DESIGNING AN AI-POWERED E-LEARNING SYSTEM TO SUPPORT EMOTIONAL REGULATION IN CHILDREN WITH AUTISM

Maxwell Palance [1], Austin Amakye Ansah [2]

[1] Stanford Online High School, 415 Broadway Academy Hall, Floor 2, 8853, 415 Broadway, Redwood City, CA 94063
[2] The University of Texas at Arlington, 701 S Nedderman Dr, Arlington, TX 76019

ABSTRACT

Students with Autism Spectrum Disorder (ASD) face unique challenges in traditional and online coding education [5]. While existing platforms offer authentic programming exercises, they often fail to address the neurodiverse learning needs of these students. This paper proposes an AI-powered e-learning platform, CODEversity, designed to offer an engaging and gamified coding education experience [6]. Built using Flutter and Firebase, the platform integrates real-world coding exercises with interactive visual and auditory feedback to foster positive reinforcement. A chatbot assistant powered by large language models helps guide students in course selection based on their interests and progress. The system also incorporates a dynamic quiz program, utilizing a tree-based course roadmap and practical coding exercises executed via a sandboxed Python environment [7]. A robust course builder ensures seamless course creation, validation, and updates without data loss. By prioritizing engagement and inclusivity, CODEversity aims to make coding education accessible and enjoyable for neurodiverse learners.

KEYWORDS

E-learning, Neurodiversity, Autism, Programming, AI Assistant

1. INTRODUCTION

Students with Autism Spectrum Disorder (ASD) often face unique learning challenges in traditional settings [15]. This challenge is no different when it comes to learning how to code. While block coding has been a popular entry point for beginners, many students find it unengaging and disconnected from real-world programming experiences. Previous research has shown that using actual coding platforms like LeetCode can offer a more authentic and stimulating experience, leading to quicker skill development and a deeper understanding of coding concepts [8]. However, these platforms are not always designed with neurodiverse learners in mind, often lacking the tailored support and positive reinforcement needed to keep students motivated.

To address these gaps, there is a need for an engaging, accessible platform that combines real coding exercises with interactive and gamified learning elements. By fostering an environment where coding feels less like a task and more like an adventure, students can develop critical problem-solving skills in a way that is both educational and enjoyable. Furthermore, providing a

David C. Wyld et al. (Eds): SEC, DMA, AIAPP, CYBI, CoSIT, SIGL, NLPML – 2025
pp. 45-57, 2025. CS & IT - CSCP 2025 DOI: 10.5121/csit.2025.150804

positive feedback loop—reinforced by visual and auditory cues—can help students with Autism build confidence as they progress. Creating such a system would not only make coding education more inclusive but also help bridge the gap between learning and applying real-world coding skills. This foundation is essential for supporting the diverse needs of neurodiverse students as they enter an increasingly digital world.

We propose a mobile e-learning platform that leverages Firebase and Flutter to create a Duolingo-like programming education experience. The goal is to create a fun platform that makes learning how to code enjoyable and gamified. The app leverages Firebase for data storage, and CODEversity's dart APIs leverage Firebase to create interactions between the user and the course data [9]. The system includes a course creation tool that can be run on the web or desktop. Since both the course creation tool and mobile app are built with Flutter, they can theoretically be deployed on multiple platforms if optimized properly. The goal was to design an app that provided programming education to students with Autism in a manner that creates a positive feedback cycle. CODEversity leverages fun animations and sounds to make learning more interesting and game-like rather than the traditional methods of education.

2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

2.1. Updating Courses

When updating courses, it is necessary to synchronize the enrolled courses of users who have already downloaded them so that they are up to date. Otherwise, not all of the information would sync properly, and some of the data would be overwritten. Firebase uses JSON for data storage, and JSON unions are not easy to implement since every key needs to be handled individually to prevent overwrites.

2.2. Creating a Scalable JSON Schema

At first, creating a scalable JSON schema to store course information was a huge issue [10]. This stage of development required many hours of planning to ensure that data could be stored long-term and updated. This involved scrapping several JSON schema ideas until an optimal one could be devised.

2.3. Creating a Tool to Dynamically Create and Update Courses

The most taxing task was creating a tool to dynamically create and update courses, as it required building numerous sub-components for activities and courses. Every course had to be validated for errors before being published, which involved writing code to manually ensure that all fields that could not be null were not left blank. Since the number of fields grows linearly with the number of activities or quizzes added to the course from the tool, it is up to the course creator to ensure that the JSON schema is error-free before validation occurs.

3. SOLUTION

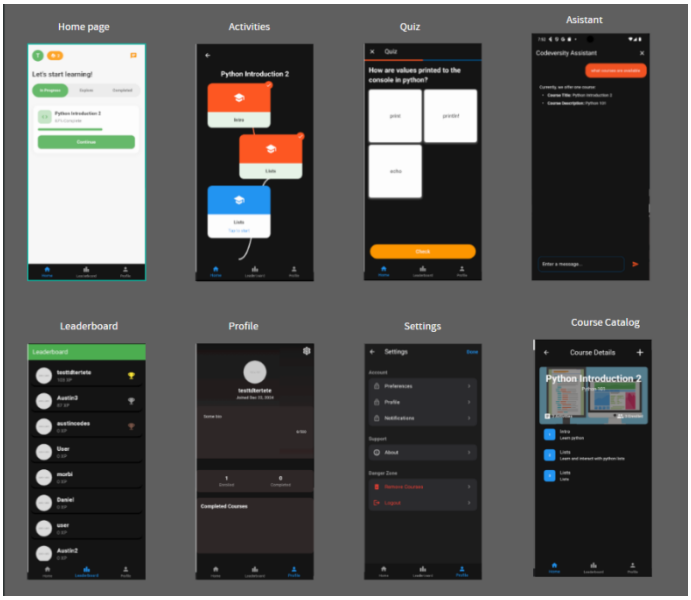

Figure 1. Overview of the solution

The CODEversity system consists of two parts: The administrative course creation tool and the mobile app that users interact with. The tool can be used to change most aspects of courses, notably editing, deleting, or adding questions and creating or updating course content and catalogs. The mobile app is where the user can view the courses and enroll in them. Every course has an activity that the user can take to earn XP. XP is calculated using a scaling function, which calculates an XP value based on the number of questions in an activity.

The XP is scaled logarithmically so that large activities do not disproportionately max out the user's XP level. The app features a global leaderboard where users can see the people with the highest XP. If a course is reset or deleted by the user, the xp they attained for it is subtracted from their account. The following formula is used to scale XP after an activity:

$$\mathrm{XP} = \left(\sqrt{(1200 + 100 \cdot \frac{\log(n+1)}{\log(2)})} \cdot n \right)$$

– where 1200 is the base difficulty, n is the number of questions, and 100 is the weight. The course creation tool features two pages. The JSON course builder and the graphical course builder. This allows experienced course creators to choose if they want to manually create courses with raw JSON or build the courses graphically with the tool.

After a quiz is completed, the user is greeted with a congratulatory animation. The screen blasts confetti as an effect, and the user is allowed to return to the main activity screen to observe their milestone map.

When a course is updated, a hand-implemented algorithm overwrites the user's saved course metadata with the updated metadata; however, fields relevant to the user's progress are preserved when the algorithm is run. This is the course union algorithm.

Every piece of data is stored using JSON, and Dart-JSON algorithms have been implemented to efficiently work with this format [14].

In-App Assistant

The CODEversity app features a chatbot that guides users towards the right course. Users can ask questions about which courses are available. Because LLM technology is used, users can provide information about their personal experiences and interests, which can be used to recommend appropriate programming courses [13].

For the sake of simplicity, message history is not stored in the database. Instead, the chat widget state is stored intermittently on the device.

The app sends a request to the Google Gemini API. Gemini is used because it offers a free tier and is affordable. After a message is sent, a dart Message object is created, storing the response text and determining whether the message came from the user or the language model. Based on the message sender, the message bubble is aligned to either the left or right and is given a distinct color to differentiate the user from the model.

The model is given a system prompt consisting of context regarding all of the course information as well as instructions on how to speak to users.

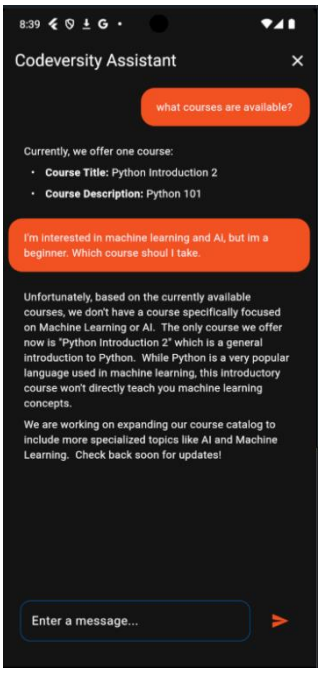

Figure 2. Screenshot of the assistant

Activities and Course Roadmap

The quiz system consists of a course journey and quiz program. Quizzes are designed based on Duolingo activities. Wrong or correct answers trigger different sound effects, and successful quizzes end with a congratulatory animation and optional confetti.

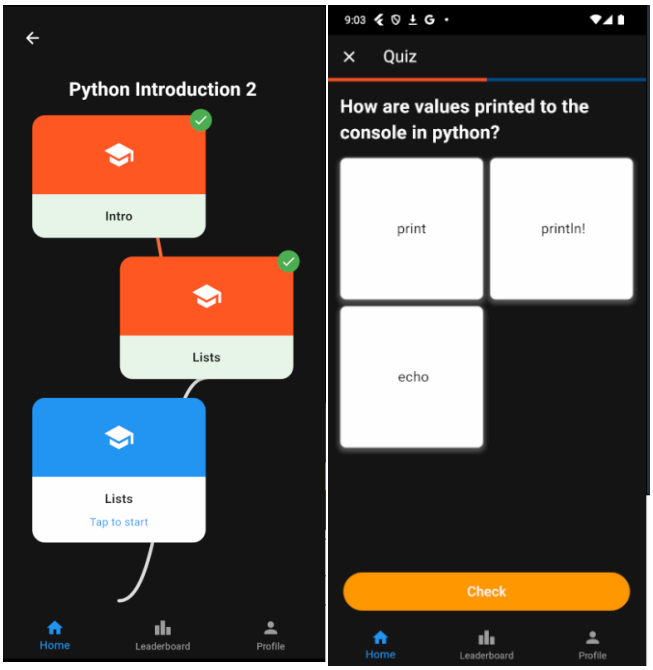

Figure 3. Screenshot of introduction and quiz

Quiz Roadmap design

Course activities are laid out based on their index. Even and odd indices are positioned oppositely on the map.

```
final double nodeSpacing = size.height / (itemCount - 1);
final path = Path();
path.moveTo(110, 50); // Start from center of first node

for (int i = 0; i < itemCount - 1; i++) {
  final currentY = 50 + (nodeSpacing * i);
  final nextY = 50 + (nodeSpacing * (i + 1));

  if (i.isEven) {
    // Path to right node1
    path.cubicTo(
      size.width - 160, currentY, // First control point
      160, nextY, // Second control point
      size.width - 110, nextY, // End point
    );
  } else {
    // Path to left node
    path.cubicTo(
      160, currentY, // First control point
      size.width - 160, nextY, // Second control point
      110, nextY, // End point
    );
  }
}
```

Figure 4. Screenshot of code 1

The Flutter custom paint API creates a tree-like structure representing course modules [12]. A predecessor module must be completed before its successor is unlocked. The next module after a completed module is marked as available, whereas the next module after an incomplete activity is locked and greyed out.

Course Builder

Using the same UI framework, a desktop course builder was created to streamline the process of adding activities, quizzes, and courses to the system.

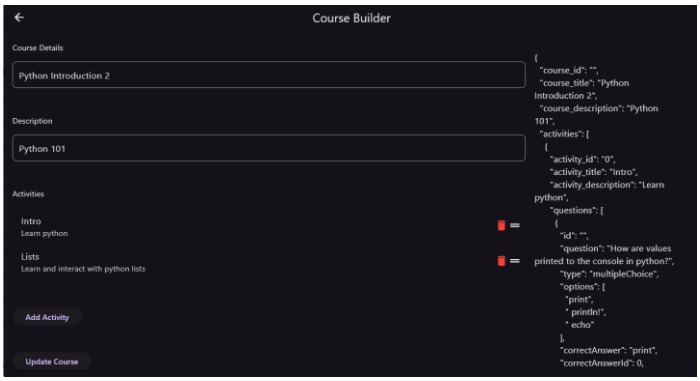

Figure 5. Screenshot of course builder

```
Map<String, dynamic> convertCourseToJson(Course course) {
  List<Map<String, dynamic>> activities = [];
  course.activities.forEach((activity) {
    activities.add({
      'activity_id': activity.activity_id,
      'activity_title': activity.activity_title,
      'activity_description': activity.activity_description,
      'activityCompleted': activity.activityCompleted,
      'questions': activity.questions.map((question) {
        // check question type, and ensure only the required fields

        switch (question.type) {
          case QuestionType.multipleChoice:
            return {
              'id': question.id,
              'question': question.question,
              'type': question.type.toString().split('.').last,
              'options': question.options,
              'correctAnswer': question.correctAnswer,
              'correctAnswerId': question.correctAnswerId,
            };

          case QuestionType.trueFalse:
            return {
              'id': question.id,
              'question': question.question,
              'type': question.type.toString().split('.').last,
              'options': question.options,
```

Figure 6. Screenshot of code 2

Course information is manually converted to JSON when exporting courses to Firebase to prevent errors. For this reason, when adding new course types in the builder, a new case for that type must be included in the converter function's switch statement.

Likewise, when converting JSON to dart Course objects when a course is opened for editing in the builder, the process must be reversed.

Using a sandboxed environment to run remote Python code

There was a necessity to have hands-on coding assignments. Ultimately, the best option was to use a pre-built sandbox environment that functioned over the web. An open-source project called Snekbox was made available to handle this task. Snekbox is a self-hostable server that can be used to run remote Python code in a sandboxed environment [2]. The goal here was to utilize this, in combination with the current quiz system, to create an environment that tests practical coding skills. By running a dockerizedSnekbox instance in a digital ocean droplet, it became possible to use the sandbox features from any device rather than a local network. The Snekbox instance is loaded into a docker server that is scheduled to begin at every boot. Using a TCP connection, the evaluation endpoint can be accessed to run Python code securely.

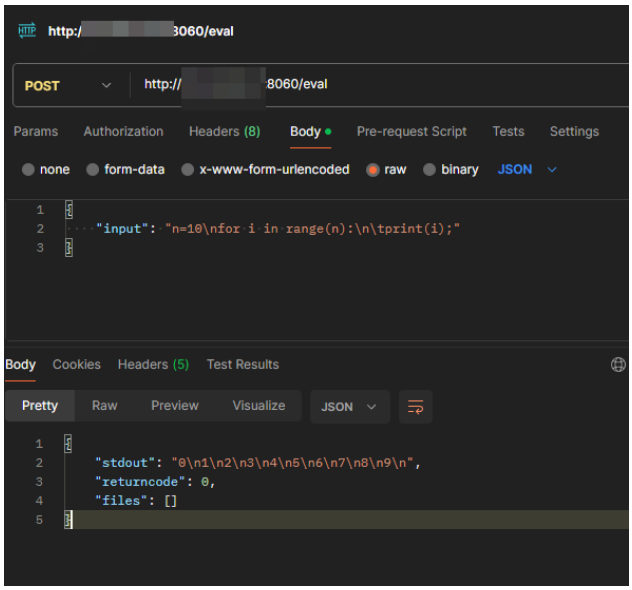

Figure 7. Screenshot of code 3

With this, it became possible to create questions that were based on the result obtained from the Snekbox request. The reliance on fill-in-the-blank questions to teach programming discipline diminished with the introduction of this feature, and the goal of having practical programming quizzes that mimicked the real thing was met.

Figure 8. Screenshot of the quiz

Every question has some context. For coding questions, the piece of information used to check correctness is the expected output.

Verifying the correctness of coding questions involves n steps:
Running the code on the Snekbox instance.

Checking to make sure that the raw output equals the expected raw output provided in the question context

Calling the _checkAnswer method to perform the standard procedure for question success checking.

Using conditional statements, every question type can dynamically be checked for correctness by checking whether the selected option (in the case of multi-choice questions) or user-inputted state matches the correct or expected output. This makes it easy to extend activities with new question types if needed.

4. EXPERIMENT

4.1. Experiment 1

One potential blind spot in CODEversity is whether the gamified learning experience effectively improves engagement and emotional regulation for students with Autism Spectrum Disorder (ASD). It is essential to determine if positive reinforcement mechanisms (animations, XP rewards, and chatbot guidance) genuinely enhance learning and retention. Without direct user feedback, it is difficult to validate the system's impact on motivation and coding proficiency.

To assess the platform's effectiveness, we conducted a user survey targeting students with ASD, their parents, and educators. The survey included:

Engagement Metrics: How often students used CODEversity compared to other platforms (1-5 scale).

Emotional Regulation: Whether the platform's design helped users feel less frustrated while coding.

Comprehension Improvement: A self-assessment of how well students understood coding concepts before and after using CODEversity.

Preferred Features: Which elements (chatbot, XP rewards, animations) had the most positive impact.

The survey was conducted online, and responses were collected over four weeks from 10 participants (students, parents, and teachers) to ensure a balanced perspective.

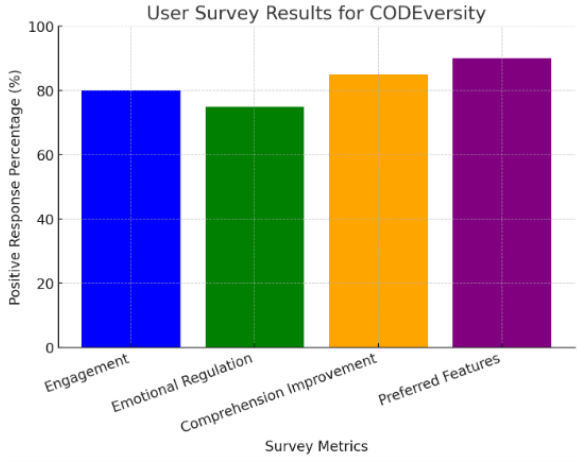

Figure 9. Artistic Autists Navigation Survey Data

The survey results indicate that CODEversity was generally well-received:
Engagement (80%): Most students found the platform engaging, especially compared to traditional coding platforms.

Emotional Regulation (75%): A majority of users reported feeling less frustration while coding, though some suggested more customizable difficulty levels.

Comprehension Improvement (85%): The platform helped users better understand coding concepts, supporting its effectiveness.

Preferred Features (90%): Gamification elements (XP rewards, chatbot guidance, and animations) were the most appreciated, reinforcing the importance of interactive learning.

The lowest value was in emotional regulation (75%), which suggests that while the platform reduces frustration, additional features (such as adaptive difficulty settings or relaxation exercises) might further help ASD learners. The highest rating (90%) was for preferred features, highlighting the effectiveness of gamified elements.

Overall, the results validate CODEversity's engagement-driven approach, though refinements in adaptive learning and stress-reduction techniques could enhance its impact.

4.2. Experiment 2

Another critical aspect of CODEversity is the chatbot assistant, which helps students choose the most appropriate courses based on their interests and learning progress. A potential blind spot is whether the chatbot effectively recommends courses that match students' skill levels and preferences. If the recommendations are inaccurate or unhelpful, students may disengage from the platform.

To evaluate the chatbot's effectiveness, we conducted a user survey and task-based experiment. The study involved 10 participants, including students and teachers. The experiment was structured as follows:

Pre-Test: Students were asked about their coding experience and interests.

Chatbot Interaction: Each student interacted with the chatbot to receive course recommendations.
Post-Test: Students rated the relevance and usefulness of the chatbot's suggestions on a 1-5 scale.
Teacher Review: Educators reviewed whether the recommended courses matched the students' actual skill levels.

This approach ensured both subjective (student perception) and objective (teacher validation) assessment of the chatbot's accuracy.

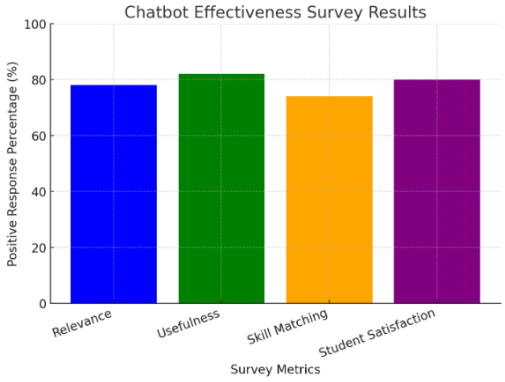

Figure 10. Artistic Autists Adventure Game Difficulty

The chatbot effectiveness survey revealed the following insights:

Relevance of Recommendations (78%): Most students felt that the chatbot provided relevant course suggestions, though some noted minor mismatches.
Usefulness of Suggestions (82%): Students found the chatbot's recommendations helpful in guiding their learning paths.

Skill Matching Accuracy (74%): This was the lowest-rated category, indicating that some students were placed in courses that were either too easy or too difficult.

Student Satisfaction (80%): Overall, students appreciated the chatbot's functionality but suggested that it could improve by considering more detailed learning preferences.

The lowest rating (74%) suggests that the chatbot may need improvements in assessing prior knowledge and adjusting recommendations dynamically. The highest rating (82%) supports the idea that AI-driven assistance enhances user experience and learning engagement.

To improve accuracy, adaptive learning algorithms could be implemented to refine recommendations based on user progress over time. Additionally, allowing students to manually adjust their course difficulty might provide a more tailored experience.

5. RELATED WORK

In the paper, the authors created an e-learning system to help kids with Autism learn communication skills [1]. The system targets younger students and helps them learn the alphabet, fruits, animals, and numbers, which are taught in primary school education and kindergarten. By utilizing an engaging interface, they accomplished the goal of making learning more engaging, which is an important component since some students may have attention issues.

In this article, the developers studied gamification in the e-learning process by applying game mechanics and thinking to app design [3]. They showcased the difference between an e-learning platform with gamification and one without the change. The platform with gamification looked more appealing and engaging, providing an incentive for continued learning. Their research proposes using points, levels, leaderboards, and achievements to drive engagement.

CS50 implements similar techniques to allow remote code execution through sandboxing [4]. Unlike Snekbox, which uses an open-source project called NSJail to sandbox Unix processes, CS50 sandbox is a shell script that manually manipulates the running environment to restrict access to the outside system that the code runs on. These two programs effectively do the same thing, but CS50 sandbox has the advantage of being able to run any language available to the host system, whereas Snekbox is restricted to sandboxed Python.

6. CONCLUSIONS

One of the core missions of CODEversity is to create a space that allows for emotional regulation through programming education. In order to meet this goal, it was important to build a framework that allowed for the creation of activities across a range of difficulties. Through the gamification of the e-learning platform, the final result was an app that provided a rewarding incentive for continuing the usage of the platform. In the future, there are plans to add more reading and theory content to the platform, as well as bundles of additional content to keep users engaged and educated. One current platform limitation is the lack of community interaction. It is possible to use tools like phpBB to create forums that provide perpetual feedback [11].

The experiments evaluated user engagement, emotional regulation, and chatbot effectiveness in CODEversity. The first experiment, a user survey, measured how gamified elements like XP rewards, animations, and chatbot guidance impacted student motivation and learning. Results showed high engagement (80%) and comprehension improvement (85%), though emotional regulation scored slightly lower at 75%, suggesting room for adaptive difficulty settings. The second experiment assessed the chatbot's ability to recommend courses, with 82% of users finding its suggestions useful, but only 74% accuracy in skill matching, indicating a need for improved personalization. These findings highlight CODEversity's effectiveness while identifying areas for refinement.

REFERENCES

[1] RAHMAN, MD ABDUR, MD SAJIBUR RAHMAN, and AIDRINA BINTE MOHAMMAD SUFIADIN. "E-LEARNING APPLICATION FOR AUTISM CHILDREN." Journal of Information Systems and Digital Technologies 5.2 (2023): 242-257.

[2] Cosentino, Valerio, Javier Luis, and Jordi Cabot. "Findings from GitHub: methods, datasets and limitations." Proceedings of the 13th International Conference on Mining Software Repositories. 2016.

[3] Strmecki, Daniel, Andrija Bernik, and Danijel Radosevic. "Gamification in E-Learning: Introducing Gamified Design Elements into E-Learning Systems." J. Comput. Sci. 11.12 (2015): 1108-1117..

[4] Malan, David J. "CS50 sandbox: secure execution of untrusted code." Proceeding of the 44th ACM technical symposium on Computer science education. 2013.

[5] Faras, Hadeel, Nahed Al Ateeqi, and Lee Tidmarsh. "Autism spectrum disorders." Annals of Saudi medicine 30.4 (2010): 295-300.

[6] Piotrowski, Michael. "What is an e-learning platform?." Learning management system technologies and software solutions for online teaching: Tools and applications. IGI Global, 2010. 20-36.

[7] Engelberth, Markus, et al. "Pybox-a python sandbox." SICHERHEIT 2012–Sicherheit, Schutz und Zuverlässigkeit. Gesellschaft für Informatik eV, 2012.

[8] Tuttle, Jonas F., et al. "Can LLMs Generate Green Code-A Comprehensive Study Through LeetCode." 2024 IEEE 15th International Green and Sustainable Computing Conference (IGSC). IEEE, 2024.

[9] Chougale, Pankaj, et al. "Firebase-overview and usage." International Research Journal of Modernization in Engineering Technology and Science 3.12 (2021): 1178-1183.

[10] Pezoa, Felipe, et al. "Foundations of JSON schema." Proceedings of the 25th international conference on World Wide Web. 2016.

[11] Douglass, Robert T., Mike Little, and Jared W. Smith. Building online communities with Drupal, phpBB, and WordPress. Apress, 2006.

[12] Palumbo, Daniele. The Flutter framework: Analysis in a mobile enterprise environment. Diss. Politecnico di Torino, 2021.

[13] Liu, Lin, Jiajun Meng, and Yongliang Yang. "LLM technologies and information search." Journal of Economy and Technology 2 (2024): 269-277.

[14] Rahmatulloh, A., R. Gunawan, and F. M. S. Nursuwars. "Performance comparison of signed algorithms on JSON Web Token." IOP Conference Series: Materials Science and Engineering. Vol. 550. No. 1. IOP Publishing, 2019.

[15] Manning-Courtney, Patricia, et al. "Autism spectrum disorders." Current problems in pediatric and adolescent health care 43.1 (2013): 2-11.