, from https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Llama-8B