AI-Augmented Personalized Feedback

Supporting Scalable, Human-Guided Feedback

This attachment is intended to visually illustrate the project submitted for the CU Innovation & Efficiency Awards. The project is an **actively used, AI-augmented feedback workflow** developed for ASEN 1030 to help instructional staff provide timely, individualized feedback on concept questions, code reports, and other open-ended student work in large-enrollment courses.

Rather than replacing instructor judgment, the system uses an AI-driven pipeline to generate personalized feedback that is **reviewed, edited, and approved by human instructors and graders** before release. This shifts the instructional workflow from resource intensive manual composition of feedback to a faster expert review-and-refinement process, making detailed, personalized feedback more operationally feasible at scale.

This project was developed within the broader BOBPE vision (*BOt-Based Personalized Education & Productivity Enhancement*; additional context available at: bobpe.com) and reflects a practical, fully implemented example of using AI to improve instructional operations while preserving rigor, consistency, and human oversight.

In the Fall 2025 semester, this system provided approximately 100 students with high-quality, personalized feedback across 969 concept quiz responses and 3 lab reports, with a total OpenAI API cost of less than $7 for the entire semester.

It is also important to emphasize that the aim of this project is **not** to replace human graders, but to equip the instructional team with a tool that enables highly personalized feedback in large-scale courses at an extremely low cost, making this impactful instructional practice operationally feasible.

What the images show

The screenshots included in this attachment demonstrate the project in practice:

- A **student-facing concept feedback report** showing individualized explanation-centered feedback.
- A **concept assessment grading interface** where instructional staff review student responses alongside AI-generated feedback.
- A **lab report feedback example** illustrating detailed, criterion-based comments on coding and reasoning.
- An **instructor review interface** supporting efficient human oversight and refinement before feedback is returned to students.

Together, these artifacts show how the innovation improves both **efficiency** and **service quality**: instructional staff can provide richer, more consistent feedback without requiring the same level of fully manual writing for every response, and students receive clearer guidance on their reasoning and next steps.

Concept Report Student Example

This image shows a typical student evaluation report that is automatically generated and returned as a PDF. Each report includes the student's original response alongside AI-augmented, instructor-reviewed feedback. Feedback is personalized to each student's short-answer response and includes targeted explanations as well as encouraging guidance, supporting reflection and conceptual understanding.

Student Report -

Original student response:

The correct answer is A because it breaks down the process of testing paper airplane distances into relevant and repeatable steps that allow for a clear way to qualify/quantify aspects of the plane.One incorrect answer is E because none of the steps are relevant to testing airplane distances. Changing paper weight between throws and a lack of measuring distances make it difficult to differentiate aspects which make one plane better than another.

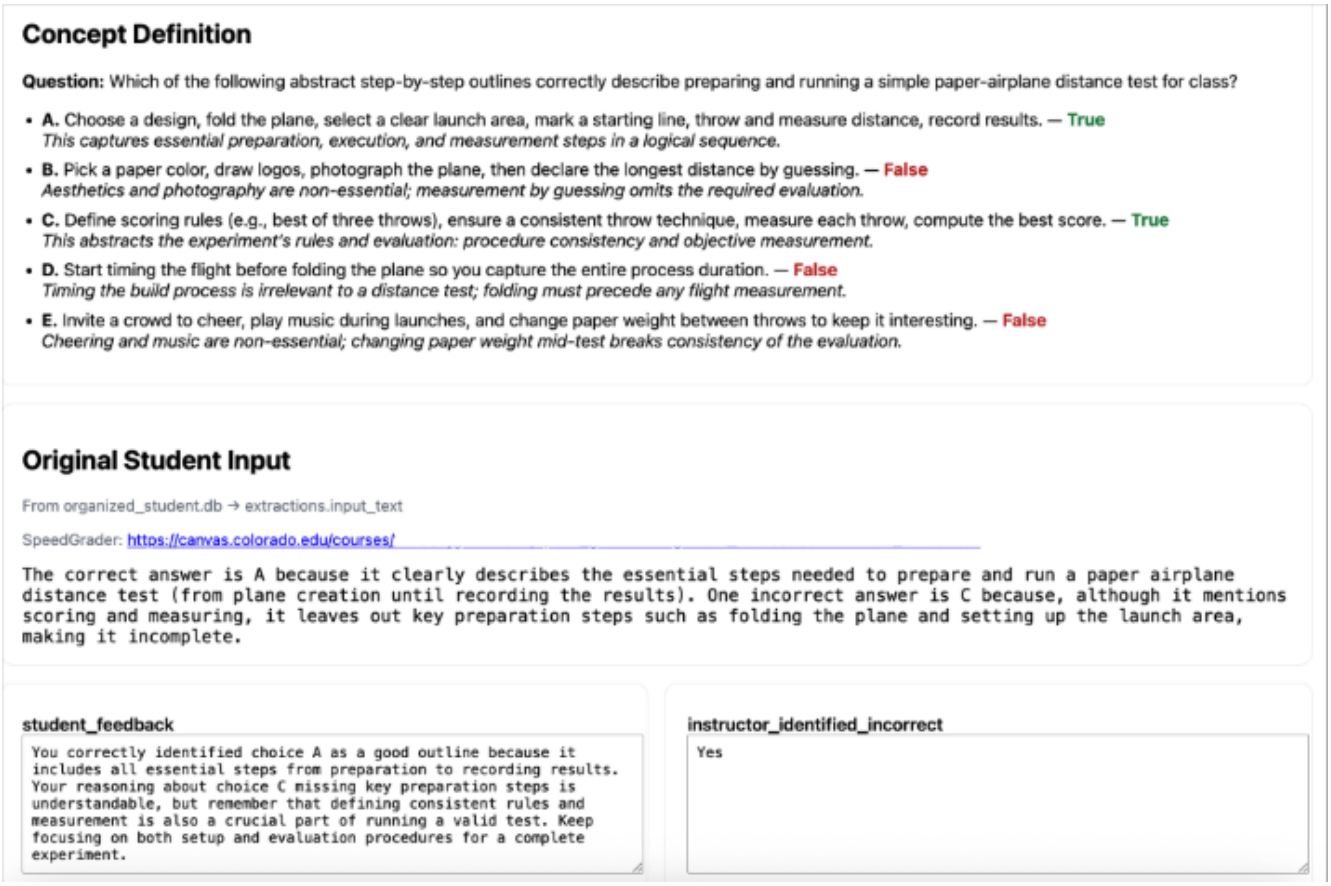
Instructor feedback:

You correctly identified choice A as a good outline because it includes clear, repeatable steps for measuring distance. Your rejection of choice E is also valid since cheering and music do not affect the test's measurement and changing paper weight disrupts consistency. Keep focusing on steps that ensure objective and consistent evaluation.

Concept Assessment Grader Interface

This image shows the human grader interface for weekly concept quizzes. The question and answer key are displayed at the top, followed by the student's verbatim response and a direct link to their Canvas submission. The `student_feedback` field is fully editable, allowing graders to quickly review and refine AI-generated feedback before proceeding.

During the Fall 2025 semester, nearly 80% of AI-generated feedback was released unchanged after human review, with the remaining responses requiring only minor edits. Overall, grader effort translated to an effective feedback rate exceeding 80 words per minute, enabling high-quality personalized feedback with professional typist speed.



Code Report Grader Interface

This image shows the code review grading interface used for programming-based assignments. The student's submitted code is displayed alongside the program output, with AI-augmented feedback provided below. Both the feedback and pass/fail determinations are fully editable by the grader. This interface allows instructional staff to efficiently evaluate code correctness, execution, and structure within a single, integrated view.

task3.m

2_Evaluation/matlab_based_eval/Lab3/filtered[redacted]/auto_extracted/task3.m

```
v = 20:2:30;                    %Row vector for all the possible velocities in m/s
g = 9.81;                       %Constant acceleration due to gravity on Earth
theta = 45;                     %degrees

if (theta >= 31) && (theta <= 60) %degrees
    disp('Medium Arc')
elseif (theta >= 0) %degrees
    disp('Low arc')
elseif (theta >= 61) && (theta <= 90)
    disp('High arc')
elseif (theta > 90) || (theta < 0) %degrees
    disp('Launch failure')
end

%Calculating Hang Time
for value = v
    t = (2 * value * sind(theta)) / g;
    disp(['Hang Time for v = ', num2str(value), ' m/s: ', num2str(t), ' seconds']);
end

%Calculating Maximum Height
for value = v
    h = ((value^2) * (sind(theta))^2) / (2 * g);
    if (h > 25) %meters
        disp('Exceeds saftey threshold')
    elseif (h <= 25) %meters
        disp('Meets saftey threshold')
    end
    disp(['Maximum Height for v = ', num2str(value), ' m/s: ', num2str(h), ' meters']);
end

%Calculating Range
for value = v
    R = ((value^2) * sind(2 * theta)) / g;  % Range formula
    if (R > 40) %meters
        disp('Target Reached!')
    elseif (R >= 35 && R <= 40)  %meters
        disp('Close, but not enough')
    elseif (R < 35)  %meters
        disp('Too short!')
    elseif (R < 0)  %meters
        disp('Launch Failure')
    end
    disp(['Range for v = ', num2str(value), ' m/s: ', num2str(R), ' meters']);
end
```

task3.txt

2_Evaluation/matlab_based_eval/Lab3/filtered[redacted]

```
Medium Arc
Hang Time for v = 20 m/s: 2.8832 seconds
Hang Time for v = 22 m/s: 3.1715 seconds
Hang Time for v = 24 m/s: 3.4598 seconds
Hang Time for v = 26 m/s: 3.7482 seconds
Hang Time for v = 28 m/s: 4.0365 seconds
Hang Time for v = 30 m/s: 4.3248 seconds
Meets saftey threshold
Maximum Height for v = 20 m/s: 10.1937 meter
Meets saftey threshold
Maximum Height for v = 22 m/s: 12.3344 meter
Meets saftey threshold
Maximum Height for v = 24 m/s: 14.6789 meter
Meets saftey threshold
Maximum Height for v = 26 m/s: 17.2273 meter
Meets saftey threshold
Maximum Height for v = 28 m/s: 19.9796 meter
Meets saftey threshold
Maximum Height for v = 30 m/s: 22.9358 meter
Target Reached!
Range for v = 20 m/s: 40.7747 meters
Target Reached!
Range for v = 22 m/s: 49.3374 meters
Target Reached!
Range for v = 24 m/s: 58.7156 meters
Target Reached!
Range for v = 26 m/s: 68.9093 meters
Target Reached!
Range for v = 28 m/s: 79.9185 meters
Target Reached!
Range for v = 30 m/s: 91.7431 meters
```

Edit task3_output

Current Result (from JSON)

FAIL

Override Result FAIL Student Feedback

- The angle classification message is printed once, but the format and casing differ from the expected "Medium arc" message; check the exact message and casing.
- The hang time, max height, and range are computed correctly but are printed in separate loops rather than combined per trial; consider consolidating calculations and outputs per velocity iteration.
- The safety threshold message is printed for each height, but the "Meets saftey threshold" message is not required by the intent and differs from the expected output; verify the safety message logic and wording.
- The range classification messages are correct in logic but are printed after the height loop, not per trial; ensure all messages for a given velocity are grouped together.

Code Report Student Feedback Example

This image shows a sample lab report feedback form provided to students. The report includes pass/fail evaluations for rubric components along with detailed, personalized feedback on code structure, logic, and clarity. This approach enables the instructional team to assess not only code output but also the quality and organization of student-written code—something that is typically impractical to evaluate consistently in large-scale programming courses.

Overall Score: 0.67 (2/3 tasks passed)

Task 3

FAIL

Student Feedback:

- Your code references an undefined variable `trialNum` when printing trial information; this causes a runtime error.

Task 4

PASS

Student Feedback:

- Your nested loops correctly cover all combinations of v0 and theta with proper trial numbering.
- The range-based and safety threshold messages match the expected classification rules.
- Angle classification uses a switch-like structure and correctly assigns arcs per the specified inclusive bounds.
- Calculations and printed outputs are consistent with the physics formulas and expected results.

Task 5

PASS

Student Feedback:

Parsed `<22,60>` (interpreted as `v0,theta`) computed range 42.727 m is within ±1.000 m of the 43.000 m target.

Summary

Task	Result	Score Contribution
Task 3	FAIL	0.00
Task 4	PASS	0.33
Task 5	PASS	0.33
Total Score		**0.67**