A Rule-based System Integrating Case-Based Reasoning for Adaptive User Interfaces in Personalized Educational Software

Christos Troussas, Akrivi Krouska, Phivos Mylonas, Cleo Sgouropoulou
Department of Informatics and Computer Engineering
University of West Attica
Egaleo, Greece
{ctrouss, akrouska, mylonasf, csgouro}@uniwa.gr

***Abstract*—This article introduces a personalized learning framework which employs case-based reasoning (CBR) and rule-based logic to facilitate adaptive user interfaces within educational software, designed for Java programming. While there has been progress in adaptive learning, it is limited when effectively combining cognitive theories and personalization strategies to adapt the instructional process with fidelity. Case-based reasoning serves to model human problem-solving through the process of retrieving and reusing solutions of prior similar cases. Rule-based logic can apply structured pedagogical rules that shape the cue to initiate the adaptation. Our framework consists of three components, each serving an important purpose in the cycle of interpersonal interactions that occur in educational settings. The Case Retrieval Module (CRM) identifies and selects a similar case from a prior learner interaction. The Rule-Based Inference Engine (RIE) identifies an appropriate option for adaptation in response to the prior case. Lastly, the Adaptive UI Controller (AUIC), executes the adaptations onto the user interface in real time. Real-time adaptations may include task difficulty, the provision of hints, or structure content presented based on the assessed proficiency of the learner. For more proficient learners, the task may be complex and hints omitted, while the adaptation might provide mediating support for novices. The evaluation of the prototype showed improved learner engagement, reduction in errors, and positive learning outcomes. The study highlights the role of cognitively grounded adaptations within an interactive learning environment and introduces a transparent and modular framework that extends prior case-based and rule-driven educational systems with real-time, interface-level personalization.**

Keywords—Adaptive learning systems; Case-based reasoning; Rule-based personalization; Intelligent user interfaces; Human-computer interaction

I. INTRODUCTION

The proliferation of educational software has reshaped the landscape of learning, offering dynamic, accessible, and scalable platforms that cater to a broad range of academic disciplines [1]. These tools have evolved from static content delivery systems to interactive environments that support diverse learning activities, including programming exercises, quizzes, simulations, and collaborative tasks [2]. As digital learning environments become more integrated into formal education, their design must go beyond content delivery to actively support the learning process.

A critical factor in enhancing the efficacy of educational software is personalization—the capability to tailor learning experiences to the unique needs, preferences, and behaviors of individual students [3, 4]. Personalized learning environments have shown to increase motivation, engagement, and academic performance by presenting content that aligns with a learner's current knowledge state, cognitive style, and pace of progression. As students differ in prior experience, confidence levels, and response to feedback, static instructional strategies often fail to provide optimal support for all learners [5-8].

To address this challenge, researchers have proposed the use of adaptive user interfaces (UIs) [9] within educational software [10, 11]. Adaptive UIs dynamically modify system behavior and presentation based on real-time user data, thus supporting differentiated instruction [12]. Such adaptations may include adjusting task difficulty, changing the presentation format, offering context-sensitive hints, or modifying feedback strategies. These mechanisms aim to reduce cognitive overload for novices while offering sufficient challenge for advanced learners.

However, achieving effective adaptation may require the integration of cognitive theories with personalization techniques to ensure pedagogically sound and user-aware behavior [13]. Cognitive theories such as constructivism, metacognition, and cognitive load theory inform how learners process information and respond to different forms of assistance. Techniques employed in personalized learning systems range from machine learning and bayesian networks to fuzzy logic and learner modeling [14]. These techniques are instrumental in inferring learner states and making informed adaptation decisions, yet their application is often disconnected from established cognitive frameworks.

While there is increased interest, the literature indicates there are few developed systems that legitimate integrate cognitive theories into personalization, particularly in terms of adaptive UIs for learning [15-20]. While there has been some

work to combine artificial intelligence with modelling of learners, the adoption of human-like reasoning paradigms, such as case-based reasoning (CBR), and rule-based pedagogical logic is less developed. This area is ripe for innovation in developing adaptive systems that respond to behavior in ways that are not only credible, but that act in cognitively plausible and transparent ways.

To meet this demand, the paper proposes a framework that combines case-based reasoning (CBR) with rule-based logic for the purpose of supporting adaptive user interfaces in educational software for the teaching of the Java programming language. CBR is a way of accomplishing the human ability to solve new problems by looking back on previous experiences that have similar characteristics. The rule-based logic provides these systems with the ability to impose structured pedagogical decisions using pre-defined rules. The proposed architecture consists of three primary components; (i) a Case Retrieval Module (CRM), which practices interaction data based on time on task, error rates and hint use to sort through and return previously used cases, (ii) a Rule-based Inference Engine (RIE) which reviews the case(s) and utilizes rules tied to the domain to recommend appropriate instructional actions, and (iii) an Adaptive UI Controller (AUIC) which implements the recommendation by changing the interface in real time such as modifying problem complexity, hints and/or content framing techniques. The system has been used and piloted with a postgraduate course on the Java programming language which produced favorable outcomes for student engagement and their learning.

II. Ease of Use

The proposed system is a smart educational platform for providing personalized learning experiences for Java programming. The main purpose of these features is to adapt the user interface and instructional content to suit learner characteristics and real-time interaction data. The system's intelligence relies on CBR in combination with Rule-Based Logic, support experience-based and structured approaches for decision-making.

The system architecture is organized into three modules that are interdependent to achieve this functionality:

- The CRM collects, stores and retrieves data from previous student actions in learning environments to find suitable learning cases.
- The RIE takes and processes these retrieved cases to determine the most suitable adaptive actions according to pedagogically-based rules.
- The AUIC implements the chosen adaptations in real-time by changing different components of the user interface for instance task difficulty, hints, and types of content based on the outcomes of the inferences.

All of the modules have unique but designed in a complementary manner to contribute to the adaptive behavior of the configuration system (Fig. 1). Each of the subsections will go into detail specifying how each component works and the rationale they followed into designing each section.

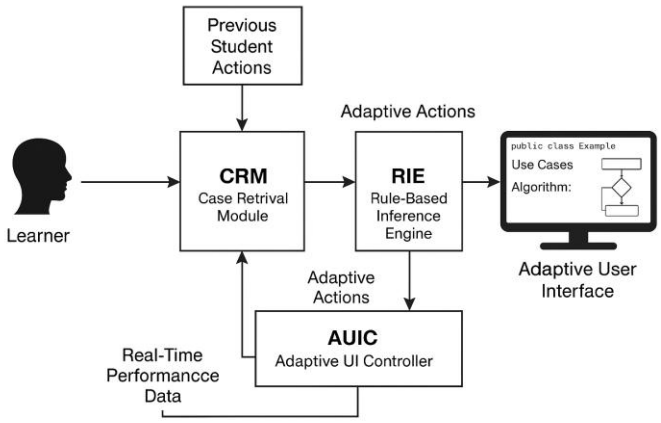

Fig. 1. System architecture.

A. Case Retrieval Module (CRM)

The CRM forms the foundational component of the reasoning architecture of the system. It implements the principles of CBR by maintaining an ongoing case base that serves as a structured archive of records of past experiences between the learner and the system. Each instance in this case base reflects a single learning experience, and it is recorded in a standard form, by a case object. Each case is a record of the state and performance of a learner in a specific activity. Formally, a case C is structured as a tuple:

$$C = \langle T_s, E_r, H_u, O_c, N_p, A \rangle$$

where

- T_s: Time spent on the activity (in seconds),
- E_r: Error rate, defined as the ratio of incorrect submissions to total attempts,
- H_u: Hint usage pattern, represented by both frequency and type (e.g., "on-request", "auto-displayed"),
- O_c: Outcome classification, such as "success", "partial success", or "failure",
- N_p: Navigation path, i.e., the sequence of UI elements accessed,
- A: Adaptation actions previously applied (e.g., "difficulty increased", "hints suppressed").

When a learner initiates a new activity, the CRM computes a feature vector $F_{current}$ representing the learner's real-time interaction profile. It then performs nearest-neighbor retrieval over the case base using a weighted similarity function:

$$Sim(C_i, F_{current}) = \sum_{k=1}^{n} w_k \cdot sim_k\left(v_k^{(i)}, v_k^{current}\right)$$

Here, $v_k^{(i)}$ is the value of the k-th attribute in case i, $v_k^{current}$ is the corresponding value from the current session, w_k is the weight assigned to attribute k, and sim_k is the similarity function for that attribute (e.g., inverse Euclidean distance for numeric values, Jaccard similarity for categorical patterns).

Suppose a current learner spends 7 minutes on an exercise, makes 2 syntax errors, and uses 1 on-request hint. The system encodes this as:

$$F_{current} = \langle 420, 0.25, \text{"low-on-request"}, -, - \rangle$$

The CRM compares this profile with historical cases. For instance:

- Case A: ⟨400, 0.20, "low-on-request", "success", "[hint, compile, retry]", "difficulty increased"⟩
- Case B: ⟨800,0.50, "high-auto", "partial success", "[hint, hint, give up]", "show scaffold"⟩

The CRM may be inclined to choose Case A because of the close time and error rate as well as the same hint behavior within Case A. The similarities will factor into the selection of precedent adaptation actions in that case (i.e., increased difficulty) and the next reasoning stage.

B. Rule-based Inference Engine

The Rule-Based Inference Engine (RIE) acts as the system's main decision-making process. The RIE elements convert the learner interaction patterns that are retrieved into specific, adaptive UI actions relying on a predetermined collection of manually constructed inference rules. These inference rules are based on pedagogy theory, and aim to imitate human-tutor decision heuristics in a clear and reproducible way.

The rule base was developed through an iterative, knowledge-based process that involved

- An empirical analysis of student interaction logs (from previous deployments of programming courses in Java) concentrating on time-on-task, clustered errors, and hint usage patterns.
- Conversations with domain experts (instructors and instructional designers) to identify possible pedagogical actions in response to observable behaviours.
- Integration of cognitive and instructional theories to inform the design, including Cognitive Load Theory, the Zone of Proximal Development, and Scaffolding Theory, to provide a theoretical foundation
- Simulation of scenarios with rule validation through walkthroughs and mock sessions with synthetic learners to confirm coverage and resolution of rules.

The resulting rule base consists of 24 inference rules, divided into three main domains of rules:

- Difficulty Management (9 rules): Modifies the level of difficulty for the next learning task.
- Guidance and Scaffolding (10 rules): Manages hint visibility, step-based support, and prioritization of concepts.
- Motivational and Engagement Support (5 rules): Initiates motivational feedback and simpler UI when users disengage. All rules follow a regular IF-AND-THEN format, and each rule is embedded in the system as a triplet:

$$R_i = \langle \text{Conditions, Actions, Priority} \rangle$$

The Conditions specify whether the parameter is from the current learner profile or a retrieved case or cases. The Actions are deterministic and can be directly executed by the AUIC. The Priority is there to address conflicts. The priorities were established through expert consensus, based on the pedagogical significance, and intended instructional effect, of each rule within scenario-based simulations.

Five illustrative rules are shown below that demonstrate a range of adaptation intentions (Table 1).

TABLE I. REPRESENTATIVE RULES

Rule Name	Category	Rule
R1 - Difficulty Escalation	Difficulty Management	IF time_spent < 300 seconds AND error_rate < 0.15 AND hint_usage = "none" THEN increase difficulty_level by one tier, disable hints by default.
R2 - Maintain Progression	Difficulty Management	IF 300 ≤ time_spent ≤ 900 AND 0.15 ≤ error_rate ≤ 0.40 AND hint_usage = "on-request" THEN keep current difficulty, keep hints available on demand.
R3 - Activate Scaffolding Mode	Guidance and Scaffolding	IF error_rate > 0.50 AND hint_usage = "auto-displayed" AND task_completed = false THEN enable step-by-step guidance, show syntax highlight, display related solved example.
R4 - Decrease Complexity & Visual Simplification	Guidance and Scaffolding	IF time_spent > 900 seconds AND navigation_pattern = "repeated back-and-forth" THEN decrease difficulty_level, simplify layout, show only key concepts.
R5 - Provide Motivational Feedback	Motivation	IF inactivity_period > 120 seconds AND last_action = "task_abort" THEN display motivational message ("You' re making progress!"), highlight next recommended task.

The RIE executes evaluations of multiple rules for each instance, and selects non-conflicting rules using a confidence-

based and priority-aware prioritization approach. When multiple rules are triggered with conflicting outcomes, the engine will utilize one or more of the following criteria:

- Preference for specificity (the rules with the most conditions are preferred),
- Pruning based on outcomes (the rules demonstrate poor outcomes historically are deprioritized),
- Priority tags assigned to rules as defined.

The resulting output is a single coherent adaptation plan that is then passed to the AUIC for immediate execution.

C. Adaptive UI Controller

The AUIC is the last operational phase of the adaptive system pipeline and acts as the executor at the interface level for pedagogical decisions made through reasoning upstream. While the RIE enacts a pedagogical action plan given retrieved cases and rules, the AUIC interprets and executes these plans as concrete, visible, and interactive elements of the educational software interface. In other words, the AUIC serves as a connection between intelligent reasoning and the user's immediate experiences, enabling responses that are adaptive and purposeful in real time for each learner.

To support responsive adaptation of the interface, the AUIC is built upon a modular, state-aware UI architecture that can respond in real time without interrupting the user's task. This system receives structured adaptation directives, including precise values for parameters from the RIE, for example, the complexity level of the next task, hint delivery mode, the extent to which instructional scaffolding is warranted, and formatting of the UI components. The AUIC receives and parses the directives from the RIE to implement them in real time using a component-based rendering engine that can selectively enable or disable UI features.

A major asset of the AUIC is its formal integration of Human-Computer Interaction (HCI) design principles into the adaptation logic. Adaptations are not made arbitrarily, or mechanically; instead, the system guarantees all interface alterations abide by the consistency, continuity, and usability expectations that are prerequisites for the "inflow moments" of learning. For example, layout consistency is maintained even as content changes in real time, preventing spatial disorientation. Visual transitions, for instance when a scaffolded hint or an annotation on a piece of code appears, are animated/rendered in a fluid manner, making the change apparent, but without distorting the learner's attention. Similarly, responsiveness of the interface is key—adaptation decisions are implemented with minimum delay, and signals/feedback are given to the user as quickly as possible, enabling perceived seamless flow and awareness.

The AUIC allows for a broad range of adaptive behaviors to support the user. For example, when a learner is performing at a high level, indicated by a consistent error rate well below the threshold, short completion times, and not having heavily relied on hints, the AUIC will sometimes modify the task to become more complex. This could be in the form of an extend exercise to one that is more abstract, seamlessly transitioning to a more expansive programming activity, disabling inline hints, or optimizing coding interfaces to eliminate unnecessary scaffolding. Equally important, these changes are made without losing the predictability of the interface—buttons, panels, and interaction zones are all still anchored in the same familiar positioning, so the user is never visually lost, despite the challenge increasing.

On the other hand, for learners who show signs of difficulty (e.g., repeated submission errors, too long of a pause, or reliance on hints), the AUIC decreases the difficulty gradient and supports incremental scaffolding (i.e., turning long task instructions into sequenced input, providing on-demand code examples, or using syntax-aware highlights in the editor). These scaffolding factors are thoughtfully designed into the existing UI platform, creating a compromise between visibility and cognitive load while maintaining a consistent visual hierarchy to direct attention without overloading the learner.

The AUIC enacts alterations in a transparent and explainable way. All alterations of the interfaces are recorded with specificity with timestamp, triggers, and adaptations made. The recorded alterations support instructor review and evaluation of the system while providing learner-facing features like tooltips, or an expandable feature with the rationale for the change (e.g. "the hints were turned off because you were efficient at solving similar problems."). This credible transparency builds trust and helps encourage learner interrogation of one's own progress.

Additionally, the AUIC was designed with incremental extensibility in mind. The system is modular, so that developers and instructional designers can create new UI adaptation patterns, or integrate other modalities of input (e.g., emotion detection, eye-tracking) without having to change or remake the core system. This ultimately positions the AUIC to be more than just an adaptive controller but flexible experimentation layer for future research in designing adaptive interaction.

In summary, the AUIC offers more than merely a device to reconfigure the interface. The system represents an HCI-aware philosophy of adaptation, in which the decisions make by the system are realized in a pedagogically informed, interactionally fluid, and responsive to learner perception and behavior. The AUIC is an active force in sustaining a coherent, motivating, and personalized learning experience, that advances the intersection of intelligent decision making and user-centered design of educational interface.

III. EVALUATION

In order to assess the utility of the proposed adaptive framework, we incorporated it in a custom-designed educational software platform where the subject matter is the Java programming language. This platform contains all of the system modules, including the CRM, RIE, and AUIC, and served as the primary learning context during a postgraduate course in object-oriented programming.

A. *Experimental Setup*

The assessment involved 60 students who were postgraduate students with at least basic programming experience, although Java proficiency varied. The research was conducted over a four-week period and consisted of compulsory weekly lab sessions, which students were also encouraged to use outside of class. Participants were assigned to either a testing group or a control group:

- The experimental group (n = 40) utilized the complete adaptive version of the system, where the interface was adapted to the learner model in real-time.
- The control group (n = 20) used the same system with a fixed interface. In the fixed interface, task difficulty was fixed with difficulty not connected to its level, hints were provided and could be obtained on demand and the interface provided no scaffolding or progression logic.

Both participant groups received the same instructional content and were asked to complete the same sequence of similar programming exercises.

We gathered both quantitative and qualitative data. Our key performance indicators included:

- Percentage of successful task completion
- Percentage of incorrect submissions for each task
- Minutes on task
- Number of hints used for each task
- Percentage of students who completed all tasks
- Learner satisfaction, assessed with a post-study questionnaire on a 5-point Likert scale.

We anonymized and analyzed interaction logs to determine behavior patterns and trends in engagement. The AUIC (Adaptive User Interface Components) also provided logs of adaptation traces that could be analyzed after the study.

B. *Results*

The quantitative evidence suggests that the adaptive system had a positive impact on performance and engagement. The main findings are represented in Table 2 and Figure 2. The control group used an adaptive version of the system with a fixed UI and no personalized features. The success rate of the adaptive version (87.2%) was significantly higher than the static version (74.5%), confirming that learners were benefiting from personalized difficulty and scaffolding. Furthermore, errors decreased almost 50%, which suggested that adaptive guidance exemplified fewer errors and better conceptual understanding. Learners using the adaptive version spent less time on a task, and therefore, gained efficiency, primarily among more proficient learners who were adequately challenged. The number of hints in the adaptive version decreased, which mainly occurred due to students who struggled needing scaffolding and students who were proficient having their hints suppressed.

The exercises completion rate was significantly higher for the adaptive group than the non-adaptive group; while this was true (95%), almost all subject also scored the survey measures higher using the adaptive system (mean score: 4.42 out of 5).

TABLE II. QUANTITATIVE RESULTS.

Metric	Adaptive System (n=60)	Control Group (n=20)	p-value (t-test)
Task Success Rate (%)	89.5	74.0	p<0.01
Average Error Rate (%)	10.3	20.6	p<0.01
Average Time per Task (min)	8.4	10.7	p<0.01
Hint Requests per Task	1.1	2.5	p<0.01
Completion Rate (%)	95.0	82	n.s.
Satisfaction Score (1–5)	4.4	3.6	p<0.01

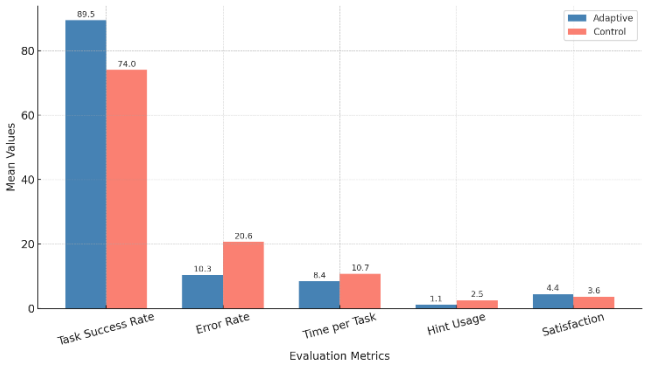

Fig. 2. Performance comparison between Adaptive and Control Group.

Statistical comparisons were conducted with independent two-sample t-tests test with unequal variance. The results showed statistical differences improving task success, error counts, time, and customer satisfaction in the adaptive condition. The difference in completion rates, despite the adaptive group having more successful completion rates over the traditional group, was not statistically significant.

C. *User Feedback and Observations*

Qualitative feedback received from open response questions and interviews shed some light on user experience. A number of learners describe feeling “guided but not bombarded” and appreciated the step-by-step explanations as well as the hints that were context-sensitive in the environment. Advanced users noted the system “removed the unnecessary help” and “the level of challenge was just right”. A small percentage (about 8%) reported that they preferred more control over seeing hints in the environment, suggesting there may be an opportunity for customizable adaptation preferences in the next iteration.

D. Discussion

The evaluation findings lend support to the hypothesis that the inclusion of case-based reasoning into rule-based logic may serve to enhance personalization of learning experiences. In terms of pedagogy, the system was able to both accelerate proficient learners and scaffold struggling learners, which aligns well with the goals of differentiated instruction. With the reduced error rates and the reduced frequency and dependence on hints, together with greater engagement and satisfaction, suggest that the adaptive decisions were, at least largely appropriate and well-executed by the AUIC. Overall, this study appears to provide some educational benefit and could potentially apply to other content areas that have structured problem-solving tasks. While the outcome of the evaluation is promising, it does have limitations in that it was of relatively short duration and was only focused on one subject area. Future studies might examine long-term retention, the transparency of adequate adaptation, and the leverage of learner-controlled personalization in conjunction with automated adaptation.

IV. Conclusions And Future Work

This paper introduced a rule-driven, case-based adaptive framework for personalized user interfaces in educational environments, to improve the learning experience within programming education. The framework leverages CBR and rule-based logic to provide context-sensitive adaptations to users' learning, behavioral patterns, and pedagogical principles of the learning task. The architecture was developed with three main modules in parallel, the CRM, the RIE, and AUIC, which worked together in real time to monitor, reason, and adapt to learners.

The framework was implemented within an educational software system for learning about Java programming, and evaluated in a study with 60 postgraduate students. The analysis demonstrated significantly better learning performance, reduced errors, increased task completion, and positive levels of user satisfaction. The results corroborated the advantages of integrating cognitively plausible reasoning mechanisms to adaptive user interfaces.

While we are encouraged by the initial findings, several directions remain available for future work. First, we designed and validated the rule base manually, and next, we will examine data-driven optimization methods, such as reinforcement learning, to tune the weights of rules or even identify new rules. Second, the current version of the system is limited to unidimensional adaptation, primarily based on the use of time and errors, and hints, but future iterations could include multi-modal interaction features such as eye tracking, keystroke dynamics, and emotional feedback to better assess cognitive states and affective engagement.

Next, we plan to investigate learner-controlled personalization, so that learners can modify some of adaptive behaviors or override them altogether. This blended approach may be able to take advantage of some of the benefits of automation, however balance these advantages with transparency and agency, which may be preferred by some users. A final contribution for future iterations of the system is expanding the framework to process additional subject areas and learner populations to check for generalizability and robustness across educational contexts.

In summary, our proposed system represents a promising advancement in the direction toward bringing together cognitive models and adaptive educational technologies. With its transparent, and interpretable logic, and modular design our proposed system provides both pedagogical grounding and technical scalability for a promising path forward in adaptive learning environments.

References

[1] N. Chondamrongkul, G. Hristov, and P. Temdee, "Addressing technical challenges in large language model-driven educational software system," *IEEE Access*, vol. 13, pp. 12846–12858, 2025. [Online]. Available: http://dx.doi.org/10.1109/ACCESS.2025.3531380

[2] A. Venugopal, "Impact of digital learning platforms on student academic performance," *Medicon Eng. Themes*, vol. 6, no. 5, pp. 19–21, 2024. [Online]. Available: http://dx.doi.org/10.55162/MCET.06.208

[3] D. Vallet, P. Mylonas, M. A. Corella, J. M. Fuentes, P. Castells, and Y. Avrithis, "A semantically-enhanced personalization framework for knowledge-driven media services," in *Proc. IADIS Int. Conf. WWW/Internet (ICWI)*, 2005. [Online]. Available: http://hdl.handle.net/10486/665989

[4] C. Troussas, A. Krouska, and C. Sgouropoulou, "Dynamic detection of learning modalities using fuzzy logic in students' interaction activities," in *Proc. 15th Int. Conf. Intelligent Tutoring Systems (ITS)*, V. Kumar and C. Troussas, Eds., Lecture Notes in Computer Science, vol. 12149, Springer, Cham, pp. 276–286, 2020. [Online]. Available: http://dx.doi.org/10.1007/978-3-030-49663-0_24

[5] E. T. Khor and K. M., "A systematic review of the role of learning analytics in supporting personalized learning," *Educ. Sci.*, vol. 14, no. 1, p. 51, 2024. [Online]. Available: http://dx.doi.org/10.3390/educsci14010051

[6] R. I. Fariani, K. Junus, and H. B. Santoso, "A systematic literature review on personalised learning in the higher education context," *Tech. Knowl. Learn.*, vol. 28, pp. 449–476, 2023. [Online]. Available: http://dx.doi.org/10.1007/s10758-022-09628-4

[7] M. Gunawardena, P. Bishop, and K. Aviruppola, "Personalized learning: The simple, the complicated, the complex and the chaotic," *Teach. Teach. Educ.*, vol. 139, p. 104429, 2024. [Online]. Available: http://dx.doi.org/10.1016/j.tate.2023.104429

[8] K. Bayly-Castaneda, M.-S. Ramirez-Montoya, and A. Morita-Alexander, "Crafting personalized learning paths with AI for lifelong learning: a systematic literature review," *Front. Educ.*, vol. 9, Aug. 2024. [Online]. Available: http://dx.doi.org/10.3389/feduc.2024.1424386

[9] D. Gaspar-Figueiredo, M. Fernández-Diego, R. Nuredini, S. Abrahao, and E. Insfran, "Reinforcement learning-based framework for the intelligent adaptation of user interfaces," in *Companion Proc. 16th ACM SIGCHI Symp. Eng. Interactive Computing Systems (EICS '24 Companion)*, ACM, New York, NY, pp. 40–48, 2024. [Online]. Available: http://dx.doi.org/10.1145/3660515.3661329

[10] S. V. Kolekar, R. M. Pai, and M. P. M. M., "Rule based adaptive user interface for adaptive e-learning system," *Educ. Inf. Technol.*, vol. 24, pp. 613–641, 2019. [Online]. Available: https://doi.org/10.1007/s10639-018-9788-1

[11] B. A. Bagustari and H. B. Santoso, "Adaptive user interface of learning management systems for Education 4.0: a research perspective," in *Proc. 3rd Int. Conf. Computing and Applied Informatics (ICCAI)*, Medan, Indonesia, Sep. 18–19, 2018, *J. Phys.: Conf. Ser.*, vol. 1235, p. 012033, 2019. [Online]. Available: http://dx.doi.org/10.1088/1742-6596/1235/1/012033

[12] S. Abrahão, E. Insfran, A. Sluÿters, *et al.*, "Model-based intelligent user interface adaptation: challenges and future directions," *Softw. Syst.*

Model., vol. 20, pp. 1335–1349, 2021. [Online]. Available: http://dx.doi.org/10.1007/s10270-021-00909-7

[13] C. Troussas, A. Krouska, and C. Sgouropoulou, "Cognitive styles as a factor of effective learning," in *Human-Computer Interaction and Augmented Intelligence*, Cognitive Systems Monographs, vol. 34, Springer, Cham, 2025. [Online]. Available: http://dx.doi.org/10.1007/978-3-031-84453-9_4

[14] A. Almalawi, B. Soh, A. Li, and H. Samra, "Predictive models for educational purposes: a systematic review," *Big Data Cogn. Comput.*, vol. 8, no. 12, p. 187, 2024. [Online]. Available: http://dx.doi.org/10.3390/bdcc8120187

[15] A. Ezzaim, A. Dahbi, A. Haidine, and A. Aqqal, "AI-based adaptive learning: a systematic mapping of the literature," *J. Univers. Comput. Sci.*, vol. 29, no. 10, pp. 1161–1198, 2023. [Online]. Available: http://dx.doi.org/10.3897/jucs.90528

[16] M. Y. Mustafa, A. Tlili, G. Lampropoulos, *et al*., "A systematic review of literature reviews on artificial intelligence in education (AIED): a roadmap to a future research agenda," *Smart Learn. Environ.*, vol. 11, p. 59, 2024. [Online]. Available: http://dx.doi.org/10.1186/s40561-024-00350-5

[17] K. I. Vorobyeva, S. Belous, N. V. Savchenko, L. M. Smirnova, S. A. Nikitina, and S. P. Zhdanov, "Personalized learning through AI: pedagogical approaches and critical insights," *Contemp. Educ. Technol.*, vol. 17, no. 2, p. ep574, 2025. [Online]. Available: http://dx.doi.org/10.30935/cedtech/16108

[18] D. Koutsantonis, K. Koutsantonis, N. P. Bakas, V. Plevris, A. Langousis, and S. A. Chatzichristofis, "Bibliometric literature review of adaptive learning systems," *Sustainability*, vol. 14, no. 19, p. 12684, 2022. [Online]. Available: http://dx.doi.org/10.3390/su141912684

[19] H. Lin and Q. Chen, "Artificial intelligence (AI)-integrated educational applications and college students' creativity and academic emotions: students and teachers' perceptions and attitudes," *BMC Psychol.*, vol. 12, p. 487, 2024. [Online]. Available: http://dx.doi.org/10.1186/s40359-024-01979-0

[20] L. Major and G. A. Francis, "Technology-supported personalised learning: a rapid evidence review," *EdTech Hub Rapid Evidence Review*, 2020. [Online]. Available: http://dx.doi.org/10.5281/zenodo.4556925

[21] N. Md Noh, A. Ahmad, S. Ab. Halim, and A. Mohd Ali, "Intelligent tutoring system using rule-based and case-based: a comparison," *Procedia Soc. Behav. Sci.*, vol. 67, pp. 454–463, 2012. [Online]. Available: http://dx.doi.org/10.1016/j.sbspro.2012.11.350

[22] N. B. M. Noh, R. Yusof, O. Ono, and T. Tojo, "Feedback preferences in case-base construction for intelligent lab tutor," in *Proc. AsiaSim 2014*, S. Tanaka, K. Hasegawa, R. Xu, N. Sakamoto, and S. J. Turner, Eds., *Commun. Comput. Inf. Sci.*, vol. 474, Springer, Berlin, Heidelberg, 2014. [Online]. Available: http://dx.doi.org/10.1007/978-3-662-45289-9_25

[23] D. Soto Forero, S. Ackermann, M.-L. Betbeder, and J. Henriet, "The intelligent tutoring system AI-VT with case-based reasoning and real-time recommender models," in *Proc. Int. Conf. Case-Based Reasoning*, Mérida, Mexico, Jul. 2024. [Online]. Available: https://hal.science/hal-04693047

[24] V. Aleven, "Rule-based cognitive modeling for intelligent tutoring systems," in *Advances in Intelligent Tutoring Systems*, R. Nkambou, J. Bourdeau, and R. Mizoguchi, Eds., *Stud. Comput. Intell.*, vol. 308, Springer, Berlin, Heidelberg, pp. 33–62, 2010. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-14363-2_3

[25] Shweta, *et al*., "Agent-based distributed intelligent tutoring system using case-based reasoning," in *Artificial Intelligence Applications in Distance Education*, U. Kose and D. Koc, Eds., IGI Global, pp. 211–236, 2015. [Online]. Available: http://dx.doi.org/10.4018/978-1-4666-6276-6.ch013