TABLE I
LAYER 1: STUDENT NEEDS PRINCIPLES FOR STRUCTURING LLM-GENERATED FEEDBACK IN NOVICE PROGRAMMING

| ID | Prompt Design Principle | Pedagogical Basis |
|---|---|---|
| N1 | Guide the prompt to generate constructive feedback focused on improvement rather than merely identifying errors. | [1], [2] |
| N2 | Include instructions in the prompt that ensure feedback explains what the mistake is, why it occurred, and how to fix it. | [1], [3], [4] |
| N3 | Direct the prompt to provide quick fix suggestions for minor, unintentional mistakes such as missing symbols or formatting errors. | [5], [6] |
| N4 | Instruct the prompt to provide example-based feedback when student errors indicate a misunderstanding of syntax or structure. | [4], [5], [7] |
| N5 | Design the prompt to generate feedback using clear and direct language that novice learners can easily understand and apply. | [1], [4], [7], [8] |
| N6 | Instruct the prompt to provide hints instead of solutions, supporting student independence and critical thinking. | [3], [4], [9] |
| N7 | Guide the prompt to include motivational feedback that acknowledges student effort and the achievement of subgoals. | [3], [10] |
| N8 | Structure the prompt to deliver feedback progressively, guiding students step-by-step through the correction process. | [3], [4], [7] |
| N9 | Design the prompt to generate feedback that includes metacognitive questions prompting students to reflect on their work. | [3], [11] |
| N10 | Instruct the prompt to generate feedback that adjusts to the student's progress, acknowledges effort, and supports continued improvement. | [3], [4] |
| N11 | Design the prompt to generate feedback segmented into distinct, specific parts to enhance clarity and support student comprehension. | [1], [4], [7], [8] |
| N12 | Instruct the prompt to address common novice-level errors related to syntax, semantics, and conceptual misunderstandings. | [1], [12] |

The prompt design principles presented in Layer 1, as shown in Table I, contribute to the development of three key components within the PPE-LLM framework: *Provide Student Background* **(Comp 3)**, *Structure Feedback Using Pedagogical Frameworks* **(Comp 6)**, and *Implement Additional Feedback Guidelines* **(Comp 7)**. They guide prompt engineering to ensure that the generated feedback is context-aware, pedagogically structured, and aligned with essential feedback qualities such as clarity, and tone.

TABLE II
LAYER 2: PEDAGOGICAL PRINCIPLES FOR STRUCTURING LLM-GENERATED FEEDBACK IN NOVICE PROGRAMMING

| ID | Pedagogical Principle | Pedagogical Basis |
|---|---|---|
| P1 | Guide the prompt to include step-by-step worked examples that support conceptual understanding before students attempt independent problem-solving (e.g., pseudocode). | [8] |
| P3 | Structure prompts to follow a Guidance-to-Independence approach: identify the issue, offer hints, and promote self-correction. | [4], [9] |
| P4 | Prompt the model to encourage students to self-verify, reflect, and optimize their work to enhance self-regulation and understanding. | [11], [13] |
| P5 | Ensure the prompt generates feedback that addresses task-level correctness, process-level strategies, and self-regulation for independent application. | [2] |
| P6 | Guide the prompt to organize feedback into setting learning goals (Feed Up), evaluating current progress (Feed Back), and guiding the next steps for improvement (Feed Forward). | [2] |
| P7 | Avoid prompts that generate personal evaluation; ensure feedback focuses on task improvement. | [4], [14] |
| P8 | Instruct the model to extend feedback beyond simple verification by explaining what the issue is, how it occurs, and why it matters. | [4] |
| P9 | Guide the prompt to segment feedback into clear, manageable parts to support student comprehension. | [4], [7], [8] |
| P10 | Ensure the prompt elicits feedback that is specific, clear, and actionable. | [4] |
| P11 | Avoid complex or abstract language in prompts; tailor explanations to match the learner's level. | [4], [7], [8] |
| P12 | Design prompts that ensure objectivity and focus on performance outcomes, not personal traits. | [4], [10] |
| P13 | Shift the focus in prompts from performance goals to learning goals to encourage growth-oriented thinking. | [4], [11] |
| P14 | Encourage the model to highlight student strengths and suggest improvements without assigning grades. | [4], [14] |
| P15 | Set the prompt to use a supportive and encouraging tone to maintain motivation and build confidence. | [4], [10] |
| P16 | Guide the prompt to promote independent problem-solving through scaffolding rather than direct answers. | [4], [9] |
| P17 | Limit feedback scope in the prompt by focusing on key errors and avoiding unnecessary detail. | [4], [7], [8] |
| P18 | Tailor prompts to provide corrective guidance and foundational support for low-achieving learners. | [4] |
| P19 | Adjust the prompt to challenge high-achieving learners through deeper questions and advanced hints rather than giving solutions directly. | [4], [10] |

The prompt design principles presented in Layer 2, as shown in Table II, contribute to the development of four key components within the PPE-LLM framework: *Defining the Goal of Feedback* **(Comp 2)**, *Define Evaluation Criteria* **(Comp 5)**, *Structure Feedback Using Pedagogical Frameworks* **(Comp 6)**, and *Implement Additional Feedback Guidelines* **(Comp 7)**. They guide prompt engineering to ensure that the generated feedback is purpose-driven, tailored to the learner's level, structured according to pedagogical best practices, and delivered with supportive tone and clarity.

TABLE III
BEST PRACTICES FOR DESIGNING LLM-GENERATED FEEDBACK

| ID | Best Practices Principle | References |
|---|---|---|
| B1 | Customize feedback according to a student's achievement level. | Nguyen & Allan (2024); Nguyen et al. (2024) |
| B2 | Structure feedback in tiers to scaffold learning effectively. | Nguyen & Allan (2024) |
| B3 | Explicitly instruct LLMs to avoid direct answers, using keywords like "hint." | Roest et al. (2024); Nguyen et al. (2024) |
| B4 | Avoid prompting for compliments, as it leads to overly lengthy and irrelevant responses. | Roest et al. (2024) |
| B5 | Use keywords like "hint" and "student" for a friendly, personalized tone and clearer explanations. | |
| B6 | Avoid over-constrained prompts, such as word or sentence limits, which reduce flexibility and feedback quality. | |
| B7 | Avoid providing model solutions, as they restrict flexibility and limit alternative approaches. | |
| B8 | Identify students' prior knowledge levels for LLMs. | Hellas et al. (2023) |
| B9 | Avoid overly general or vague praise (e.g., "You're on the right track") as it may seem insincere or irrelevant. | |
| B10 | Make sure comments contain clear instructions that avoid providing direct answers, complete solutions, or snippets of code. | |
| B11 | Differentiate feedback on progress (achievements) from feedback suggestions (areas for improvement and next steps). | |

TABLE IV
TECHNICAL PRINCIPLES FOR STRUCTURING LLM-GENERATED FEEDBACK

| ID | Technical Principle | References |
|---|---|---|
| T1 | Clearly state your objective to ensure the response matches your needs. | OpenAI (2024) |
| T2 | Specify the target audience and desired style or tone. | OpenAI (2024); Gemini (2024); Google Cloud (2024) |
| T3 | Provide context and reference specific sources. | OpenAI (2024); AWS (2024) |
| T4 | Add external sources dynamically to provide relevant information. | OpenAI (2024); Gemini (2024) |
| T5 | Break complex tasks into simpler subtasks for better accuracy. | |
| T6 | Define the output format (e.g., table, list, or paragraph) for clear structure. | OpenAI (2024) |
| T7 | Begin the prompt with clear instructions and use separators such as ‘‘ or ‘‘‘‘‘‘ to distinguish instructions from the context. | |
| T9 | Use precise language and avoid ambiguity. | OpenAI (2024); Gemini (2024) |
| T10 | Avoid vague or overly detailed descriptions. | |
| T11 | Verify responses with self-checks to reduce errors and improve accuracy. | Harrington et al. (2024) |
| T12 | Refine and adjust prompts continuously to enhance relevance and clarity. | |

## REFERENCES


[1] A. Luxton-Reilly, B. A. Becker, Y. Cao, R. McDermott, C. Mirolo, A. Mühling, A. Petersen, K. Sanders, Simon, and J. Whalley, "Developing assessments to determine mastery of programming fundamentals," in *Proceedings of the 2017 ITiCSE Conference on Working Group Reports*, ser. ITiCSE-WGR '17. New York, NY, USA: Association for Computing Machinery, 2018, p. 47–69. [Online]. Available: https://doi.org/10.1145/3174781.3174784

[2] J. Hattie and H. Timperley, "The power of feedback," *Review of Educational Research*, vol. 77, no. 1, pp. 81–112, 2007.

[3] J. Jeuring, H. Keuning, S. Marwan, D. Bouvier, C. Izu, N. Kiesler, T. Lehtinen, D. Lohr, A. Peterson, and S. Sarsa, "Towards giving timely formative feedback and hints to novice programmers," in *Proceedings of the 2022 Working Group Reports on Innovation and Technology in Computer Science Education*, ser. ITiCSE-WGR '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 95–115. [Online]. Available: https://doi.org/10.1145/3571785.3574124

[4] V. J. Shute, "Focus on formative feedback," *Review of educational research*, vol. 78, no. 1, pp. 153–189, 2008.

[5] U. Z. Ahmed, N. Srivastava, R. Sindhgatta, and A. Karkare, "Characterizing the pedagogical benefits of adaptive feedback for compilation errors by novice programmers," in *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering: Software Engineering Education and Training*, ser. ICSE-SEET '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 139–150. [Online]. Available: https://doi.org/10.1145/3377814.3381703

[6] S. Narciss, "Feedback strategies for interactive learning tasks," in *Handbook of research on educational communications and technology*. Routledge, 2008, pp. 125–143.

[7] R. E. Mayer, "Applying the science of learning: evidence-based principles for the design of multimedia instruction." *American psychologist*, vol. 63, no. 8, p. 760, 2008.

[8] J. Sweller, "Cognitive load during problem solving: Effects on learning," *Cognitive science*, vol. 12, no. 2, pp. 257–285, 1988.

[9] D. Wood, J. S. Bruner, and G. Ross, "The role of tutoring in problem solving," *Journal of child psychology and psychiatry*, vol. 17, no. 2, pp. 89–100, 1976.

[10] J. Hattie, *Visible learning for teachers: Maximizing impact on learning*. Routledge, 2012.

[11] D. J. Nicol and D. Macfarlane-Dick, "Formative assessment and self-regulated learning: A model and seven principles of good feedback practice," *Studies in higher education*, vol. 31, no. 2, pp. 199–218, 2006.
[12] A. Robins, J. Rountree, and N. Rountree, "Learning and teaching programming: A review and discussion," *Computer science education*, vol. 13, no. 2, pp. 137–172, 2003.
[13] D. H. Schunk and B. J. Zimmerman, *Self-regulated learning: From teaching to self-reflective practice*. Guilford Press, 1998.
[14] P. Black and D. Wiliam, "Assessment and classroom learning," *Assessment in Education: principles, policy & practice*, vol. 5, no. 1, pp. 7–74, 1998.