

# Extending Fraud Detection in Students Exams Using AI

Georgi Cholakov<sup>1</sup>, Asya Stoyanova-Doycheva<sup>1</sup>

<sup>1</sup>*Department of Computer Systems, Plovdiv University "Paisii Hilendarski", "Bulgaria" Blvd. 236, Plovdiv, Bulgaria*

**Abstract** – The objective of the research is to enhance the functionality of the FraudDetector software agent within the Distributed eLearning Center (DeLC), a platform providing extensive support for e-learning activities. DeLC assists students and teachers in organizing learning materials, addressing knowledge gaps, conducting exams, and fostering personalized e-learning environments. The project scope encompasses various extensions, including an agent-oriented environment that enriches functionalities with reactive and proactive intelligent components, referred to as agents or assistants. This paper focuses on the latest evolution of the FraudDetector software agent, transitioning from its base functionality for fraud detection to leveraging artificial intelligence (AI) capabilities. The goal is to integrate AI, specifically the knowledgebase provided by ChatGPT, to enhance FraudDetector's effectiveness. This integration is the primary contribution of the research, aimed at improving fraud detection precision. Experimentation reveals promising results, suggesting that involving ChatGPT enriches FraudDetector's functionality and enhances the agent's precision. Moving forward, the agent's architecture should remain open for collaboration with external AI providers, with efforts to decouple components responsible for integration. The real-world implementation of these findings is pending, warranting further validation through production environment testing.

**Keywords** – E-learning, software agents, fraud detection, artificial intelligence, ChatGPT.

## 1. Introduction

Since e-learning became a standard for distance learning, almost every school and university try to enrich its educational processes with an educational platform that provides various forms of assistance in the learning process (the term e-learning refers to computer-based learning supported and conducted using electronic media). Among the most popular choices are Google Classroom [1], Moodle [2], to name a few. Making the right choice relies on a wide range of factors – price, functionalities, support efforts, user-friendliness of the interface, and more. However, when it comes to having partial or overall control over developed components, investing in a custom solution starts to sound much more attractive. It would provide the ability to implement only the necessary functionality, tailored to the institution's needs, without requiring the institution to adapt its processes to fit the business logic of an out-of-the-box system. Furthermore, when the choice was made, open-source systems did not have enough functionality or lacked features required by the business. These arguments prevailed in making the decision in many institutions, including the Faculty of Mathematics and Informatics at Plovdiv University "Paisii Hilendarski," Bulgaria.

The concept emerged many years ago, and the developed system has been serving various needs for over a decade. Initially established as the Distributed eLearning Center (DeLC) [3], it began as a research project dedicated to the creation of a new context-oriented and adaptive architecture. Its objectives include addressing the requirements for distance learning, exams, and a range of educational and organizational activities. Additionally, a significant goal was to engage in the development and experimentation of diverse prototypes within the e-learning domain. Through a series of iterations, a hybrid service and an agent-oriented environment were fashioned to provide educational materials and electronic services in the field.

DOI: 10.18421/TEM134-41

<https://doi.org/10.18421/TEM134-41>


**Corresponding author:** Georgi Cholakov,  
*Department of Computer Systems, Plovdiv University "Paisii Hilendarski", "Bulgaria" Blvd. 236, Plovdiv, Bulgaria*  
**Email:** [gcholakov@uni-plovdiv.bg](mailto:gcholakov@uni-plovdiv.bg)

*Received: 20 May 2024.*

*Revised: 09 September 2024.*

*Accepted: 28 October 2024.*

*Published: 27 November 2024.*

 © 2024 Georgi Cholakov & Asya Stoyanova-Doycheva; published by UIKTEN. This work is licensed under the Creative Commons Attribution-NonCommercial-NoDerivs 4.0 License.

The article is published with Open Access at <https://www.temjournal.com/>

Opting for an in-house customized solution carried the added advantage of making the codebase accessible to researchers within the institution. This accessibility facilitated development, reengineering, and enhancement of most features, allowing internal exploration of the system's workings, stored data, and the execution of analytical processes to extract valuable information and knowledge. Over time, the user base expanded beyond the institution to include other universities, evolving the system into a comprehensive platform that integrates functionalities and data from multiple satellite systems, thereby enhancing its capabilities. This brought additional problems to take care of – how to organize the conditions for confidentiality, integrity, accessibility, and security of information, as it is crucial to protect sensitive data and ensure a smooth learning experience. As a result, there is currently an ongoing process for implementing some key principles (briefly mentioned here as this is not the focus of this article):

- Risk management – to identify potential risks and vulnerabilities in the e-learning system; conduct regular risk assessments to stay ahead of emerging threats;
- Access controls – revision of currently implemented authentication mechanisms and role-based access;
- Regular software updates – apply security updates on a regular basis to minimize vulnerabilities;
- Data backups – regularly backup critical data to prevent data loss in case of system failures, cyber-attacks, or other emergencies;
- Incident response plan – to develop a comprehensive incident response plan to address security incidents promptly;
- Monitoring and auditing – to implement monitoring tools to track user activities, system logs, and potential security incidents;
- User training and awareness – educate users about security best practices and the importance of confidentiality, integrity, and accessibility. Promote strong password policies and ensure users understand the risks associated with sharing login credentials.

The primary objective of this system was to enhance the quality of the educational process by providing interactive and proactive personalized services for students (those studying in higher education institutions), fostering creative thinking. This response was driven by the escalating standards in university education, technological advancements, and heightened expectations of students regarding the quality of their education.

The system aimed to engage students in a personalized, creative, and flexible approach to self-education, promoting their activity and collaboration. Moreover, the system was designed to be open for extensions and experiments with prototypes, such as service and agent-oriented architectures, aligning it with the interactive, reactive, and proactive educational processes seen in other systems [4], [5], [6].

Throughout its lifecycle, the architecture underwent expansion with the inclusion of various subsystems, including IntelliDeLC, which ensures the provision of a personalized e-learning environment with reactive and proactive behavior [7]. Proactivity, enhancing usability and friendliness, is achieved through the reinforcement of the service-oriented architecture with intelligent components – essentially, software agents. This agent-oriented extension creates an environment housing these software agents, constantly developed and improved. Their functionalities, behavior, and the latest results are discussed in [8]. Recently, in the era of pervasive artificial intelligence (AI), an exploration was undertaken to determine how these agents could benefit from the integration of AI.

The main objective of this study is to enhance the functionality of a specific module within IntelliDeLC known as FraudDetector. This software tool is designed to identify and monitor potential instances of cheating during exams, particularly in open-response quizzes where students provide concise answers. Currently, FraudDetector relies on a knowledge base, essentially acting as a dictionary, to perform its tasks. The goal now is to expand this knowledge base by integrating the intelligence provided by artificial intelligence (AI) systems like ChatGPT. These sophisticated AI models, often referred to as large language models, offer valuable insights and support to FraudDetector. By leveraging the capabilities of these AI systems, FraudDetector can improve its effectiveness in identifying cheating behavior during exams, thereby enhancing its overall functionality and efficiency. This involves integrating FraudDetector with ChatGPT, allowing the agent to ask the AI the same question posed to each student and compare the responses for similarities. If the similarity percentage exceeds a predetermined threshold, the agent flags the answer as suspicious.

In the "Materials and Methods" section is discussed the architecture of IntelliDeLC, where the intelligent agents, including the FraudDetector agent, operate. This section is briefly presented to familiarize the reader with the overall idea of the IntelliDeLC intelligent environment.

In Section 4, "Implementation of Fraud Detection Improvement," the extended architecture of FraudDetector is presented, including its added functionality that utilizes ChatGPT. The "Results" section presents some data obtained in a test environment of the new functionality.

## 2. Related Works

Current state-of-the-art reveals that using AI in education and particularly in e-learning recently becomes a trend, a modern approach, a must-have solution if the adopting institution wants to keep extending and improving its e-learning systems and quality of education. It even looks like an approach that everyone is trying to benefit from, and the fields to apply it to vary in many aspects. Without precise statistic it can be assumed that AI is predominantly utilized to support learners in their learning path, though not exclusively. Here are some areas in e-learning where AI is applied to:

- Generally, for improving learning process, for example, making it adaptive [9];
- Using chatbots to accelerate learning process [10] and later measure the effectiveness of each feature [11];
- Improving accessibility of e-learning and academic connectivity [12];
- Personalized learning pathways [13] and adaptive assessment [14];
- For conversational agents for classroom use [15];
- For creating virtual assistants/teachers, even trying to replace teachers when there is deficit of such [16];
- And more, like automating learning processes by building teaching materials, curriculum, training, evaluating student performance [17], and so on.

On the other hand, the problem with academic dishonesty is a matter that every university should consider. Usage of different approaches and systems is observed:

- For prevention of cheating, plagiarism, and collusion [18];
- For online detection for learning time in distance learning systems [19];
- For detecting cheating in electronic exams [20].

There are not many tools that utilize generative artificial intelligence for fraud detection. The most popular ones are created by companies that have developed tools using large language models (LLMs), such as OpenAI, for example.

The detection tool, which OpenAI calls its AI Text Classifier [21], analyzes texts and then gives it one of five grades: "very unlikely, unlikely, unclear if it is, possibly, or likely AI-generated". The company said the tool would provide a "likely AI-generated" grade to AI-written text 26% of the time.

Other fraud detection environments using artificial intelligence are designed to use machine learning. One such environment is presented in [22]. The article discusses the prevalence of cheating in exams and proposes a novel method for detection using machine learning (ML) approaches. It utilizes the 7WiseUp behavior dataset, combining surveys, sensor data, and institutional records to predict academic success, identify at-risk students, and detect problematic behavior. The model approach achieves 90% accuracy by employing a long short-term memory (LSTM) technique with dropout layers, dense layers, and the Adam optimizer. Further analysis is needed to understand the factors behind the model's success. Paper [23] describes the aim to develop a fully automated online cheat detection system to identify cheating behavior during exams. The system is utilizing deep learning techniques, such as face recognition, sound detection, and active-window detection. Specifically, a CNN-based module will be employed for face recognition due to its high accuracy and stability. In paper [24] authors present a deep learning-based cheating detection system using a YOLOv7 model trained on a custom dataset. The dataset contains images of cheating and non-cheating behaviors, collected from various sources. Evaluation metrics such as precision, F1 score, recall, and mAP are employed to assess the model's performance, which achieved an mAP@0.5 of 0.719. The proposed method demonstrates promising capabilities in identifying cheating behaviors, potentially reducing human monitoring errors by alerting authorities to suspicious behavior during academic tests. In article [25] the authors introduce an innovative method that employs machine learning techniques to identify potential instances of cheating in final exams. They view cheating detection as an outlier detection challenge, utilizing continuous assessment data to pin-point abnormal scores. Their approach combines recurrent neural networks with anomaly detection algorithms to address the sequential nature of student assessment data. Experimental findings across diverse datasets illustrate the efficacy of their approach in precisely identifying instances of cheating. The authors suggest that their method could be a valuable resource for educators and administrators aiming to maintain academic integrity in course assessments during remote teaching.

Of course, some of the features enlisted above that fit perfectly in one system, would not fit well in another. This is true also for DeLC portal – some of the fields mentioned above are also under consideration, but the current study focuses on another possibly useful way of AI application – fraud detection in students’ exams. In DeLC there is such an internal implementation tailored to its specific needs, developed as a software agent.

### 3. Materials and Methods

The goal of the experiment is to enrich the existing functionality for fraud detection, as illustrated by the FraudDetector agent in Figure 1. This software component maintains its own data dictionary, comprising lexemes from each test question, keywords, answers, and other sources, as described in [7], [8], and further elaborated in the article. As this data is structured, it is stored in a relational database (currently MySQL [26]).

By utilizing external AI, it is possible to leverage its vast knowledge base and functionalities, thereby enhancing the capabilities of the agent and subsequently evaluating its performance. The expected results include an increase in the number of detected frauds during exams; for instance, if students use general phrases generated by AI like ChatGPT, the FraudDetector would detect and mark the answer as suspicious.

The part of the current architecture concerning the agent-oriented extension called IntelliDeLC is presented in Figure 1. It provides a simplified depiction of interactions between software agents and components from the DeLC portal, aiming to illustrate the existing components discussed in this article and how they interact with each other. The agents, also referred to as assistants, 'live' and operate in the back-end agent-oriented server, built with the JADE framework [27]. Their environment is known as the Agent Village (AV).

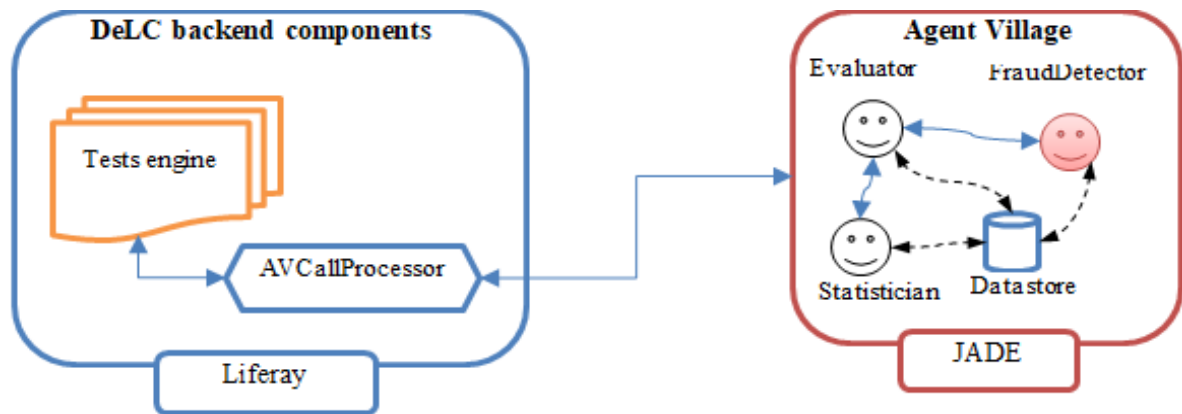


Figure 1. IntelliDeLC extension of DeLC

DeLC runs on the Liferay portal [28], and its architecture consists of a set of components and satellite extensions [29], developed by various researchers within the team. These extensions are omitted from the figure for readability and because they are not related to the current topic, unlike the tests’ engine, which contains the functionality responsible for students' exams. Simply put, when there are test questions with free text answers, the depicted workflow comes into play.

When DeLC needs to utilize the functionality of the assistants, it calls their services using AVCallProcessor, which acts as an adapter, making the technical details of the call transparent and providing a single point of implementation for the actual calls. Positioned within the portal as a system service, it serves as a mediator for all requests from the portal to the agents. The communication process operates as follows: when DeLC requires a service from Agent Village, it calls a specific method from AVCallProcessor.

This action triggers the generation of a SOAP request to Agent Village by AVCallProcessor. Subsequently, this request undergoes transformation into an ACL message [30], a format comprehensible to the agents. On the return journey, the corresponding agent formulates the result as an ACL message. Agent Village then transforms this message into a SOAP response, which is transmitted to AVCallProcessor. AVCallProcessor, in turn, parses the response and generates the result in the expected format for the portal.

There are some workflows that require agents to cooperate, for example:

- During the estimation of answers, the Evaluator communicates with the Statistician to select the algorithm that best matches the approach and behavior of the test creator (teacher) in order to simulate his/her manual estimation;

- When FraudDetector attempts to check suspicious activity during exams, it also communicates with the Statistician to compare similarities with previous cases to decide whether further investigation is necessary.

These internal communications between agents rely on standard ACL communication messages, which are the default mechanism for Agent Village (JADE) environment. When the agents need to access the datastore, they use the JDBC (Java Database Connectivity [31]) driver, as it is currently a relational database (MySQL).

### 3.1. Agent Village

This architectural component serves as the physical environment where the software agents (assistants) operate. The environment is established and operational on the JADE framework, with the entire internal communication among agents relying on ACL messages. Outside of Agent Village, the services provided by the agents are encapsulated and accessible as SOAP web services [32]. Further information on this subject is extensively covered in [7].

### 3.2. Evaluator

The Evaluator Agent (EA) functions as an expert, aiding the teacher in the assessment of electronic tests. While DeLC's test engine includes a system service for the automated assessment of "multiple choice questions", EA contributes by analyzing responses to short free-text questions, providing a rating for each answer and deferring the final decision to the teacher. When external assessment is needed, the test engine sends a request for expert assistance to EA. It then leverages its knowledge base to search for matches generated from keywords and phrases associated with each test question. Typically, these keywords and phrases are supplied by the test maker. The precision of the test maker in specifying keywords directly impacts the quality of results produced by the agent. Essentially, the agent needs to be appropriately "educated" to be effective. The keywords in the knowledge base carry no priorities; they are considered equal in searches. Currently, EA employs two distinct algorithms to calculate points, as previously detailed [7]. In the evaluation process for each answer, EA also considers the points (estimations) previously assigned by the teacher for that specific answer in prior exam runs. This approach allows EA to refine its estimation based on the teacher's style and approach. Post-evaluation, EA stores data about each answer, including the awarded points.

Further details on the latest validation of this agent can be found in [8], but in general the effectiveness of this agent is constantly being subject of monitoring.

The application of this agent is during exams with students from Faculty of mathematics and informatics, Plovdiv University "Paisii Hilendarski", primarily in the subjects "Database management systems", "Software engineering", and "Design patterns".

### 3.3. Statistician

The Statistician is responsible for storing comprehensive information on all processed answers, maintaining a complete history of details from all calculation methods employed by the Evaluator Agent. Currently, two methods, namely pessimistic and optimistic, are utilized as described in [7]. This assistant actively seeks feedback on the final points assigned by the teacher for each answer. Consequently, it accumulates a knowledge base for each teacher, enabling it to determine which method aligns best with the assessment style of the current evaluating teacher. Upon the Evaluator Agent returning its results, the Statistician plays a crucial role in determining which results are deemed suitable for presentation to the teacher. It identifies eligible results, while the remaining outcomes from other methods are presented as alternative options. This approach aims to provide the teacher with a comprehensive view of the assessment, incorporating alternative perspectives from different calculation methods. Currently, the satisfaction of teachers and students is not strictly measured. Instead, improvements are planned based on constructive feedback from both parties.

### 3.4. FraudDetector

FraudDetector's purpose is to help recognize attempts for cheating in the answers, given by the students. Among mostly used ways for cheating are:

- Using portal's integrated chat system to share answers – easy to trace and react to, and it is already covered by existing functionality;
- Copy/paste results from Internet search engines, e.g., Google, Bing – harder to recognize, most common so far;
- Copy/paste results from ChatGPT – not so rare recently. This is an area that is currently undergoing preliminary analysis and could be developed further.

The first point has been successfully implemented and rigorously tested over the years, with recent results detailed in [8].

This assistant primarily relies on its own knowledge base, which is derived from several sources: the words constituting each question, the keywords specified for each question (provided by the test creator), the words found in students' answers, and messages exchanged between students in the portal's chat system – particularly those flagged by an operator as potentially involving cheating. This process results in the creation of a substantial dictionary that expands with each examination, as both the number of answers and chat messages increase. This necessitates another key functionality for the agent—its ability to be “self-learning”. This crucial feature ensures the agent stays abreast of new potential cheating trends. An interesting side effect of this self-learning capability is that the agent may become somewhat stringent, flagging messages and answers as suspicious even when they are unrelated to the subject. However, this behavior was anticipated and discussed in [8].

The second point is large to cover with custom functionality, but since Google and Bing provide APIs for searching programmatically, there are some ideas in this direction, so far just in plans and analysis/investigation phase.

The third point is very interesting now, as ChatGPT also provides API for accessing its functions. In fact, this is the subject of the current study: to determine whether an out-of-the-box AI solution can be effectively utilized to meet the specific requirements in this context.

#### **4. Implementation of Fraud Detection Improvement**

Recently, there has been a noticeable increase in cases at the university where students attempt to use ChatGPT during exams to aid in passing the tests. As this way of cheating is becoming more and more popular, detecting fraud from this direction becomes important goal to stay ahead of emerging new ways of cheating. The idea is to extend FraudDetector's dictionary by adding functionality for collaborating with ChatGPT – the agent, in addition to its current functionality, will ask the AI the same question, answered by each student, and compare result with the answer for similarities – if the percentage is higher than a parameterized value, the agent marks that answer as suspicious – it is important to point out here that this agent does not provide the final and ultimate decision, it only advises the teachers to look at the answer more attentively. It is perfectly clear that some students would use AI to learn, and even by a chance the student's answer could match entirely without cheating, so the teacher makes final decision, after discussion with the student, if necessary.

Of course, there are many aspects here for improvement, e.g., selecting correct percentage threshold requires education of the agent, based on many runs over the same data until final results are reliable enough and could be used in real environment, eventually; using multiple answers from AI to compare student's one with, and so on. But as a starting point, for first iteration the explained functionality is considered as sufficient. Such an extension of this agent would bring indirect access to an enormous knowledge base, because the scale of data amount, used by ChatGPT, is in fact incomparable to current one, used by the agent, increasing the expected reliability of fraud detection in times. The topic of the proper threshold is not subject of interest in this article and selecting it is still in development and experiments continue.

Integrating with ChatGPT was not the sole option for selecting a third-party provider; developing an in-house solution would be more costly and would require efforts in a direction that is not aligned with current primary objectives. Thinking in this direction, research was conducted on how most popular AI providers could be used for educational purposes, and particularly in fraud detection during exams. Among intriguing ones were ChatGPT [33], Perplexity [34] and Google Bard (now renamed to Gemini) [35] – a good comparison could be found in [36], [37]. Many arguments were taken into account when the choice was made, but in summary:

- Gemini (previously Google Bard) was still in development at the time of choosing. In the near future, it could be a very good alternative. It produces also additional information, which currently does not add value to the experiment. Nevertheless, it will be monitored closely, as it excels at answering questions with up-to-date information, having real-time access to Google. In contrast, ChatGPT (with GPT 3.x) sometimes returns a bit outdated information. Also, Gemini so far is free, while new version of GPT 4 is paid. There is another feature here, that attracts – Gemini can produce multiple answers (variations) to a single question. It makes it particularly interesting for experiments in this area, as it allows for the investigation multiple answers within a single roundtrip – that would perform better and also would return more results to search within;
- Perplexity would be actually a very good choice, its advanced answer engine considers the entire conversation history and uses predictive text algorithms to generate concise responses from multiple sources, which is helpful for generating answers bound to a specific context.

It provides real-time information from multiple sources, just like Bard and in contrast with ChatGPT. This approach is expected to yield more topic-oriented results and would be considered the second choice for the experiment;

- ChatGPT tries to mimic human conversation and its training methodology involves learning from human feedback. Communication with it provides a possibility to tune up the search for answer – more deterministic or more creative, which was interesting for the experiment. Consequently, the focus was placed on ChatGPT due to its ease of use, comprehensive support, and recent progress. A downside is that the new version requires a paid subscription, unlike Bard and Perplexity. However, since the price is affordable and the primary concern is the correctness of the results, this factor is not significant.

To summarize this comparison and as final consideration for the choice, any of the aforementioned AI systems would be suitable for the case, as only a small subset of their capabilities is utilized.

Integration architecture is presented in Figure 2. The communication with ChatGPT is achieved by REST calls, since the existing ChatGPT API exposes its functions in this way. But implementing integration with external systems in existing and working component (FraudDetector) directly could be error prone, that is was achieved by developing another adapter, to incorporate the entire communication in it. Thus, if changes are required in some technical details in REST calls, or even replace external system with another one, no changes will affect the FraudDetector assistant, only the adapter will accommodate the implementation change.

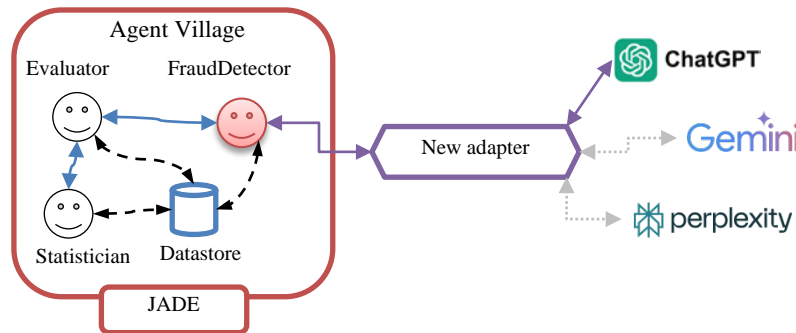


Figure 2. Improved architecture for accessing external systems e.g., ChatGPT, Gemini, Perplexity

As of now, introducing integration with multiple systems is scheduled for future analysis and development, current implementation is focused on integration with ChatGPT only – there are many parameters to tune up for getting reliable results, that’s why adding integrations will be accomplished iteratively, keeping control over incrementing complexity.

As illustrated in Figure 3, the request and response appear straightforward and in human-readable format. While the generated result is accurate, it is important to examine the request attribute "temperature" more closely. This parameter enables users to fine-tune the trade-off between creativity and coherence during text generation.

Increasing temperatures injects more randomness, leading to more imaginative but possibly less coherent text output. It ranges from 0 to 2, as the higher the temperature, the higher the randomness of the result, and vice-versa. For more focused and deterministic responses, the value should not exceed 0.7, as the tests revealed, otherwise the response could become nonsensical. According to the documentation, higher values yield more creative responses, but this approach does not seem to perform well in scientific contexts. Conversely, setting the value too low tends to produce formulaic responses, which are uncommon in test questions that require short free-text answers. Of course, choosing the right value depends heavily on context, and setting it up for specific needs rely on many experiments and statistical analysis, not finished yet.

```

{
  "model": "gpt-3.5-turbo",
  "temperature": 0.7,
  "messages": [
    {
      "role": "user",
      "content": "Which functional dependency third normal form deals with"
    }
  ]
}

{
  "id": "chatcmpl-8Huvvm2wXhBoIjd2G4161g75tepKqm",
  "object": "chat.completion",
  "created": 1699281110,
  "model": "gpt-3.5-turbo-0613",
  "choices": [
    {
      "index": 0,
      "message": {
        "role": "assistant",
        "content": "The third normal form (3NF) deals with functional dependencies that are transitive in nature. \n\nIn 3NF, a table is considered to be in third normal form if it meets the following conditions:\n\n1. It is in second normal form (2NF).\n2. There are no transitive dependencies between non-key attributes.\n\nA transitive dependency"
      }
    }
  ]
}
    
```

Figure 3. Raw result from ChatGPT to a particular question

## 5. Results

This section provides experimental results over ChatGPT’s behavior using different settings and FraudDetector’s functionality over existing data, collected from past periods of time. These results are important as they provide directions how to proceed further with the implementation of described functionality so far.

### 5.1. Tests for ChatGPT’s Temperature Parameter

The tests for temperature parameter were conducted with simulation, including only the new adapter and ChatGPT. The adapter is implemented as separate microservice with Spring Framework 6 [38]. In the simulation, 89 questions were used, given to students in the tests in exams for one particular subject – Database Management Systems, which subject is intended for pilot version for real environment. The appropriate accuracy in results, returned by ChatGPT, is summarized on Figure 4. It is important to clarify that the correctness mentioned here is not a general or overall assessment of ChatGPT’s functionality. It actually performs surprisingly well for the specific purposes of this study. And this evaluation is subjective, as this AI is used in a very specific domain, and its success is estimated by individuals, each with their own perspectives on the topics.

The results presented in Figure 4 were produced by the new adapter, iterating over those 89 questions, each one sent to ChatGPT with all possible values for the temperature with step 0.1. As indicated by the graphic, the most accurate results for the intended purposes were obtained with a parameter value around 0.7, which is why this value is currently set.

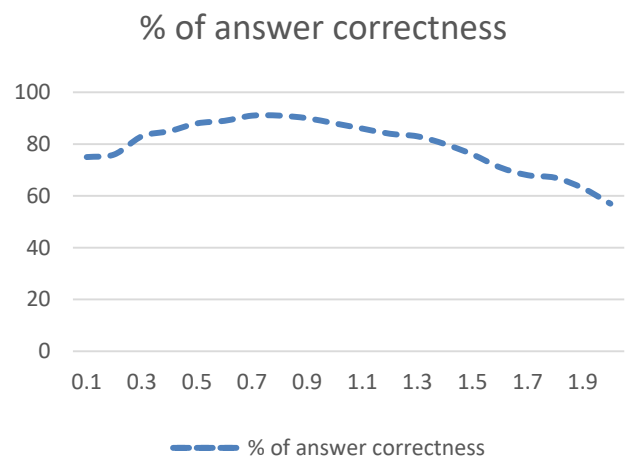


Figure 4. Results accuracy of the answers by ChatGPT

At some point, even after getting first results in the real environment, this value could be amended, to compare different behaviors and results in empirical way.

### 5.2. Tests for Fraud Detection Over Existing Data

This simulation was conducted over existing data from previous periods (kept in portal’s database), consisting of selected top suspicious 5000 messages from the integrated chat messaging system in DeLC portal. This system is often used by the students for helping each other during exams, and it was the first target of FraudDetector for investigation.

Having the results from FraudDetector before the new functionality to take place, the results were compared now with the new functionality, running a simulation over the same data. These results are summarized in Table 1.



*Table 1. Comparison between the results from FraudDetector prior and after the new changes over the same dataset*

<b>Suspected messages</b>	<b>Reasonably suspected</b>	<b>Without reason</b>
Before: 235	167 (71%)	68 (29%)
After: 258	179 (69%)	79 (31%)

From the results in Table 1 it looks like after integration with ChatGPT the correctness is dropping, and the suspiciousness is rising, but it is only percentage – after the integration the agent suspected more messages, and it found 12 messages reasonably suspected more than it did it previously, which is a progress. The percentage of its suspiciousness increased, as well, and that reveals another room for improvement – the agent’s functionality should be revised further to find its weaknesses in marking a message as suspicious.

## 6. Discussion

The following main results can be concluded of the applied approach with the FraudDetector agent integrated with ChatGPT - the integration with ChatGPT led to an increase in the total number of suspected messages identified by FraudDetector. This suggests that leveraging ChatGPT's intelligence resulted in a broader detection of potential cheating behavior during exams. Despite the increase in suspected messages, the proportion of reasonably suspected messages remained stable before and after the integration. This highlights the reliability of FraudDetector in accurately identifying truly suspicious behavior, further emphasizing the importance of utilizing ChatGPT. However, there was an uptick in the number of messages flagged without clear reason post-integration. This underscores the challenge of false positives introduced by ChatGPT, indicating the need for ongoing refinement of FraudDetector's algorithms to minimize such instances. The results obtained are based on testing in a controlled environment and have not yet been deployed in a production environment. Therefore, real-world testing is essential to validate the performance of the integrated system and identify any additional improvements needed.

## 7. Conclusion

Keeping the quality of education high relies on many parameters, going far beyond the teaching skills of the personnel and students’ awareness that building their skills needs dedication and self-motivation. As the technologies go further, entire process of university education needs to keep up with recent standards and trends – not only to attract students’ attention, but to provide a healthy environment for building professionals from students. It also includes the quality of examination, part of which is to guarantee fair process of assessment – which is subject of the current study. Improvement of software components, that work in background and support in fact the entire educational process, is a major part of building a strong foundation of education nowadays.

Extending fraud detection with artificial intelligence will provide contemporary functionalities to the activities that FraudDetector is responsible for. Having its own knowledge base extended is not an easy task, that why integration with external systems is justified, leaving focus on precision of fraud detection methods instead of building and extending its own knowledge base.

If the integration with ChatGPT proves successful, as suggested by the modest improvement in numbers shown in Table 1, Evaluator agent would be the next candidate for such integration as well, enriching its answers estimation algorithms with another source of truth for building larger knowledge base for its purposes. Furthermore, evaluator could use another AI provider, for example Google Bard, which is able to produce multiple results. These results could be used to form several evaluations, depending on the results number, providing the teacher with different estimations for the same answer – and when the teacher makes a choice, evaluator could be educated what answer is preferred.

## Acknowledgements

Special thanks to the system and network administrators in Faculty of mathematics and informatics, Plovdiv university “Paisii Hilendarski”, who supported used hardware and software through all these years of experiments. Also, thanks to people involved in the project European Union – NextGenerationEU, through the National Recovery and Resilience Plan of the Republic of Bulgaria, project № BGRRP-2.004-0001-C01, which made this study possible.

## References:

- [1]. Google for Education. (n.d.). *Google Classroom*. Google for Education. Retrieved from: <https://edu.google.com/workspace-for-education/classroom/> [accessed: 03 May 2024]
- [2]. Moodle. (n.d.). *Online learning, delivered your way*. Moodle. Retrieved from: <https://moodle.com/> [accessed: 04 May 2024]
- [3]. Stoyanov, S., et al. (2010). DeLC educational portal. *Cybernetics and Information Technologies*, 10(3), 49-69.
- [4]. Goyal, M., & Krishnamurthi, R. (2019). Pedagogical Software Agents for Personalized E-Learning Using Soft Computing Techniques. In *Nature-Inspired Algorithms for Big Data Frameworks*.
- [5]. Farzaneh, M., Vanani, I. R., & Sohrabi, B. (2012). Utilization of Intelligent Software Agent Features for Improving E-Learning Efforts: A Comprehensive Investigation. *International Journal of Virtual and Personal Learning Environments (IJVPLE)*, 3(1), 55-68. Doi: 10.4018/jvple.2012010104
- [6]. Rani, M., Nayak, R., & Vyas, O. P. (2015). An ontology-based adaptive personalized e-learning system, assisted by software agents on cloud storage. *Knowledge-based systems*, 90, 33-48. Doi: 10.1016/j.knosys.2015.10.002
- [7]. Cholakov, G. (2013). *Hybrid Architecture for Building Distributed Center for e-Learning* [PhD thesis, Plovdiv University “Paisii Hilendarski”]. Plovdiv (Bulgaria).
- [8]. Cholakov, G. (2020). Approbation of software agent Evaluator in a nonspecific environment for extension of its purpose. *International Conference Automatics and Informatics (ICAI)*, 1–5. Doi:10.1109/ICAI50593.2020.9311346
- [9]. Benkhalfallah, F., & Laouar, M. (2023). Artificial Intelligence-Based Adaptive E-learning Environments. *Novel & Intelligent Digital Systems: Proceedings of the 3rd International Conference (NiDS 2023)*, 62–66. Doi: 10.1007/978-3-031-44097-7\_6
- [10]. Benachour, P., Emran, M., & Alshafut, A. (2023). Assistive Technology and Secure Communication for AI-Based E-Learning. *AI-Based Digital Health Communication for Securing Assistive Systems*, 1-21. IGI Global.
- [11]. Raghavendruchar, S., Anand, VS., Anushree, H., & Manjunath, R. (2023). “E-Learning Management System with AI Assistance”. *International Journal for Research in Applied Science and Engineering Technology*, 11(XI), 1233–1238.
- [12]. Sinha, M., Fukey, L., & Sinha, A. (2021). “AI in e-learning”. *E-learning Methodologies: Fundamentals, technologies and applications*, 107-131.
- [13]. Tapalova, O., & Zhiyenbayeva, N. (2022). Artificial intelligence in education: AIED for personalised learning pathways. *Electronic Journal of e-Learning*, 20(5), 639-653. Doi: 10.34190/ejel.20.5.2597
- [14]. Tanjga, M. (2023). “E-learning and the Use of AI: A Review of Current Practices and Future Directions”. *Qeios*. Doi: 10.32388/AP0208.2
- [15]. Alfehaid, A., & Hammami, M. A. (2023). Artificial Intelligence in Education: Literature Review on The Role of Conversational Agents in Improving Learning Experience. *International Journal*, 10(3), 3121-3129. Doi:10.15379/ijmst.v10i3.3045
- [16]. Muzurura, O., Mzikamwi, T., Rebanowako, T. G., & Mpini, D. (2023). Application of artificial intelligence for virtual teaching assistance (Case study: Introduction to Information Technology). *International Research Journal of Engineering and Technology (IRJET)*, 10.
- [17]. Arun Kumar, U., Mahendran, G., & Gobhinath, S. (2022). A review on artificial intelligence based E-learning system. *Pervasive Computing and Social Networking: Proceedings of ICPCSN 2022*, 659-671. Doi:10.1007/978-981-19-2840-6\_50
- [18]. Bylieva, D., Lobatyuk, V., Tolpygin, S., & Rubtsova, A. (2020). Academic Dishonesty Prevention in E-learning University System. *Trends and Innovations in Information Systems and Technologies*, 1161, 225–234. Doi: 10.1007/978-3-030-45697-9\_22
- [19]. Ueno, M. (2004). Online Outlier Detection System for Learning Time Data in E-Learning and Its Evaluation. *Proceedings of the International conference on computers and advanced technology in education*, 248–253.
- [20]. Bawarith, R., Basuhail, A., Fattouh, A., & Gamaleldin, S. (2017). E-exam cheating detection system. *International Journal of Advanced Computer Science and Applications*, 8(4). Doi: 10.14569/IJACSA.2017.080425
- [21]. AI Text Classifier. (n.d). Home page. Free Ai text classifier. Retrieved from: <https://freeaitextclassifier.com> [accessed: 10 May 2024]
- [22]. Alsabhan, W. (2023). Student cheating detection in higher education by implementing machine learning and LSTM techniques. *Sensors*, 23(8). Doi: 10.3390/s23084149
- [23]. Soltane, M., & Laouar, M. (2021). A Smart System to Detect Cheating in the Online Exam. *2021 International Conference on Information Systems and Advanced Technologies (ICISAT)*, 1–5. Doi:10.1109/ICISAT54145.2021.9678418
- [24]. Trabelsi, Z., Parambil, M. M. A., Alnajjar, F., & Ali, L. (2023, November). Behavioral-based real-time cheating detection in academic exams using deep learning techniques. In *AIP Conference Proceedings*, 2909(1). Doi:10.1063/5.0181921

- [25]. Kamalov, F., Sulieman, H., & Calonge, D. (2021). Machine learning based approach to exam cheating detection. *Plos one*, 16(8), Doi:10.1371/journal.pone.0254340
- [26]. MySQL. (n.d.). *MySQL: The world's most popular open-source database*. MySQL. Retrieved from: <https://www.mysql.com/> [accessed: 16 May 2024].
- [27]. Jade. (n.d.). *JADE: Java agent development framework*. Jade Tilab . Retrieved from: <https://jade.tilab.com/> [accessed 27 April 2024].
- [28]. Liferay. (n.d.). *Digital Experiences, Your Way*. Liferay. Retrieved from: <https://www.liferay.com/> [accessed: 28 April 2024].
- [29]. Doychev, E. (2013). *Environment for Provision of eLearning Services*. [PhD thesis, Plovdiv University "Paisii Hilendarski"]. Plovdiv (Bulgaria).
- [30]. FIPA. *Agent Communication Language Specifications*. FIPA. Retrieved from: <http://www.fipa.org/repository/aclspecs.html> [accessed: 12 May 2024].
- [31]. Wikipedia. (n.d.). *Java Database Connectivity*. Wikipedia. Retrieved from: [https://en.wikipedia.org/w/index.php?title=Java\\_Database\\_Connectivity&oldid=1199135695](https://en.wikipedia.org/w/index.php?title=Java_Database_Connectivity&oldid=1199135695) [accessed: 16 May 2024].
- [32]. W3C. (n.d.). *SOAP Version 1.2 Part 1: Messaging Framework (Second Edition)*. W3.org. Retrieved from: <https://www.w3.org/TR/soap12> [accessed: 17 May 2024].
- [33]. Chatbot App. (n.d.). *Chatbot App: All-in-One AI Chatbot*. Chatbot App. Retrieved from: <https://chatbotapp.ai/> [accessed: 18 May 2024].
- [34]. Perplexity. (n.d.). *Home page*. Perplexity Retrieved from: <https://www.perplexity.ai/> [accessed: 18 May 2024].
- [35]. Wikipedia. (n.d.). *Gemini (chatbot)*. Wikipedia. Retrieved from: [https://en.wikipedia.org/w/index.php?title=Gemini\\_\(chatbot\)&oldid=1206112332](https://en.wikipedia.org/w/index.php?title=Gemini_(chatbot)&oldid=1206112332) [accessed: 19 May 2024].
- [36]. Java, W. (2023). *Battle of the AI's: Chat GPT vs. Perplexity AI vs. Google Bard*. Medium. Retrieved from: <https://medium.com/@jamva/battle-of-the-ai-chat-gpt-vs-perplexity-ai-vs-google-bard-bca76474a30d> [accessed: 19 May 2024].
- [37]. Horsey, J. (2023). *Perplexity vs Bard vs ChatGPT*. Geeky Gadgets. Retrieved from: <https://www.geeky-gadgets.com/perplexity-vs-bard-vs-chatgpt> [accessed: 17 April 2024].
- [38]. Spring. (n.d.). *Microservices*. Spring.io. Retrieved from: <https://spring.io/microservices> [accessed: 11 May 2024].