AI-Powered Grading Agent for Question-Answer and Programming Assignments with Human-in-the-Loop Validation

[Prabal Shrestha, Sujan Khadka, Shashwat Gautam, Aayush Bikram Bista]
Computer Science Department
Boise State University

October 2025

1 Executive Summary

The exponential growth in computer science enrollment presents significant challenges for educators managing large-scale assignment grading. Computer science bachelor's degree completions have more than doubled over the last decade, from 51,696 in 2013-2014 to 112,720 in 2022-2023 [1]. Traditional manual grading processes are time-intensive, subjective, and difficult to scale as enrollment continues growing at 4.9% annually [2].

As teaching assistants at BSU, we collectively spend most of our teaching hours grading homework and coursework in Canvas, often at the expense of more meaningful student interaction or personal study. While Canvas and SpeedGrader provide efficient tools for manual workflows, the sheer assignment volume and turnaround expectations have revealed persistent problems: grading fatigue, subjectivity, slow feedback, and workload bottlenecks during busy weeks.

Our project aims to address these bottlenecks by developing an AI-powered grading agent that is purpose-built for seamless integration into **Canvas** and **SpeedGrader** at BSU. By automating rubric-based grading, generating explainable feedback, and learning from instructor corrections, our system supports a transparent, time-saving workflow that improves both grading consistency and student satisfaction. Recent studies demonstrate that AI-assisted grading can reduce grading time by up to 75% while maintaining comparable accuracy to human graders [3].

2 Relevance to Course and AI

This project directly applies multiple AI techniques covered in the course, including natural language processing for text evaluation, machine learning for code quality and assignment analysis, and multi-agent architecture for handling different assignment types. The system leverages state-of-the-art Large Language Models (LLMs) including GPT-4, Gemini, and Claude for robust evaluation and integration with BSU's educational stack.

The project incorporates several core AI concepts: knowledge representation through rubric formalization, automated reasoning for grade assignment, machine learning for continuous improvement through human feedback, and natural language understanding for text analysis. These components demonstrate practical application of AI methodologies to solve real-world educational challenges.

3 Project Significance and Personal Motivation

A significant portion of our teaching responsibilities often more than half of our working hours are spent grading and commenting on student submissions in Canvas. This repetitive workload detracts from time spent tutoring, preparing lessons, or supporting students individually. We have seen firsthand how grading delays, inconsistent feedback, and subjective assessments can impact student motivation and course momentum.

The AI in education market is experiencing unprecedented growth, with the global market valued at USD 5.88 billion in 2024 and projected to reach USD 32.27 billion by 2030, growing at a CAGR of 31.2% [4]. This growth is driven by the exact challenges we face daily as teaching assistants: increasing class sizes, limited grading resources, and the need for timely, consistent feedback.

Our project directly targets these issues in **BSU's real-world setting**:

- **Canvas and SpeedGrader Integration:** The agent will natively work with Canvas and SpeedGrader, allowing effortless import and export, annotation, and grade synchronization for assignments submitted on BSU's official platforms. This integration means instructors and TAs can continue using familiar workflows without learning entirely new systems.
- **Real Data, Real Impact:** Unlike generic grading tools trained on standardized datasets, our system uses actual, anonymized BSU assignments that we have already graded ourselves. This ensures the models and interface are tuned to our specific grading practices, rubrics, and the types of assignments common in BSU computer science courses.
- **TA-Centered Design:** All features, rubrics, and feedback templates are informed by our lived experience as teaching assistants. We understand the pain points of grading at 2 AM before a deadline, the challenge of maintaining consistency across hundreds of submissions, and the difficulty of providing meaningful feedback when time is limited.
- **Time Reclamation:** By automating the initial grading pass, we aim to reclaim teaching assistant hours for higher-value activities: holding office hours, developing better course materials, providing personalized student support, and engaging in meaningful educational interactions rather than repetitive marking.

Research supports the potential impact of our approach. A comprehensive study at the University of Ljubljana involving over 100 students found that LLMs can achieve grading accuracy and feedback quality comparable to human graders when well-designed prompts are used [5]. Furthermore, automated programming assignment grading has been shown to result in higher student scores with lower standard deviation compared to manual grading, with immediate feedback improving learning outcomes [6].

This deeply local, practical perspective, grounded in the needs of the BSU teaching community, is rare in the grading agent ecosystem, making our tool uniquely valuable for BSU and similar institutions using Canvas.

4 Novelty and BSU Alignment

4.1 Existing Work Summary

Current automated grading solutions fall into several categories:

Essay and Text Grading: Tools like Grammarly AI Grader, StarGrader, and CoGrader focus primarily on essay evaluation using rubric-based assessment. Gradescope offers AI-assisted grouping of similar answers but relies heavily on human intervention.

Programming Assignment Grading: Platforms like CodeGrade, Codio, and various autograder frameworks primarily focus on correctness testing through unit tests and static code analysis, with limited semantic evaluation capabilities.

Recent LLM-Based Approaches: Studies have shown promising results with LLM-based grading. GPT-4 has achieved 75-90% accuracy in grading programming assignments [7], while for algebra problems, GPT-4.1-mini achieved 94.47% accuracy with self-consistency approaches [8]. However, most implementations lack transparency in their decision-making process and are not designed for integration with specific institutional systems.

4.2 Our Unique Contributions

Our grading agent's core innovations specifically address gaps in existing solutions:

- **BSU-First Integration:** Purpose-built for Canvas and SpeedGrader, not a standalone product or generic tool. The system will support Canvas API for assignment retrieval, submission download, and grade upload, making adoption frictionless for BSU instructors already using these tools.

- **LangChain and LangGraph Agent Architecture:** We employ **LangChain** and **LangGraph** frameworks to build sophisticated, stateful multi-agent workflows [15][16]. LangGraph's graph-based orchestration enables complex control flows including loops, conditional branching, and human-in-the-loop checkpoints, essential for grading tasks that require iterative review and validation. This architecture provides native support for state management, memory persistence across grading sessions, and transparent debugging through LangGraph Studio visualization tools.

- **Explainable AI Grading:** Every grade and comment comes with detailed, rubric-linked reasoning. Students can see exactly why they received specific marks, and instructors can verify that grading decisions align with their rubric criteria. This transparency increases trust and reduces grade disputes.

- **Human-in-the-Loop Adaptivity:** Using LangGraph's built-in checkpoint and interruption capabilities, the system can pause execution for human review at any point. When a TA corrects a grade or adjusts feedback, these modifications are captured in the agent's state and used to improve future grading accuracy through adaptive learning [15][17]. This creates a feedback loop where the tool becomes increasingly aligned with instructor preferences over time.

- **Authentic Training Data from BSU Courses:** Uses assignment data previously graded by our team from actual BSU computer science courses, anonymized to protect student privacy. This ensures grading style, feedback tone, and rubric interpretation match BSU standards and expectations rather than generic educational benchmarks.

- **Multimodal Assignment Support:** Handles both question-answer format assignments (short answer, essays, explanations) and programming assignments (code correctness, style, documentation) within a unified LangGraph workflow, reflecting the diverse assignment types common in CS courses [16].

This approach ensures the solution is immediately deployable, scalable, and valuable starting with our own classrooms, with potential for expansion across BSU's computer science department and eventually to other Canvas-using institutions.

5 Objectives and Deliverables

1. **Develop BSU-Integrated Grading Engine:** Implement LLM-based evaluation system using LangChain and LangGraph for BSU assignments and rubrics with seamless Canvas API integration for assignment import and grade export.
2. **Create Explainable Feedback System:** Build reasoning generation module that provides rubric-linked explanations for every grade and deduction, making the grading process transparent to students and reviewers.
3. **Implement Human-in-the-Loop Workflow:** Develop instructor review interface leveraging LangGraph's checkpoint system that allows TAs to easily verify, modify, and approve AI-generated grades, with all corrections feeding back into the learning system.
4. **Conduct BSU-Based Evaluation:** Test system performance on anonymized BSU assignment data, benchmarking accuracy and consistency against our own previous human grading to ensure the system meets BSU quality standards.
5. **Prepare Pilot Deployment:** Ready the implementation for course pilots in actual BSU classes using Canvas and SpeedGrader, with documentation and training materials for other TAs and instructors.

6 Scope and Feasibility

6.1 In Scope

- Grading BSU computer science assignments via Canvas API integration
- Programming assignment evaluation (Python, Java) and question-answer format assignments
- Rubric-based feedback with explainable reasoning using LangChain evaluation chains
- TA correction and review interface with feedback learning via LangGraph state management
- Use of anonymized BSU assignment data from courses we have TAed
- SpeedGrader-compatible output format

6.2 Out of Scope

- Integration with LMS platforms other than Canvas in phase one
- Production deployment beyond BSU pilot courses
- Advanced plagiarism detection (will rely on existing Canvas tools)
- Multi-language support beyond English
- Real-time grading during student submission

6.3 Resources

Datasets:

- Anonymized assignment submissions and grading data from BSU computer science courses, including assignments we have personally graded in our TA roles
- BSU instructor rubrics and grading criteria from multiple CS courses
- Feedback templates and comment patterns from actual BSU grading
- ASAP++ dataset (13,000+ essays with attribute scores) for supplementary training and comparison [9]

Technical Resources:

- OpenAI API for GPT-4 evaluation
- Google Cloud credits for Gemini Pro
- Anthropic API for Claude integration
- **LangChain and LangGraph frameworks** for agent orchestration and workflow management [15][16]
- Canvas API access and documentation
- BSU SpeedGrader workflow specifications
- Python backend development (Flask/FastAPI)
- React.js for web interface
- Docker for containerization
- MySQL for data storage and grade tracking

7 Methods

Our technical approach combines multiple AI techniques with modern agent frameworks for robust, explainable grading:

- **LangGraph Multi-Agent Architecture:** We implement a graph-based state machine using LangGraph [15][16] where different specialized agents handle specific grading tasks (rubric evaluation, code analysis, feedback generation). LangGraph's node-edge structure allows us to model complex workflows with conditional branching. For example, routing difficult cases to human review while auto-approving straightforward submissions. The framework's native persistence ensures grading state is maintained across sessions, critical for handling large assignment batches.
- **LangChain Evaluation Chains:** Utilize LangChain's built-in evaluation tools [18] to systematically assess grading quality, including accuracy, relevance, and feedback helpfulness metrics. These chains enable automated quality control by comparing AI-generated grades against ground truth.

- **Multi-Model LLM Ensemble:** Use GPT-4, Gemini, and Claude in parallel for robust evaluation, comparing outputs to identify consensus and flag uncertain cases for human review.
- **Rubric Formalization:** Convert BSU rubrics into structured formats that LLMs can consistently apply, with clear criteria mappings and point allocations.
- **Chain-of-Thought Prompting:** Generate step-by-step reasoning for each grading decision, making the AI's logic transparent and verifiable.
- **Programming Analysis:** Combine traditional autograder functionality (test cases, correctness checks) with LLM-based evaluation of code quality, documentation, and design choices.
- **Human-in-the-Loop Learning with LangGraph Checkpoints:** Implement feedback collection using LangGraph's interrupt and checkpoint features [15][17], where TA corrections trigger state updates that inform prompt refinement and improve future accuracy.
- **Evaluation Methodology:** Measure accuracy, speed, feedback consistency, and bias through comparison with ground-truth human grading on BSU assignments using LangChain evaluation frameworks [18].

8 Ethical Considerations

Student Privacy: All BSU data will be fully anonymized with student identifiers removed. Data will be used only for research and prototyping purposes. No personally identifiable information will be retained beyond the project duration. All data handling will comply with FERPA regulations.

Informed Consent: All datasets will be sourced from assignments previously graded by project team members in their TA roles, with appropriate permissions from course instructors and department administration.

Transparency and Fairness: The explainable AI approach ensures students understand how grades are determined. TAs and instructors maintain final authority to review, appeal, and override any AI-generated grades.

Bias Monitoring: The system will include monitoring for systematic grading biases across different student populations, assignment types, and response styles to ensure equitable assessment.

Academic Integrity: This tool is designed as a TA assistance system, not a replacement for human judgment in educational assessment. It augments rather than replaces the teaching assistant role.

9 Team Responsibilities

We emphasize that all team roles are **flexible**, and every member will contribute to all aspects of the project throughout its lifecycle. The assignments below reflect each member's current expertise to maximize workflow efficiency and mentorship, but deliverables and milestones will always be achieved collaboratively.

- **Prabal (LLM Integration, LangGraph Agent Architecture)**:
  - Design and development of LangChain evaluation chains and LangGraph agent workflows.
  - LLM integration and prompt-tuning for grading tasks.

  - Architect multi-agent system to coordinate evaluation, feedback generation, and Canvas/SpeedGrader endpoints.
  - Contribute to documentation, workflow debugging, and both frontend/backend integration.

- **Sujan (Backend Development, Canvas & API Integration)**:
  - Build and maintain backend services (`Flask/FastAPI`) and manage server deployment.
  - Develop integration with Canvas LMS, SpeedGrader, and all external APIs.
  - Support agent orchestration endpoints (LangGraph/LLMs) on the backend.
  - Participate in sprint planning, code review, and prototyping across the stack.

- **Shashwat (Frontend and User Experience Developer)**:
  - Develop TA/instructor interface matched to SpeedGrader workflows.
  - Implement interface for explainable reasoning, feedback, and correction/override.
  - Contribute to API integration, interface debugging, and end-user documentation.

- **Aayush (Evaluation and Data Analysis)**:
  - Prepare and anonymize BSU datasets for training and evaluation.
  - Lead comparative and bias analysis between AI- and human-generated grades.
  - Develop pipelines for performance metrics and consistency analysis.
  - Collaborate on iterative tuning, reporting, and end-user feedback testing.

All members will participate in coding, testing, design review, and presentations, ensuring shared ownership and learning. Roles are structured for current efficiency, but task boundaries will remain fluid as the project evolves.

10 Timeline

October 20 - November 10 (3 weeks): Project Setup and Foundation

- Complete comprehensive literature review of automated grading systems and LangGraph applications
- Gather and anonymize BSU assignment datasets from our TA courses
- Obtain Canvas API access and explore SpeedGrader integration points
- Formalize BSU rubrics into structured formats for AI processing
- Set up development environment with LangChain, LangGraph, and API credentials
- Design initial LangGraph workflow architecture

November 11 - November 20 (1.5 weeks): Core Engine Development

- Implement basic LangGraph agent nodes for grading workflow
- Develop LangChain evaluation chains for text-based assignment grading

- Build initial prompt engineering strategies for BSU rubrics
- Create autograder framework for programming assignment evaluation
- Test accuracy on small subset of anonymized BSU data
- Implement state persistence using LangGraph checkpoints

November 21 - November 29 (1 week): Integration and Interface Development

- Integrate Canvas API for assignment retrieval and grade submission
- Build TA review interface with SpeedGrader-compatible workflows
- Implement explainable reasoning display and feedback templates
- Create human-in-the-loop correction mechanism using LangGraph interrupts
- Add LangGraph Studio integration for workflow visualization
- Conduct initial user testing with team members

December 1 - December 7 (1 week): Buffer, Testing, and Documentation

- Bug fixes and system refinement based on testing
- Complete evaluation study comparing AI vs human grading accuracy
- Optimize LangGraph workflow performance and reduce latency
- Prepare demonstration with real BSU assignment examples
- Write final project report and documentation
- Create user guides for pilot deployment in BSU courses

References

1. National Student Clearinghouse Research Center. Computer Science Has Highest Increase in Bachelor's Earners. May 2024. https://www.studentclearinghouse.org/nscblog/computer-science-has-highest-increase-in-bachelors-earners/
2. Validated Insights. Validated Insights Report on Computer Science and Information Technology Released: Tech Degrees Growing, Some Significantly. June 2025. https://edtechchronicle.com/validated-insights-report-on-computer-science-and-information-technology-released-tec
3. NVIDIA Developer Blog. Artificial Intelligence Helps Grade Exams 90% Faster. August 2022. https://developer.nvidia.com/blog/artificial-intelligence-helps-grade-exams-90-faster/
4. Grand View Research. AI In Education Market Size & Share — Industry Report, 2030. October 2024. https://www.grandviewresearch.com/industry-analysis/artificial-intelligence-ai-ed
5. Policar, Pavlin G. et al. Automated Assignment Grading with Large Language Models: Insights From a Bioinformatics Course. arXiv preprint arXiv:2501.14499. 2025. https://arxiv.org/html/2501.14499v1

6. St. Aubin, A. J. An Empirical Investigation into the Impact of Automated Grading. University of Nevada, Las Vegas. Dissertation. 2022. https://oasis.library.unlv.edu/thesesdissertations/4476/

7. DiVA Portal. GPT-4 as an Automatic Grader. Master's Thesis, KTH Royal Institute of Technology. 2024. https://www.diva-portal.org/smash/get/diva2:1779778/FULLTEXT01.pdf

8. Bhandari, Shreya; Pardos, Zach. Can Language Models Grade Algebra Worked Solutions? Evaluating LLM-Based Autograders Against Human Grading. EDM 2025 Proceedings. Educational Data Mining Society. 2025. https://educationaldatamining.org/EDM2025/proceedings/2025.EDM.poster-demo-papers.290/

9. Mathias, Sandeep Albert; Bhattacharyya, Pushpak. ASAP++: Enriching the ASAP Automated Essay Grading Dataset with Essay Attribute Scores. LREC 2018. https://aclanthology.org/L18-1187.pdf

10. Tian, Z. et al. Implementation Considerations for Automated AI Grading of Student Work. University of Washington & Hensun Innovation. 2024. https://arxiv.org/html/2506.07955v1

11. Floden, J. Grading Exams Using Large Language Models: A Comparison Between Human and AI Grading of Exams in Higher Education Using ChatGPT. British Educational Research Journal, 51, 201–224. Wiley. 2025.

12. Messer, M. et al. AI and Auto-Grading in Higher Education: Capabilities, Ethics, and the Evolving Role of Educators. Ohio State University Academic Technologies. 2024. https://ascode.osu.edu/news/ai-and-auto-grading-higher-education-capabilities-ethics-and-evolving

13. Dubois, M. et al. Skewed Score: A Statistical Framework to Assess Autograders. UK AI Security Institute. 2024. https://arxiv.org/html/2507.03772v1

14. Taylor & Francis. Promises and Breakages of Automated Grading Systems. Educational Assessment, Evaluation and Accountability. February 2025.

15. Campos, Nuno. Building LangGraph: Designing an Agent Runtime from first principles. LangChain Blog. October 2025. https://blog.langchain.com/building-langgraph/

16. Runkle, Sydney; LangChain OSS Team. LangChain and LangGraph Agent Frameworks Reach v1.0 Milestones. LangChain Blog. October 2025. https://blog.langchain.com/langchain-langgraph-1dot0/

17. LangChain. LangGraph - Build Controllable Cognitive Architecture. 2025. https://www.langchain.com/langgraph

18. LangChain. Evaluation Module Documentation. LangChain API Reference. October 2025. https://python.langchain.com/api_reference/langchain/evaluation.html