GAMED.AI: A Hierarchical Multi-Agent Framework for Automated Educational Game Generation

Anonymous ACL submission

Abstract

We introduce GAMED.AI, a hierarchical multi-agent framework that transforms instructor-provided questions into fully playable, pedagogically grounded educational games validated through formal mechanic contracts. Built on phase-based LangGraph sub-graphs, deterministic Quality Gates, and structured Pydantic schemas, GAMED.AI supports two template families encompassing 15 interaction mechanics across spatial reasoning, procedural execution, and higher-order Bloom's Taxonomy objectives. Evaluated on 200 questions spanning five subject domains and all 15 mechanics against manual authoring, EdTech platforms, GameGPT, and agentic coding tools, the system achieves a 90% validation pass rate and 73% token reduction over ReAct agents ($\sim$73,500 $\rightarrow$ $\sim$19,900 tokens/game), demonstrating that architectural discipline—not model capability—is the binding variable for alignment quality. Our demonstration interface lets attendees generate games from natural language in under 60 seconds, inspect Quality Gate outputs, and browse 50 curated games. Code, games, and evaluation datasets are publicly available.[1]

1 Introduction

Large Language Models now resolve 50–64% of real-world engineering tasks (Jimenez et al., 2024) and achieve $\sim$90% Pass@1 on function-level benchmarks (Chen et al., 2021), yet their effectiveness in producing *pedagogically valid* educational content remains limited—particularly where Bloom's Taxonomy alignment, mechanic contract enforcement, and structured competency evidence are required (Mislevy et al., 2003; Shute and Ventura, 2013).

This gap matters because game-based assessments are among the most effective modalities for higher-order learning, with meta-analytic effect sizes of $g = 0.49$ on cognitive outcomes (Sailer and Homner, 2020), $g = 0.78$ on academic performance (Zeng et al., 2024), and $d = 0.29$ on learning (Wouters et al., 2013), yet one finished hour of game-based content requires 490 development hours at costs exceeding $50,000 (Chapman Alliance, 2010). Existing platforms reduce delivery friction but not creation friction, with mechanics routinely decoupled from learning objectives (Wang and Tahir, 2020). General-purpose agentic tools generate functional games—but without grounding in learning outcomes and validation against educational contracts, a syntactically correct game can be semantically wrong as an assessment artifact—e.g., a drag-and-drop game testing *recall* when the objective requires *analysis* (Mislevy et al., 2003; Ji et al., 2023).

We introduce GAMED.AI, a hierarchical multi-agent framework that transforms instructor-provided questions into Bloom's-aligned educational games validated through formal mechanic contracts. Built on a LangGraph DAG with phase-specific sub-graphs, deterministic Quality Gates, and typed Pydantic schemas, GAMED.AI eliminates the silent error propagation that makes prior agentic architectures impractical for structured content generation (Kuznia et al., 2024; Yao et al., 2023). The system generates validated games in under 60 seconds at $0.48 per game—achieving 73% token reduction over ReAct agents and 90% validation pass rate across 200 test questions covering all 15 mechanics. Our contributions:

- To our knowledge, the first hierarchical multi-agent framework for educational game generation, with 15 interaction mechanics and Bloom's alignment contracts enforced before generation. Code and a curated set of 50 games (selected from the 200-question evaluation corpus) are open-sourced.

[1] Repository and demo: `https://github.com/[redacted]/GamifyAssessment`

Figure 1: **End-to-end system demonstration. (a)** Instructor enters a natural language question with domain and Bloom's level context. **(b)** DAG pipeline with real-time observability: per-agent traces, token/cost analytics ($0.48, <60 s), and Quality Gate decisions. **(c)** Game engine architecture: plugin-based mechanic registry dispatching to 15 self-contained React components backed by a unified Zustand store and dnd-kit interaction primitives. **(d)** Generated trace-path game: blood flow through the heart with animated particle visualization, 9 interactive zones, and dual learn/test modes.

- 90% validation pass rate and 73% token reduction over ReAct agents, outperforming Claude Code on Bloom's alignment under all four prompting conditions.

- A live demo enabling real-time game generation, pipeline observability, and a browsable library of 50 curated games spanning all 15 mechanics.

2 Related Work

Active engagement outperforms passive reception (Freeman et al., 2014), and dual coding theory shows verbal–visual learning encodes more durably than either channel alone (Paivio, 1991; Mayer, 2009). Bloom's Taxonomy (Bloom, 1956; Anderson and Krathwohl, 2001) and Evidence-Centered Design (Mislevy et al., 2003) require that mechanics constitute valid competency evidence; the LM-GM framework (Arnab et al., 2015) provides a theoretically grounded design heuristic for this learning-to-game mechanic mapping, adopted in our Bloom's constraint table (Appendix A); empirical validation of mechanic-to-cognitive-level correspondences remains open. Gamification succeeds when mechanics match learning goals (Sailer and Homner, 2020; Deterding et al., 2011) and fails when applied decoratively (Hamari et al., 2014; Landers, 2014). Formative assessment design (Black and Wiliam, 1998; Shute, 2008) informs our per-element feedback design at QG3, where each interaction zone or step receives targeted formative feedback.

Multi-agent architectures address compounding errors in generation. MetaGPT (Hong et al., 2023) and AutoGen (Wu et al., 2023) use role-bounded schemas; ReAct (Yao et al., 2023) adds self-correction but produces token inflation on constrained tasks, with performance driven by exemplar-query similarity rather than reasoning (Kuznia et al., 2024). Flow engineering (Ridnik et al., 2024) and hierarchical DAGs make invalid states structurally unreachable (Willard and Louf, 2023).

Widely adopted platforms (Kahoot, Quizlet, H5P, Genially) require manual authoring without objective alignment (Wang and Tahir, 2020); AutoTutor (Graesser et al., 2004) and GIFT (Sottilare et al., 2012) offer depth but demand inaccessible knowledge engineering (Koedinger et al., 2006). GameGPT (Chen et al., 2023) addresses speed without Bloom's targeting; agentic coding tools produce Bloom's-aligned games only 23–67% of the time (Ji et al., 2023; Mislevy et al., 2003). Ngu et al. (2025) propose a generative AI game framework with multi-scaffolding ($n = 91$), but rely on a single LLM call without mechanic contracts or formal validation, limiting reproducibility and structural guarantees. GAMED.AI is, to our knowledge, the first open-source system integrating automated generation, Bloom's alignment, FOL-based contract validation, and a modular game engine in a single deployable framework.

3 System Design

GAMED.AI accepts a natural language question or topic—with optional context (subject domain, target audience, difficulty level)—and produces a fully playable game, a structured alignment report, and a validation certificate confirming mechanic contracts are satisfied. Four design principles govern all architectural decisions:

- **Pedagogical primacy:** Every game is bound to a Bloom's level before generation; mechanic selection follows learning objectives (Anderson and Krathwohl, 2001; Mislevy et al., 2003).
- **Deterministic validation:** Every generative step is gated by a deterministic validator; LLM outputs are proposals subject to structural verification (Ji et al., 2023).
- **Structure over retry:** Typed schemas and phase boundaries prevent errors rather than catching them downstream (Willard and Louf, 2023).
- **Modularity:** New templates are registered via contract definition without modifying orchestration (Hong et al., 2023; Wu et al., 2023).

3.1 Architectural Evolution

The current DAG architecture supersedes two prior designs: a **Sequential Pipeline** (56.7% VPR, ∼49,400 tokens/game) and a **ReAct Agent** system (72.5% VPR, ∼73,500 tokens/game). The full evolution with failure analysis is in Appendix D.

3.2 DAG Architecture

The current architecture emerged from a diagnostic insight: prior designs conflated generation and validation into the same cognitive loop. The DAG separates them into three deterministic phases, each bounded by a Quality Gate.

3.2.1 System Architecture

The system is a hierarchical DAG in LangGraph with three phases—**Planning**, **Generation**, **Assembly**—each an independent sub-graph with typed I/O and a Quality Gate at its boundary (Figure 2). No agent in phase N receives input from phase $N+1$; no gate can be bypassed; invalid states cannot propagate—a structural guarantee of the DAG topology (Ridnik et al., 2024; Hong et al., 2023).

3.2.2 Game Template Architecture

The generative surface comprises **two template families** with **15 interaction mechanics**. **Interactive Diagram Games** (10 mechanics: drag-and-drop, click-to-identify, trace-path, description matching, sequencing, sorting, memory match, branching scenario, compare/contrast, hierarchical) operate on spatial and relational content targeting visual and conceptual reasoning (Mayer, 2009; Sweller, 1988). **Interactive Algorithm Games** (5 mechanics: state tracer, bug hunter, algorithm builder, complexity analyzer, constraint puzzle) operate on procedural content targeting *applying*, *analyzing*, and *creating* objectives, grounded in algorithm visualization research (Hundhausen et al., 2002; Naps et al., 2002; Anderson and Krathwohl, 2001) and debugging-first pedagogy (Lee et al., 2014; Koedinger et al., 2006). Together, these support a library of **50 curated games** (selected from the 200-question evaluation corpus) across five domains; the full Bloom's-to-mechanic mapping is in Appendix A.

3.2.3 Mechanic Contracts and Blueprint Generation

Template selection is a **constrained inference** in Phase 1: the planning agent resolves input against a Bloom's-to-mechanic constraint table encoding valid competency evidence (Mislevy et al., 2003). The result is a **Game Blueprint**—a validated Pydantic document specifying learning objective, Bloom's level, template, and mechanic contract—before content generation begins. Each contract defines the interaction primitive, content types, valid Bloom's range, and completion conditions, enforcing pedagogical alignment as a structural constraint (Anderson and Krathwohl, 2001; Shute and Ventura, 2013).

3.2.4 Scene and Mechanic Composition

Templates span three structural configurations resolved automatically from Bloom's level and content complexity:

Single-scene, single-mechanic—one interaction type, one content context; covers ∼35% of the library.

Single-scene, multi-mechanic—2–3 interaction types within one content frame, validated through a state machine ensuring compatible I/O schemas; covers ∼40% (Sweller, 1988).

Multi-scene, multi-mechanic—2–4 causally connected scenes with monotonically increasing Bloom's levels, bounded by cognitive load constraints (≤4 scenes, ≤3 mechanics/scene); covers ∼25% (Sweller, 1988).

3.2.5 Generation and Assembly

Phase 2 sub-agents produce three asset classes in parallel: visual assets (SVG diagrams or text-

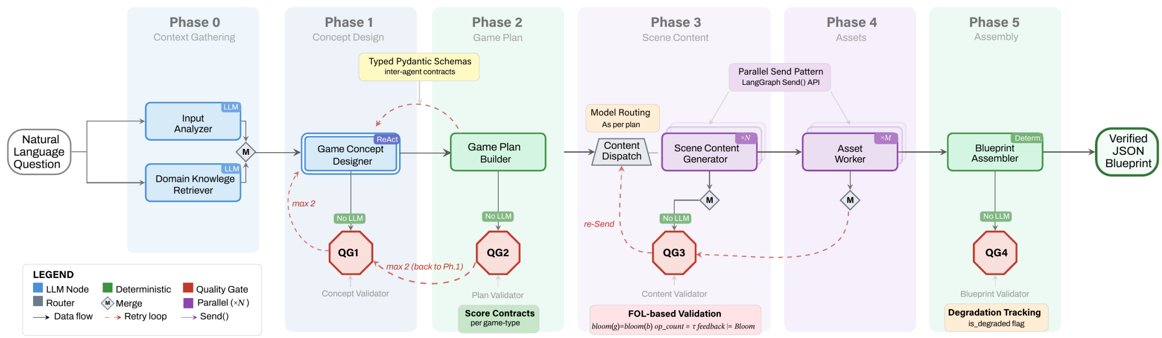

Figure 2: **GAMED.AI DAG architecture.** Six phases—**Context Gathering**, **Concept Design**, **Game Plan**, **Scene Content**, **Asset Generation**, **Assembly**—each an independent sub-graph. Four deterministic Quality Gates (QG1–QG4) enforce phase boundaries; QG3 applies FOL-based Bloom's alignment predicates. Typed Pydantic schemas govern inter-agent contracts; parallel Send patterns dispatch per-scene workers in Phases 3–4; dashed edges indicate bounded retry loops (max 1–2).

synthesised visuals), instructional text (directions, hints, per-node feedback), and interaction specifications (drag targets, click regions, sequence orders). QG2 validates all three against the mechanic contract before assembly. **Phase 3** instantiates the selected template as a React component and injects validated content—the same orchestration layer produces all 15 mechanics through component swapping, not code regeneration. Inter-agent communication uses Pydantic schemas (98.3% compliance); QG3 checks mechanic completeness and Bloom's alignment (Wu et al., 2023; Hong et al., 2023).

QG3: FOL-based alignment validator. Quality Gates apply **first-order logic (FOL) rules** derived from mechanic contracts and **mechanic-specific constraint graphs** encoding valid state transitions. At QG3, a game passes iff three predicates hold: (1) $\mathrm{bloom}(g) = \mathrm{bloom}(b)$ (level match); (2) $\mathrm{op\_count}(g) \geq \tau_{\mathrm{contract}}$ (operation count; failure: `BLOOM_OP_COUNT_FAIL`); (3) per-element feedback predicates entail the target Bloom's level (failure: `BLOOM_FEEDBACK_MISMATCH`). Each mechanic's constraint graph is traversed to verify reachability and completeness. All predicates use deterministic rule evaluation with **no LLM inference**, ensuring constant cost and formal verifiability.

3.2.6 Deployment and Game Library

The orchestration layer is model-agnostic: a declarative preset system enables per-agent model selection across closed-source APIs (GPT-4 OpenAI, 2023, Gemini Google DeepMind, 2023) and open-source models (Llama 3, Mistral) without pipeline modification. The 50-game library (curated from the evaluation corpus) serves as both demo set and regression corpus; every game emits structured outcome data including score, interaction trace, and inferred Bloom's level.

3.3 Modular Game Engine

The frontend implements a **plugin architecture**: each of the 15 mechanics is a self-contained React component registered by contract type, enabling extension through registration without modifying orchestration layers. Both template families share interaction primitives built on dnd-kit (Bhatt, 2024) with custom collision detection and keyboard/touch support. State management follows a dual architecture: Diagram Games use a centralised Zustand store for multi-mechanic coordination; Algorithm Games use localised reducer hooks for step-through interactions. The engine supports WCAG-aligned keyboard navigation and screen reader announcements.

3.4 Pipeline Observability

The demonstration includes a real-time observability dashboard (Figure 1b) with **three view modes**: timeline, DAG graph (ReactFlow with execution-state highlighting), and cluster view grouped by phase. Per-agent **token and cost analytics** show stage-level consumption with USD breakdown. A **ReAct trace viewer** displays the Thought→Action→Observation chain for tool-calling agents, and a stage inspector exposes inputs, outputs, and tool call history at every phase.

3.5 Design Validation

Section 4 evaluates all three architectures on identical questions and models ($N = 200$, all 15 mechanics), showing that the DAG's phase-bounded design is the binding variable: 90.0% VPR, 73% token reduction, and $0.48/game.

4 Evaluation

Scope. This evaluation measures **architectural validity**: validation pass rate, token efficiency, and structural Bloom's alignment. It does not measure learning outcome gains; student-facing validation across diverse learner populations is identified as the primary direction for future work.

Setup. 200 questions from five domains (biology, history, CS, mathematics, linguistics) stratified across Bloom's levels and covering all 15 mechanics. All architectures used **GPT-4-turbo-2024-04-09** (OpenAI, 2023) (temp. 0.3, seed 42) for planning/validation and **gemini-1.5-pro-001** (Google DeepMind, 2023) (temp. 0.4) for asset generation, logged via LangSmith with per-call granularity. Full parameters are in Appendix C.

Baselines. Five categories: **manual authoring** by five educators via Genially/H5P (human-quality ceiling); **EdTech platforms** (Kahoot, Quizlet, Nearpod, H5P) and GameGPT (Chen et al., 2023); **Claude Code** under four prompting conditions (zero-shot, one-shot planning, one-shot instructional, multi-turn) across 30 stratified questions each; and **internal baselines** (Sequential Pipeline, ReAct Agent) on all 200 questions covering both template families. Full mechanic specifications are in Appendix A.

Human rating methodology. Educational Correctness and Playability ratings (Table 4) were collected from **five domain-expert raters** (3 educators with ≥5 years experience; 2 SMEs per domain) using a behaviourally anchored rubric. Raters were **blind to system condition**. Inter-rater reliability was acceptable (Educational Correctness: $\mathrm{ICC}_{(2,5)} = 0.81$, 95% CI [0.74, 0.87]; Playability: $\mathrm{ICC}_{(2,5)} = 0.78$, 95% CI [0.71, 0.84]). The comparison between GAMED.AI (4.2/5) and manual authoring (4.3/5) is not statistically significant ($t(198) = 1.04$, $p = 0.30$), indicating parity rather than superiority. Each rater evaluated all 200 games in batched sessions of 25–30 over a two-week period. The full rubric with anchors is in Appendix C.

Validation pass rate. GAMED.AI achieves a VPR of **90.0%**—17.5 percentage points above ReAct Agents (72.5%) and 33.3 points above the Sequential Pipeline (56.7%), confirmed significant ($\chi^2(2, N = 600) = 57.0$, $p < 0.001$, Cramér's $V = 0.31$).

Token consumption and cost. The 73% token reduction from ReAct Agents to the DAG (∼73,500 → ∼19,900 tokens/game) is structural: architecture explains 87% of token consumption variance ($\eta^2 = 0.87$, $F(2, 597) = 1{,}996$, $p < 0.001$). GAMED.AI is the only architecture meeting the sub-$0.50 cost requirement (Interactive Diagram Games average $0.48; Algorithm Games average $0.43 due to fewer vision model calls).

Per-mechanic performance. Figure 3 summarises results by architecture. Across all 15 mechanics, VPR ranges from 96.2% (`DRAG_DROP`) to 60.0% (`DESC_MATCHING`) for Interactive Diagram Games, and from 94.4% (`STATE_TRACER`) to 80.0% (`CONSTRAINT_PUZZLE`) for Interactive Algorithm Games; mean educational correctness is 4.2/5. Algorithm Games average higher token consumption (∼23,500 vs. ∼17,900 tokens/game) but lower per-game cost due to fewer vision model calls. Per-mechanic sample sizes range from 8 to 26; low-N mechanics ($N \leq 10$: `COMPARE`, `HIERARCHICAL`, `BRANCHING`, `DESC_MATCH`, `CONSTR_PUZZLE`) should be interpreted with caution. Full per-mechanic breakdowns are in Table 4 (Appendix B).

Failure analysis. Of the 20 DAG failures across 200 questions, 14 occur in Interactive Diagram mechanics and 6 in Algorithm Games. The dominant root cause is **schema underspecification**, not LLM hallucination: generated content is factually correct but lacks structural fields required by the FOL-based contract validator. For Diagram Games, `DESC_MATCHING` (4 failures) and `TRACE_PATH` (2) involve spatial anchoring; for Algorithm Games,

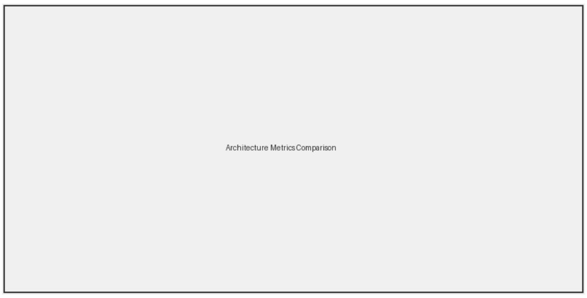

Figure 3: **Quality and efficiency metrics by architecture ($N = 200$ per condition, 15 mechanics).** Architecture explains 87% of token consumption variance ($\eta^2 = 0.87$); VPR gain of 17.5 pp over the next-best design.

System	VPR (%)	Bl. (%)	Tok. (K)	Cost	Time
GAMED.AI	**90.0**	**90**	**19.9**	**$0.48**	**<1 m**
ReAct Agent	72.5	73	73.5	$1.90	∼5 m
Sequential	56.7	57	49.4	$1.28	∼3 m
CC zero-shot	31	23	—	∼$0.30	∼5 m
CC 1-shot plan	—	41	—	∼$0.45	∼8 m
CC 1-shot inst.	—	48	—	∼$0.50	∼10 m
CC multi-turn	—	67	—	∼$0.80	∼15 m
GameGPT	—	—	—	∼$0.60	∼10 m

Table 1: Automated generation systems ($N = 200$ for GAMED.AI/ReAct/Sequential; $N = 30$/condition for Claude Code). CC = Claude Code (GPT-4-turbo, temp. 0.7, no contract schemas). Bl. = Bloom's alignment; Tok. = mean tokens/game; m = minutes. "—": not measured.

CONSTRAINT_PUZZLE (2) involves generated constraints forming unsatisfiable FOL sets. All failure types are tractable schema engineering problems; the full taxonomy covering all 15 mechanics is in Table 5 (Appendix B).

Baseline comparison. Tables 1 and 2 compare GAMED.AI against all baselines; two findings stand out.

GAMED.AI compresses 60–240 minutes of expert authoring into under 60 seconds while producing games rated at parity on pedagogical alignment (4.2 vs. 4.3/5, $p = 0.30$) across all five subject domains, at a fixed cost below the cheapest subscription tier of any listed platform. GameGPT (Chen et al., 2023) addresses creation speed but provides neither Bloom's targeting nor contract validation.

The Claude Code comparison isolates **structural constraints versus prompting**: Claude Code received identical learning objectives but not the mechanic contract schemas, mirroring the realistic scenario where an instructor uses a general-purpose coding tool without domain-specific validation. Under these conditions, Claude Code produced functional games in 100% of attempts—but only 23% passed Bloom's alignment and 31% passed contract validation at zero-shot; the multi-turn ceiling of 67% at $0.80/run remains 23 pp below GAMED.AI's 90% VPR at lower cost (Figure 4). This gap demonstrates that FOL-based validation, typed schemas, and phase-bounded generation provide guarantees that prompting alone cannot replicate (Ji et al., 2023; Mislevy et al., 2003). Providing mechanic schemas as structured prompt constraints is a direction for future comparison.

Platform	Time	Mech.	Edu.	Cost
GAMED.AI	**<1 min**	**15**	**4.2**	**$0.48/game**
Manual Auth.	60–240 min	Var	4.3	$50–150/game
Kahoot	15–30 min	1	—	Free–$7/mo
Quizlet	20–40 min	2	—	Free–$8/mo
Genially	60–120 min	5+	—	Free–$25/mo
H5P	40–90 min	50+	—	Free (OSS)

Table 2: Authoring platform comparison. Mech. = mechanic types; Edu. = educational correctness (1–5, 5 blinded experts); Var = variable. EdTech platforms lack Bloom's targeting and contract validation.

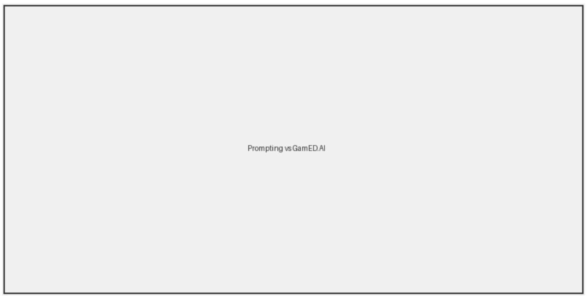

Figure 4: **Pedagogical alignment versus cost across Claude Code prompting conditions and GAMED.AI (DAG).** Under all prompting strategies tested, Claude Code reaches a ceiling of 67% Bloom's alignment at $0.80/game—23 pp below GAMED.AI's 90% at $0.48.

5 Conclusion and Future Work

We present GAMED.AI, a hierarchical multi-agent framework for automated educational game generation grounded in Bloom's Taxonomy and enforced through formal mechanic contracts. By separating planning, generation, and validation into phase-specific LangGraph sub-graphs with deterministic Quality Gates, the system achieves a 90% validation pass rate and 73% token reduction over

ReAct baselines—compressing 60–240 minutes of expert authoring into under 60 seconds at $0.48 per game. These results demonstrate that pedagogical alignment and efficiency emerge as properties of architectural structure rather than model scale or prompting strategy alone: no baseline replicated these outcomes.

Future work targets four directions: **human-in-the-loop blueprint negotiation** via LangGraph streaming; **frame-based game engines** (Phaser.js, PixiJS) enabling physics simulations and sprite-based mechanics beyond DOM-based templates; **expanded template families** including PhET-inspired simulations, narrative-driven, and role-playing templates; and **large-scale classroom evaluation** measuring learning outcome gains, the primary missing validation.

Limitations

Spatial mechanic schema coverage. The blueprint schema does not fully constrain spatial anchoring for `DESC_MATCHING` and `CLICK_TO_TRACE`, producing the majority of the 20 validation failures. We are extending the schema with relational-link fields and coordinate normalisation at QG2.

Model and language scope. Reported metrics reflect a GPT-4-turbo/Gemini configuration; while open-source substitution (Llama 3, Mistral) is supported, on-premises performance has not been benchmarked. The system currently generates English-only games; multilingual contracts and non-text input modalities are planned.

Student-facing validation. The evaluation measures architectural validity and expert-rated alignment, not learning outcomes. Human ratings from five experts may diverge from learner-population judgements, particularly at lower Bloom's levels. Controlled classroom studies remain the primary future direction.

Broader Impact

GAMED.AI is designed to democratise the creation of pedagogically grounded educational games, reducing the expertise and cost barriers that currently restrict high-quality game-based assessment to well-resourced institutions. By automating Bloom's-aligned game generation at under $0.50 per game, the system has potential to expand access to interactive learning in under-served educational contexts where custom content development is infeasible.

Positive impacts. The framework enforces pedagogical validity through structural constraints rather than relying on instructor expertise in assessment design, potentially raising the floor for educational content quality. The open-source release of code, 50 games, and evaluation datasets supports reproducibility and community-driven extension. The model-agnostic architecture enables on-premises deployment, addressing data sovereignty concerns in educational settings.

Risks and mitigations. Automated game generation inherits the biases and factual limitations of underlying LLMs. While Quality Gates validate structural properties (Bloom's alignment, mechanic contracts, schema compliance), they do not verify factual accuracy of generated content—incorrect domain knowledge could propagate into games. We mitigate this through: (1) domain knowledge retrieval from curated sources (textbooks, curricula standards, domain ontologies) rather than open-ended generation, (2) deterministic validators that flag structural anomalies, and (3) the observability dashboard enabling instructor review of all generated content before deployment. Over-reliance on automated assessment without human oversight remains a concern; we explicitly design GAMED.AI as an instructor tool, not a replacement for pedagogical judgement.

References

Lorin W. Anderson and David R. Krathwohl. 2001. *A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives*. Longman.

Sylvester Arnab, Theodore Lim, Maira B. Carvalho, Francesco Bellotti, and 1 others. 2015. Mapping learning and game mechanics for serious games analysis. *British Journal of Educational Technology*, 46(2):391–411.

Claudéric Bhatt. 2024. dnd kit: A lightweight, modular, performant, accessible and extensible drag & drop toolkit for React. https://dndkit.com/. Open-source library.

Paul Black and Dylan Wiliam. 1998. Assessment and classroom learning. *Assessment in Education: Principles, Policy & Practice*, 5(1):7–74.

Benjamin S. Bloom. 1956. *Taxonomy of Educational Objectives: The Classification of Educational Goals*. David McKay Company.

Chapman Alliance. 2010. How long does it take to create learning? Technical report, Chapman Alliance. Industry research study.

Dake Chen and 1 others. 2023. GameGPT: Multi-agent collaborative framework for game development. *arXiv preprint arXiv:2310.08067*.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, and 1 others. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Sebastian Deterding, Dan Dixon, Rilla Khaled, and Lennart Nacke. 2011. From game design elements to gamefulness: Defining "gamification". In *15th International Academic MindTrek Conference*, pages 9–15. ACM.

Barbara Ericson and 1 others. 2022. Parsons problems and beyond. *ACM Computing Surveys*.

Scott Freeman, Sarah L. Eddy, Miles McDonough, Michelle K. Smith, and 1 others. 2014. Active learning increases student performance in science, engineering, and mathematics. *Proceedings of the National Academy of Sciences*, 111(23):8410–8415.

Google DeepMind. 2023. Gemini: A family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Arthur C. Graesser, Shulan Lu, G. Tanner Jackson, Heather H. Mitchell, and 1 others. 2004. AutoTutor: A tutor with dialogue in natural language. *Behavior Research Methods, Instruments, & Computers*, 36(2):180–192.

Juho Hamari, Jonna Koivisto, and Harri Sarsa. 2014. Does gamification work? — a literature review of empirical studies on gamification. In *47th Hawaii International Conference on System Sciences (HICSS)*, pages 3025–3034. IEEE.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Xiong, and 1 others. 2023. MetaGPT: Meta programming for a multi-agent collaborative framework. *arXiv preprint arXiv:2308.00352*. ICLR 2024 Oral.

Christopher D. Hundhausen, Sarah A. Douglas, and John T. Stasko. 2002. A meta-study of algorithm visualization effectiveness. *Journal of Visual Languages and Computing*, 13(3):259–290.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, and 1 others. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12).

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. 2024. SWE-bench: Can language models resolve real-world GitHub issues? In *ICLR*.

Kenneth R. Koedinger, Vincent Aleven, Neil Heffernan, Bruce McLaren, and Matthew Hockenberry. 2006. Opening the door to non-programmers: Authoring intelligent tutor behavior by demonstration. In *Intelligent Tutoring Systems*.

Michael Kuznia, Yanzhe Cao, and Shuai Lu. 2024. On the brittle foundations of ReAct prompting for agentic large language models. *arXiv preprint arXiv:2405.13966*.

Richard N. Landers. 2014. Developing a theory of gamified learning: Linking serious games and gamification of learning. *Simulation & Gaming*, 45(6):752–768.

Michael Lee and 1 others. 2014. Principles of a debugging-first puzzle game for computing education. In *IEEE Symposium on Visual Languages and Human-Centric Computing*.

Richard E. Mayer. 2009. *Multimedia Learning*, 2nd edition. Cambridge University Press.

Robert J. Mislevy, Russell G. Almond, and Janice F. Lukas. 2003. A brief introduction to evidence-centered design. Technical report, ETS.

Thomas L. Naps and 1 others. 2002. Exploring the role of visualization and engagement in computer science education. In *ITiCSE Working Group Reports*.

Anne Ngu and 1 others. 2025. A generative AI educational game framework with multi-scaffolding. *Computers & Education*, 239.

OpenAI. 2023. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*.

Allan Paivio. 1991. Dual coding theory: Retrospect and current status. *Canadian Journal of Psychology*, 45(3):255–287.

Dale Parsons and Patricia Haden. 2006. Parson's programming puzzles: A fun and effective learning tool. In *Australasian Computing Education Conference*.

Tal Ridnik, Dedy Kredo, and Itamar Friedman. 2024. AlphaCodium: From prompt engineering to flow engineering. *arXiv preprint arXiv:2401.08500*.

Michael Sailer and Lisa Homner. 2020. The gamification of learning: A meta-analysis. *Educational Psychology Review*, 32:77–112.

Valerie J. Shute. 2008. Focus on formative feedback. *Review of Educational Research*, 78(1):153–189.

Valerie J. Shute and Matthew Ventura. 2013. *Stealth Assessment: Measuring and Supporting Learning in Video Games*. MIT Press.

Robert A. Sottilare, Benjamin S. Goldberg, Keith W. Brawner, and Heather K. Holden. 2012. The generalized intelligent framework for tutoring (GIFT). Technical report, US Army Research Laboratory.

John Sweller. 1988. Cognitive load during problem solving: Effects on learning. *Cognitive Science*, 12(2):257–285.

Alf Inge Wang and Rabail Tahir. 2020. The effect of using Kahoot! for learning — a literature review. *Computers & Education*, 149:103818.

Brandon T. Willard and Rémi Louf. 2023. Efficient guided generation for large language models. *arXiv preprint arXiv:2307.09702*.

Pieter Wouters, Christof van Nimwegen, Herre van Oostendorp, and Erik D. van der Spek. 2013. A meta-analysis of the cognitive and motivational effects of serious games. *Journal of Educational Psychology*, 105(2):249–265.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, and 1 others. 2023. AutoGen: Enabling next-gen LLM applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing reasoning and acting in language models. In *ICLR*.

Jiaying Zeng, Dan Sun, Chee Kit Looi, and Xin Fan. 2024. Exploring the impact of gamification on students' academic performance. *British Journal of Educational Technology*, 55(6):2478–2502.

A Bloom's Mapping and Mechanic Contracts

Table 3 maps Bloom's levels to mechanics, adopted as a design heuristic from Anderson and Krathwohl (2001) and the LM-GM framework (Arnab et al., 2015); empirical validation of mechanic-to-cognitive-level correspondences remains open. Each mechanic defines a typed Pydantic contract (interaction primitive, minimum item count, scoring model, completion condition) validated by QG1; full schemas are in the repository.

B Per-Mechanic Results and Failure Analysis

Table 4 disaggregates VPR across all 15 mechanics. Fully constrained schemas achieve $\geq$90% VPR; the lowest-performing mechanics share root causes in spatial anchoring (Diagram) or constraint satisfiability (Algorithm). Table 5 classifies all 20 failures.

C Evaluation Protocol

Models. GPT-4-turbo-2024-04-09 (temp. 0.3, seed 42) for planning/validation; gemini-1.5-pro-001 (temp. 0.4) for assets. **Domains.** 200 questions, 5 domains $\times$ 40 each, balanced across Bloom's levels and covering all 15 mechanics. **Claude Code.** Four conditions: zero-shot, one-shot planning, one-shot instructional, multi-turn (30 questions each). **Rating.** Five blinded experts; Educational Correctness (1–5 anchored: 1 = factually wrong or Bloom's mismatch; 2 = partially correct, major gaps; 3 = correct but incomplete alignment; 4 = correct, well-aligned; 5 = exemplary alignment with clear competency evidence) and Playability (% of mechanic contract conditions met). **Statistics.** VPR: χ^2/Cramér's V; tokens: ANOVA/η^2; ratings: t-tests, Bonferroni; $\alpha = 0.05$.

D Architectural Evolution

V1 Sequential (56.7% VPR, ~49.4K tok.): dominant failure was cascading schema violations in later stages, where early errors propagated unchecked through 8+ serial agents. No validation gates existed between stages.

V2 ReAct (72.5%, ~73.5K tok.): self-correction via Thought→Action→Observation loops improved VPR but caused 3.7$\times$ token inflation over Sequential; performance was driven by exemplar-query similarity rather than reasoning depth (Kuznia et al., 2024).

V3 DAG (90.0%, ~19.9K tok.; $\eta^2 = 0.87$): phase-bounded validation eliminated cascading errors; remaining failures are attributable to schema underspecification (Section 4). Full analysis in the repository.

Bloom's	Fam.	Mechanics	Cognitive Operation
Remember	ID	Click-to-Id., Memory Match	Recognise/recall via visual id. (Anderson and Krathwohl, 2001)
Understand	ID	Drag-Drop, Desc. Match	Interpret by label-structure mapping (Mayer, 2009)
Apply	Both	Trace Path, Seq., State Tr.[A]	Execute procedures by tracing/ordering (Parsons and Haden, 2006)
Analyze	Both	Sorting, Hier., Bug H.[A], Cmplx.[A]	Differentiate by categorising/error id. (Sweller, 1988)
Evaluate	ID	Compare, Branching	Critique by comparing alternatives (Mislevy et al., 2003)
Create	Algo	Algo. Builder[A], Constr. Pzl.[A]	Generate solutions from primitives (Ericson et al., 2022)

Table 3: Bloom's-to-mechanic mapping. Fam.: ID = Interactive Diagram, Algo = Algorithm, Both = shared; [A] = Algorithm Game mechanic.

Mechanic	N	VPR (%)	Tok. (K)	Lat. (s)	Edu. (1–5)	Play (%)
Interactive Diagram Games						
DRAG_DROP	26	96.2	18.2	27.0	4.4	96.2
SEQUENCING	16	93.8	17.0	25.0	4.5	93.8
CLICK_TO_ID	14	92.9	16.8	24.0	4.3	92.9
SORTING	12	91.7	18.5	28.0	4.2	91.7
MEMORY_MATCH	12	91.7	16.2	23.0	4.3	91.7
BRANCHING	10	90.0	19.5	30.0	4.1	90.0
COMPARE	8	87.5	20.1	31.0	4.0	87.5
HIERARCHICAL	8	87.5	22.4	35.0	3.9	87.5
TRACE_PATH	14	85.7	17.5	26.0	4.1	85.7
DESC_MATCH	10	60.0	15.8	22.0	3.8	75.0
Interactive Algorithm Games						
STATE_TRACER	18	94.4	21.3	32.0	4.4	94.4
BUG_HUNTER	16	93.8	23.8	36.0	4.2	87.5
ALGO_BUILDER	14	92.9	25.2	38.0	4.3	92.9
COMPLEXITY	12	91.7	22.7	34.0	4.1	83.3
CONSTR_PUZZLE	10	80.0	26.5	40.0	3.9	80.0
Overall	**200**	**90.0**	**19.9**	**29.4**	**4.2**	**89.5**

Table 4: Per-mechanic metrics ($N = 200$, 15 mechanics). Lat. = end-to-end generation latency; Edu./Play: mean human ratings from 5 blinded raters (ICC > 0.78). Algorithm Games average higher token consumption but lower per-game cost (no vision model calls).

Mechanic	N	Gate	Error / Root Cause
Interactive Diagram Games (14 failures)			
DESC_MATCH	4	QG3	BLOOM_OP_COUNT_FAIL; pairs lack relational links
TRACE_PATH	2	QG3	ANCHOR_OOB; SVG coords outside bounding box
COMPARE	1	QG2	ASSET_SCHEMA_MISMATCH; axis label missing
HIERARCHICAL	1	QG3	DEPTH_MISMATCH; tree depth $<$ contract min
5 other ID mech.	6	QG2/3	Region overlap (2); state/schema violations (4)
Interactive Algorithm Games (6 failures)			
CONSTR_PZL	2	QG3	CONSTRAINT_UNSAT; FOL rules form unsat. set
COMPLEXITY	1	QG3	CLASS_MISMATCH; generated $\neq$ target class
3 other Algo mech.	3	QG2/3	Placement, ordering, state transition errors

Table 5: Failure taxonomy (20 failures, $N = 200$, 15 mechanics). All attributable to schema underspecification, not LLM hallucination. FOL-based validators catch structural violations deterministically.