

Communication

# Automated Test Creation Using Large Language Models: A Practical Application

Stanka Hadzhikoleva <sup>1,\*</sup>, Todor Rachovski <sup>1</sup>, Ivan Ivanov <sup>1</sup>, Emil Hadzhikolev <sup>1</sup> and Georgi Dimitrov <sup>2</sup>

<sup>1</sup> Faculty of Mathematics and Informatics, University of Plovdiv Paisii Hilendarski, 236 Bulgaria Blvd., 4027 Plovdiv, Bulgaria; todormr@gmail.com (T.R.); ivanivanov1040@gmail.com (I.I.); hadjikolev@uni-plovdiv.bg (E.H.)

<sup>2</sup> Faculty of Information Science, University of Library Studies and Information Technologies, 1784 Sofia, Bulgaria; g.dimitrov@unibit.bg

\* Correspondence: stankah@uni-plovdiv.bg

**Abstract:** The article presents work on developing a software application for test creation using artificial intelligence and large language models. Its main goal is to optimize the educators' work by automating the process of test generation and evaluation, with the tests being stored for subsequent analysis and use. The application can generate test questions based on specified criteria such as difficulty level, Bloom's taxonomy level, question type, style and format, feedback inclusion, and more, thereby providing opportunities to enhance the adaptability and efficiency of the learning process. It is developed on the Google Firebase platform, utilizing the ChatGPT API, and also incorporates cloud computing to ensure scalability and data reliability.

**Keywords:** test generation; AI in education; ChatGPT API; prompt engineering; automated assessment; artificial intelligence; student evaluation; LLM

## 1. Introduction



**Citation:** 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>

Received: 14 August 2024

Revised: 21 September 2024

Accepted: 27 September 2024

Published: 9 October 2024



**Copyright:** © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

The rapid development of various software technologies utilizing artificial intelligence (AI) in recent years has unveiled significant potential in the field of education [1,2]. AI offers numerous methods, technologies, and tools for learning and assessment, as well as for administering educational processes. This includes the use of technologies such as natural language processing through large language models [3,4], image recognition technologies through computer vision [5,6], data analysis and prediction using neural networks [7,8] to enhance the learning process, and more.

The growing interest in new trends for the use of artificial intelligence in education is evidenced by the increasing number of scientific publications on the topic [9,10]. These publications address a wide range of issues—from technical details related to the integration of AI into software applications [11], to ethical concerns, risks, and threats associated with the use of AI [12]. Research shows that AI has the potential to enhance the quality of education, making it more accessible and tailored to the needs of individual learners [13,14].

AI finds widespread application in various fields of education—social sciences [15], engineering education [16], science education [17], medical education [18], life sciences [19], language learning [20], and others [21].

In general, five main research directions have emerged: Assessment/Evaluation, Predicting, AI Assistant, Intelligent Tutoring System (ITS), and Managing Student Learning [22], with each of them revealing a distinct area with significant potential for innovation in the educational field.

The use of AI spans various educational activities and processes. AI tools are employed to create intelligent tutoring systems, profiling and prediction systems, as well as for adaptive learning [23]. Research demonstrates the positive role of AI in enhancing personalized learning experiences, identifying at-risk students, and automating administrative

tasks [24]. Various AI algorithms and technologies are also applied to ensure the quality of higher education. The analysis of data from different indicators provided by the internal quality systems of educational institutions, along with the evaluation of their alignment with target indicators, enables the early identification of weaknesses and gaps [25].

AI chatbots are finding increasingly widespread applications in education. They offer substantial capabilities for research support, automated assessment, and improved human-computer interaction [26]. Virtual assistant tools provide learners with interactive and engaging ways to master complex theories and conduct independent research [27,28].

One of the most popular AI chatbots is ChatGPT. It can be used for teaching, learning, and conducting scientific research [29]. ChatGPT has the potential to write homework, essays, and coursework assignments [30,31]. In some cases, it is not possible to determine whether a given piece of work was created by a student or ChatGPT [32]. When used without the establishment of appropriate rules and procedures, it can lead to the erosion of traditional mechanisms for assessing student exam work. Moreover, some researchers argue that the chatbot could render certain traditional forms of assignments, such as essay writing, obsolete [33]. The lack of critical thinking skills and difficulties in evaluating information generated by ChatGPT are other issues [34]. It can lead to various problems in the academic development of learners [35].

ChatGPT can be used by educators for various activities, including generating educational materials, creating exam materials and tests, analyzing and providing feedback on essays and other assignments, creating educational documentation such as curricula, programs, and schedules, and offering personalized explanations for complex educational content. It is particularly popular for generating tests, as educators can either provide specific content or rely on the chatbot's knowledge base [36].

The integration of ChatGPT's capabilities into a software system for automated test creation and management is a natural step forward in optimizing learning processes. The advantages of automatically generating test questions based on provided educational content, offering possible answers and marking the correct ones, managing the execution of tests, and evaluating students' responses as an alternative to the manual work of creating questions, structuring tests, and grading student tests are evident. However, the use of AI-driven software applications for test management brings additional benefits. They can offer greater adaptability by generating tests with varying levels of difficulty to assess different cognitive skills, supporting modern educational paradigms focused on differentiated and adaptive learning. ChatGPT's ability to create various types of questions—from multiple choice to essays—enables the creation of tests that encourage critical and creative thinking. The chatbot's capability to provide differentiated, immediate feedback with brief or detailed explanations for correct and incorrect answers helps learners understand their mistakes and fosters their progress.

This article presents an approach for developing a software application that leverages the capabilities of ChatGPT for generating and managing tests. The goal is to facilitate the work of educators by creating test questions based on given texts, structuring tests, and automating the process of conducting test assessments. The tests are stored in a database for future use and are automatically evaluated upon completion. In this context, the developed application is an innovative tool that optimizes assessment processes and supports personalized learning.

## 2. The Evolution of Software Applications for Test Creation

Test creation typically requires significant time and human resources. Additionally, the tests must be evaluated for reliability and validity, with pilot testing being essential for identifying weaknesses and gaps, and addressing them [37]. This motivates educators to actively seek ways to automate the process, including the generation of test questions, construction, and evaluation of tests [38]. Initially, applications were developed using traditional methods and technologies.

The first such applications relied on the manual input of questions and answers. Educators created questions manually using text editors or specialized software applications for test creation, which required considerable time and effort. Over time, new developments emerged that offered templates and pre-prepared questionnaires, which could be modified and adapted to meet the users' needs [39]. The templates define the structure of various types of test questions and include sentences with fixed text in which there are placeholders. To create a specific question, these placeholders are replaced with appropriate text [40].

The drive for quick test creation led to the development of applications with fixed questions. These applications often utilize static databases with pre-created questions and answers. Educators can select questions from these databases to create tests. In some cases, the questions are categorized and labeled according to the topic, difficulty level, and type of question (multiple choice, open-ended, etc.), which significantly facilitates the creation of well-balanced tests with randomly generated questions [41].

The need for more flexible online learning solutions led to the development of complex Learning Management Systems (LMS) such as Moodle, Blackboard, Google Classroom, Cornerstone, OpenEDX, and others [42]. These platforms support a variety of tools for creating and managing tests, including basic features for question creation, rule setting, and result tracking. More advanced systems use algorithms to create adaptive tests based on predefined rules, where questions are selected based on the user's previous answers. However, these systems require complex algorithms and significant manual configuration.

Some automated test generation systems follow a methodology similar to manual test creation. To create a test question, a sentence is first selected, and a keyword or phrase is identified. The sentence is then transformed into a question format based on the selected key, and possible answers are determined. The formalization of these steps can be roughly divided into six stages: (1) Pre-processing, (2) Sentence selection, (3) Key selection, (4) Question generation, (5) Distractor generation, and (6) Post-processing [43].

In many cases, pre-processing of the text from which questions are generated is applied. Various techniques are used for this purpose, such as text normalization [43], structural analysis [44], sentence simplification [45], lexical analysis [46], statistical analysis [47], syntactic analysis [48], and others. The selection of sentences to be used for constructing test questions is also the subject of numerous studies. Different criteria are employed, such as sentence length [49], occurrence of a particular word [50], parts-of-speech information [47], parse information [51], semantic information [52], and others. For constructing certain types of test questions, keywords are selected based on frequency [53], parts-of-speech information [47], parse information [51], semantic information [54], and others. Older applications rely mainly on basic statistical and syntactic information, while more recent ones increasingly use semantic information [55].

The generation of test questions can be performed using various algorithms. In the simplest case, one or more words from the original sentence are removed, thus creating a "fill-in-the-blank" type question. Other approaches include forming questions by selecting an appropriate Wh-word [56], analyzing the relationships between the subject, verb, and object [57], dependency-based patterns [58], syntactic transformation [59], and others. The scientific literature describes various methods for generating distractors for test questions. These include techniques based on parts-of-speech information [60], word frequency [61], domain ontology [62], semantic analysis [44], and others. Detailed reviews of test generation technologies are presented in [55,63–65].

The classic approaches to test creation have some significant limitations:

- Traditional systems are highly dependent on predefined rules and templates, which limits the ability to generate diverse questions, especially when more complex or adaptive test formats are required.
- These systems require significant manual configuration, both in creating rules and selecting appropriate keys and distractors, making them time-consuming and labor-intensive.
- Adaptation is limited, usually based on prior answers and not easily adjustable to dynamically changing conditions.

- The applications use limited semantic information processing, which reduces their ability to understand and interpret deeper contexts and relationships within the text.

Modern technologies based on artificial intelligence and large language models (LLMs) provide a rich set of tools that allow for easy overcoming of the aforementioned limitations. Thanks to these technologies, the process of automated creation of personalized tests can be fully innovated, increasing not only efficiency but also the quality of the tests themselves.

Artificial intelligence and LLMs can analyze and understand texts at a deeper level, enabling the generation of questions with a high degree of accuracy and relevance. They can create diverse questions tailored to the individual needs and knowledge levels of learners by utilizing contextual information and semantic relationships within the text.

Interesting possibilities in this direction are offered by AI-powered chatbots such as ChatGPT, Gemini, Claude, and others. They can be integrated into educational platforms and provide dynamic tests. These chatbots not only generate questions but also offer distractor answers that are sufficiently challenging and relevant, increasing the difficulty level and preventing easy guessing.

### 3. Using ChatGPT for Test Creation

In recent months, we have conducted numerous experiments through which we tested various use cases of ChatGPT (Version gpt-3.5turbo) for educational purposes. Our experience has been positive, and we see great potential for further development.

In 2023, we tested ChatGPT's ability to create assignments aimed at training and assessing higher-order thinking skills according to Bloom's Taxonomy. Tasks and educational resources were successfully created for the course on non-relational databases, with specific requirements set for engaging skills such as analysis, synthesis, and evaluation [36].

In [66], our experience with the use of AI tools in teaching the topic of "Electric Current" in Physics to 7th-grade students is presented. Special attention is given to ChatGPT in supporting students during the self-learning process and in developing coursework.

In another experiment in the Database course, three generative AI tools were integrated into Moodle: "AI Text to Questions Generator" for automated test generation, "AI Text to Image" for generating images based on a given text, and "OpenAI Chat Block"—a virtual assistant based on ChatGPT. The students were assigned to complete five tasks sequentially: to complete one task independently without using AI; to research a new technology on their own, using the integrated virtual AI chatbot; to answer a short AI-generated test with five questions assessing their understanding of the new technology; to demonstrate their ability to use the new technology by extending their solution to the first task. The final task was to write a short essay, without using AI technologies, explaining the advantages and technologies involved in using the new technology. The experiment was successful, and the results were published in [67].

Our subsequent work focuses on creating educational games with ChatGPT. In [68], the process of developing educational games with ChatGPT is described. It is iterative and includes several steps: initial formulation of requirements, testing, adding new functionalities or design elements, improvements, and bug fixing. In [69], a systematic approach to creating educational games with ChatGPT is proposed. Examples of different types of games are explored—flashcards, quizzes, puzzles, matching games, and fill-in-the-blanks—emphasizing certain specific features in the creation of prompts and the iterative process of game improvement and bug fixing.

All of our experiments unequivocally proved that ChatGPT can be effectively used in education. For this reason, we focused on developing a software system for automated test creation. Our idea is to personalize the entire testing process while providing the teacher with the ability to set a variety of specific criteria for test questions in order to support their teaching strategies and approaches.

When developing tests, educators usually take into account many factors, such as the studied content, the age group of the learners, their level of knowledge, cultural characteristics and differences, and many others. This complexity can be transferred to

ChatGPT by specifying numerous specific requirements for generating tests. These can include, for example:

- Purpose of the test: the specific purpose of the test to be considered when creating the questions (e.g., assessment of understanding, self-preparation, competition preparation);
- Topic coverage: distribution of questions across different subtopics (e.g., five questions for each of the four subtopics within one main topic);
- Difficulty level: the complexity level of the questions (e.g., easy, medium, hard);
- Question type: the type of questions to be generated (e.g., multiple choice, open-ended questions, true/false, fill-in-the-blank);
- Question format: additional requirements for the format of the questions (e.g., use of diagrams or charts, inclusion of case studies or scenarios);
- Question style: the type of expression (e.g., academic, conversational, technical);
- Number of answers: how many answers the multiple-choice questions should have (e.g., four possible answers, one correct).
- Feedback: inclusion of explanations or comments for correct and incorrect answers (e.g., a brief explanation for each correct answer);
- Specific guidelines: additional instructions or criteria for creating the questions (e.g., avoiding questions with misleading answers, including questions that require critical thinking);
- Use of sources: inclusion of questions based on specific sources or texts (e.g., questions based on a specific article, textbook, or study);
- Context and examples: inclusion of questions with context or examples (e.g., questions that use real-life situations or case studies);
- Test format: the output format in which ChatGPT should provide the questions and answers (e.g., table, text file, Excel file);
- Target audience: the age group or education level of the learners (e.g., high school students, university students, professionals), etc.

An example request for test creation might look like this (Box 1):

**Box 1.** Create a test with the following characteristics.

Subject: Databases
Subtopic: Relational Algebra
Difficulty Level: Medium
Question Types: Multiple choice, true/false, fill-in-the-blank, short answer, graphical questions
Number of Questions: 25
Answer Format: docx file
Target Audience: University students
Language: Bulgarian
Number of Answers (for multiple choice): 4 possible answers, 1 correct
Topic Coverage: 5 questions on selection, 5 questions on projection, 5 questions on union, 5 questions on difference, 5 questions on Cartesian product
Purpose of the Test: Assess understanding and application of relational algebra
Question Style: Academic
Feedback: Explanation for the correct answer
Specific Guidelines: Avoid misleading questions, include examples and case studies, focus on critical thinking and application of theoretical knowledge, use ER diagrams and SQL queries as part of the questions
Training for Specific Skills: Data analysis and synthesis, data retrieval and manipulation through relational algebra, problem-solving skills in the context of databases

By fulfilling this request, the chatbot will create a test and display it in the workspace. The educator can copy and use it appropriately. The inconvenience in this case is that the educator will find it difficult to manage the tests created in this manner. Their work can be facilitated by developing a specialized application for test generation based on ChatGPT. The potential benefits are:

- Reusability: Questions and tests can be stored in a database and reused multiple times. This eliminates the need for the educator to store tests in files, which makes searching for and reusing tests more difficult.
- Personalization: The application can include specific criteria for creating tests that are tailored to a particular subject area, educator, or educational institution. This allows for the creation of tests adapted to specific needs and teaching styles.
- Flexibility and adaptability: Developing a custom application can provide greater flexibility in design and features, allowing adaptation to various test formats and question types.
- Data protection: Using a proprietary software product allows for control over the security and privacy of learners' data. This helps avoid the risks associated with using confidential data by third parties.
- Specific educational methodologies: The educator can use unique teaching methods that are not supported by existing applications, and the custom application can be designed in a way that supports their use.
- Integration with other systems: The custom application can be more easily integrated with other internal systems that the educational institution or teacher already uses.

## 4. AI-Powered Quiz Creation and Management Tool

### 4.1. Technologies and Methods Used

The main technologies used for the development of the application are the Google Firebase cloud platform and the ChatGPT API for generating test questions.

Google Firebase is a platform for developing mobile and web applications provided by Google. It offers a set of services and tools that help developers create, test, manage, and scale their applications. These include a user authentication system, NoSQL databases, hosting for web applications, server services for running backend code in the cloud, tools for analyzing user behavior, tools for monitoring and optimizing application performance, and more.

For data management in the application, the NoSQL database management system Firebase Realtime Database has been used. Besides standard functionalities, it includes built-in security mechanisms, data encryption, and automatic backup creation mechanisms, ensuring data reliability and security. It supports resource scaling capabilities according to the application's needs. The adaptation of resources based on the number of active users, data volume, etc., ensures optimal efficiency and performance. Firebase Database integrates easily with other services, enabling future support and development of the educational platform.

The OpenAI API is a platform that provides access to various OpenAI models, such as GPT-3.5, GPT-4, DALL-E, Whisper, Codex, and more. The ChatGPT API is a part of the OpenAI API, which allows developers to integrate ChatGPT's capabilities into applications, websites, services, and more. Through this API, developers have access to various language models, can create interactive chatbots, generate texts for different purposes, use them for text analysis and interpretation, and more. To use the OpenAI API, registration with OpenAI is required, from which an API key can be obtained to access the service. Requests to the API are made via HTTP POST requests. The request typically includes a text input (prompt) to which the model should respond. The responses from the API are returned as JSON objects containing the generated text.

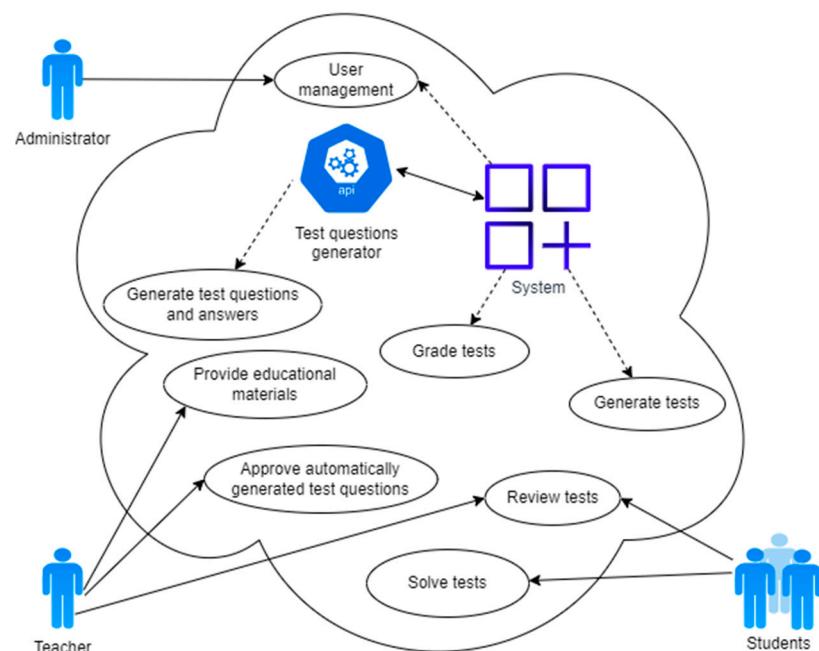
### 4.2. Key Roles and Functionalities

The main roles in the quiz management system are administrator, educator, and learner. An important element in the system is the module for automated test generation, which in the current solution uses the OpenAI API but, in the future, can integrate other AI tools. The key activities related to working with quizzes performed by these actors are:

- User management: The administrator can manage user accounts, including creating, modifying, and deleting accounts.

- Entering text with educational content: The educator enters text—educational content, based on which test questions and answers should be created. The correct answer is marked as such.
- Generating questions and answers: The system uses the specialized module and ChatGPT API to generate multiple questions and answers based on the entered text.
- Approving test questions: The educator reviews the generated questions and answers and approves the appropriate ones according to their own judgment. The approved answers are saved in the database. There is an option for subsequent editing of questions and answers.
- Creating quizzes: The educator can create quizzes by selecting from the test questions stored in the database.
- Taking quizzes: Learners take quizzes as assigned by the educator.
- Grading quizzes: The system automatically grades the quiz submissions provided by the learners.
- Reviewing quizzes: Learners have the opportunity to review the graded quizzes and the mistakes they made.

The main roles and functionalities are illustrated in Figure 1.



**Figure 1.** Key roles and functionalities.

When generating test questions, in addition to the standard aspects concerning the topic and target group, the educator has the option to specify the following characteristics for generating test questions:

- Bloom's taxonomy level, e.g., Knowledge, Comprehension, Application, Analysis, Synthesis, Evaluation
- Difficulty of questions, e.g., easy, medium, difficult
- Question style, e.g., academic, conversational, technical
- Feedback, e.g., no feedback, correct/incorrect (i.e., only indicates if the answer is correct or incorrect), a brief explanation for each correct answer, etc.
- Specific guidelines for questions, e.g., avoiding questions with misleading answers, creating questions that require critical thinking, including real-life examples, incorporating interdisciplinary questions, short and precise formulations, etc.
- Training for specific skills, e.g., computational skills, problem-solving, reading comprehension, data interpretation skills, language skills, research skills, etc.

It should be noted here that these test characteristics are chosen by a specific educator. Any other characteristics can be added in a similar manner.

#### 4.3. Communication between the Application and LLM APIs. Prompt Engineering

A Large Language Model is a type of machine learning model trained to understand and generate text in natural language. LLMs are based on deep neural networks and typically contain billions of parameters, enabling them to handle complex text-related tasks such as translation, summarization, question-answering, text generation, foreign language translation, and more [70]. LLMs are widely used in chatbots, auto-completion systems, search engines, and other natural language processing services.

The process of creating and refining input text prompts, which are fed to LLMs like GPT-3, GPT-4, and others, to achieve the desired outcomes is known as Prompt Engineering [71]. This is a crucial aspect of interacting with LLMs, as well-formulated prompts can significantly improve the quality and accuracy of the responses generated by the model.

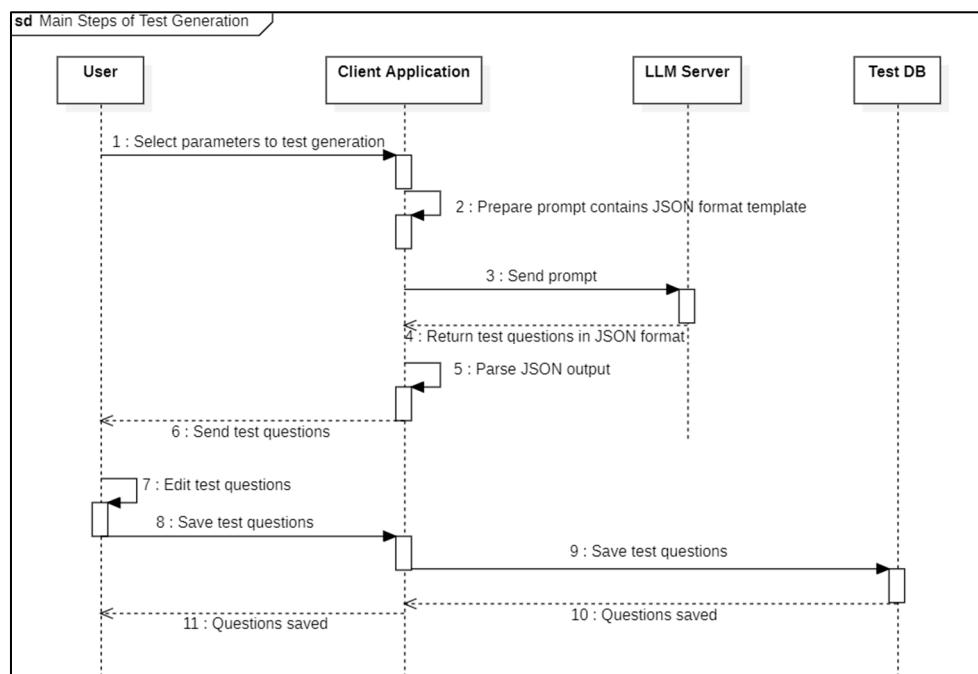
Communication between the client application and the LLM requires careful formulation of the questions directed to the model. The usual approach, where the user asks a question in natural language and receives a free-text response, is unsuitable for subsequent automated processing of this text by the application. If the results are returned in a structured format, it will significantly facilitate their automated processing by the client application.

JSON is a widely used standard for data transfer in internet communication between applications. In the context of LLMs, generating output in JSON format provides a clear and organized data structure that the client application can easily interpret and process. Many of the latest versions of leading LLM APIs, including those from OpenAI and Gemini, offer the capability to generate structured JSON output. However, different APIs use different functions and parameters for this purpose, which can complicate the logic of the client application and hinder the future integration of new LLMs into it.

Our research shows that all major LLMs, such as ChatGPT [72], Gemini [73], Llama [74], Claude [75], Mistral [76], Cohere [77], Reka [78], DeepSeek [79], and others, can generate structured JSON output if the query is properly formulated and includes instructions for the desired output format. This allows the client application to work uniformly with different LLM APIs using the same prompt and to receive results in a format defined by the developers.

Figure 2 presents a UML sequence diagram illustrating the main steps in the process of automated test question generation using an LLM. These steps include:

- Teacher Configuration: A user with the role of a teacher configures the parameters for generating test questions, such as the topic, target group, test complexity, question types, and other specific requirements.
- Prompt Generation: Upon starting the process, the application generates a prompt that includes the teacher-defined parameters and a JSON template for the response structure.
- Prompt Submission: The client application sends this prompt to the LLM through the appropriate model's API.
- LLM Response: The LLM returns an output that contains the generated test questions and answers in JSON format.
- Output Processing: The client application processes the received JSON output and displays the questions and answers in a user-friendly format.
- Teacher Review and Storage: The teacher reviews and edits the questions, after which they are saved in the application's database.



**Figure 2.** Main steps of test generation process.

A prompt template that can be used for all of the reviewed LLMs is presented in Figure 3. It returns a structured JSON result containing test questions and answers.

Create <n> sample test questions with <m> answers in the field of <subject area> in the following JSON format:

```
{
  "questions": [
    {
      "text": "Question text",
      "answers": [
        {
          "text": "Answer text",
          "isCorrect": true or false,
          "feedback": "Explanation of why the answer is correct or incorrect"
        }
      ]
    }
  ]
}
```

**Figure 3.** A prompt designed for an LLM that specifies a format for JSON output.

Sample values for the parameters <n>, <m>, and <subject area> are “5”, “4”, and “Databases”, respectively.

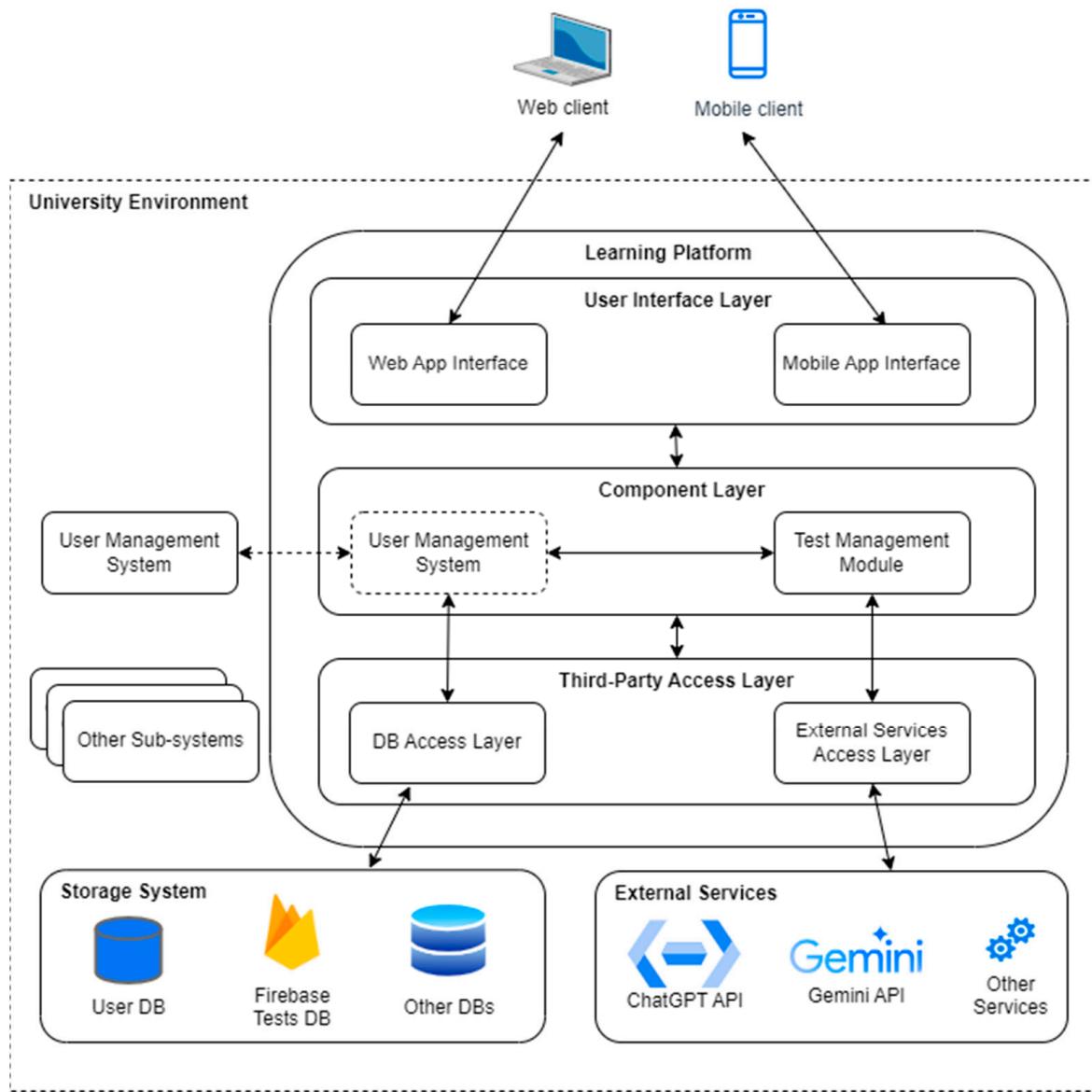
The presented JSON structure consists of an array of questions, each of which has two sub-elements: the question text and an array of answers. The answers, in turn, have three sub-elements: the answer text, a boolean field indicating whether the answer is correct or incorrect, and feedback explaining why the answer is correct or incorrect.

Using the context provided by the JSON template, LLMs can determine the logic of the structure and populate the corresponding JSON elements with appropriate content. If only data types are specified without detailed explanations for the fields, this could make it difficult for the LLM to “understand” and correctly apply the logic of the JSON template.

Although JSON is a widely used and convenient format, other formats for structured output can also be chosen, such as XML, YAML, Markdown, Protobuf, SQL, etc.

#### 4.4. System Architecture

The quiz generation and management application can operate independently or be integrated into a learning platform. The learning platform, in turn, can be integrated into a comprehensive university system, providing standardized user access to the services of all university subsystems. A summarized architecture of a learning platform is presented in Figure 4 and includes the Client Level, User Interface, Component Level, Database and External Services Access Level, Database Management Systems, and other external services.



**Figure 4.** System Architecture.

Users access the system's functionalities through an interface provided by a web or mobile application. Regardless of the interface used, the user should be able to freely access all functionalities. A key advantage of using mobile applications is easy access from any location, while for web applications, it is the better organization of work screens.

Important functionalities for different user roles (administrators, educators, and learners) at the presentation level are:

- Managing user profile information
- Entering data and text for generating test questions; approving and editing test questions and answers; configuring quizzes

- Taking quizzes, reviewing received results, and more.

At the second level are the components that implement the system's core business logic. The user management module needs to coordinate its data with the external environment. The test management module provides capabilities for:

- Generating test questions and configuring tests
- Executing and automatically grading tests
- Storing the tests in the database
- Personalization for users, utilizing the User Management System.

An important part of the test management system is the test question generation module, which is developed using Node.js technology. In the current version, the test questions and their answers are generated using the ChatGPT API based on textual materials provided by the educator. The generated questions are approved by the respective educator and then stored in the database. They can be edited and used to create tests.

A comprehensive learning management system can support many other components, such as "Course and Educational Material Management", "Interactive Lessons", "Forums", and more [80].

The second-level components communicate with database management systems and other external systems and services through modules from the third-party access level.

The Google Firebase database management system provides reliable and efficient storage for all essential data [81], including information about quizzes, their questions, user profiles, and the results of completed quizzes. Communication between all components is carried out via RESTful requests, ensuring security and efficiency in the system's communication [82]. The system offers possibilities for future expansion through the integration of other databases and external services, including other AI systems, such as Gemini, Llama, Claude, etc.

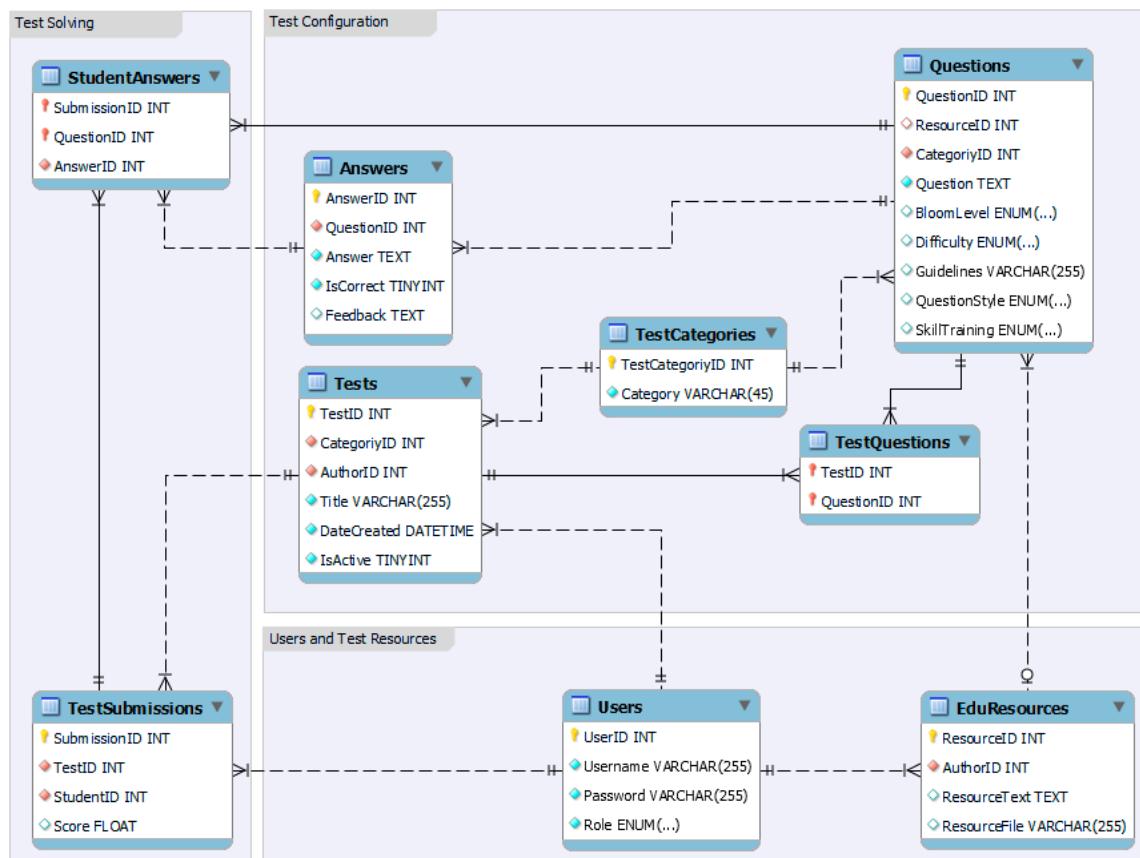
#### 4.5. System Database

The main entities in the database for storing the generated test questions, the composed tests, and their execution are presented in Figure 5. For visual representation of the database, an Enhanced Entity-Relationship diagram was created during the design process using MySQL Workbench.

The main entities in the quiz database are:

- Users: Contains information about all system users, including administrators, teachers, and learners. Attributes include a unique identifier, username, password, and role in the system.
- Educational Resources: Stores resources in the form of text or files, which are used for generating test questions. Includes a unique resource identifier, text, file name, and author identifier, which is a foreign key to the user who provided the resource.
- Test Categories: Stores test categories that define both individual questions and the overall test. Attributes include a category identifier and its name.
- Questions: Contains the questions generated through the OpenAI API. Includes a question identifier, question category identifier, question content, and an optional foreign key to the educational resource, as questions can be created without being tied to specific educational resources. Optional attributes for additional categorization of the question, used when generating it from the OpenAI API, include Bloom Level, Difficulty, Guidelines, Question Style, and Skill Training.
- Answers: Stores all answers associated with the questions. Consists of an answer identifier, question identifier, answer text, feedback for the answer, and a boolean flag indicating whether the answer is correct.
- Tests: Stores information about the created tests, including a unique test identifier, test title, test author identifier, test category identifier, test creation date, and a boolean flag indicating whether the test is active.

- Test Questions: Establishes a relationship between tests and questions and contains the identifiers of the test and the question.
- Test Submissions: Records the test solutions submitted by learners and includes a unique submission identifier, test identifier, student identifier, and score.
- Student Answers: Contains the answers given by learners to the test questions, including the submission identifier, question identifier, and an optional identifier of the given answer.



**Figure 5.** Enhanced Entity-Relationship diagram of the data model.

## 5. Experiments

During the development of the application, numerous tests and analyses were conducted to evaluate the applicability of the test questions generated by ChatGPT. The system was experimented with using educational materials on the Database course. An example of the request to the ChatGPT API and the obtained result with the automatically generated questions and answers is presented in Figures 6 and 7. The request does not include the text of the educational material, and only the generated test questions are shown in the result.

In Figure 6, a textual request (prompt) for creating 3 test questions is shown, with the following characteristics: Difficulty Level: High; Bloom's Taxonomy Level: Evaluation; Question Type: Multiple Choice; Number of Answers: 4 possible answers, 1 correct; Style of the questions: Technical; Feedback: Detailed explanation for the correct answer; Specific Guidelines: Inclusion of real-life examples; Training for Specific Skills: Problem-solving skills in the context of databases. In Figure 7, the response is shown in JSON format.

In Figures 8 and 9, screens of key application functionalities are shown, concerning the generation of questions based on a given text, selection of questions to be saved in the database for future use, and selection of test questions for creating a specific quiz.

```

var text = `Generate three questions with the difficulty level of high. The style of the questions: Technical;
Bloom's Level: Evaluation; Question Type: Multiple Choice; Number of Answers: four possible answers, one correct;
Feedback: Detailed explanation for the correct answer; Specific Guidelines: Include real-life examples;
Training for Specific Skills: Problem-solving skills in the context of databases from the following text:
${questionText}. The questions should be in the following format: {"questions": [{"question": "", "answers": [{"answer": "", "isCorrect": false, "feedback": ""}, {"answer": "", "isCorrect": true, "feedback": ""}]}]}`;

```

**Figure 6.** OpenAI API request for generating test questions with a specified answer format.

```

{
  "questions": [
    {
      "question": "Why is duplicate information considered bad in a database design?",
      "answers": [
        {
          "answer": "It takes up extra space and increases the likelihood of errors and inconsistencies.", "isCorrect": true, "feedback": "Duplicate information wastes space and can lead to errors and inconsistencies in the database, making it less efficient and reliable."},
        {
          "answer": "It helps to organize information more effectively.", "isCorrect": false, "feedback": "While organizing information is important, duplicate information can actually hinder the efficiency and accuracy of the database."},
        {
          "answer": "It allows for easier data manipulation.", "isCorrect": false, "feedback": "Duplicate information can actually complicate data manipulation and lead to inaccuracies."},
        {
          "answer": "It ensures data integrity.", "isCorrect": false, "feedback": "While data integrity is important, duplicate information does not contribute to it."}]
    },
    {
      "question": "What is the purpose of a primary key in a database table?",
      "answers": [
        {
          "answer": "To provide a way to quickly associate data from multiple tables.", "isCorrect": true, "feedback": "The primary key is used to uniquely identify each row in a table and to establish relationships between tables."},
        {
          "answer": "To store calculated data.", "isCorrect": false, "feedback": "The primary key is not used to store calculated data, but to uniquely identify each row."},
        {
          "answer": "To contain redundant information.", "isCorrect": false, "feedback": "The primary key should not contain redundant information, as it is meant to be a unique identifier."},
        {
          "answer": "To ensure data completeness.", "isCorrect": false, "feedback": "While data completeness is important, the primary key's main purpose is to uniquely identify each row."}]
    },
    {
      "question": "Why is it important to apply the normalization rules in a database design?",
      "answers": [
        {
          "answer": "To ensure that each cell in the table contains a single value.", "isCorrect": false, "feedback": "While this is a rule of normalization, it is not the main reason for applying the normalization rules."},
        {
          "answer": "To eliminate data redundancy and improve data integrity.", "isCorrect": true, "feedback": "Normalization helps to reduce data redundancy, improve data integrity, and ensure data consistency across the database."},
        {
          "answer": "To create complex data calculations.", "isCorrect": false, "feedback": "Normalization is more focused on data organization and structure, rather than complex calculations."},
        {
          "answer": "To speed up data processing.", "isCorrect": false, "feedback": "Normalization may improve data processing efficiency, but its primary goal is to reduce redundancy and improve data integrity."}]
    }
  ]
}

```

**Figure 7.** OpenAI API response with 3 generated test questions.

AIED

Role: teacher [Logout](#)

Category <b>Databases</b>	Feedback Type <b>Detailed explanation ...</b>
Number of questions <b>3</b>	Specific Guidelines Include real-life examples
Bloom's Level <b>Evaluation</b>	Skill Training Problem-solving skills in the context of databases
Difficulty <b>high</b>	
Question Style <b>technically</b>	

Text for question generation  
**WHAT IS GOOD DATABASE DESIGN?**

Certain principles guide the database design process. The first principle is that duplicate information (also called redundant data) is bad, because it wastes space and increases the likelihood of errors and inconsistencies. The second principle is that the correctness and completeness of information is important. If your database contains incorrect information, any reports that pull information from the database will also contain incorrect information. As a result, any decisions you make that are based on those reports will then be misinformed.

[Choose File](#) [Generate Questions](#)

Question 1: Why is duplicate information considered bad in a database design?

1. It takes up extra space and increases the likelihood of errors and inconsistencies. - Duplicate information wastes space and can lead to errors and inconsistencies in the database, making it less efficient and reliable.

2. It helps to organize information more effectively.

3. It allows for easier data manipulation.

4. It ensures data integrity.

Question 2: What is the purpose of a primary key in a database table?

1. To provide a way to quickly associate data from multiple tables. - The primary key is used to uniquely identify each row in a table and to establish relationships between tables.

2. To store calculated data.

3. To contain redundant information.

4. To ensure data completeness.

Question 3: Why is it important to apply the normalization rules in a database design?

1. To ensure that each cell in the table contains a single value.

2. To eliminate data redundancy and improve data integrity. - Normalization helps to reduce data redundancy, improve data integrity, and ensure data consistency across the database.

3. To create complex data calculations.

4. To speed up data processing.

[Save selected questions](#)

**Figure 8.** Screen for generating and approving test questions.

AIED

Role: teacher [Logout](#)

**Selected questions: 8 of 16**

Name of the test <b>Databases 2024</b>	<b>Questions:</b>
Choose category <b>Databases</b>	<input type="checkbox"/> What is the purpose of a primary key in a database table?
Bloom Level <b>Evaluation</b>	<input checked="" type="checkbox"/> What is one of the principles of good database design mentioned in the text?
Difficulty <b>high</b>	<input checked="" type="checkbox"/> Why is it important to apply the normalization rules in a database design?
Question Style <b>technically</b>	<input checked="" type="checkbox"/> Why is duplicate information considered bad in a database design?

[Save Test](#)

<input checked="" type="checkbox"/> Why is it important to specify primary keys in database tables?
<input checked="" type="checkbox"/> What is the purpose of determining the purpose of the database at the beginning of the design process?
<input checked="" type="checkbox"/> Why is duplicate information considered bad in database design?
<input type="checkbox"/> What should be considered when determining the columns in a table during database design?
<input checked="" type="checkbox"/> Why is it important to divide information into tables in database design?

**Figure 9.** Selection of questions for constructing a quiz.

The experiments conducted showed that the application can significantly facilitate the work of educators. The functionality for editing the created test questions is practical. Automatically generated questions save a lot of time, allowing educators to focus on other important tasks. The quick and easy generation of a large number of questions enables the maintenance of up-to-date tests and the avoidance of repetitive questions. Additionally, the educator can select the most appropriate questions depending on the specific educational goals set, such as difficulty, cognitive skills, etc.

## 6. Future Work

The initial experiments showed that the developed software prototype can be a useful assistant in the teacher's work. To assess its usability, effectiveness, and impact on the learning process, experiments in a real educational environment need to be conducted. Therefore, the work is planned to continue in several directions:

- Empirical study: The application will be used for generating and conducting tests in existing courses in computer sciences, allowing it to be evaluated by both teachers and students. Through observation and conducting surveys, the usability of the application, its impact on the teachers' work, the learning process, and the learning outcomes will be assessed.
- Comparative analysis with traditional test creation methods: Studies will be conducted to gather opinions from both teachers and students regarding the quality, relevance, adaptability, and evaluation of questions generated with the help of AI.
- Exploration of the application's potential in different fields of education: This research will focus on the adaptability of the application across various disciplines outside of computer and technical sciences. Experiments will be conducted with courses in social and humanities fields to test its ability to generate questions in different contexts and at varying levels of complexity.
- Evaluation of applicability in different teaching strategies: The application will be studied in various teaching strategies and approaches, such as problem-based and project-based learning, self-learning, group work, etc. Experiments and analyses will be conducted to explore how the application can be used to support different teaching strategies and forms of learning.

These planned studies will provide a clearer understanding of the applicability and effectiveness of the developed prototype and offer ideas and directions for its improvement.

## 7. Conclusions

The article presents the development of a software application for automated test generation and management, utilizing generative artificial intelligence. It leverages Google Firebase and the ChatGPT API to integrate AI into the educational process—generating questions based on textual materials, evaluating responses in real time, and providing personalized feedback to learners. This significantly reduces the time and effort required to prepare tests and improves the quality of assessment.

One of the key contributions of our work is the development of a flexible and user-friendly application that can be easily adapted to various subjects and different levels of complexity. The ability to automatically generate questions based on various criteria set by the instructors, as well as the storage and reuse of tests, greatly facilitates the work of educators.

The experiments conducted with the application demonstrate that it can effectively generate test questions that meet various educational needs. Tests were generated using educational materials on the Databases course, and questions and answers were evaluated in terms of their relevance to the course content and their level of cognitive complexity. The results indicate that the application can significantly facilitate the work of educators and improve the efficiency of the learning process.

In conclusion, the developed application offers an innovative approach to automating the process of creating and evaluating tests in the educational space. It combines the power of AI and cloud technologies to provide faster and more accurate assessment solutions while offering opportunities for personalizing the learning process. In the future, the system can be expanded by integrating new AI tools and technologies, increasing the adaptability and personalization of tests according to the specific needs of learners.

**Author Contributions:** Conceptualization, E.H., T.R. and G.D.; methodology, S.H. and T.R.; software, I.I. and T.R.; validation, E.H. and T.R.; investigation, E.H. and T.R.; resources, T.R. and I.I.; writing—original draft preparation, I.I., E.H. and S.H.; writing—review and editing, T.R. and G.D.; visualization, T.R. and I.I.; supervision, G.D. and S.H.; project administration, T.R.; funding acquisition, E.H. and S.H. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was partly funded by the MUPD23-FMI-021 and SP23-FMI-008 projects of the Research Fund of the University of Plovdiv “Paisii Hilendarski”.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

1. Zhang, K.; Aslan, A.B. AI technologies for education: Recent research & future directions. *Comput. Educ. Artif. Intell.* **2021**, *2*, 100025. [[CrossRef](#)]
2. Rojas, M.P.; Chiappe, A. Artificial Intelligence and Digital Ecosystems in Education: A Review. *Technol. Knowl. Learn.* **2024**, *1*–18. [[CrossRef](#)]
3. Kasneci, E.; Sessler, K.; Küchemann, S.; Bannert, M.; Dementieva, D.; Fischer, F.; Gasser, U.; Groh, G.; Günemann, S.; Hüllermeier, E.; et al. ChatGPT for good? On opportunities and challenges of large language models for education. *Learn. Individ. Differ.* **2023**, *103*, 102274. [[CrossRef](#)]
4. Jeon, J.; Lee, S. Large language models in education: A focus on the complementary relationship between human teachers and ChatGPT. *Educ. Inf. Technol.* **2023**, *28*, 15873–15892. [[CrossRef](#)]
5. Chen, W.; Samuel, R.; Krishnamoorthy, S. Computer Vision for Dynamic Student Data Management in Higher Education Platform. *J. Mult.-Valued Log. Soft Comput.* **2021**, *36*, 5.
6. Agbo, G.C.; Agbo, P.A. The role of computer vision in the development of knowledge-based systems for teaching and learning of English language education. *ACCENTS Trans. Image Process. Comput. Vis.* **2020**, *6*, 42–47. [[CrossRef](#)]
7. Kucak, D.; Juricic, V.; Dambic, G. Machine Learning in Education—A Survey of Current Research Trends. In Proceedings of the 29th DAAAM International Symposium, Vienna, Austria, 24–27 October 2018. [[CrossRef](#)]
8. Hadzhikolev, E.; Hadzhikoleva, S.; Yotov, K.; Borisova, M. Automated Assessment of Lower and Higher-Order Thinking Skills Using Artificial Intelligence Methods. *Commun. Comput. Inf. Sci.* **2021**, *1521*, 13–25.
9. Chui, K.T.; Lee, L.K.; Wang, F.L.; Cheung, S.K.S.; Wong, L.P. A Review of Data Augmentation and Data Generation Using Artificial Intelligence in Education. *Commun. Comput. Inf. Sci.* **2024**, *1974*, 242–253. [[CrossRef](#)]
10. Ayeni, O.O.; Al Hamad, N.M.; Chisom, O.N.; Osawaru, B.; Adewusi, O.E. AI in education: A review of personalized learning and educational technology. *GSC Adv. Res. Rev.* **2024**, *18*, 261–271. [[CrossRef](#)]
11. Hwang, G.J.; Xie, H.; Wah, B.W.; Gašević, D. Vision, challenges, roles and research issues of Artificial Intelligence in Education. *Comput. Educ. Artif. Intell.* **2020**, *1*, 100001. [[CrossRef](#)]
12. Borenstein, J.; Howard, A. Emerging challenges in AI and the need for AI ethics education. *AI Ethics* **2021**, *1*, 61–65. [[CrossRef](#)]
13. Sofianos, K.C.; Stefanidakis, M.; Kaponis, A.; Bukauskas, L. Assist of AI in a Smart Learning Environment. *IFIP Adv. Inf. Commun. Technol.* **2024**, *714*, 263–275. [[CrossRef](#)]
14. Harry, A.; Sayudin, S. Role of AI in Education. *Interdisciplinary J. Hummanity* **2023**, *2*, 260–268. [[CrossRef](#)]
15. Nurhayati, T.N.; Halimah, L. The Value and Technology: Maintaining Balance in Social Science Education in the Era of Artificial Intelligence. In Proceedings of the International Conference on Applied Social Sciences in Education, Bangkok, Thailand, 14–16 November 2024; Volume 1, pp. 28–36. [[CrossRef](#)]
16. Nunez, J.M.; Lantada, A.D. Artificial intelligence aided engineering education: State of the art, potentials and challenges. *Int. J. Eng. Educ.* **2020**, *36*, 1740–1751.
17. Darayeh, A.A. Acceptance of artificial intelligence in teaching science: Science teachers' perspective. *Comput. Educ. Artif. Intell.* **2023**, *4*, 100132. [[CrossRef](#)]
18. Briganti, G.; Le Moine, O. Artificial intelligence in medicine: Today and tomorrow. *Front. Med.* **2020**, *7*, 27. [[CrossRef](#)]
19. Kandlhofer, M.; Steinbauer, G.; Hirschmugl-Gaisch, S.; Huber, P. Artificial intelligence and computer science in education: From kindergarten to university. In Proceedings of the 2016 IEEE Frontiers in Education Conference (FIE), Erie, PA, USA, 12–15 October 2016. [[CrossRef](#)]
20. Edmett, A.; Ichaporla, N.; Crompton, H.; Crichton, R. Artificial Intelligence and English Language Teaching: Preparing for the Future. *British Council*. 2023. Available online: [https://www.teachingenglish.org.uk/sites/teacheng/files/2024-08/AI\\_and\\_ELT\\_Jul\\_2024.pdf](https://www.teachingenglish.org.uk/sites/teacheng/files/2024-08/AI_and_ELT_Jul_2024.pdf) (accessed on 21 September 2024).

21. Hajkowicz, S.; Sanderson, C.; Karimi, S.; Bratanova, A.; Naughtin, C. Artificial intelligence adoption in the physical sciences, natural sciences, life sciences, social sciences and the arts and humanities: A bibliometric analysis of research publications from 1960–2021. *Technol. Soc.* **2023**, *74*, 102260. [[CrossRef](#)]
22. Crompton, H.; Burke, D. Artificial Intelligence in Higher Education: The State of the Field. *Int. J. Educ. Technol. High. Educ.* **2023**, *20*, 22. [[CrossRef](#)]
23. Xu, W.; Ouyang, F. The application of AI technologies in STEM education: A systematic review from 2011 to 2021. *Int. J. STEM Educ.* **2022**, *9*, 59. [[CrossRef](#)]
24. Rahiman, H.; Kodikal, R. Revolutionizing education: Artificial intelligence empowered learning in higher education. *Cogent Educ.* **2024**, *11*, 2293431. [[CrossRef](#)]
25. Mishra, R. Usage of Data Analytics and Artificial Intelligence in Ensuring Quality Assurance at Higher Education Institutions. In Proceedings of the 2019 Amity International Conference on Artificial Intelligence (AICAI), Dubai, United Arab Emirates, 4–6 February 2019; pp. 1022–1025. [[CrossRef](#)]
26. Dempere, J.; Modugu, K.; Hesham, A.; Ramasamy, L. The impact of ChatGPT on higher education. *Front. Educ.* **2023**, *8*, 1206936. [[CrossRef](#)]
27. Chaudhry, I.; Sarwary, S.; Refae, G.; Chabchoub, H. Time to Revisit Existing Student’s Performance Evaluation Approach in Higher Education Sector in a New Era of ChatGPT—A Case Study. *Cogent Educ.* **2023**, *10*, 2210461. [[CrossRef](#)]
28. Pradana, M.; Elisa, H.; Syarifuddin, S. Discussing ChatGPT in education: A literature review and bibliometric analysis. *Cogent Educ.* **2023**, *10*, 2243134. [[CrossRef](#)]
29. Chinonso, O.; Theresa, A.; Aduke, T. ChatGPT for Teaching, Learning and Research: Prospects and Challenges. *Glob. Acad. J. Humanit. Soc. Sci.* **2023**, *5*, 33–40. [[CrossRef](#)]
30. Aecharungroj, V. “What Can ChatGPT Do?” Analyzing Early Reactions to the Innovative AI Chatbot on Twitter. *Big Data Cogn. Comput.* **2023**, *7*, 35. [[CrossRef](#)]
31. Akiba, D.; Fraboni, M.C. AI-Supported Academic Advising: Exploring ChatGPT’s Current State and Future Potential toward Student Empowerment. *Educ. Sci.* **2023**, *13*, 885. [[CrossRef](#)]
32. O’Connor, S. Open artificial intelligence platforms in nursing education: Tools for academic progress or abuse? *Nurse Educ. Pract.* **2023**, *66*, 103537. [[CrossRef](#)]
33. Stokel-Walker, C. AI bot ChatGPT writes smart essays—should academics worry? *Nature*, 9 December 2022. [[CrossRef](#)]
34. Rahman, M.M.; Watanobe, Y. ChatGPT for Education and Research: Opportunities, Threats, and Strategies. *Appl. Sci.* **2023**, *13*, 5783. [[CrossRef](#)]
35. Grassini, S. Shaping the Future of Education: Exploring the Potential and Consequences of AI and ChatGPT in Educational Settings. *Educ. Sci.* **2023**, *13*, 692. [[CrossRef](#)]
36. Borisova, M.; Hadzhikoleva, S.; Hadzhikolev, E.; Gorgorova, M. Training of higher order thinking skills using ChatGPT. In Proceedings of the International Conference on Virtual Learning, Bucharest, Romania, 15 November 2023; Volume 18, pp. 15–26. [[CrossRef](#)]
37. Osterlind, S.J. What is constructing test items? In *Constructing Test Items. Evaluation in Education and Human Services*; Springer: Dordrecht, The Netherlands, 1998; Volume 25. [[CrossRef](#)]
38. Bugbee, A.C. The Equivalence of Paper-and-Pencil and Computer-Based Testing. *J. Res. Comput. Educ.* **1996**, *28*, 282–299. [[CrossRef](#)]
39. Serbedzija, N.; Kaiser, A.; Hawryszkiewycz, I. E-Quest: A Simple Solution for e-Questionnaires. In Proceedings of the IADIS International Conference e-Society, Ávila, Spain, 16–19 July 2004; pp. 425–432.
40. Bennett, R.E.; Bejar, I.I. Validity and automad scoring: It’s not only the scoring. *Educ. Meas. Issues Pract.* **1998**, *17*, 9–17. [[CrossRef](#)]
41. Thelwall, M. Computer-based assessment: A versatile educational tool. *Comput. Educ.* **2000**, *34*, 37–49. [[CrossRef](#)]
42. Sanchez, L.; Penarreta, D.; Poma, X. Learning Management Systems for Higher Education: A Brief Comparison. *TechRxiv*. 2023. Available online: <https://www.techrxiv.org/doi/full/10.36227/techrxiv.23615523.v1> (accessed on 26 September 2024).
43. Bednarik, L.; Kovács, L. Implementation and assessment of the automatic question generation module. In Proceedings of the 2012 IEEE 3rd International Conference on Cognitive Infocommunications (CogInfoCom), Kosice, Slovakia, 2–5 December 2012; pp. 687–690. [[CrossRef](#)]
44. Pino, J.; Heilman, M.; Eskenazi, M. A selection strategy to improve cloze question quality. In Proceedings of the Workshop on Intelligent Tutoring Systems for Ill-Defined Domains. In Proceedings of the 9th International Conference on Intelligent Tutoring Systems, Montreal, QC, Canada, 23–27 June 2008; pp. 22–32.
45. Das, B.; Majumder, M.; Phadikar, S. A novel system for generating simple sentences from complex and compound sentences. *Int. J. Mod. Educ. Comput. Sci.* **2018**, *10*, 57–64. [[CrossRef](#)]
46. Pabitha, P.; Mohana, M.; Suganthi, S.; Sivanandhini, B. Automatic Question Generation system. In Proceedings of the 2014 International Conference on Recent Trends in Information Technology, Chennai, India, 10–12 April 2014; pp. 1–5. [[CrossRef](#)]
47. Aldabe, I.; Maritxalar, M.; Mitkov, R. A study on the automatic selection of candidate sentences distractors. In Proceedings of the 2009 Conference on Artificial Intelligence in Education: Building Learning Systems that Care: From Knowledge Representation to Affective Modelling, Brighton, UK, 20 July 2009; pp. 656–658.
48. Lin, Y.-C.; Sung, L.-C.; Chen, M.C. An automatic multiple-choice question generation scheme for English adjective understanding. In Proceedings of the Workshop on Modeling, Management and Generation of Problems/Questions in eLearning, 15th

- International Conference on Computers in Education, Hiroshima, Japan, 5–9 November 2007; pp. 137–142. Available online: <https://api.semanticscholar.org/CorpusID:239993403> (accessed on 26 September 2024).
49. Correia, R.; Baptista, J.; Eskenazi, M.; Mamede, N. Automatic Generation of Cloze Question Stems. In *Computational Processing of the Portuguese Language. PROPOR 2012. Lecture Notes in Computer Science*; Caseli, H., Villavicencio, A., Teixeira, A., Perdigão, F., Eds.; Springer: Berlin/Heidelberg, Germany, 2012; Volume 7243. [[CrossRef](#)]
50. Smith, S.; Avinesh, P.; Kilgarriff, A. Gap-fill tests for language learners: Corpus-driven item generation. In Proceedings of the 8th International Conference Natural Lang Process, Kharagpur, India, 8–11 December 2010; pp. 1–6.
51. Mitkov, R.; An Ha, L.; Karamanis, N. A computer-aided environment for generating multiple-choice test items. *Nat. Lang. Eng.* **2006**, *12*, 177–194. [[CrossRef](#)]
52. Araki, J.; Rajagopal, D.; Sankaranarayanan, S.; Holm, S.; Yamakawa, Y.; Mitamura, T. Generating Questions and Multiple-Choice Answers Using Semantic Analysis of Texts. In Proceedings of the 26th International Conference on Computational Linguistics (COLING 2016), Osaka, Japan, 11–16 December 2016; pp. 1125–1136. Available online: <https://aclanthology.org/C16-1107/> (accessed on 26 September 2024).
53. Agarwal, M.; Mannem, P. Automatic Gap-Fill Question Generation from Text Books. In Proceedings of the 6th Workshop on Innovative Use of NLP for Building Educational Applications, Portland, OR, USA, 24 June 2011; pp. 56–64. Available online: <https://aclanthology.org/W11-1407/> (accessed on 26 September 2024).
54. Fattoh, I. Automatic multiple choice question generation system for semantic attributes using string similarity measures. *Comput. Eng. Intell. Syst.* **2014**, *5*, 66–73.
55. CH, D.; Saha, S. Automatic Multiple Choice Question Generation From Text: A Survey. *IEEE Trans. Learn. Technol.* **2020**, *13*, 14–25. [[CrossRef](#)]
56. Majumder, M.; Saha, S. A system for generating multiple choice questions: With a novel approach for sentence selection. In Proceedings of the 2nd Workshop on Natural Language Processing Techniques for Educational Applications, Beijing, China, 31 July 2015; pp. 64–72.
57. Mitkov, R.; Ha, L. Computer-aided generation of multiple-choice tests. In Proceedings of the HLT-NAACL 03 Workshop on Building Educational Applications Using Natural Language Processing, Edmonton, Canada, 31 May 2003; pp. 17–22. [[CrossRef](#)]
58. Afzal, N.; Mitkov, R. Automatic generation of multiple choice questions using dependency-based semantic relations. *Soft Comput.* **2014**, *18*, 1269–1281. [[CrossRef](#)]
59. Heilman, M. Automatic Factual Question Generation from Text. Ph.D. Thesis, Carnegie Mellon University, Pittsburgh, PA, USA, 2011.
60. Goto, T.; Kojiri, T.; Watanabe, T.; Iwata, T.; Yamada, T. Automatic generation system of multiple-choice cloze questions and its evaluation. *Knowl. Manag. E-Learn.* **2010**, *2*, 210–224. [[CrossRef](#)]
61. Liu, C.-L.; Wang, C.-H.; Gao, Z.-M.; Huang, S.-M. Applications of lexical information for algorithmically composing multiple-choice cloze items. In Proceedings of the second workshop on Building Educational Applications Using NLP, Michigan, USA, 29 June 2005; pp. 1–8.
62. Papasalouros, A.; Kanaris, K.; Kotis, K. Automatic generation of multiple choice questions from domain ontologies. In Proceedings of the International Conference e-Learning 2008, Amsterdam, The Netherlands, 22–25 July 2008; pp. 427–434.
63. Das, B.; Majumder, M.; Phadikar, S.; Sekh, A. Automatic question generation and answer assessment: A survey. *Res. Pract. Technol. Enhanc. Learn.* **2021**, *16*, 5. [[CrossRef](#)]
64. Kurdi, G.; Leo, J.; Parsia, B.; Sattler, U.; Al-Emari, S. A Systematic Review of Automatic Question Generation for Educational Purposes. *Int. J. Artif. Intell. Educ.* **2020**, *30*, 121–204. [[CrossRef](#)]
65. Divate, M.; Salgaonkar, A. Automatic question generation approaches and evaluation techniques. *Curr. Sci.* **2017**, *113*, 1683–1691. [[CrossRef](#)]
66. Borisova, M.; Hadzhikoleva, S.; Hadzhikolev, E. Use of Artificial Intelligence technologies in studying the phenomenon of electric current in physics education. In Proceedings of the International Conference on Virtual Learning, Bucharest, Romania, 26–27 October 2023; Volume 18, pp. 215–224. [[CrossRef](#)]
67. Gorgorova, M.; Gaftandzhieva, S.; Hadzhikoleva, S. Use of Artificial Intelligence Tools in Moodle. In Proceedings of the Second National Scientific and Practical Conference “Digital Transformation of Education—Problems and Solutions”, Ruse, Bulgaria, 24–25 April 2024. (In Bulgarian).
68. Hadzhikoleva, S.; Gorgorova, M.; Hadzhikolev, E.; Pashev, G. AI-Driven Approach to Educational Game Creation. In Proceedings of the 16th International conference ICT Innovations, Ohrid, North Macedonia, 28–30 September 2024.
69. Hadzhikoleva, S.; Gorgorova, M.; Hadzhikolev, E.; Pashev, G. Creating Educational Games with ChatGPT. *Educ. Technol.* **2024**, *15*, 212–218.
70. Zhang, Y.; Chen, X.; Jin, B.; Wang, S.; Ji, S.; Wang, W.; Han, J. A Comprehensive Survey of Scientific Large Language Models and Their Applications in Scientific Discovery. *arXiv* **2024**, arXiv:2406.10833. [[CrossRef](#)]
71. Sahoo, P.; Singh, A.; Saha, S.; Jain, V.; Mondal, S.; Chadha, A. A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications. *arXiv* **2024**, arXiv:2402.07927. [[CrossRef](#)]
72. ChatGPT. Available online: <https://chatgpt.com/> (accessed on 14 August 2024).
73. Gemini. Available online: <https://gemini.google.com/> (accessed on 14 August 2024).
74. Llama. Available online: <https://llama.meta.com/> (accessed on 14 August 2024).

75. Claude. Available online: <https://claude.ai/> (accessed on 14 August 2024).
76. Mistral. Available online: <https://chat.mistral.ai/> (accessed on 14 August 2024).
77. Cohere. Available online: <https://coral.cohere.com/> (accessed on 14 August 2024).
78. Reka. Available online: <https://chat.reka.ai/> (accessed on 14 August 2024).
79. DeepSeek. Available online: <https://chat.deepseek.com/> (accessed on 14 August 2024).
80. Shannon, L.-J.; Rice, M. Scoring the open source learning management systems. *Int. J. Inf. Educ. Technol.* **2017**, *7*, 432–436. [[CrossRef](#)]
81. Gaurav, S.; Shrivastava, V.; Pandey, A.; Shrivastava, V. A Survey of Firebase Technology and It's Features. *SSRN Electron.* **2024**. [[CrossRef](#)]
82. Biehl, M. *RESTful API Design: Best Practices in API Design with REST*; ASIN: B01L6STMVW; API-University Press: Zürich, Switzerland, 2016.

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.