QA Automation of Canvas Courses

Natalia Echeverry[1]

[1] University of Pittsburgh; nae81@pitt.edu

Abstract. The increasing demand for online education calls for cost-effective and scalable systems that help instructional designers and faculty ensure the quality of their courses. The QA Bot addresses this need by conducting rapid evaluations of courses against established standards and practices, including those created by Quality Matters (QM), the Online Learning Consortium (OLC), and the Center for Applied Special Technology (CAST). The QA Bot working prototype follows a hybrid approach that combines programmatic file extraction with LLM-based evaluation of course data. This project emphasizes accessibility by supporting smaller, high-performing models (e.g., Llama 3.1 8B, DeepSeek-R1 8B) that enable an open-source release. The system's performance validation includes comparisons of course quality evaluations conducted by human experts (the benchmark) with those conducted by the QA Bot using open-source and commercial LLMs. This paper presents a work-in-progress system and its integration into the learning engineering process.

Keywords: Instructional Design; Quality Assurance; Learning Engineering; Large Language Models; QA Automation; Canvas LMS; Course Quality Standards.

Introduction

With the growing demand for online education, we recognize the need for a rapid diagnostic tool that helps instructional designers, faculty, and administrators identify opportunities for improvement in online courses. This need is particularly pressing for those working in low-resource contexts, who are often constrained by tight development timelines that make comprehensive manual course review impractical. The QA Bot is designed to support these teams by providing automated, efficient quality diagnostics that can be integrated into various instructional design processes, whether traditional (e.g., ADDIE, Dick & Carey model) or iterative (e.g., rapid prototyping) (Brown & Green, 2024).

Often, widely recognized standards and frameworks such as the Universal Design for Learning Guidelines (CAST, n.d.), the OLC Course Review Scorecard (Online Learning Consortium, n.d.), and the QM Course Design Rubric Standards (Quality Matters, n.d.), which codify evidence-based teaching practices, receive insufficient attention during online course design. In addition to resource constraints, this may occur because of a lack of experience in translating these frameworks into concrete instructional activities and assessments. The QA Bot addresses these challenges by evaluating courses against various frameworks and providing targeted feedback and actionable recommendations that enhance course quality.

This paper presents the design, architecture, and initial development of the QA Bot, contributing three key innovations: (1) a hybrid architecture that grounds LLM outputs in verifiable course content, (2) support for resource-efficient, open-source models that democratize access to automated course evaluation, and (3) explicit integration with learning

engineering's nested cycle approach for systematic instructional improvement. We describe the technical approach, preliminary validation methods comparing model performance, and a study design for a future comprehensive empirical evaluation with instructional design practitioners.

Background: Integration with Learning Engineering

Learning engineering provides a compatible theoretical framework for the systematic and practical use of the QA Bot, given its shared focus on identifying opportunities to improve learning conditions. As an engineering-oriented and agile approach, learning engineering guides instructional designers and learning engineers through three nested cycles: creation, implementation, and investigation (Totino & Kessler, 2024). The QA Bot's ease of use and relevance across these three cycles make it a practical tool for teams seeking systematic, scalable course-quality assurance.

Following a learning engineering nested cycle approach (Goodell & Kolodner, 2023; Totino & Kessler, 2024), the QA Bot integrates into each cycle:

- **Creation cycle:** Before learner data collection, the team analyzes prototype or beta-course exports to validate the design against established standards.
- **Implementation cycle:** After learner data collection, the team analyzes the production course export, combining QA Bot diagnostics with learning analytics to identify challenges.
- **Investigation cycle:** The team conducts targeted analysis of specific design elements or investigates identified issues in the production course.

The QA Bot output report can serve as direct input for learning design artifacts used by learning engineering teams, such as conjecture mappings (Totino & Kessler, 2024), design backlogs (Scrum.org, n.d.), and decision trackers (e.g., LEED Tracker; Totino & Kessler, 2024).

System Design and Architecture

To address the accuracy limitations of processing entire course exports in a single pass, the current design employs a hybrid architecture combining programmatic file extraction with targeted LLM evaluation. The system uses a Python-based extraction process that:

- Parses the Canvas course export (.imscc format) to identify relevant course components
- Maps specific evaluation criteria from instructional design frameworks to corresponding course files (e.g., syllabus files, content pages, assignment specifications)
- Extracts and prepares targeted content for LLM analysis
- Submits focused prompts to the LLM, each addressing specific standards with relevant course data

Model Specifications

The QA Bot supports several open-source language models that can be run locally, including Llama 3.1 8B and DeepSeek-R1 8B, as well as Llama 3.1 70B and DeepSeek-R1 70B for teams with greater computational resources. These general-purpose models were selected for their balance of performance and accessibility, enabling deployment on standard laptop hardware without requiring expensive API access or cloud infrastructure. The tool requires Python and Ollama installed locally. Hardware requirements scale with model size: 8B-parameter models run on standard laptops (8GB+ RAM), while 70B-parameter models require more computational resources. Detailed setup instructions will be provided upon open-source release.

A key focus of the project is making this tool accessible to learning engineering teams in

low-resource contexts. The project plans to release the tool as open-source software, allowing the academic community to improve, customize, and freely distribute it. This approach aligns with the goal of making expertise and resources for educational quality improvement widely accessible.

Evaluation Framework

We assess model performance by calculating percentage agreement and inter-rater reliability metrics, comparing small open-source models (Llama 3.1 8B, DeepSeek-R1 8B) against both large models (Llama 3.1 70B, DeepSeek-R1 70B, Claude Sonnet 4, GPT-4o) and human expert evaluations. This framework aims to address two questions: (1) How do small open-source models compare to large models in generating course evaluations? (2) To what extent do small open-source models (Llama 3.1 8B, DeepSeek-R1 8B) achieve comparable agreement with human experts?

Future Work

Future work involves optimizing the QA Bot's parser architecture and prompting to improve accuracy and performance, expanding support to other Learning Management Systems (e.g., Moodle) and standards (e.g., SCORM, xAPI, and cmi5), and supporting other languages. Moreover, adapting QA testing automation frameworks and practices from the gaming industry may provide valuable insights (Donat, 2005; Harty, 2011; Roche, 2013).

Citations

Brown, A., & Green, T. (2024). The essentials of instructional design: Connecting fundamental principles with process and practice. https://doi.org/10.4324/9781003404835

CAST. (n.d.). Universal Design for Learning guidelines. Retrieved January 31, 2026, from https://udlguidelines.cast.org/

Donat, M., Husain, J., & Coatta, T. (2014). Automated QA testing at EA: Driven by events. ACM Queue, 12(5). https://queue.acm.org/detail.cfm?id=2627372

Donat, M. (2005, February 16). Orchestrating an automated test lab. ACM Queue, 3(1). https://queue.acm.org/detail.cfm?id=1046946

Goodell, J., & Kolodner, J. (Eds.). (2023). Learning engineering toolkit: Evidence-based practices from the learning sciences, instructional design, and beyond. Taylor & Francis.

Goodell, J., Kessler, A., & Schatz, S. (2023). Learning engineering at a glance. Journal of Military Learning. https://www.armyupress.army.mil/Journals/Journal-of-Military-Learning/Journal-of-Military-Learning-Archives/Conference-Edition-2023-Journal-of-Military-Learning/Engineering-at-a-Glance/

Harty, J. (2011, January 12). Finding usability bugs with automated tests. ACM Queue, 9(1). https://queue.acm.org/detail.cfm?id=1925091

Koedinger, K. R., Corbett, A. T., & Perfetti, C. (2012). The knowledge-learning-instruction framework: Bridging the science-practice chasm to enhance robust student learning. Cognitive Science, 36(5), 757–798. https://doi.org/10.1111/j.1551-6709.2012.01245.x

McDonald, N., Schoenebeck, S., & Forte, A. (2019). Reliability and inter-rater reliability in qualitative research: Norms and guidelines for CSCW and HCI practice. Proceedings of the ACM on Human-Computer Interaction, 3(CSCW), Article 72. https://doi.org/10.1145/3359174

Online Learning Consortium. (n.d.). OLC course review scorecard. Retrieved January 31, 2026, from https://onlinelearningconsortium.org/quality/scorecards/course-review/

Quality Matters. (n.d.). QM course design rubric standards. Retrieved January 31, 2026, from https://www.qualitymatters.org/qa-resources/rubric-standards/higher-ed-rubric

Roche, J. (2013, October 30). Adopting DevOps practices in quality assurance: Merging the art and science of software development. ACM Queue, 11(9). https://queue.acm.org/detail.cfm?

id=2540984
Scrum.org. (n.d.). What is a product backlog? Retrieved January 18, 2026, from https://www.scrum.org/resources/what-is-a-product-backlog
Totino, L., & Kessler, A. (2024). "Why did we do that?" A systematic approach to tracking decisions in the design and iteration of learning experiences. The Journal of Applied Instructional Design, 13(2). https://doi.org/10.59668/1269.15630
Zhao, X., Wang, H., Ding, J., Hu, Z., Tian, Q., & Wang, Y. (2025). Augmenting software quality assurance with AI and automation using PyTest-BDD. Automated Software Engineering, 33(1). https://doi.org/10.1007/s10515-025-00566-w

Acknowledgments
The author would like to thank Rae Mancilla and Stephen Butler for their valuable feedback and contributions to this work.
Claude Sonnet 4.5 (Anthropic) was used as a writing assistance tool to improve manuscript clarity, organization, and grammatical accuracy. All research design, technical implementation, data analysis, and intellectual contributions are the original work of the author.

References
Brown, A., & Green, T. (2024). The essentials of instructional design: Connecting fundamental principles with process and practice. https://doi.org/10.4324/9781003404835
CAST. (n.d.). Universal Design for Learning guidelines. Retrieved January 31, 2026, from https://udlguidelines.cast.org/
Donat, M., Husain, J., & Coatta, T. (2014). Automated QA testing at EA: Driven by events. ACM Queue, 12(5). https://queue.acm.org/detail.cfm?id=2627372
Donat, M. (2005, February 16). Orchestrating an automated test lab. ACM Queue, 3(1). https://queue.acm.org/detail.cfm?id=1046946
Goodell, J., & Kolodner, J. (Eds.). (2023). Learning engineering toolkit: Evidence-based practices from the learning sciences, instructional design, and beyond. Taylor & Francis.
Goodell, J., Kessler, A., & Schatz, S. (2023). Learning engineering at a glance. Journal of Military Learning. https://www.armyupress.army.mil/Journals/Journal-of-Military-Learning/Journal-of-Military-Learning-Archives/Conference-Edition-2023-Journal-of-Military-Learning/Engineering-at-a-Glance/
Harty, J. (2011, January 12). Finding usability bugs with automated tests. ACM Queue, 9(1). https://queue.acm.org/detail.cfm?id=1925091
Koedinger, K. R., Corbett, A. T., & Perfetti, C. (2012). The knowledge-learning-instruction framework: Bridging the science-practice chasm to enhance robust student learning. Cognitive Science, 36(5), 757–798. https://doi.org/10.1111/j.1551-6709.2012.01245.x
McDonald, N., Schoenebeck, S., & Forte, A. (2019). Reliability and inter-rater reliability in qualitative research: Norms and guidelines for CSCW and HCI practice. Proceedings of the ACM on Human-Computer Interaction, 3(CSCW), Article 72. https://doi.org/10.1145/3359174
Online Learning Consortium. (n.d.). OLC course review scorecard. Retrieved January 31, 2026, from https://onlinelearningconsortium.org/quality/scorecards/course-review/
Quality Matters. (n.d.). QM course design rubric standards. Retrieved January 31, 2026, from https://www.qualitymatters.org/qa-resources/rubric-standards/higher-ed-rubric
Roche, J. (2013, October 30). Adopting DevOps practices in quality assurance: Merging the art and science of software development. ACM Queue, 11(9). https://queue.acm.org/detail.cfm?id=2540984
Scrum.org. (n.d.). What is a product backlog? Retrieved January 18, 2026, from https://www.scrum.org/resources/what-is-a-product-backlog

Totino, L., & Kessler, A. (2024). "Why did we do that?" A systematic approach to tracking decisions in the design and iteration of learning experiences. The Journal of Applied Instructional Design, 13(2). https://doi.org/10.59668/1269.15630
Zhao, X., Wang, H., Ding, J., Hu, Z., Tian, Q., & Wang, Y. (2025). Augmenting software quality assurance with AI and automation using PyTest-BDD. Automated Software Engineering, 33(1). https://doi.org/10.1007/s10515-025-00566-w