Journal of Artificial Intelligence and Technology, 2024, 4, 188-199
https://doi.org/10.37965/jait.2024.0513

ISTP
RESEARCH ARTICLE

Revolutionizing eLearning Assessments: The Role of GPT in Crafting Dynamic Content and Feedback

Michael E. Bernal
Arizona State University, Tempe, AZ, USA

(Received 19 January 2024; Revised 04 April 2024; Accepted 13 April 2024; Published online 06 May 2024)

Abstract: This study presents the Learnix project that utilizes the GPT-4 large language model (LLM) to enhance interactive learning in eLearning platforms, with a specific focus on generating dynamic multiple-choice questions (MCQs) and providing tailored feedback for Python coding problems and short-answer questions. The aim was to explore how language models can be integrated into eLearning environments to create more adaptive and engaging educational experiences. Leveraging GPT-4's advanced natural language processing capabilities, the developed platform can generate a diverse range of MCQs and offer unique feedback, significantly improving upon traditional static learning content. This approach enhances the learner's journey, offering a more engaging and individualized educational experience. The methodology involved the integration of GPT-4 into an eLearning platform, emphasizing user interaction and content relevance. The Learnix platform was designed to handle a variety of coding problems, with the LLM generating corresponding MCQs and feedback. This method's effectiveness was evaluated based on content quality and relevance. Results demonstrated that GPT-4's inclusion markedly enhanced the eLearning experience by providing diverse and up-to-date content. Customized feedback is particularly effective in reinforcing learning concepts and addressing individual learning needs. Moreover, the platform showed versatility in scaling and adapting to different educational contexts, making it a valuable tool for various learning requirements. The findings of this project emphasize the transformative potential of language models and Generative AI in redefining online education, leaning toward more adaptive and engaging learning experiences. Additionally, the Learnix project underscores the importance of continual innovation in educational technology, suggesting a new paradigm where AI becomes an integral part of the teaching and learning process. The integration of GPT-4 not only enriches the learning material but also enhances the overall effectiveness of the educational process, paving the way for future advancements in AI-driven eLearning solutions.

Keywords: automated assessments; eLearning; GPT; large language models; natural language processing

I. INTRODUCTION

The landscape of eLearning has been continuously on the cusp of revolutionary shifts [1], with promises to transform how knowledge is delivered and absorbed. This article unpacks the journey and insights gleaned from developing an innovative eLearning platform, Learnix, which stands at the intersection of artificial intelligence (AI) and educational technology. The impetus for this venture was two-fold: to bridge the persisting gaps in eLearning offerings and to harness the potential of AI, particularly generative pre-trained transformers (GPTs), to catalyze a learning paradigm that is as dynamic as it is interactive.

The backdrop against which Learnix was conceived is one where traditional eLearning platforms have plateaued [2], often circumscribed by content that ages rapidly and fails to engage learners in a meaningful way. Recognizing these constraints, Learnix was designed to transcend the static nature of conventional online courses [3], deploying a synergistic mix of established content and cutting-edge, GPT-generated material. This approach ensures that learners are not only privy to the latest developments in their field of study but are also part of a learning experience that is customized and stimulating.

Central to the project was the objective to evaluate how traditional learning modalities could be augmented with AI-driven techniques [4]. A prototype course centered around Python programming served as the prototype for this experiment, melding static educational resources with dynamic, GPT-generated assessments and real-time coding and short-answer response evaluations. However, the aspirations of Learnix were not confined to a single programming language. The ambition was to craft a scalable and adaptable framework that could serve as a blueprint for GPT-integrated learning across diverse disciplines, empowering educators to curate and evolve their content with AI as a steadfast ally [5].

The novelty of Learnix is particularly pronounced in its deployment of GPT-generated activities and the immediacy with which it assesses coding exercises and short-answer responses. This platform leaps forward by introducing a feature that permits the reloading of GPT-generated questions, thereby tackling the variability in question quality head-on. Moreover, the project is distinguished by its feedback mechanisms that provide instant, actionable insights, significantly enriching the learning trajectory.

In its developing form, Learnix was envisioned as a prototype for learning Python programming, yet its architecture carries the potential for far-reaching applications. It is meticulously engineered to accommodate a gamut of subjects and learning styles, laying the groundwork for a comprehensive eLearning ecosystem. The foresight embedded in Learnix's design anticipates an educational expanse that includes a multitude of topics, supported by a

Corresponding Author: Michael E. Bernal (e-mail: mberna13@asu.edu).

user-centric interface that allows educators to seamlessly integrate and manage AI-driven courses using large language models (LLMs) and Generative AI.

Embarking on this relatively untrodden path, the project has underscored the untapped possibilities of AI in redefining educational methodologies. The empirical evidence gathered through the development and deployment of Learnix speaks to the transformative impact AI can have on eLearning. As a harbinger of innovations yet to come, Learnix offers a tangible demonstration of how educational content can evolve in tandem with the rapid advancements in technology and knowledge domains [6].

Considering the outcomes realized through Learnix, this article posits that the way forward lies in the continuous exploration and integration of AI within the learning sphere. It advocates for an expansion of the Learnix framework to a broader spectrum of subjects, coupled with the refinement of AI algorithms to yield more tailored and responsive learning experiences. Furthermore, it calls for an enhanced synergy between educational institutions and technology developers to further this pioneering forefront of eLearning. As such, Learnix is not merely a culmination of research and development; it is a compelling call for a collective stride toward an era of AI-enriched education.

The remainder of this paper is structured as follows:

Section 2, Related Work: This section delves into the landscape of AI-driven eLearning platforms, examining how various LLMs, including GPT-4, Claude-3-Opus, and Google Gemma-7b, are shaping or can potentially shape educational technology. We discuss the integration and capabilities of these models, analyzing their role in course management, interactive learning, and content creation. Special attention is given to contrasting the functionality and impact of these LLMs with the innovations introduced by the Learnix platform, thereby underscoring the platform's unique approach and contributions to the field. Furthermore, the evolution of AI in education is explored, evaluating how the interplay of these advanced technologies fosters new educational paradigms.

Section 3, Methodology: This section outlines the technical approach that underpins the Learnix platform. This outline includes the architecture of the system, the integration of GPT-4 for content generation, and the design philosophy that ensures the platform remains user-centric and responsive to learners' needs. A detailed discussion delves into the technical implementation and architecture of Learnix, explaining the specific technologies and frameworks utilized. It sheds light on the back-end C# RESTful services, the ReactJS front-end, and the mechanisms by which these components work together to provide a seamless learning experience. Additionally, the user interface design and the rationale behind design choices are discussed. It reflects on the ethical considerations of AI in education and the steps taken to address them within the Learnix platform.

Section 4, Discussion: This section presents a critical discussion on the incorporation of GPT-4 within Learnix, reflecting on the project's achievements and challenges. It evaluates the effectiveness of Learnix, examining how the platform's use of GPT-4 for generating content and feedback has impacted learners' experiences.

Section 5, Conclusion: This section summarizes the key findings, contributions, and implications for future research. This section reiterates the significance of the Learnix project in the broader context of AI-driven eLearning solutions and suggests directions for future advancements in this rapidly evolving field.

II. RELATED WORK

Exploring the emerging field where AI meets eLearning platforms uncovers a landscape brimming with possibilities, yet largely untapped. As the realms of technology continually reveal new opportunities, the introduction and integration of GPT into educational environments emerges as a notable innovation. The inclusion of AI in the education and learning domain was projected to grow by 43% between 2018 and 2022 [7]. However, this innovation is more in its infancy [8] than fully illuminating the way forward.

In the present scenario, integrating GPT models into online educational platforms is approached with both eagerness and caution [9]. Those navigating the complex world of educational software often encounter a difficult journey marked by a steep learning curve and ethical challenges [10] in effectively implementing AI. Developers and engineers are navigating the complexities of AI implementation, focusing on how these advanced technologies can be integrated into existing educational platforms to deliver robust and consistent learning experiences [11]. Central to this technological intersection is not just the logistical aspects of embedding GPT into existing platforms, but also ensuring that these integrations are robust and consistently effective.

Various initiatives aiming to merge AI with eLearning have cautiously begun, primarily focusing on how to implement GPT for generating and assessing content that enhances self-taught learning approaches. The benefits of using models like ChatGPT-3.5/4 is clear, offering a unique capability to interact with learners in a manner that is both clear and natural. This integration goes beyond simply delivering information by providing an interactive, responsive learning experience tailored to individual needs and learning paths.

Despite its clear potential, the journey to fully integrate GPT models into eLearning is still in its early stages [12]. The broader community of educational technology is collectively tackling the challenges and prospects of such integration. The questions are complex: How to seamlessly weave these intelligent models into the fabric of eLearning platforms, and how to ensure this integration is not only reliable but also significantly enhances the learning experience? Initial forays into these questions have provided fascinating insights and established a foundation for further research and development [13]. Yet, as is typical with innovative ventures, this path is strewn with both unexpected challenges and untapped possibilities. Thus, this project does not simply follow a well-worn path but boldly ventures into relatively unknown territory, aiming to build upon the initial efforts of its predecessors and carve a new path in the fusion of AI/GPT with eLearning. At this early stage of combining AI with eLearning, this project is positioned to not only add to the ongoing discussion and exploration in this field but also to unveil new possibilities, tackle emerging challenges, and ultimately help steer the future course of AI's role and impact within the eLearning sector.

A. COMPARISON AND CONTRAST WITH EXISTING WORK

The current landscape of AI in education is marked by a limited variety of AI applications partly due to the reluctance of educators to fully embrace and adopt such tools within their teaching methodologies [14]. Many educational platforms integrate AI for specific functions, but these are often narrow in scope. This limitation might be due to various factors, such as the complexity of

educational needs, the cost of developing more sophisticated AI tools, or the challenges in ensuring that AI systems are pedagogically effective and ethically sound. Below is a sample of popular applications and their use of AI.

1. AMIRA LEARNING AND DUOLINGO. Amira Learning utilizes AI for speech recognition and oral fluency assessment in early education, where monitoring reading skills and fluency is critical. The AI provides instant feedback and progress tracking, which is particularly beneficial in medium to large classrooms. By integrating speech recognition with the Science of Reading [15], Amira offers personalized and immediate tutoring to students practicing reading aloud, helping to identify and address reading skill gaps [16].

Duolingo employs AI [17] to customize the learning experience, adjusting exercise difficulty based on learner performance and evaluating progress in foreign languages. Its adaptive learning approach maintains user engagement and personalizes learning paths. Duolingo has recently incorporated GPT-4 to provide highly personalized lessons through its Duolingo Max subscription service, enhancing English language assessments and expanding its AI use for cognitive modeling and content development [18].

2. COURSERA AND KHAN ACADEMY. Both platforms leverage AI [19,20] to personalize learning experiences by analyzing user data to adapt content, suggest courses, and provide feedback based on a learners' pace and challenge areas. Such personalization's aim to make online learning more effective by catering to individual needs. Both platforms have harnessed AI to offer personalized feedback and tutoring or coaching services, aligning with the learners' requirements.

A comparative overview of the eLearning platforms discussed, focusing on their integration of AI and educational offerings, is provided in Table I. This table illustrates the varied approaches and features of existing platforms, setting the context for the Learnix project's unique contribution in this field.

3. LEARNIX. In the evolving landscape of AI-enhanced eLearning, Learnix emerges as a pioneering platform distinguished by its specialized focus on technology education and its use of the more advanced GPT-4 model. Unlike the general language learning and early education emphases of Duolingo and Amira Learning, Learnix is engineered to address the intricate challenges of technology education, offering dynamic feedback and interactive learning tailored to the nuances of programming and other technology subjects. While Coursera and Khan Academy have set a precedent in course personalization and tutoring, Learnix extends these concepts into the realm of coding challenges and short-answer questions. It provides an enriched learning experience through its array of dynamic multiple-choice questions (MCQs) and static exercises, designed to reinforce programming and technology concepts actively. This approach underscores Learnix's commitment to delivering a comprehensive and customized educational experience, leveraging the latest in AI advancements to foster a robust and adaptive learning environment for budding programmers and technologists.

In the future, other OpenAI capabilities, like Whisper, can be utilized in conjunction with GPT-4 to revolutionize how language learners and those aiming to improve their speaking skills are assisted. Whisper, an advanced speech-to-text model, can first be employed to accurately transcribe spoken language into text. This transcribed content then becomes the foundation for GPT-4's comprehensive analysis. GPT-4 excels in evaluating grammatical structure, syntax, vocabulary usage, and overall coherence of the transcribed text, providing critical insights into a speaker's language proficiency. While it does not directly analyze audio nuances like pronunciation and intonation, patterns in transcription errors can indirectly highlight areas for improvement in these aspects. Furthermore, GPT-4 can generate personalized feedback and custom language exercises tailored to the learner's specific needs. This synergy of advanced transcription and analytical capabilities offers a potent and customizable tool for enhancing language-speaking abilities. However, for a holistic learning experience, especially in areas like pronunciation and oral fluency, integrating this technology with human instruction is the most sensible approach. Additionally, ensuring data privacy and the suitability of content for all learners, particularly minors, remains a paramount consideration in such applications.

B. COMPARATIVE ANALYSIS OF LLMs IN eLearning

This section delves into the capabilities of two prominent LLMs alongside GPT-4: Claude-3-Opus and Google Gemma-7b. Claude-3-Opus, developed by Anthropic, is known for its strong performance in various natural language tasks, including question-answering, summarization, and creative writing [21]. Its ability to generate coherent and contextually relevant responses makes it a promising candidate for generating educational content and providing personalized feedback to learners. On the other hand, Google Gemma-7b, a smaller variant of the Gemma model series, has shown impressive results in language understanding and generation tasks [22]. Its compact size and efficiency make it an attractive option for integrating into eLearning platforms, especially in resource-constrained environments.

By comparing the strengths and limitations of these LLMs with GPT-4, this section aims to provide a comprehensive understanding of their potential to enhance various aspects of eLearning, such as interactive content creation, automated assessment, and adaptive learning experiences. Through this analysis, we explore how the unique capabilities of each model can contribute to the development of more engaging and effective educational platforms, ultimately transforming the way learners acquire knowledge and skills in the digital age.

Table I. Comparative overview of eLearning platforms focusing on AI integration and educational offerings

Platform	Focus area	Special features
Amira Learning	Early Education	Instant feedback, reading skill gap identification
Duolingo	Language Learning	Difficulty adjustment, Duolingo Max with GPT-4
Coursera	Higher Education & Professional Development	Personalized course suggestions, diverse course offerings
Khan Academy	General Education	Personalized learning dashboard, tutoring services
Learnix	Programming and Diverse Subjects	Dynamic MCQs, short-answer and coding assessments, GPT-4 integration

1. LLM PERFORMANCE EVALUATION. To compare the performance and capability of the three models (GPT-4, Claude-3-Opus, and Google Gemma-7b), standardized Python scripts that are in line with the code templates provided in the Application Programming Interface (API) dashboards of each respective model were utilized. These scripts are designed to automate the sending of prompts to the models and capture their responses for analysis. Each script initiates an API call, which is structured to ensure consistent communication with the language models' servers, maintaining uniformity in the evaluation parameters such as temperature, max tokens, and the top-p probability. Additionally, all three scripts were run in the Jupyter Notebook environment hosted under Google Colab. This approach ensured that the assessment of each model's ability to understand and generate responses was conducted under comparable conditions, thus providing a fair basis for comparative analysis.

For each model, ten tests were conducted divided into two sets. The first set consisted of five tests wherein the models were tasked with generating a MCQ on regular expressions in Python. The prompt specified creating a unique, theoretical, or conceptual question without direct code snippets, four answer choices, and a correct answer with an explanation. The exact phrasing of the prompt was: ('Provide a unique plain text multiple-choice question on "Regular Expressions" in Python, focusing on different aspects of "Defining and Using Regular Expressions". The question should be theoretical or conceptual, with no direct code snippets. Provide four answer choices, only one of which should be correct. Provide the answer and a concise explanation as to why it's correct.')

The second set also included five tests per model, which were aimed at soliciting feedback on a given Python function's syntax and logical structure. The models were instructed to identify errors and offer corrective suggestions, with the prompt reading, as follows: ('Here is a Python function. Please check the syntax and logic of the code and provide feedback on whether it is correct. If there are any issues, explain what they are and suggest how to fix them. `\n\n"'python\ndef check_prime(number):\n  if number > 1:\n  for i in range(2, number):\n  if (number % i) == 0:\n  return False\n  return True\n else:\n  return False\n\n# Example usage:\nprint(check_prime(5))\n"'`).

The models were evaluated based on the following criteria: average response time, clarity of response, relevance to the prompt, adherence to the prompt's instructions, and the variability of the generated content. The testing methodology was designed to evaluate the performance of each language model quantitatively and qualitatively across several metrics. The response time was measured using Python's time module, marking the start and end of the request to the model, and calculating the duration. This measurement provided an average response time for each model, giving insight into its efficiency in processing and returning a response to the prompts. The grading criteria for the other four metrics were as follows:

Clarity of Response: This metric assessed how the model's response fulfilled the prompt requirements. A score of '0' indicated a response that was not clear, while a '1' signified that the response provided all the requested parts, fulfilling the prompt's basic requirement. A score of '2' was reserved for responses that not only fulfilled the prompt but also offered additional insights or information, such as comprehensive feedback on incorrect answers or detailed enhancements in the case of code feedback.

Response Relevance to Prompt: A binary score was applied here, with '0' indicating that the response was irrelevant to the prompt, that is, it responded with a totally different topic than the one requested and '1' denoting that the response appropriately addressed the prompt's request.

Prompt Adherence: This metric was also evaluated in a binary fashion, where a '0' indicated non-adherence and a '1' reflected full adherence to the instructions of the prompt.

Variability: The degree of variation in the model's responses to the same prompt over multiple iterations was quantified. A score of '0' meant the response was the same as previous ones, '1' indicated similar but not identical responses, and '2' was given to responses that were distinctly different from previous ones, showcasing the model's ability to generate a diverse range of answers. Each response from the models was systematically reviewed according to these criteria, and scores were assigned accordingly.

The average of these scores provided a comprehensive view of each model's capabilities in generating MCQs and providing code feedback, allowing for a detailed comparative analysis. The results are displayed in Table II.

The comparative analysis conducted on GPT-4, Claude-3-Opus, and Gemma-7b demonstrates that each model possesses a robust capacity to address the prompts given, albeit with varied proficiency across different metrics. The testing revealed that some models exhibited a faster average response time, while others provided greater clarity or variability in their responses.

GPT-4 showcased consistent performance, with a notable balance between clarity and variability, suggesting its versatility across various prompts. Claude-3-Opus was observed to strongly adhere to the prompt instructions, indicating a precise understanding of the tasks. Meanwhile, Gemma-7b excelled in providing rapid responses, which may be particularly beneficial in time-sensitive applications.

It is crucial to recognize that each model's strengths may align differently with specific use cases. For instance, in scenarios where clarity and detailed feedback are paramount, one model may be more suitable than others, whereas for applications that prioritize response diversity and speed, an alternative model might be favored. The findings underscore the importance of selecting a language model that aligns with the goals and constraints of the intended application. Rather than ranking the models, this study highlights the unique contributions each model brings to the table, encouraging a more tailored and strategic approach to their deployment in real-world scenarios.

Future research may expand upon this work by exploring additional metrics, and broader contexts, or by incorporating other emerging language models, further enriching our understanding of the capabilities and potential of AI-driven language processing tools in diverse settings.

III. METHODOLOGY

The cornerstone of the project's methodology was to seamlessly blend two distinct technologies: robust C# RESTful services and the dynamic ReactJS framework. This approach was chosen to leverage the strengths of both technologies in creating a resilient, user-friendly, and scalable eLearning platform. The central issue recognized while designing this platform was the absence of versatile and adaptive resources to assist coding learners. The aim was to focus on this area and other technology-related domains. A key aspect of this project involves leveraging ChatGPT to generate MCQs and offering feedback on coding challenges. The

Table II. Comparative analysis of LLM results

Model	Prompt type	Average response time	Clarity of response	Response relevant to prompt	Prompt adherence	Variability
gpt-4-1106-preview	MCQ Generation:	14.49s	1.6	1	1	2
	Code Feedback:	18.86s	2	1	1	1.25
claude-3-opus-20240229	MCQ Generation:	11.36s	1.6	1	1	0.75
	Code Feedback:	17.02s	1.2	1	1	1.25
google/gemma-7b	MCQ Generation:	0.81s	1	1	1	2
	Code Feedback:	0.84s	2	1	1	1.25

fundamental objective is to furnish learners with immediate feedback on their coding efforts, while also maintaining the learning experience as one that is captivating and scalable. A user-focused approach was employed in this project, with particular attention to designing the interface and feedback mechanisms that prioritize the needs and experiences of the end user.

A. INTEGRATION OF GPT-4 FOR EDUCATIONAL CONTENT

In the Learnix platform, the seamless interplay between front-end and back-end technologies is pivotal in delivering a responsive and personalized learning experience. This integration is exemplified in the process of generating educational content and feedback, as depicted in the architecture diagram illustrated in Fig. 1.

This diagram illustrates the workflow from user interaction through the ReactJS front-end, RESTful services using C#, to the GPT-4 endpoint and back to the user with generated content or feedback. Amazon API Gateway serves as a managed intermediary, facilitating secure and controlled access to the local REST services.

At the heart of the platform's front-end is ReactJS, a powerful JavaScript library known for its efficient rendering of dynamic user interfaces. When a learner interacts with the Learnix platform, their actions trigger React components to send requests for content or feedback. These requests are then processed by a set of RESTful services, developed using the robust C# language, ensuring that the communication is secure, reliable, and efficiently managed. Upon receiving a request, the RESTful services formulate a prompt based on the learner's input or requirements. This prompt encapsulates the specific educational content, or the nature of feedback required. The service then interacts with the Amazon API Gateway that acts as a conduit, forwarding the prompt to the designated GPT-4 endpoint.

The GPT-4 endpoint receives the prompt and processes it to generate a response. This response may consist of a MCQ tailored to the learner's current topic of study, or it may contain feedback on a piece of code or short-answer text the learner has submitted for evaluation. GPT-4's sophisticated algorithms ensure that the MCQs are relevant, and the feedback is insightful, contributing to an effective and engaging learning process.

Finally, the generated content or feedback is sent back through the Amazon API Gateway to the RESTful services. The RESTful services then format and relay this information to the React front-end, where it is rendered to the learner. If there is an error at any point in the process, it is also communicated back to the front-end, ensuring that the learner is informed and can take appropriate action, such as retrying the request.

Within each Python lesson, users are first presented with reading material on topics such as conditional execution, regular expressions, or strings, preparing them for the subsequent interactive component. This preparatory phase is facilitated through activity functions encapsulated within the learning platform. Upon completing a reading section and proceeding to the next phase, an activity function dedicated to MCQs is triggered. This function interfaces with QuestionComponent.js, a critical component designed to connect with the RESTful API endpoint. It sends a prompt tailored to the specific topic of the lesson to the ContentGen endpoint. The endpoint ContentGen, leveraging the GPT-4 model, generates a MCQ relevant to the lesson's material. QuestionComponent.js then processes and renders this question alongside potential answer choices (Fig. 2) for user interaction. Figures 3 and 4 show the activity function containing the prompt to be sent and core aspects of the QuestionComponent that are responsible for sending the prompt payload and rendering the result.

At the end of each lesson, learners are engaged with two to three coding activities designed to reinforce the lesson's concepts. These activities leverage the React CodeMirror library to properly

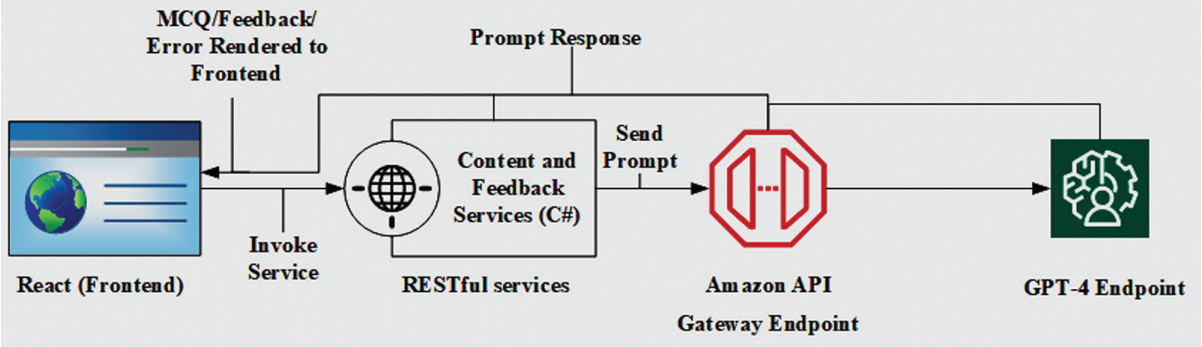

Fig. 1. Architecture and communication flow of the Learnix platform.

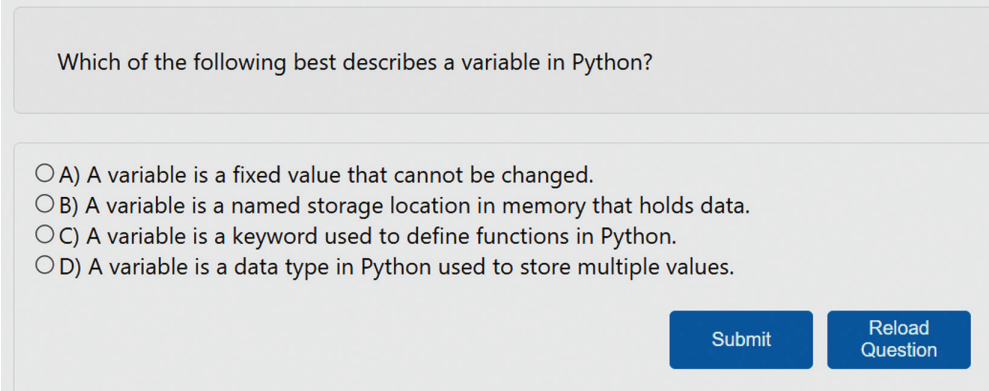

Fig. 2. GPT-4 Generated MCQ on the topic of regular expressions in Python.

format and display code snippets in Python. Users are provided with specific instructions to guide their coding efforts, focusing on tasks directly related to the lesson's topics. Upon completing the code as per the instructions, users submit it through the CodeCheckQuestion.js component. This component is responsible for sending the user's code, along with the prompt for feedback, to the InputChecker RESTful API endpoint. The CodeCheckQuestion component then processes the response from the API, rendering the feedback to help users evaluate their work, as illustrated in Fig. 5. Detailed views of the activity prompt and the core functionalities of the CodeCheckQuestion component are depicted in Figs. 6 and 7, respectively.

```
function Activity3({ enableNext }) {
    return <QuestionComponent
        enableNext={enableNext}
        prompt="Provide a unique plain text multiple-choice question on
Variables in Python. The question should be theoretical or conceptual, with
no direct code snippets in the question or the answers. Provide four answer
choices, only one of which should be correct. Provide the answer and a
concise explanation as to why it's correct."
    />;
}
```

Fig. 3. Example of an Activity function calling the QuestionComponent with a prompt for generating a multiple-choice question on Python variables.

```
function QuestionComponent({ prompt, enableNext }) {
    const [questionData, setQuestionData] = useState(null);
    const [selectedOption, setSelectedOption] = useState(null);
    // Fetches MCQ data based on the prompt
    useEffect(() => {
        fetch(API_ENDPOINTS.generateContent, {
            method: 'POST', headers: { 'Content-Type': 'application/json' },
            body: JSON.stringify({ Prompt: prompt }),
        })
        .then(response => response.json())
        .then(data => setQuestionData(data))
        .catch(error => console.error(error));
    }, [prompt]);
    // Simplified rendering logic
    return (
        <div>
            {/* Render question and choices here */}
            <ReactMarkdown>
                { questionData }
            </ReactMarkdown>
        </div>
    );
}
```

Fig. 4. Core functionality of QuestionComponent for fetching and displaying MCQs.

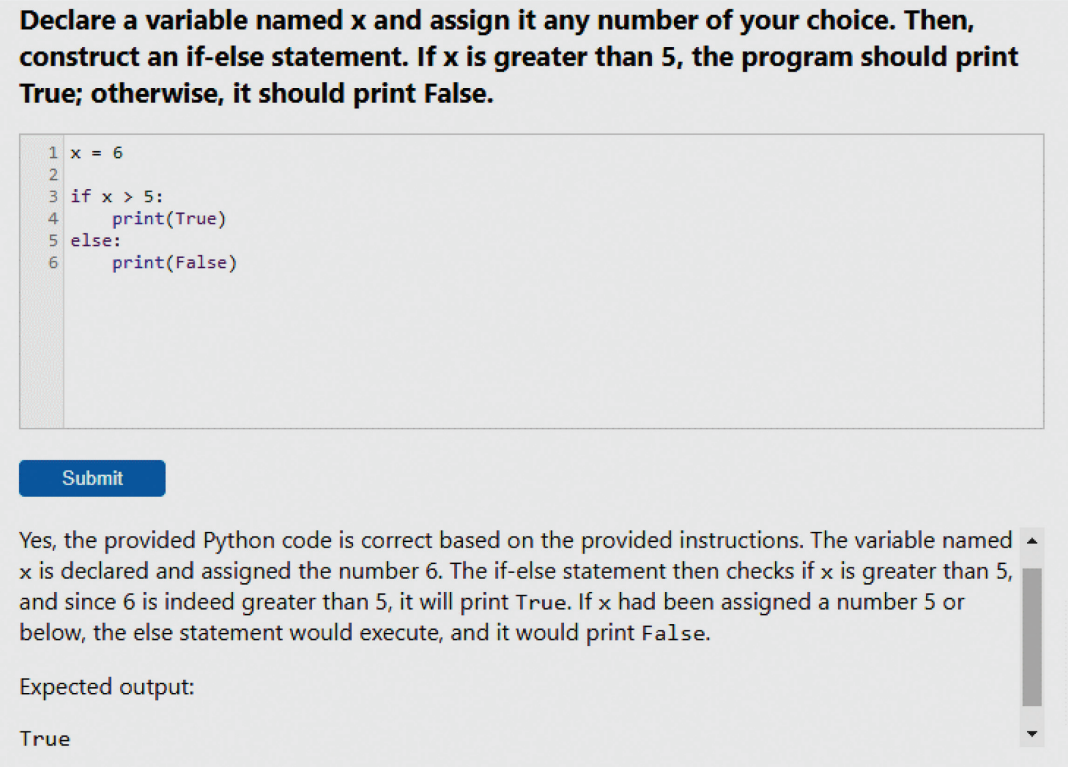

Fig. 5. GPT-4 generated feedback based on Python code input to a coding problem.

```
function Activity6({ enableNext }) {
    return <CodeCheckQuestion
        enableNext={enableNext}
        title="Declare a variable named x and assign it any number of your
choice. Then, construct an if-else statement. If x is greater than 5, the
program should print True; otherwise, it should print False."
        prompt="Please assess the Python code to ensure it adheres to these
instructions and provide a very brief explanation: 'Declare a variable named
x and assign it any number of your choice. Then, construct an if-else
statement. If x is greater than 5, the program should print True; otherwise,
it should print False.'"
    />;
}
```

Fig. 6. Activity Function Example: This figure showcases the Activity function, which uses the CodeCheckQuestion component to engage users with a coding challenge.

A RESTful service was designed to generate MCQs for any topic, like Python, as depicted in Fig. 2 utilizing the cutting-edge capabilities of GPT-4 to produce questions that are both relevant and challenging to learners.

Central to its operation is the service's ability to communicate efficiently with the GPT-4 endpoint. This communication is achieved by dispatching structured JSON payloads, meticulously crafted to outline the precise content requirements and the desired parameters essential for the generation of high-quality questions. Upon the submission of a prompt, the GPT-4 endpoint processes this data, subsequently returning a MCQ that aligns seamlessly with the educational objectives of the lesson. A notable feature of this service is the 'Reload Question' function. This innovative feature empowers users to request a fresh question in instances where the initial output does not meet their expectations, thereby guaranteeing a consistently exceptional standard of educational content.

Delving deeper into the technical intricacies, the RESTful service code implemented introduces a ContentGen class – a specialized construct dedicated to the generation of educational content. The ContentGen class is equipped with a private HttpClient object, a critical component used for transmitting HTTP requests. Additionally, it possesses a private API key, a crucial element required for secure authentication with the OpenAI API. In the constructor of the class, the HttpClient is instantiated, and its default request headers are configured. This configuration includes the integration of the authorization header, which incorporates the bearer token, which is essentially the API key.

```
function CodeCheckQuestion({ title, prompt, enableNext }) {
    const [inputFeedback, setInputFeedback] = useState('');
    const codeMirrorContainer = useRef(null);
    // Setup for code editor and handling code submission
    useEffect(() => {
        // Initialize CodeMirror instance here
    }, []);
    const handleSubmit = async () => {
        setSubmitLoading(true);
        await new Promise(resolve => setTimeout(resolve, 1000));
        const response = await fetch(API_ENDPOINTS.AltInCheck, {
            method: 'POST',
            mode: 'cors',
            headers: {
                'Content-Type': 'application/json',
            },
            // The code and prompt to be sent to the RESTful API
            body: JSON.stringify({Code: code, Prompt: prompt})
        });
        const result = await response.json();
        setInputFeedback(result.Feedback);
        setSubmitLoading(false);
    };
    // Rendering the code editor and feedback
    return (
        <div>
             <ReactMarkdown>
                    {inputFeedback}
              </ReactMarkdown>
        </div>
    );
}
```

Fig. 7. Overview of CodeCheckQuestion component, illustrating code submission for assessment and feedback rendering.

The GenerateContent method, a pivotal feature of this class, is an asynchronous task with a ContentRequest object designed to interact seamlessly with the OpenAI API for content generation. This method constructs a JSON payload, delineating the specific model to be employed, the user's prompt extracted from the ContentRequest, and the maximum token limit for the content to be generated. This payload is then asynchronously posted to the OpenAI API's GPT-4 chat completions endpoint. In instances of a successful API response, the method parses the JSON response, extracting the generated content, which encompasses a MCQ along with the correct answer. This content is meticulously structured, typically featuring the question followed by a selection of answer choices and the correct answer, all separated by newlines for clarity.

Subsequently, the method initiates the creation of a ContentResponse object. This object encapsulates the generated question, the array of answer choices, and the correct answer. In scenarios where the expected content structure is not adhered to or if anomalies occur during the HTTP request or response handling, the method adapts by returning a ContentResponse imbued with detailed error information. Moreover, if an exception arises during this process, the ContentResponse is designed to include both the exception message and the stack trace. This method is not just a technical function; it embodies a structured and refined approach to requesting and receiving generated content, and it exemplifies an elegant method of handling potential errors gracefully. Figure 8 depicts the core elements of the ContentGen RESTful service showing how endpoint communication occurs.

Another RESTful service developed, named InputChecker, provides real-time, personalized feedback on coding exercises and short-answer questions. This service also interfaces with the GPT-4 endpoint, exchanging JSON payloads that contain code snippets or short-text answers submitted by learners. GPT-4's sophisticated analysis capabilities are then applied to these submissions. It assesses the correctness of text responses or evaluates the syntax and logical structure of code snippets in relation to the specific requirements of the coding exercise. The outcome of this analysis is a comprehensive feedback mechanism that confirms whether the text response aptly answers the question or if the code fulfills the stipulated criteria. This feedback is then rendered in React to the learner (Fig. 5), constituting an instantaneous feedback loop that is pivotal for the learning process. It enables learners to swiftly comprehend their mistakes and enhance their coding skills in real time, thereby fostering an environment of continuous learning and improvement.

Furthermore, the InputChecker service accomplishes providing feedback through an integration with the CodeMirror component in React, where it evaluates code input for both syntactical and logical accuracy, based on predefined static instructions. The core functionality of this service includes a constructor for the InputChecker class and an Analyzer method. The constructor is responsible for initializing an HttpClient object and configuring its authorization header using a bearer token.

The Analyzer method, an asynchronous function, accepts an AnalysisRequest object containing a user's prompt and a code snippet or short-answer text. This method is designed to construct

```
// Initialize HttpClient and set the Authorization header with the API key
httpClient = new HttpClient();
httpClient.DefaultRequestHeaders.Add("Authorization", $"Bearer {apiKey}");
// Construct the payload with the prompt and other parameters
var payload = new
{
    model = " gpt-4-0125-preview",
    messages = new[]
    {
        new {role = "user", content = request.Prompt}
    },
    max_tokens = 220, //Length of response
    temperature = 0.7 // This setting determines the variability in responses
};
// Send the request to the OpenAI API endpoint
var response = await
httpClient.PostAsync("https://api.openai.com/v1/chat/completions",
    new StringContent(JsonConvert.SerializeObject(payload), Encoding.UTF8,
"application/json"));
// Process the API response
string responseBody = await response.Content.ReadAsStringAsync();
var jsonResponse = JObject.Parse(responseBody);
var rawContent =
jsonResponse["choices"]?[0]?["message"]?["content"]?.ToString();
```

Fig. 8. Core code elements of the ContentGen C# RESTful class.

a payload for the OpenAI API, specifically interacting with gpt-4-0125-preview for detailed code analysis. The payload encompasses instructions for the analysis, the user's code snippet, and a query regarding the correctness of the code. Upon sending this payload to the OpenAI API's chat completions endpoint via a POST request, the method awaits a response. If the response is affirmative, the method parses the content, determining the code's correctness based on the presence of a specific confirmation string ('Yes, the code is correct'). It then generates an AnalysisResponse object, which not only indicates the correctness of the code but also includes insightful feedback derived from the analysis process.

In instances where the OpenAI API's response is either unsuccessful or deviates from the expected format, such as missing content, the method returns an AnalysisResponse with an appropriate error message. Furthermore, should an exception arise during this process, such as network errors or issues with the API request, the catch block is designed to capture these exceptions and return an AnalysisResponse containing the exception message.

This method exemplifies a well-structured approach to analyzing code snippets and short answers using AI. It adeptly handles various response scenarios and encapsulates the analysis results in a response object. This object not only provides a Boolean indicating the correctness of the code but also offers valuable feedback text, thereby enhancing the overall learning experience for the user. The structure of the code deviates little from the ContentGen class with a few exceptions – the payload prompt is modified to expect code or text and to evaluate the code based on syntax and adherence to the instructions given in the lesson. Additionally, conditional logic is added to return the appropriate response based on the code's correctness. Figure 9 shows the important changes made to the implementation of the InputChecker service.

These services exemplify how Learnix utilizes GPT-4's capabilities to enrich the learning experience, providing a platform that not only educates but also adapts to the individual needs of each student. The use of JSON payloads for data exchange and the incorporation of features like the 'Reload Question' button reflects the platform's commitment to providing an engaging, interactive, and flexible learning environment.

B. TECHNICAL IMPLEMENTATION AND ARCHITECTURE

The backbone of Learnix is constructed upon a service-oriented architecture (SOA) [23] designed for modularity and flexibility. The back end is powered by C# RESTful services [24], which are tasked with handling crucial operations such as user authentication against a MySQL database, personalization of user experiences, managing user account information, and orchestrating dynamic content generation by interfacing with the GPT-4 endpoint [25].

With a commitment to security, API keys were implemented to control access to various services. AWS API Gateway was utilized as a secure store for these keys and as a conduit for proxy passthrough, ensuring that all communication between the front-end and back-end services is both secure and efficient. Sensitive data handling and credential management are conducted securely in the back end, incorporating encryption methodologies such as BCrypt to protect user information.

The platform's architecture is built to accommodate growth, allowing for the easy addition of new services and features. The scalability of the design is evidenced by the seamless inclusion of additional learning modules and user capacity as the platform expands.

The complexity of managing multiple integrated services was addressed by deploying a decoupled architecture, enhancing platform resilience, and simplifying deployment and maintenance. Contingency measures were established to mitigate the impact on the user experience in case of service downtime or failure. Additionally, rigorous security protocols were implemented to

```
// payload specifies the gptmodel, the prompt including code assessment
instructions, and the user's code.
var payload = new
{
    model = "gpt-4-0125-preview",
    messages = new[]
    {
        new { role = "user", content = $"{request.Prompt}\n\nUser's
code:\n{request.Code}\nIs this correct?" }
    },
    max_tokens = 320,
    temperature = 0.7 // Adjusted for more variability
};
// Check if the content indicates the code is correct and return the
appropriate response
if (rawContent.Contains("Yes, the code is correct"))
{
    return new AnalysisResponse { IsCorrect = true, Feedback = "Your code
looks correct!" };
}
else
{
    return new AnalysisResponse { IsCorrect = false, Feedback = rawContent };
}
```

Fig. 9. Core code elements of the ContentGen C# RESTful service, InputChecker.

protect against potential vulnerabilities, thereby ensuring the platform's robustness and reliability.

C. USER-CENTRIC DESIGN PHILOSOPHY

Throughout the development process of the Learnix eLearning platform, adherence to a user-centric design philosophy was pivotal, particularly in shaping the interface and feedback mechanisms. This approach was underpinned by the goal to develop an intuitive, engaging, and educational user experience, with a focus on smooth and rewarding interactions. The platform was characterized by its intuitive navigation, offering a clean, straightforward interface that allowed users easy access to various sections and features. Its logical structure significantly reduced the learning curve, thereby enhancing user satisfaction. Personalized and immediate feedback on coding tasks and MCQs was a standout feature, made possible through the integration of GPT-4, which facilitated a more interactive learning process and catered to individual learning styles and paces. Dynamic content generation, adapting to the user's progress and learning needs, kept the educational material fresh and relevant. Recognizing the diverse range of devices used in eLearning, the platform's responsive design ensured a seamless experience across desktops, laptops, and tablets.

Integral to its development was the continuous incorporation of user feedback, which directly influenced subsequent updates and feature enhancements, aligning the platform's evolution with user needs and preferences. Additionally, a strong commitment to accessibility and inclusivity meant that the platform was developed to be compliant with accessibility standards, thereby catering to a wide audience, including those with disabilities.

Ethical considerations in AI use and stringent data security protocols were also central to the design philosophy, ensuring high standards of data protection and privacy, and efforts to mitigate potential biases in LLM-generated content. In essence, the user-centric design of the Learnix platform was not just a choice but a strategic imperative, aiming to elevate the educational experience to be both effective and enjoyable, thus setting a new benchmark in the eLearning user experience.

IV. DISCUSSION

The incorporation of LLMs, specifically GPT-4, in the Learnix project marks a significant stride in the eLearning domain. This integration exemplifies the unique capabilities of LLMs in enriching the educational experience through dynamic content generation and sophisticated feedback mechanisms. Unlike broader AI applications, the use of an LLM like GPT-4 in Learnix facilitates nuanced, context-aware interactions with learners, which is particularly beneficial in areas like coding education and language skills development [26].

The technical implementation of Learnix harnesses the strengths of LLMs within a structured educational framework [27]. By integrating GPT-4 with C# RESTful services and the ReactJS framework, the platform strikes a balance between robust functionality and user-centered design. This synthesis not only addresses the complex needs of modern learners by offering personalized and adaptive learning experiences [28] but also demonstrates the potential of AI to revolutionize traditional learning methodologies [29].

During the development of Learnix, specific challenges related to the integration of LLMs were encountered, such as managing complex queries and ensuring the generation of relevant content. Overcoming these challenges required a detailed understanding of both the technical capabilities of GPT-4 and the pedagogical needs of the eLearning environment. This experience provides valuable insights into the practical application of LLMs in educational settings, highlighting both the potential and the limitations of this emerging technology [30].

The rigorous testing phase of Learnix, ensuring the accuracy and relevance of the LLM's responses, underscores the importance

of precision in educational technology. This aspect of the project demonstrates the critical role of thorough testing and validation in deploying LLM-based systems, ensuring reliability and educational effectiveness.

In discussing the ethical implications of using LLMs in education, the Learnix project underscores the need for responsible deployment of this technology. The project's commitment to ethical standards, including data protection and bias mitigation, is crucial given the expansive capabilities of LLMs in processing and generating language-based content [31].

Looking toward the future, the project opens avenues for further research into the use of LLMs and Generative AI across diverse educational contexts. Future explorations could extend the applications of LLMs beyond coding and technical education, offering personalized learning experiences in various subjects and disciplines. The potential for LLMs to cater to different learning styles and needs, coupled with their ability to provide up-to-date, contextually relevant content, positions them as a transformative tool in the landscape of education technology. In this capacity, the Learnix project not only highlights the current capabilities of AI in enhancing eLearning but also sets a path for future innovations. As AI continues to evolve, its integration into educational platforms like Learnix is poised to redefine the boundaries of digital learning, offering more adaptive, personalized, and effective educational experiences.

V. CONCLUSION

The information presented on the Learnix project provides a comprehensive overview of creating a cutting-edge eLearning platform. With a focus on Python programming, Learnix seeks to transform the eLearning industry by incorporating LLM and Generative AI technology, specifically to boost user involvement and learning efficiency. Its main attributes include comprehensive learning integration, advanced user management systems, diverse educational resources, sophisticated GPT-4-based assessments, and scalable architecture. Learnix tackles the issue of maintaining up-to-date educational content in a rapidly evolving tech environment by leveraging OpenAI's GPT-4 model. This strategy ensures content remains current and dynamically adjusts to technological shifts and programming language developments.

Learnix represents a significant advancement in the field of eLearning, combining language models and traditional learning methods to create a more dynamic and interactive educational experience. The project showcases the feasibility and effectiveness of integrating AI-based technology in education, particularly in the development of a scalable and versatile framework for AI-integrated courses. It stands out for its innovative use of GPT-generated activities, real-time assessment of coding exercises and short-answers, and a design that prioritizes user experience and scalability. The project sets a new benchmark in AI-enhanced eLearning, offering valuable insights and laying a foundation for future educational technologies. It is a trailblazing effort that not only addresses current educational challenges but also opens new avenues for the integration of LLMs and Generative AI in eLearning environments.

CONFLICT OF INTEREST STATEMENT

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

REFERENCES

[1] F. J. García-Peñalvo, "Avoiding the dark side of digital transformation in teaching. An institutional reference framework for eLearning in higher education," *Sustainability*, vol. 13, no. 4, p. 2023, 2021.

[2] M. Pikhart and B. Klímová, "eLearning 4.0 as a sustainability strategy for generation z language learners: applied linguistics of second language acquisition in younger adults," *Societies*, vol. 10, no. 2, p. 38, 2020.

[3] C. A. Collins and D. R. Galbreath, "A case study of teaching single variable calculus traditionally vs. virtually: a semester bifurcated," *PRIMUS*, vol. 32, no. 7, pp. 798–811, 2021.

[4] A. Johri, "Augmented sociomateriality: implications of artificial intelligence for the field of learning technology," *Research in Learning Technology*, vol. 30, p. 2642, 2022.

[5] C. Guan, J. Mou, and Z. Jiang, "Artificial intelligence innovation in education: a twenty-year data-driven historical analysis," *International Journal of Innovation Studies*, vol. 4, no. 4, pp. 134–147, 2020.

[6] R. Soler-Costa, A.-J. Moreno-Guerrero, J. López-Belmonte, and J.-A. Marín-Marín, "Co-word analysis and academic performance of the term TPACK in web of science," *Sustainability*, vol. 13, no. 3, p. 1481, 2021.

[7] O. Z. Richter, V. I. Marín, M. Bond, and F. Gouverneur, "Systematic review of research on artificial intelligence applications in higher education – where are the educators?," *International Journal of Educational Technology in Higher Education*, vol. 16, no. 1, pp. 1–27, 2019.

[8] M. E. Dogan, T. Goru Dogan, and A. Bozkurt, "The use of artificial intelligence (AI) in online learning and distance education processes: a systematic review of empirical studies," *Applied Sciences*, vol. 13, no. 5, p. 3056, 2023.

[9] Y. Wang, C. Liu, and Y.-F. Tu, "Factors affecting the adoption of AI-based applications in higher education: an analysis of teachers' perspectives using structural equation modeling," *DOAJ (DOAJ: Directory of Open Access Journals)*, vol. 24, no. 3, pp. 116–129, 2021.

[10] K. Fuchs, "Exploring the opportunities and challenges of NLP models in higher education: is chat GPT a blessing or a curse?," *Frontiers in Education*, vol. 8, p. 1166682, 2023.

[11] W. Holmes, R. Luckin, Ucl Knowledge Lab, and E. Al,*Intelligence Unleashed: An Argument for AI in Education*. London: Pearson, 2016.

[12] J. Dunnigan, D. Henriksen, P. Mishra, and R. Lake, "'Can we just please slow it all down?' School leaders take on ChatGPT," *Tech Trends*, vol. 67, no. 6, pp. 878–884, 2023.

[13] L. Chen, P. Chen, and Z. Lin, "Artificial intelligence in education: a review," *IEEE Access*, vol. 8, pp. 75264–75278, 2020.

[14] T. B. Brown et al., "Language models are few-shot learners," arXiv preprint arXiv:2005.14165, 2020.

[15] J. Li, "An empirical study on reading aloud and learning english by the use of the reading assistant SRS," *International Journal of Emerging Technologies in Learning (iJET)*, vol. 15, no. 21, p. 103, 2020.

[16] Amira Learning, "How it works?," *staging.amiralearning.com*, May 25, 2023. https://staging.amiralearning.com/how-it-works.html (accessed Jan. 06, 2024).

[17] M. Y. M. Amin, "AI and chat GPT in language teaching: enhancing EFL classroom support and transforming assessment techniques", *Intern. J., High. Educ. Pedag.*, vol. 4, no. 4, pp. 1–15, 2023.

[18] Y. Attali et al., "The interactive reading task: transformer-based automatic item generation," *Frontiers in Artificial Intelligence*, vol. 5, p. 903077, 2022.

[19] R. Kindi et al., "Comparing a-list empire educational software to Khan Academy's: increasing student performance by incorporating artificial intelligence into video-based instruction," *International Journal of Advanced Educational Research*, vol. 1, no. 6, pp. 01–06, 2016.

[20] R. Rashad Saqr, S. Abdullah Al-Somali, and M. Y. Sarhan, "Exploring the acceptance and user satisfaction of AI-driven e-learning platforms (blackboard, moodle, Edmodo, coursera and edX): an integrated technology model," *Sustainability*, vol. 16, no. 1, pp. 204–204, 2023.

[21] S. Wu et al., "A Comparative Study of Open-Source Large Language Models, GPT-4 and Claude 2: Multiple-Choice Test Taking in Nephrology," *arXiv:2308.04709*, 2023.

[22] Gemma Team et al., "Gemma: Open Models Based on Gemini Research and Technology," *arXiv:2403.08295*, 2024.

[23] N. Niknejad, W. Ismail, I. Ghani, B. Nazari, M. Bahari, and A. R. B. C. Hussin, "Understanding service-oriented architecture (SOA): a systematic literature review and directions for further investigation," *Information Systems*, vol. 91, p. 101491, 2020.

[24] L. Xu and Y. Wang, "XCloud: Design and Implementation of AI Cloud Platform with RESTful API Service.," *arXiv:1912.10344*, 2019.

[25] E. Ozdemir, "A General Overview of RESTful Web Services," *Advances in Systems Analysis, Software Engineering, and High Performance Computing Book Series*, Hershey, PA: IGI Global, 2020.

[26] S. MacNeil et al., "Automatically generating CS learning materials with large language models," *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 2*, vol. 2, pp. 1176, 2023.

[27] S. Sarsa, P. Denny, A. Hellas, and J. Leinonen, "Automatic generation of programming exercises and code explanations using large language models," *Proceedings of the 2022 ACM Conference on International Computing Education Research V.1*, vol. 1, pp. 27–43, 2022.

[28] H. Peng, S. Ma, and J. M. Spector, "Personalized adaptive learning: an emerging pedagogical approach enabled by a smart learning environment," *Smart Learning Environments*, vol. 6, no. 1, p. 9, 2019.

[29] M. J. Reiss, "The use of AI in education: practicalities and ethical considerations," *London Review of Education*, vol. 19, no. 1, pp. 1–14, 2021.

[30] A. Tamkin, M. Brundage, J. Clark, and D. Ganguli, "Understanding the Capabilities, Limitations, and Societal Impact of Large Language Models.," *arXiv:2102.02503*, 2021.

[31] S. I. Ross, F. Martinez, S. Houde, M. Muller, and J. D. Weisz, "The programmer's assistant: conversational interaction with a large language model for software development,"In *Proceedings of the 28th International Conference on Intelligent User Interfaces*. New York, NY: Association for Computing Machinery, 2023.