Volume 2, Issue 4

Innovation Series Quarterly

https://journal.academicapress.org/aps/index.php/isq

Research Article

No-Code Solution Development Framework (NSDF): An Instructional Framework for Generative AI Application Development

Wahidah Mohd Zain[1,*], Faddliza Mohd Zaki[2], Razifah Othman[3], Jashira Jamin[4], Haziah Sa'ari[5], and Amirah Haji Abu Hassan[6]

[1] Faculty of Information Science, Universiti Teknologi MARA (UiTM), Negeri Sembilan Branch, Rembau Campus; wahidahmohdzain@uitm.edu.my

[2] Faculty of Information Science, Universiti Teknologi MARA (UiTM), Negeri Sembilan Branch, Rembau Campus; faddliza@uitm.edu.my

[3] Faculty of Information Science, Universiti Teknologi MARA (UiTM), Negeri Sembilan Branch, Rembau Campus; razif879@uitm.edu.my

[4] Faculty of Information Science, Universiti Teknologi MARA (UiTM), Negeri Sembilan Branch, Rembau Campus; jashira@uitm.edu.my

[5] Faculty of Information Science, Universiti Teknologi MARA (UiTM), Negeri Sembilan Branch, Rembau Campus; haziah095@uitm.edu.my

[6] Faculty of Information Science, Universiti Teknologi MARA (UiTM), Negeri Sembilan Branch, Rembau Campus; amirah893@uitm.edu.my

* Correspondence: wahidahmohdzain@uitm.edu.my.

Keywords:
No-Code Development
Generative AI
AWS PartyRock
Teaching and Learning Framework
AI Literacy

Abstract: *Learners without programming experience can build generative AI applications with no-code tools, but many struggle to set output criteria and document their testing. The No-Code Solution Development Framework (NSDF) guides them through seven steps in three phases: Build, Sustain and Assure. Three experts rated every step relevant (I-CVI = 1.00). After an AWS PartyRock activity, 29 students rated ease of use (78.7%) and usefulness (78.2%) positively, and 89.7% would recommend the platform. Over 500 students from the faculty entered a national AI competition, and one won a category award. NSDF gives non-programmers a repeatable, assessable route to AI solutions.*

Copyright: © 2026 by the authors. Submitted for open access publication under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. INTRODUCTION

No-code platforms remove the programming requirement from early AI application development. AWS PartyRock lets students connect prompts, assemble an interface and generate responses without writing code (Amazon Web Services [AWS], 2023). A working interface, however, says little about the decisions behind it: whether the problem is adequately specified, whether outputs

meet stated criteria, whether tests are systematic and whether revisions are justified. These decisions connect application development with the creation and critical evaluation competencies of AI literacy (Long & Magerko, 2020; Ng et al., 2021).

1.1 Problem statement

AI use in higher education is now widespread, but confidence has not kept pace with adoption. A UNESCO survey of 400 UNESCO Chairs and UNITWIN Network members across 90 countries found that nine in ten respondents used AI tools in their professional work, while confidence in applying them remained uneven (UNESCO, 2025). If educators report uneven confidence, learners without technical backgrounds are even less likely to use AI tools well without explicit pedagogical support.

At Universiti Teknologi MARA (UiTM) Rembau Campus, Negeri Sembilan, Malaysia, lecturers observed that students without programming experience proposed practical applications for everyday problems but struggled to specify output criteria and to keep evidence of testing and revision. These observations were informal and were not recorded as baseline data. What these students needed was a repeatable procedure linking a defined problem to response testing, user feedback and documented revision decisions. NSDF was developed at the Faculty of Information Science, UiTM Rembau Campus, to meet this need.

1.2 Objectives

The main objective of this innovation is to develop, validate and implement a structured no-code teaching and learning framework that guides learners without programming experience in turning everyday ideas into working AI-powered solutions. Specifically, this innovation aims to:

1. Design the No-Code Solution Development Framework (NSDF), comprising three phases (Build, Sustain and Assure) and seven steps (Problem, Design, Build, Test, Deploy, Feedback and Validate), implemented using AWS PartyRock.

2. Establish the content validity of NSDF through expert evaluation of the relevance and clarity of each step and of the framework as a whole.

3. Implement NSDF in a classroom setting and describe learners' perceptions of the AWS PartyRock learning experience in terms of perceived usefulness, ease of use, reported use, perceived impact and willingness to recommend.

1.3 Novelty

NSDF adapts established development activities for learners who build through natural-language prompts instead of code. Its distinctive design decision is to place post-deployment Feedback (Sustain) and process-level Validate (Assure) in separate phases, each answering a different assessment question. Feedback asks how users experienced the application and what the learner revised in response. Validate asks whether the original problem, output criteria, tests and revisions form a traceable process. Every step therefore specifies an action that an instructor can request and review. Although NSDF was implemented with AWS PartyRock, its steps are intended to transfer to other no-code platforms and disciplines.

2. LITERATURE REVIEW

2.1 No-code learning and development processes

Research on low-code and no-code platforms describes the features that make rapid application development possible without programming (Sahay et al., 2020). In teaching, Sundberg and Holmström (2024) organised no-code machine-learning instruction around five principles, including problem-centred learning, application and integration. Matook et al. (2025) linked experiential learning factors in a low-code project with students' metacognitive reflection. These studies address how activities are structured and how students reflect on them; NSDF adds task-specific checkpoints for assessing the student's development record.

General-purpose process models were written for other audiences. Development life-cycle models organise specification, implementation and testing for software developers (Ruparelia, 2010), and design thinking emphasises problem framing and iteration for designers (Brown, 2008). NSDF draws on both but sequences the activities for prompt-based classroom tasks completed by non-programmers.

2.2 Student perceptions and AI literacy

The Technology Acceptance Model links perceptions of usefulness and ease of use to technology acceptance (Davis, 1989), and it informed the student questionnaire used here. AI literacy research treats creating with AI and critically evaluating its outputs as core competencies (Long & Magerko, 2020; Ng et al., 2021). Experiential learning theory adds that reflection on one activity should inform the next (Kolb, 1984), which is the principle behind NSDF's Feedback step. Acceptance measures describe attitudes and reported engagement, so this study keeps them separate from the documented development decisions that NSDF asks learners to produce.

3. METHODOLOGY

The study combined framework development, expert content appraisal, classroom implementation and a post-activity descriptive questionnaire at the Faculty of Information Science, UiTM Rembau Campus.

3.1 Framework development and expert review

NSDF contains seven steps organised into three phases: Build (Problem, Design, Build, Test and Deploy), Sustain (Feedback) and Assure (Validate). Table 1 sets out the learner evidence each step is intended to produce. The Build step shares its name with the Build phase; as a step, it refers specifically to constructing the prototype.

Table 1. NSDF phases, steps and intended learner evidence.

Phase	Step	Intended learner evidence
Build	Problem	Written problem, intended beneficiary and purpose
Build	Design	Expected inputs, outputs and response criteria
Build	Build	Working prompt-based application prototype
Build	Test	Test cases, observed responses and prompt changes
Build	Deploy	Shareable version available for user interaction
Sustain	Feedback	User comments and a documented revision decision
Assure	Validate	Process audit against the problem and step record

Three senior lecturers at UiTM Rembau Campus, specialising in Data Science, Information Systems and Library Science, rated each step's relevance and clarity on four-point scales. The item-level content validity index (I-CVI) was calculated as the number of relevance ratings of 3 or 4 divided by the three raters (Lynn, 1986; Polit & Beck, 2006). The same panel rated 12 framework-level statements on structure, pedagogical suitability, feasibility and adaptability. Rating counts, percentages and mean scores were summarised.

3.2 Classroom implementation

NSDF was implemented with diploma and degree students of the Faculty of Information Science, UiTM Rembau Campus, during Semester 20254. Students received step-by-step guidance while developing applications in AWS PartyRock. In the Build phase, learners identified an everyday problem and described the proposed solution's inputs, outputs and intended behaviour, then constructed prototypes through natural-language prompts, tested the outputs and refined their prompts before sharing the applications. In the Sustain phase, they gathered user comments and recorded a revision decision. In the Assure phase, the Validate step reviewed whether each learner had followed the process and addressed the intended problem. This process review is separate from the expert appraisal of the framework itself.

3.3 Initial evaluation and implementation reach

After the classroom activities, 29 students completed a questionnaire adapted from Davis (1989). Its 23 items used a five-point agreement scale across four constructs: perceived usefulness (PU, six items), perceived ease of use (PEOU, six), use of AWS PartyRock (USE, six) and perceived impact (PI, five). Five open-ended questions followed. One asked whether students would recommend AWS PartyRock; the other open-ended responses are not reported here.

The analysis covered mean item scores, the share of Agree or Strongly Agree responses across the items in each construct, Cronbach's alpha (Cronbach, 1951) and exploratory Spearman correlations between construct mean scores. Each percentage is therefore a share of item responses, not the percentage of students above a threshold. Implementation reach was recorded as faculty participation in the AWS CendekiAwan 2026 National AI Competition and the awards that followed.

4. FINDINGS

4.1 Expert ratings

All 21 relevance ratings across the seven steps were 3 or 4, giving an I-CVI of 1.00 for every step. Mean relevance scores ranged from 3.33 to 4.00 and mean clarity scores from 3.67 to 4.00. Eighteen of the 21 clarity ratings (85.7%) were 'very clear' and three (14.3%) were 'clear'. One expert suggested clearer wording for the Problem step. At framework level, 34 of 36 ratings (94.4%) were 'strongly agree' and two were 'agree', and all three experts rated the framework's overall status as 'High'.

4.2 Student ratings

Table 2 reports the four constructs. PEOU received the highest mean rating and the largest share of positive responses (78.7%), closely followed by PU (78.2%). USE was lowest at 65.5%, and one classroom-use item received 55.2% positive responses. Internal consistency was high, with alpha coefficients from .906 to .963.

Table 2. Student questionnaire results (n = 29).

Construct	Items	Mean / 5	Positive (%)	Alpha
Perceived usefulness (PU)	6	3.96	78.2	.953
Perceived ease of use (PEOU)	6	4.03	78.7	.963
Use of AWS PartyRock (USE)	6	3.70	65.5	.955
Perceived impact (PI)	5	3.84	71.7	.906

Note. Positive (%) = Agree + Strongly Agree responses across all items in the construct. Alpha = Cronbach's internal-consistency coefficient.

Twenty-six students (89.7%) said they would recommend AWS PartyRock to other students. In the exploratory correlations, the six pairwise coefficients ranged from .541 to .794, and USE and PI were positively associated (Spearman $\rho = .755$, uncorrected $p < .001$).

4.3 Implementation reach

Beyond the classroom, more than 500 students from the faculty took part in the AWS CendekiAwan 2026 National AI Competition. A Rembau Campus student won the top award in the GovTech & Social Impact category with the CareerAdvisor application (Elitery, 2026).

5. DISCUSSION

5.1 Ease of use and reported use

Students found AWS PartyRock easy to use, but their reported use was lower, a gap of 13.2 percentage points at the item-response level. The low classroom-use item suggests that an easy platform is not enough on its own to make use routine. Structured tasks of the kind NSDF specifies may help. A follow-up study could test this by pairing individual survey scores with task logs of the tests and revisions each student actually carried out.

5.2 Why Feedback and Validate are separate steps

NSDF places two evidence obligations after deployment. Feedback records the user's experience and the learner's revision decision. Validate audits whether the original problem, output criteria, tests and revisions form a traceable development process. The two steps answer different questions, since a favourable user response cannot show that the generated outputs were systematically tested. A reviewable record would contain a problem statement, response criteria, test cases, revision notes and a process audit. Records of this kind turn the creation and critical evaluation competencies of AI literacy (Long & Magerko, 2020; Ng et al., 2021) into student work that can be assessed.

5.3 Implications and adoption

For educators, NSDF gives a structured way to organise practical, learner-centred activities with existing no-code tools, and its steps can be adapted across disciplines and institutions. Community organisations and small businesses could apply the same method to routine tasks, given suitable training and testing. Costs stay modest because students use available devices and teaching resources can be reused, although platform access and training still need budgeting. The framework can be packaged as a ready-to-use teaching module with guidebooks, templates and assessment rubrics, supported by train-the-trainer programmes and licensed to other higher education institutions, schools and training providers.

5.4 Limitations

Unanimous relevance ratings support an initial content judgement, but a three-person panel gives limited precision, and the experts' endorsement of adaptability is professional judgement. The 29-student sample is small, the number of students invited was not recorded, and all student measures are self-reports from a single source. The questionnaire items referred to AWS PartyRock, not to NSDF's steps, so they describe the platform experience; completion of the steps was not scored. Alpha values above .95 may partly reflect similar item wording. The study measured neither learning gains nor application accuracy. A stronger evaluation would score process records and final applications against a transparent rubric with independent assessors, record response rates, and compare NSDF-guided tasks with comparable unguided no-code tasks.

6. CONCLUSION

NSDF proposes an assessable standard for generative AI development: learners should be able to defend the decisions behind their outputs, particularly when plausible responses fail the intended task. Its three phases and seven steps give learners without programming experience a repeatable process that runs from problem definition to user feedback and a final process audit. Expert ratings supported the relevance and clarity of every step, and students rated the AWS PartyRock experience as easy to use and useful. Faculty-wide participation of more than 500 students, including a category award at the AWS CendekiAwan 2026 National AI Competition, shows that the framework works beyond a single class. Future work will package NSDF as a licensable teaching module with assessment rubrics and compare independently scored development records and application quality under comparable teaching conditions.

Acknowledgments: The authors thank the three expert reviewers and the students of the Faculty of Information Science, UiTM Rembau Campus, who took part in this study.

References

Amazon Web Services. (2023, November 16). *Announcing PartyRock, an Amazon Bedrock playground*. https://aws.amazon.com/about-aws/whats-new/2023/11/partyrock-amazon-bedrock-playground

Brown, T. (2008). Design thinking. *Harvard Business Review, 86*(6), 84–92.

Cronbach, L. J. (1951). Coefficient alpha and the internal structure of tests. *Psychometrika, 16*(3), 297–334. https://doi.org/10.1007/BF02310555

Davis, F. D. (1989). Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Quarterly, 13*(3), 319–340. https://doi.org/10.2307/249008

Elitery. (2026, April 21). *CendekiAwan Malaysia program empowers thousands of Malaysian students to lead the generative AI revolution* [Press release]. PR Newswire Asia. https://en.prnasia.com/releases/apac/cendekiawan-malaysia-program-empowers-thousands-of-malaysian-students-to-lead-the-generative-ai-revolution-529910.shtml

Kolb, D. A. (1984). *Experiential learning: Experience as the source of learning and development*. Prentice Hall.

Long, D., & Magerko, B. (2020). What is AI literacy? Competencies and design considerations. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (pp. 1–16). Association for Computing Machinery. https://doi.org/10.1145/3313831.3376727

Lynn, M. R. (1986). Determination and quantification of content validity. *Nursing Research, 35*(6), 382–385. https://doi.org/10.1097/00006199-198611000-00017

Matook, S., Wang, Y. M., & Axelsen, M. (2025). Experiential learning for citizen developers: Training IT talent in low-code development and metacognitive reflection. *Business & Information Systems Engineering, 67*(1), 7–30. https://doi.org/10.1007/s12599-024-00921-3

Ng, D. T. K., Leung, J. K. L., Chu, S. K. W., & Qiao, M. S. (2021). Conceptualizing AI literacy: An exploratory review. *Computers and Education: Artificial Intelligence, 2*, Article 100041. https://doi.org/10.1016/j.caeai.2021.100041

Polit, D. F., & Beck, C. T. (2006). The content validity index: Are you sure you know what's being reported? Critique and recommendations. *Research in Nursing & Health, 29*(5), 489–497. https://doi.org/10.1002/nur.20147

Ruparelia, N. B. (2010). Software development lifecycle models. *ACM SIGSOFT Software Engineering Notes, 35*(3), 8–13. https://doi.org/10.1145/1764810.1764814

Sahay, A., Indamutsa, A., Di Ruscio, D., & Pierantonio, A. (2020). Supporting the understanding and comparison of low-code development platforms. In *2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)* (pp. 171–178). IEEE. https://doi.org/10.1109/SEAA51224.2020.00036

Sundberg, L., & Holmström, J. (2024). Teaching tip: Using no-code AI to teach machine learning in higher education. *Journal of Information Systems Education, 35*(1), 56–66. https://doi.org/10.62273/CYPL2902

UNESCO. (2025). *UNESCO survey: Two-thirds of higher education institutions have or are developing guidance on AI use*. https://www.unesco.org/en/articles/unesco-survey-two-thirds-higher-education-institutions-have-or-are-developing-guidance-ai-use