AI Mentors for Student Projects: Spotting Early Issues in Computer Science Proposals

Gati Aher GAHER@CS.CMU.EDU
Robin Schmucker RSCHMUCK@CS.CMU.EDU
Tom Mitchell TOM.MITCHELL@CS.CMU.EDU
Zachary C. Lipton ZLIPTON@CS.CMU.EDU
Carnegie Mellon University, Pittsburgh, PA 15213, USA

Abstract

When executed well, project-based learning (PBL) engages students' intrinsic motivation, encourages students to learn far beyond a course's limited curriculum, and prepares students to think critically and maturely about the skills and tools at their disposal. However, educators experience mixed results when using PBL in their classrooms: some students thrive with minimal guidance and others flounder. Early evaluation of project proposals could help educators determine which students need more support, yet evaluating project proposals and student aptitude is time-consuming and difficult to scale. In this work, we design, implement, and conduct an initial user study ($n = 36$) for a software system that collects project proposals and aptitude information to support educators in determining whether a student is ready to engage with PBL. We find that (1) users perceived the system as helpful for writing project proposals and identifying tools and technologies to learn more about, (2) educator ratings indicate that users with less technical experience in the project topic tend to write lower-quality project proposals, and (3) GPT-4o's ratings show agreement with educator ratings. While the prospect of using LLMs to rate the quality of students' project proposals is promising, its long-term effectiveness strongly hinges on future efforts at characterizing indicators that reliably predict students' success and motivation to learn.

Keywords: project-based learning, generative AI in education, learning analytics

1. Introduction

Project-based learning puts students in charge of their own education, giving them the freedom to decide what to learn, what to create, and what success means for them. It is an approach that intends to mimic real life, where answers are not easy to find, and success requires curiosity, critical thinking, and resilience. PBL is frequently used in computer science classes because it encourages students to test their understanding against real-world problems, develop essential project management skills, and explore industry tools and practices outside of the standard classroom curriculum.

However, PBL has shown mixed effectiveness in the classroom, with learning and motivation depending on students' prior ability, educators' ability to support specific projects and topics (Barnes and Bramley, 2008), and whether the project idea originated from students or teachers (Pucher and Lehner, 2011). Evaluating project proposals may help educators identify which students require additional guidance or alert educators to major issues that could consume limited motivation, time, and resources. Yet evaluating project proposals

is challenging and time-consuming, requiring educators to review freeform text on a broad range of topics and infer student aptitude from incomplete or vague details.

In this work, we design, implement, and conduct a user study for a software system that collects project proposals and aptitude information that may help educators determine whether a student is ready to engage with PBL. Our design elicits proposals for a high school-level computer science project where students build and design an interactive web application or video game. Our project proposal form asks users to (1) describe a problem they want to work on, (2) propose a solution, (3) recall and analyze design inspirations, (4) predict the effects of their design, (5) plan to evaluate and iterate on their project, (6) describe skills they want to develop, and (7) connect those skills to computer science careers, technical tasks, and popular industry technologies.

Our preliminary user study ($n = 36$) showed our system could engage users' intrinsic motivation, with 88.8% of users wanting to use our system in the future to choose skills and technologies to learn more about, and 91.6% of users wanting to use our system in the future to design project ideas that motivate them to learn more. Two educators and GPT-4o independently graded the quality of each project proposal according to a 29-item rubric. Educators and GPT-4o showed promising agreement on the relative quality of proposals and tended to give lower quality scores to users with less computer science experience.

Our findings suggest that LLMs show promise for scaling the automatic grading of project proposals; however, the effectiveness of using LLM grades to guide instructional design decisions hinges on whether project proposals and grading criteria contain information that reliably predicts whether a student can benefit from project-based learning.

2. Related Work

Barnes and Bramley (2008) found that student engagement in an Algebra 1 class improved when activities incorporated opportunities for choice, goal-setting, and one-on-one interactions with teachers. Similarly, Pucher and Lehner (2011) reported increased motivation among students when employing PBL in computer science classes. However, both studies highlighted challenges tied to students' limited meta-cognitive skills and domain knowledge. For instance, many students struggled to maintain their goal portfolios (Barnes and Bramley, 2008), and student-defined projects received significantly lower grades from faculty (Pucher and Lehner, 2011). Kirschner et al. (2006) critique PBL's effectiveness in supporting student learning, arguing that under minimal guidance, inexperienced students often struggle to develop effective strategies for independently searching for and applying information. Recent advances in LLMs enabled new types of learning technologies in various educational domains (Kasneci et al., 2023). LLMs' ability to provide feedback to student submissions (Botelho et al., 2023) and to support student self-reflection (Yazici et al., 2024) inspires the present development of a system to support effective execution of PBL.

3. System Design

Our software system is implemented as an React.js and firebase-based web application that collects project proposals and aptitude information that may help educators determine whether a student is ready to engage with PBL. The system logs the time and type of

user actions, such as keystroke patterns, answers to multiple-choice selections, and when users navigate away from and back to the web page. Our system first asks users questions about their aptitude for problem-solving, experience with computer science technologies, and prior experience with PBL. Then the system asks users to write down their idea for a high school-level computer science project on developing an interactive web application or video game. Our project proposal form, adapted from CS Pathway (2024); Byrdseed (2009), asks users to (1) describe a problem they want to work on, (2) describe the software they want to build, (3) recall and analyze design inspirations, (4) predict the effects of their design, (5) make a plan to evaluate and iterate on their project, (6) self-evaluate using a 10-point quality checklist, (7) describe skills they want to develop, and (8) connect those skills to computer science careers, technical tasks, and popular industry technologies. Due to space limitations, we included UI images in Appendix A.

4. System Evaluation

Each project proposal was independently evaluated by two human domain experts with experience as head teaching assistant (TA) for undergraduate computer science classes and GPT-4o (Hurst et al., 2024). All raters used the same 23-item quality checklist with four subtasks: (1) 10-question quality checklist students had used to self-assess their own work, (2) 3 questions judging the quality of 3 skill descriptions written by students, (3) 9 questions judging the appropriateness of skill-career pairings (3×3), and (4) an overall quality judgment question ("I would recommend a student include this project on their resume."). We did not perform any prompt-engineering on GPT-4o beyond evaluating subtasks and each of the student's three written skill descriptions in a separate prompt to avoid holistic evaluations. We provide the evaluation rubric in Appendix B and GPT-4o prompts in Appendix C.

This evaluation protocol captures two main signals of user readiness for PBL. First, by comparing users' self-assessments to expert human evaluations on the same 10-point **quality checklist** adapted from WPI CS 4518 (2024) and Lawlor (2012), we could determine whether users can accurately report on the quality of their own work.

Second, by asking human expert raters to classify the **quality of skill descriptions** ("Good" vs. "Irrelevant", "Vague", or "Not Core Computer Science Skill") of three skills users want to develop by working on their project and the **appropriateness of how users matched those skills to predefined mentor profiles** (such as "Data Scientist" or "Software Developer"), career-specific tasks, and trending technologies sourced from the O*Net Online Database (National Center for O*NET Development, 2024), we could capture signal on whether users' are capable of identifying perspectives and types of industry knowledge relevant to their project ideas. Users were explicitly instructed to leverage internet resources for assistance if needed, so this metric was intended to measure general reasoning and searching ability rather than ability to recall specific definitions. However, misinterpreting instructions, inability to parse technical language used by O*Net Online Database (National Center for O*NET Development, 2024), and shoehorning skills to match one of a limited number of options could weaken the strength of this measurement. We sanity check the quality of our grading criteria by comparing scores from users with higher levels

of computer science experience against users with less experience. We expect that students with computer science experience should be able to write higher quality project proposals.

Additionally, we asked the human experts raters and GPT-4o to answer a question meant to capture the overall quality of the project proposal ("I would recommend a student include this project on their resume. Yes/No").

We checked the motivational benefits of our project proposal writing activity with a post-activity experience survey.

Experimental Procedure. We recruited 40 participants online via Prolific. Our recruitment call was shown to crowd workers (i) 18 years or older; (ii) located in the USA; (iii) fluent in English; (iv) possess at least a high-school (HS) degree or equivalent; (v) using a desktop device (no mobile or tablets); and (vi) answered yes to "Are you a student?" on Prolific (see recruitment call in Appendix D). 2 participants were filtered out for not completing the activity, and 2 were filtered for failing attention checks. Of the 36 remaining participants, 17 were 18-25 years old, 14 were 26-35 years old, and 5 were over 46 years old. 25 were male, 11 were female. 11 participants reported Computer Science as their field of work/study, and 19 claimed to have computer science experience, i.e, have built a website or interactive application before using "Programming Language (e.g., Python, JavaScript)", "Backend Technologies (e.g., Node.js, Django, Flask)", "Cloud Platforms (e.g., Firebase, AWS)", and/or "Testing and Deployment (e.g., unit testing libraries, Docker)". Interestingly, some of these users claimed experience in categories like Backend Technologies or Cloud Platforms while simultaneously reporting no experience with Programming Languages. This discrepancy suggests a potential need for more fine-grained and specific questions to accurately capture their technical experience.

All participants encountered the same set of activities and questions in the same order, and could not go back to previous stages. Participants were told the activity would take 30-45 minutes, but we did not control for time in any phase. The median time to complete the activity was 40 minutes. The recruitment and study process was approved by CMU's Institutional Review Board (research study 2024_00000405).

4.1. Results

As shown in Table 1, human expert raters and GPT-4o showed promising agreement on the relative quality of proposals. As shown in Table 2, human expert raters and GPT-4o tended to give lower quality scores to users with less computer science experience.

Skill Classification. Human raters and GPT-4o classifed 50% of skills as irrelevant, vague, or not relevant to computer science ($> 80\%$, $\kappa > 0.6$). Consistent across all three graders, students with less computer science knowledge had an average of 1 out of 3 skills accepted, while students with more computer science knowledge had an average of 2 of 3 skills accepted.

Appropriateness of Skill Pairing Classification. The low quality of skills written down by the students might have made the subsequent activity to pair skills with careers, tasks, and technologies more challenging to grade. Human rater agreement was minimal ($68\%, \kappa = 0.25$). GPT-4o showed weak agreement with TA 1 ($74\%, \kappa = 0.46$) and minimal agreement with TA 2 ($67\%, \kappa = 0.20$).

Table 1: Rater Agreement. We report agreement percentage and Cohen's κ value between the two human expert raters (TA1 and TA2) and GPT-4o with respect to the 4 grading subtasks. As an overall index of agreement, we compute kappa for all rater pairs then report the arithmetic mean of these estimates.

	TA1 / TA2	TA1 / GPT-4o	TA2 / GPT-4o	Avg. κ
Skill Quality Classification	86.1%, $\kappa = 0.72$	84.3%, $\kappa = 0.68$	81.5%, $\kappa = 0.63$	0.68
Skill Pairing Classification	68.6%, $\kappa = 0.26$	74.2%, $\kappa = 0.46$	67.2%, $\kappa = 0.20$	0.29
Quality Checklist	87.2%, $\kappa = 0.49$	71.4%, $\kappa = 0.28$	74.7%, $\kappa = 0.28$	0.38
Recommend for Resume	80.6%, $\kappa = 0.50$	75.0%, $\kappa = 0.43$	77.8%, $\kappa = 0.45$	0.46

Table 2: Mean Positive Grades for Experienced vs. Novice Students. Each cell represents the mean proportion of student responses graded positively per subtask and rater, grouped by whether students reported having previous computer science experience. Subtasks averaged over 3, 6, 10, and 1 rater items respectively

Task	Experience	TA1	TA2	GPT-4o	Self-Rating
Skill Classification	Novice	35.3%	39.2%	31.4%	-
	Experienced	70.2%	68.4%	68.4%	-
Skill Pairing Classification	Novice	60.8%	86.9%	57.5%	-
	Experienced	53.7%	92.4%	64.9%	-
Quality Checklist	Novice	83.5%	82.4%	58.2%	85.9%
	Experienced	84.7%	90.0%	71.1%	95.8%
Recommend for Resume	Novice	58.8%	70.6%	58.8%	-
	Experienced	78.9%	84.2%	73.7%	-

Quality Checklist. GPT-4o's scores tended to be lower than either human rater's scores, leading to lower agreement between GPT-4o and humans ($71 - 75\%, \kappa = 0.28$). However, as seen in Figure 1, GPT-4o roughly maintained the rank of student project proposal quality relative to each other, shown by GPT-4o's Spearman correlations with TA 1's (Spearman=0.70) and TA 2's (Spearman=0.53). In contrast, students' self-evaluations had a much weaker correlation with either human expert's ratings (Spearman=0.16, Spearman=0.38).

Experience Survey. As shown in Figure 2, the large majority of users enjoyed the process of writing project proposals, with 88.8% of users wanting to use our system in the future to choose skills and technologies to learn more about, and 91.6% of users wanting to use our system in the future to design project ideas that motivate them to learn more.

5. Discussion and Limitations

Our findings suggest that LLMs show promise in scaling the automatic grading of project proposals; however, the effectiveness of using LLM grades to guide instructional design decisions hinges on whether project proposals and grading criteria contain information that

	TA 1	TA 2	GPT-4o	Self
TA 1	1.00	0.74	0.70	0.16
TA 2	0.74	1.00	0.53	0.38
GPT-4o	0.70	0.53	1.00	0.23
Self	0.16	0.38	0.23	1.00

Figure 1: Spearman correlation of students' total score on the 10-point quality checklist. Although GPT-4o's scores for the quality checklist task are lower than both teaching assistants and student self-evaluations, GPT-4o's scores preserve the rank order of teaching assistants' scores better than students' self-evaluation scores do.

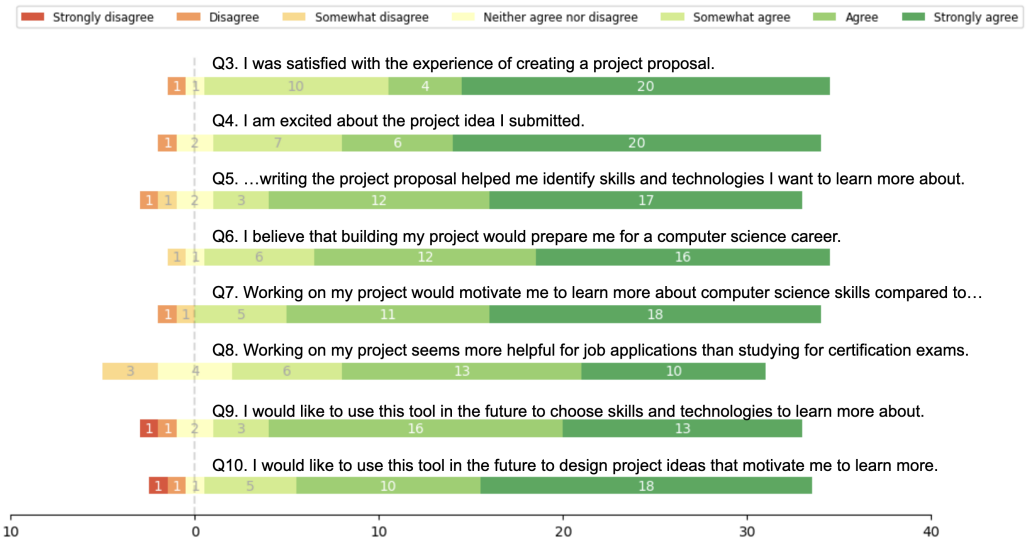

Figure 2: **Experience Survey, Response Counts.** The majority of users reported high level of excitement, motivation, and wanting to use the activity in the future to choose skills and technologies to learn more about.

reliably predicts whether a student can benefit from project-based learning. There are several limitations of our work that we now discuss.

First, in an optional feedback text field, human expert raters noted that several project proposals contained vague implementation details and skills, making it challenging for raters to determine whether crowd workers had appropriately paired their skills to computer sci-

ence careers, tasks, or technologies (inter-TA agreement of 68.61%, $\kappa = 0.26$). To address these concerns, we gave raters a skill classification rubric to quantify the vagueness and irrelevance of skills, achieving moderate agreement among human expert raters (inter-TA agreement of 86.11%, $\kappa = 0.72$). Consistent across both human expert ratings and GPT-4o ratings, students with less computer science knowledge wrote an average of 1 out of 3 acceptable skills, while students with more computer science knowledge wrote an average of 2 of 3 acceptable skills. The effectiveness of project-based learning relies on students' recognizing the necessary skills to develop and maintaining motivation to learn them. While our current work focuses on leveraging LLMs to evaluate issues in project proposals, future iterations of our system may be able to improve students' success in project-based learning by helping beginners gain an awareness of specific and contextually relevant computer science skills.

Second, our study examines project proposals written by crowd workers who were 18 years or older and identified as students (answered yes to "Are you a student?"). In this preliminary work, our aim was to gather project proposals from students with diverse backgrounds and varying degrees of familiarity with computer science. This approach attempted to capture the potential diversity of students that might enroll in introductory computer science courses at the high school or undergraduate level. The trends observed in the collected proposals can inform the design of future tools to support students in writing project proposals. Future work will investigate whether trends and issues observed in crowd workers' project proposals transfer to classroom assignments.

Third, while generative AI-based learning technologies like our GPT-4o-based system show promise, they also incur regular costs due to API calls, raising important questions about equity and accessibility in educational contexts. While we use GPT-4o, we should be able to use our LLM-powered grading methodology with any generative large language model. Future work could explore the application of different LLMs for this task, including open-source models.

References

Jessica L Barnes and Scott A Bramley. Increasing high school student engagement in classroom activities by implementing real-world projects with choice, goals portfolios, and goals conferencing. *Online Submission*, 2008.

Anthony Botelho, Sami Baral, John A Erickson, Priyanka Benachamardi, and Neil T Heffernan. Leveraging natural language processing to support automated assessment and feedback for student open responses in mathematics. *J. of Computer Assisted Learning*, 2023.

Byrdseed. Don't Jump Straight to "Create"! https://www.byrdseed.com/creating-board-games-animation-comics/, 2009. Retrieved November 22, 2024.

CS Pathway. CS Pathway – An Innovative Computer Science Career Pathway Program for High School Students in Massachusetts. https://cspathway.org/, 2024. Retrieved November 22, 2024.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

Enkelejda Kasneci, Kathrin Seßler, Stefan Küchemann, Maria Bannert, Daryna Dementieva, Frank Fischer, Urs Gasser, Georg Groh, Stephan Günnemann, Eyke Hüllermeier, et al. Chatgpt for good? on opportunities and challenges of large language models for education. *Learning and individual differences*, 103:102274, 2023.

Paul Kirschner, John Sweller, and Richard E Clark. Why unguided learning does not work: An analysis of the failure of discovery learning, problem-based learning, experiential learning and inquiry-based learning. *Educational Psychologist*, 41(2):75–86, 2006.

K Blaine Lawlor. Smart goals: How the application of smart goals can contribute to achievement of student learning outcomes. In *Developments in business simulation and experiential learning: Proceedings of the annual ABSEL conference*, volume 39, 2012.

National Center for O*NET Development. O*NET OnLine Help: The Database. https://www.onetonline.org/help/onet/database, 2024. Retrieved November 22, 2024.

Robert Pucher and Martin Lehner. Project based learning in computer science–a review of more than 500 projects. *Procedia-Social and Behavioral Sciences*, 29:1561–1566, 2011.

WPI CS 4518. CS 4518 Final Project Proposal Grading Sheet. http://web.cs.wpi.edu/~emmanuel/courses/cs4518/C18/projects/final_project/project_proposal_grading_sheet.pdf, 2024. Retrieved November 22, 2024.

Aybars Yazici, Paola Meija-Domenzain, Jibril Frej, and Tanja Käser. Gelex: Generative ai-hybrid system for example-based learning. In *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*, pages 1–10, 2024.

Appendix A. Illustrations of User Interface

Here, we provide additional details regarding the system workflow and user interface. The system guides users through a series of steps prompting the user to (1) describe a problem they want to work on (Figure 3 and 4), (2) propose a solution (Figure 5), (3) recall and analyze design inspirations (Figure 4), (4) predict the effects of their design (Figure 6), (5) plan to evaluate and iterate on their project, (6) describe skills they want to develop, and (7) connect those skills to computer science careers, technical tasks, and popular industry technologies (Figure 7, 8, 9, 10 and 11).

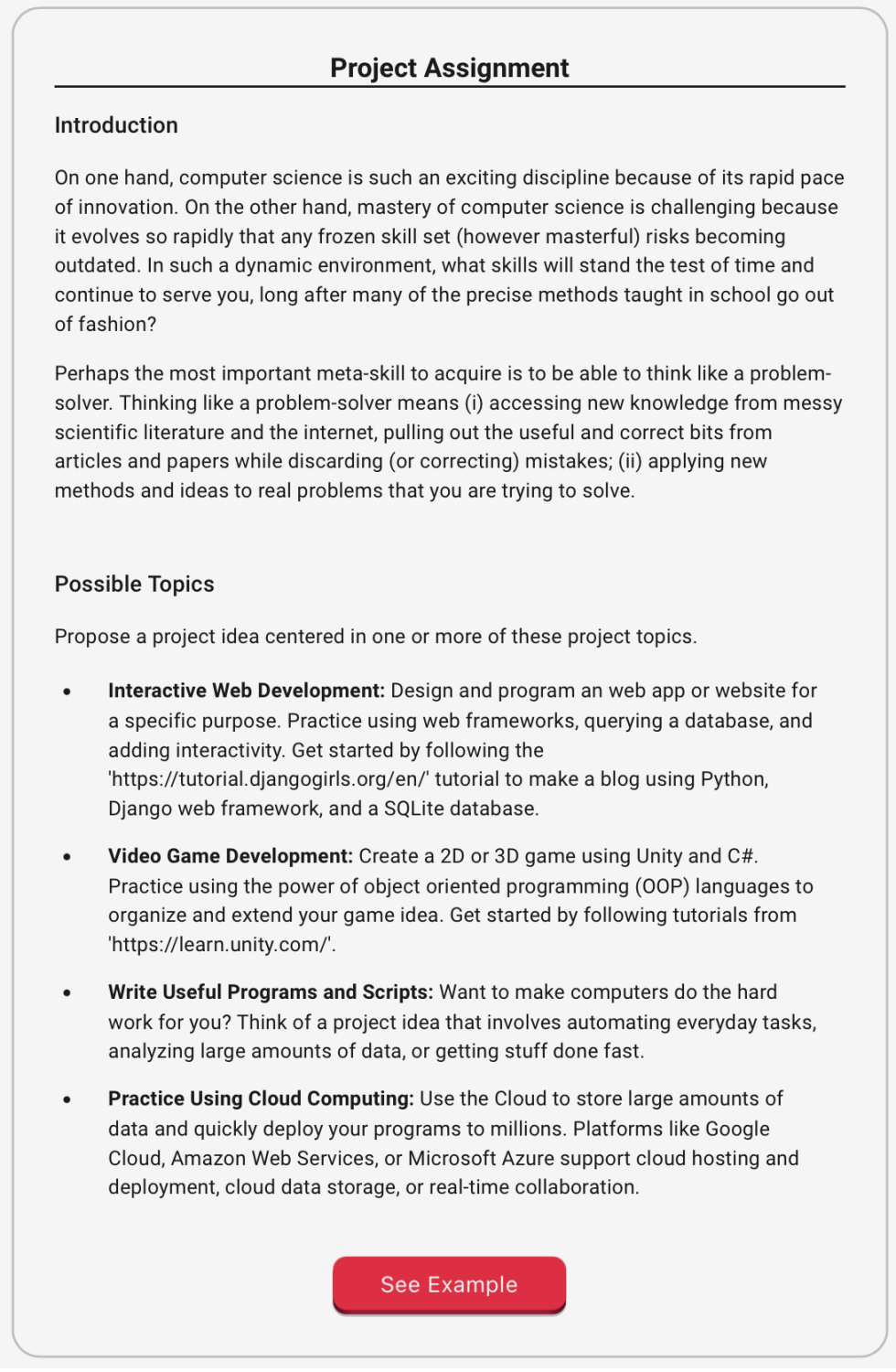

Figure 3: Introduction of project-based learning (PBL) task and possible projects.

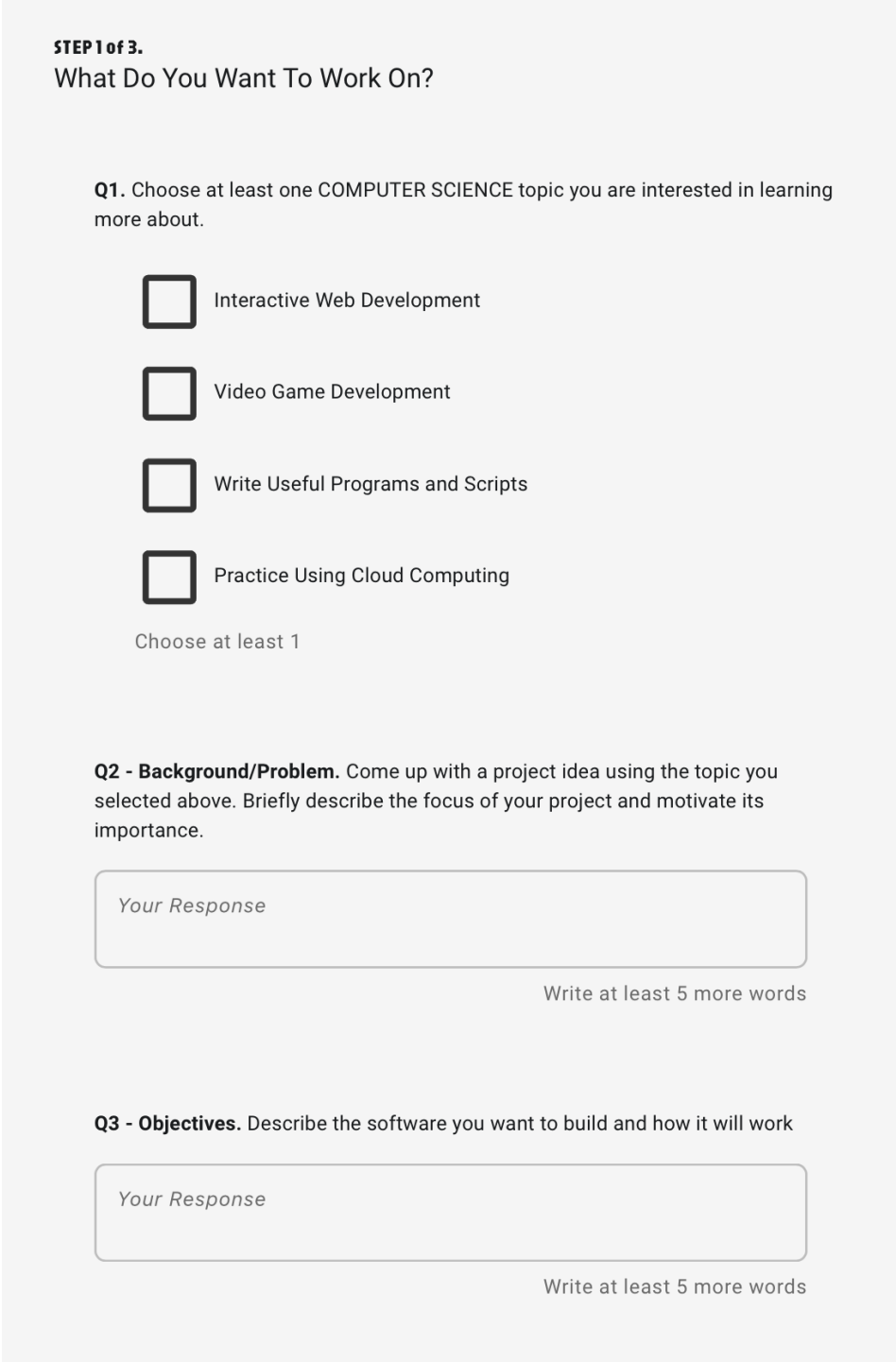

Figure 4: In the first step, the system prompts the student to specify a project and describe background and objectives.

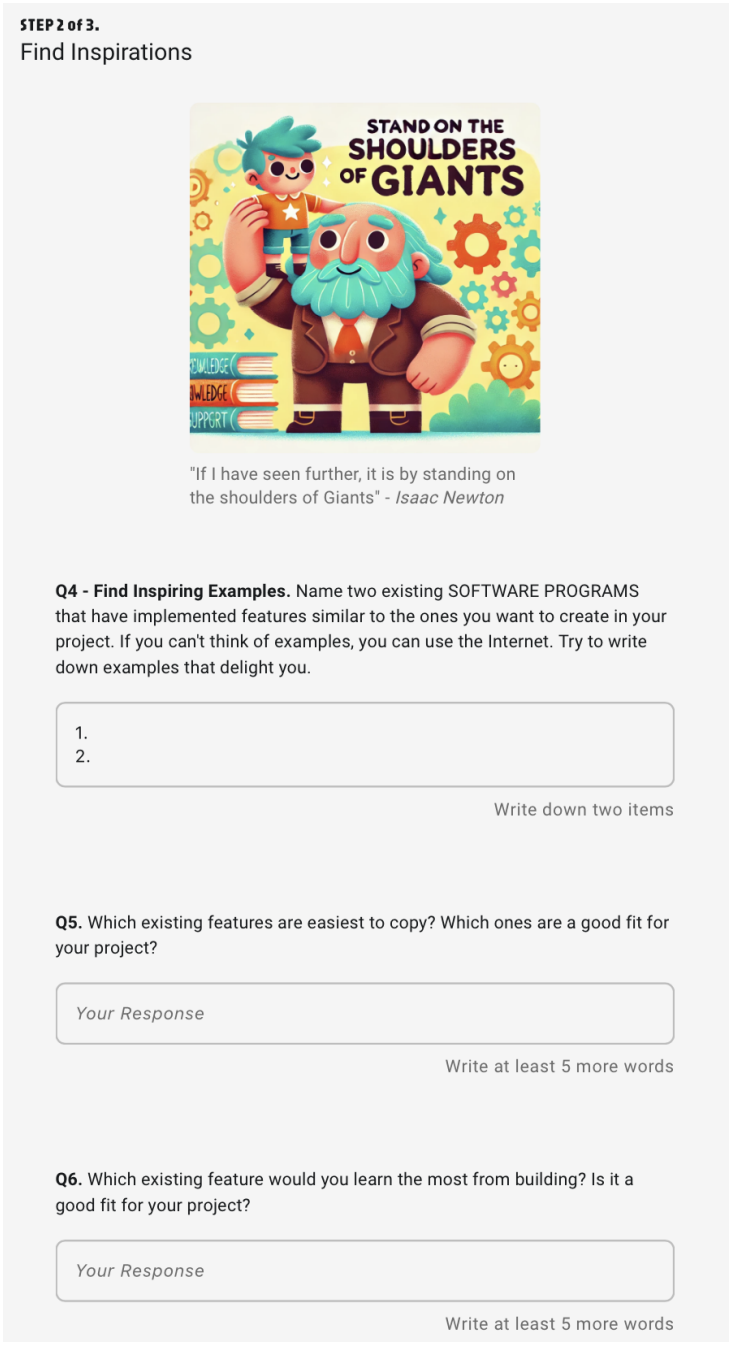

Figure 5: In the second step, the system prompts the student to find inspiration and to connect the problem to existing examples.

Figure 6: In the third step, the system prompts the student to specify their design in greater detail.

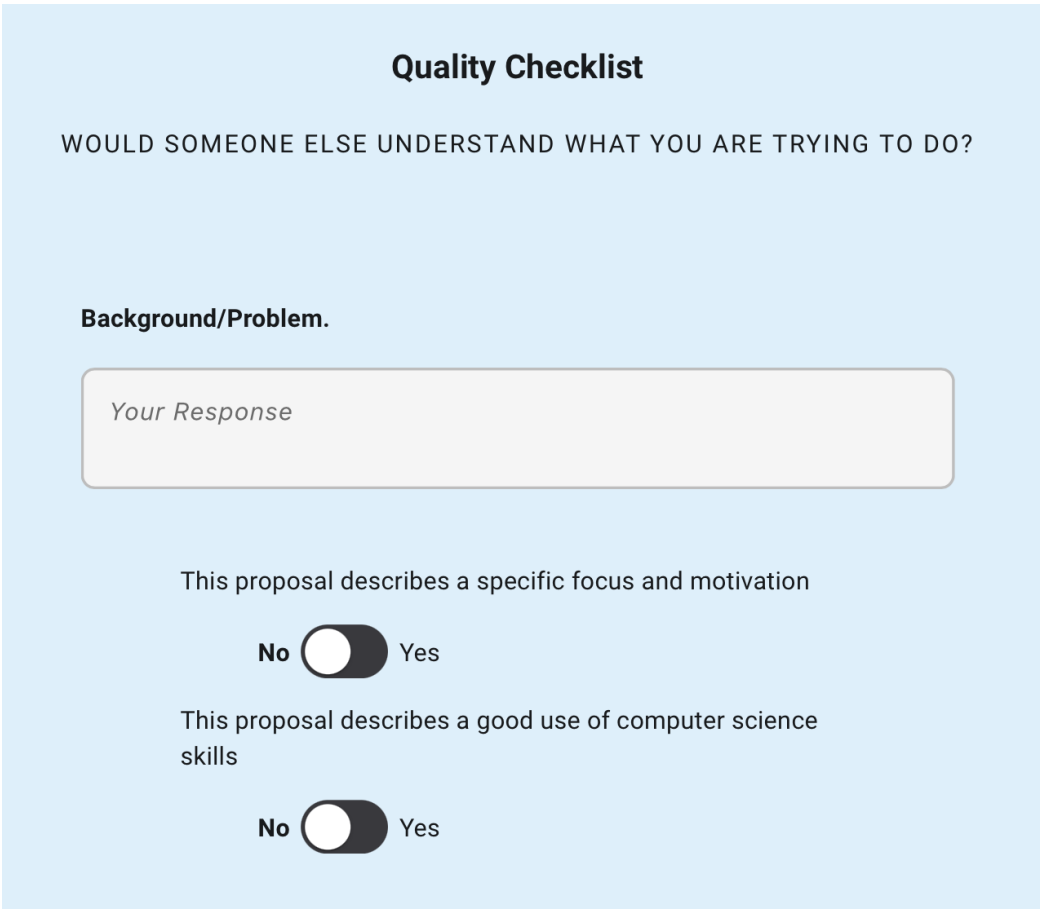

Figure 7: The system prompts the student to reflect on their proposal in terms of background and problem.

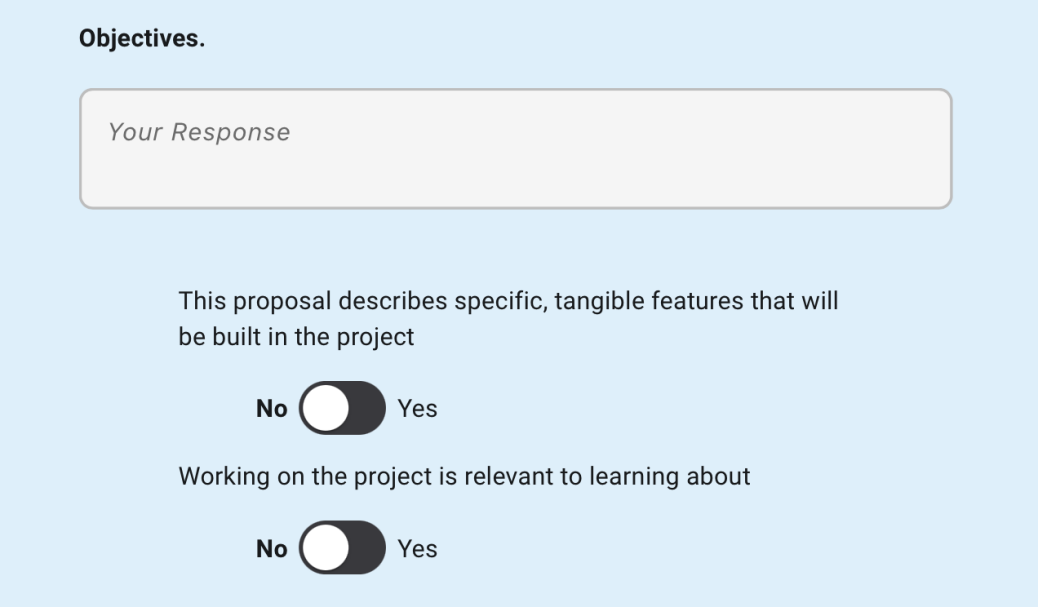

Figure 8: The system prompts the student to reflect on their proposal in terms of project objectives. After answering Q1 (choosing topic), the second question in the checklist is populated with the chosen topic's name.

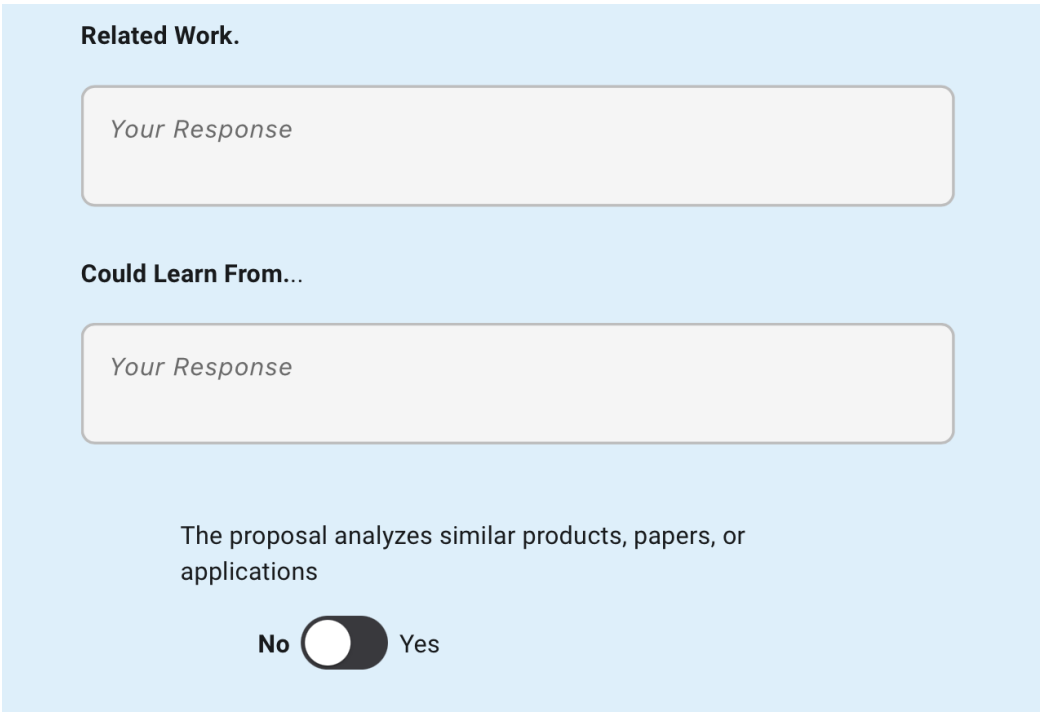

Figure 9: The system prompts the student to reflect on their proposal in terms of related work and references.

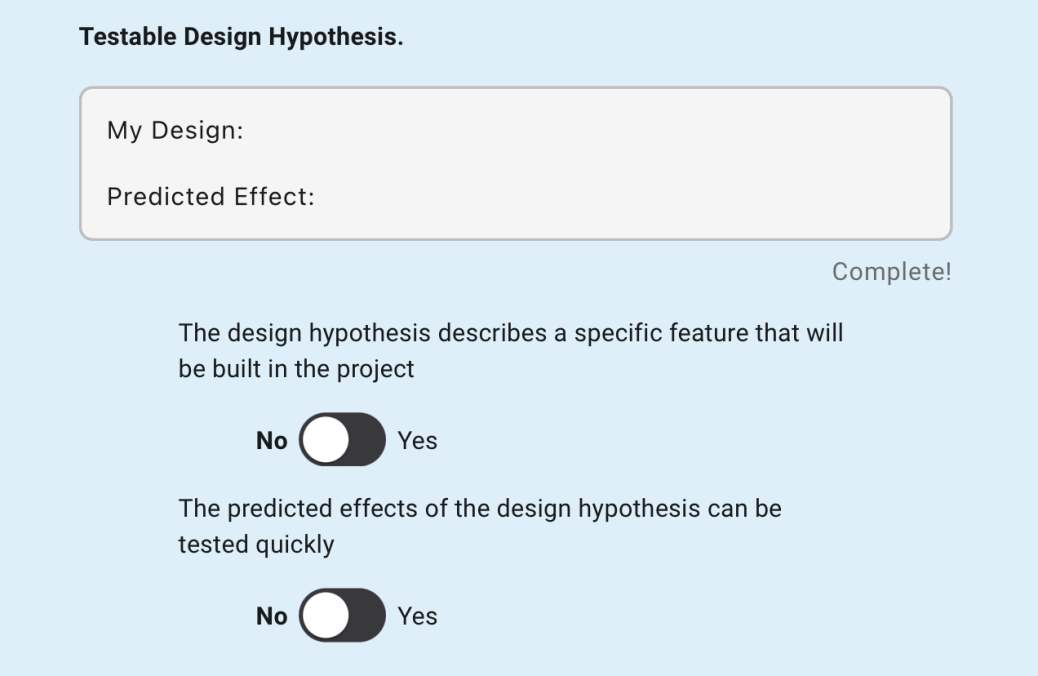

Figure 10: The system prompts the student to reflect on their proposal in terms of design hypothesis.

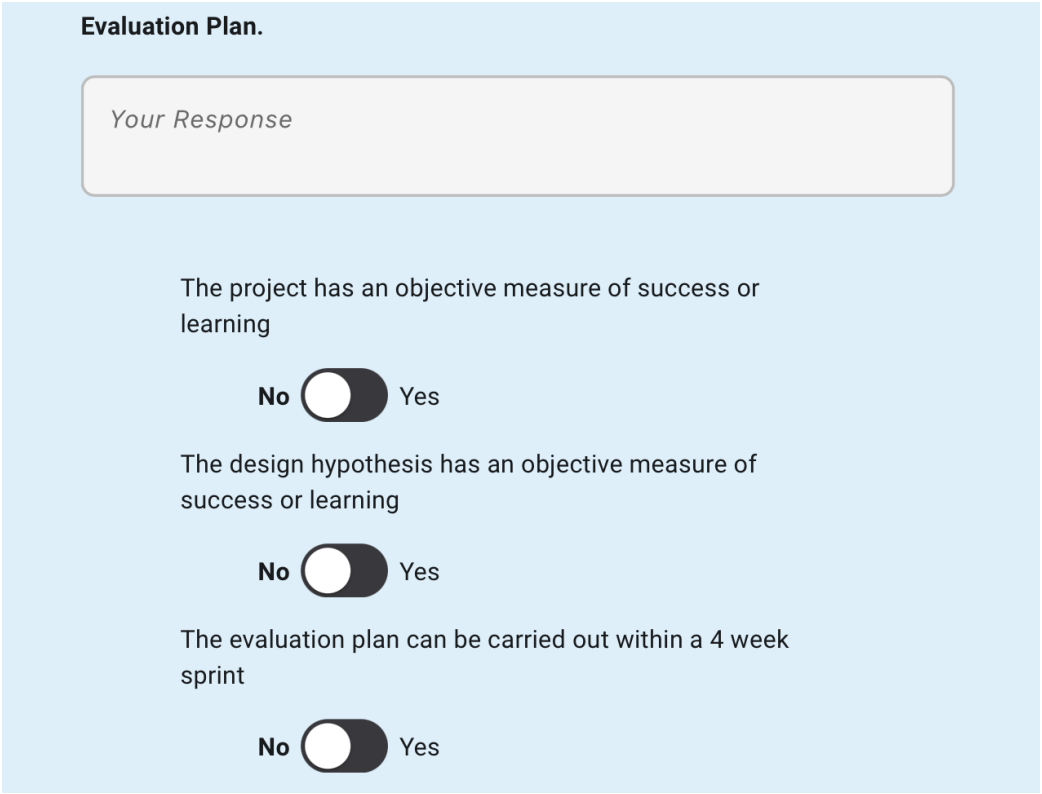

Figure 11: The system prompts the student to reflect on their proposal in terms of evaluation plan.

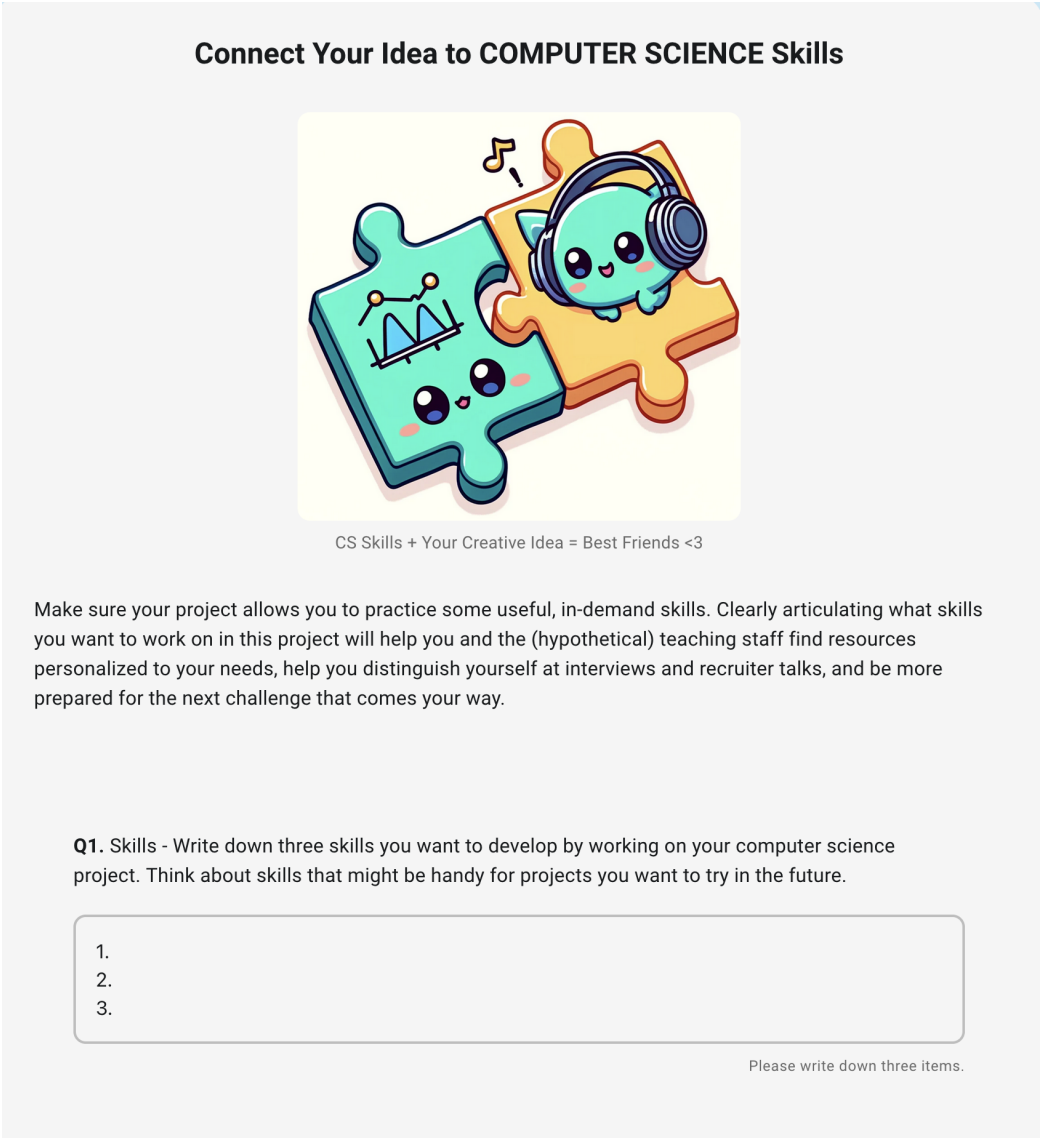

Figure 12: The system prompts the student to describe three skills they want to develop while working on their project.

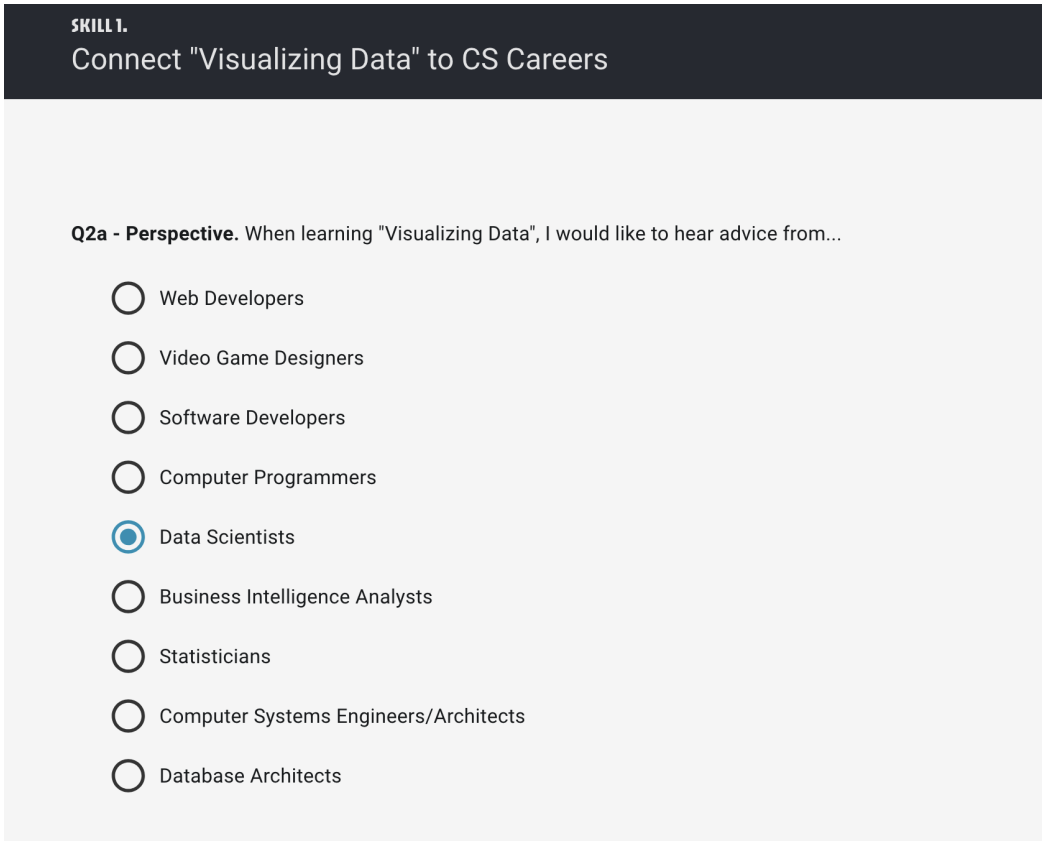

Figure 13: The system prompts the student to connect the first skill to a mentor whose perspective would be valuable.

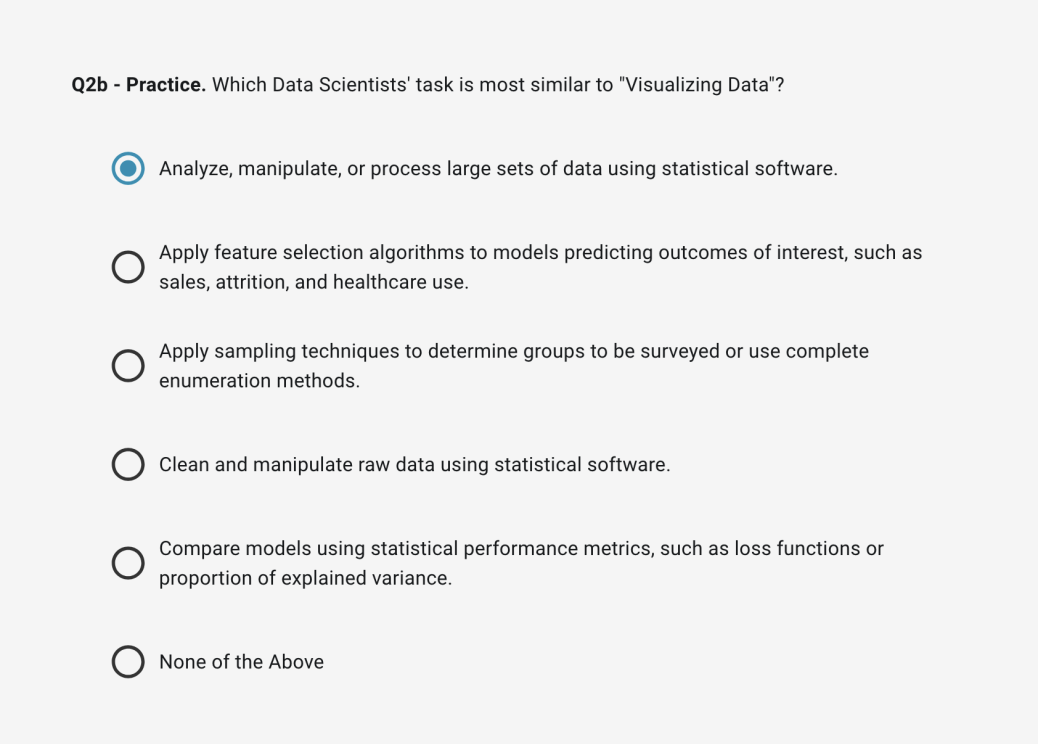

Figure 14: The system prompts the student to connect the first skill to one of the top 5 important tasks for their selected mentor's career (tasks sourced from O*Net Online).

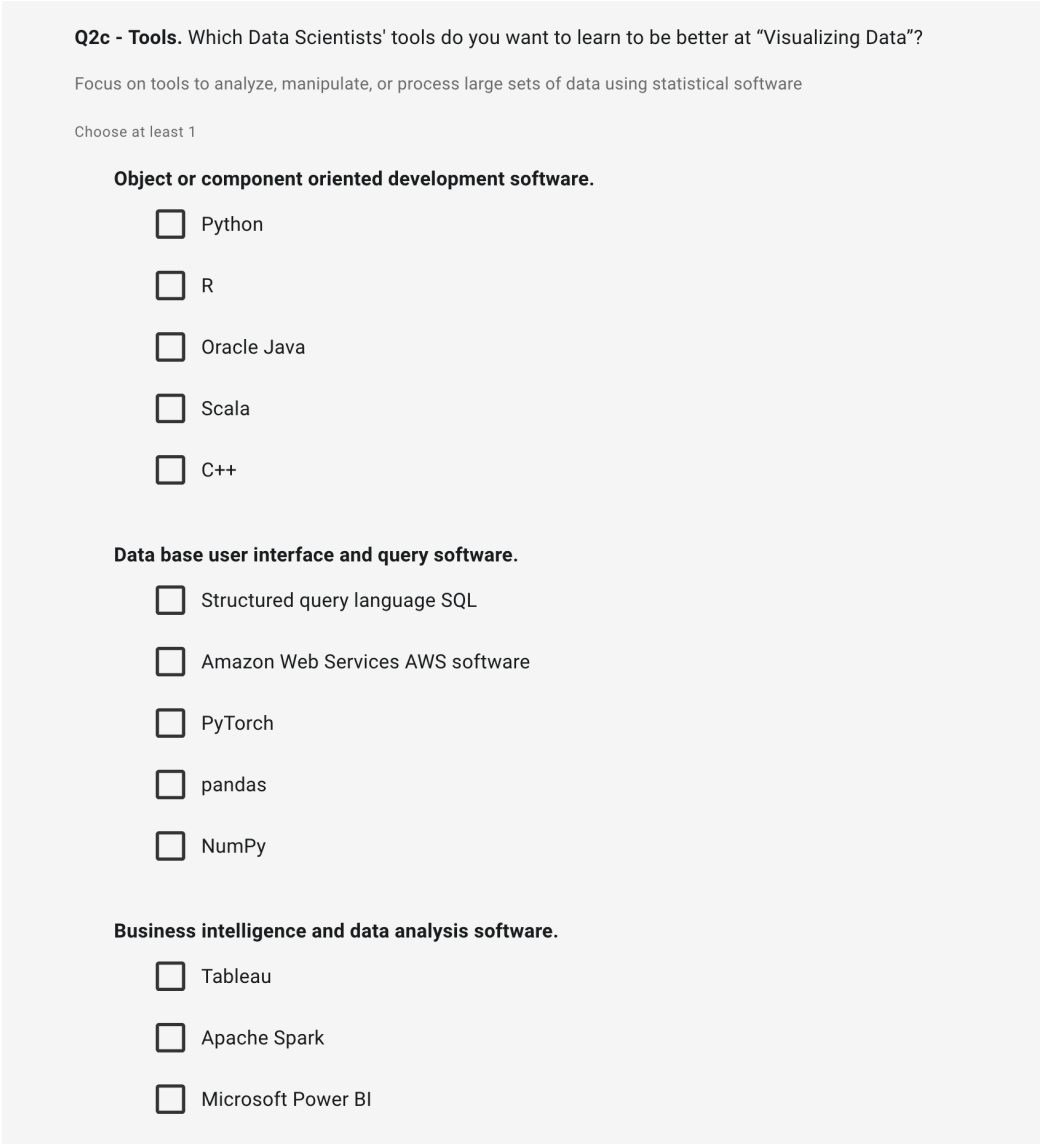

Figure 15: The system prompts the student to connect the first skill to tools and technologies associated with their selected mentor's career (technologies sourced from O*Net Online).

Appendix B. Human Expert Evaluation Rubric

Here, we provide a crowd worker's project proposal and the 23-item quality checklist that human experts and GPT-4o used to grade the project proposals. The rubric had four subtasks:

- Using the same quality checklist students used to self-assess their own work (10 items).
- Judging the appropriateness of skill-career pairings (9 items; 3 items per skill).
- An overall quality judgment question ("I would recommend a student include this project on their resume") (1 item).
- Judging the quality of skill descriptions written by students (3 items; 1 item per skill). This set of questions was asked in a separate form developed to capture factors that could explain low rater agreement on the appropriateness of skill-career pairings.

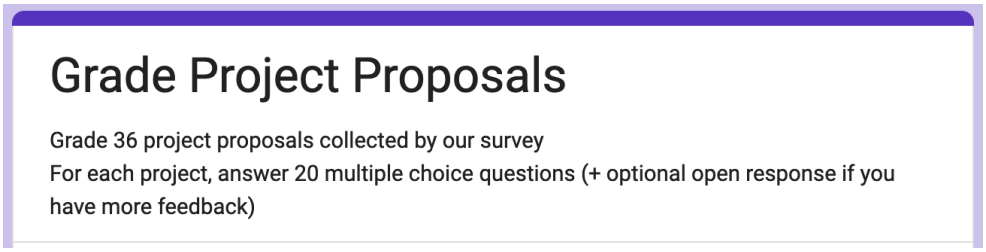

Figure 16: Evaluation task description, shown to the human experts.

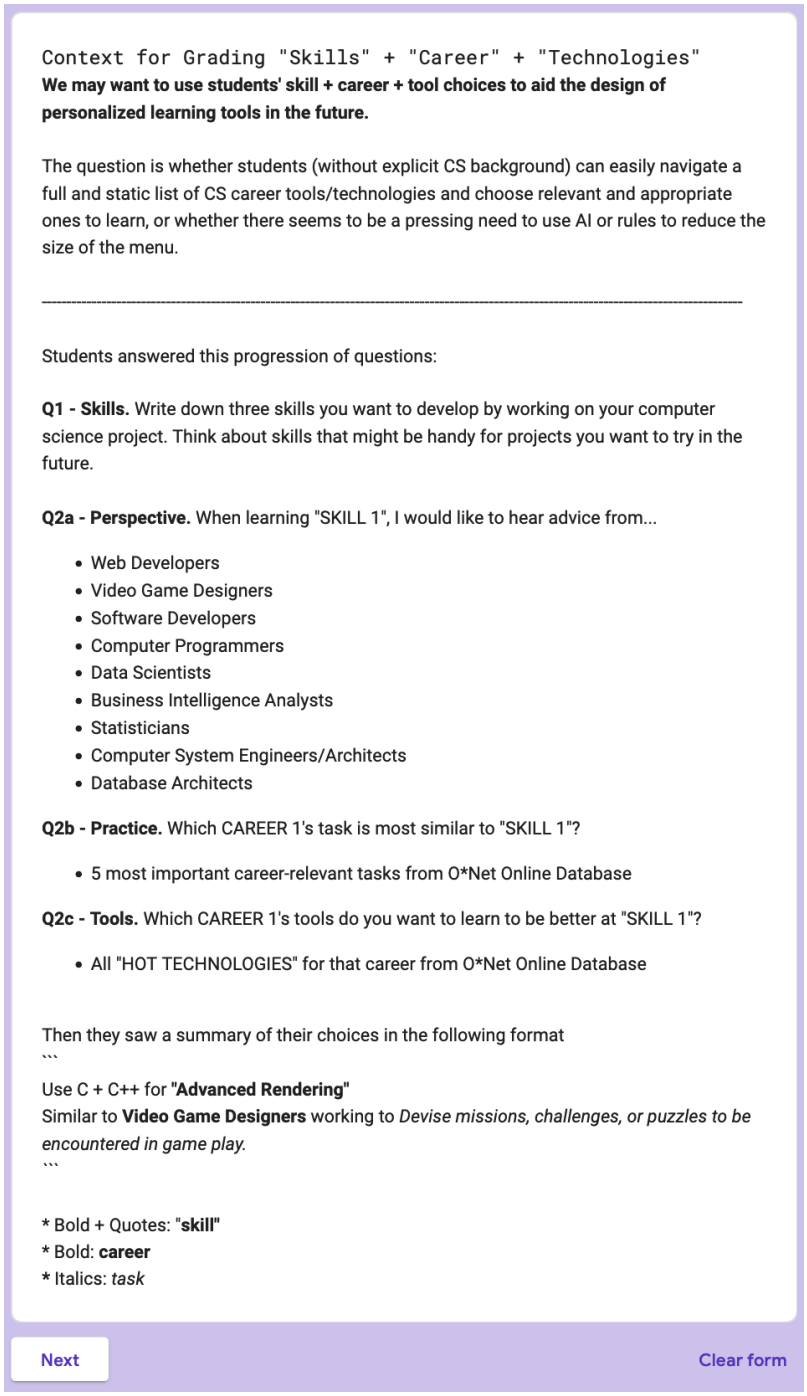

Figure 17: Instructions for evaluating project proposals, shown to the human experts.

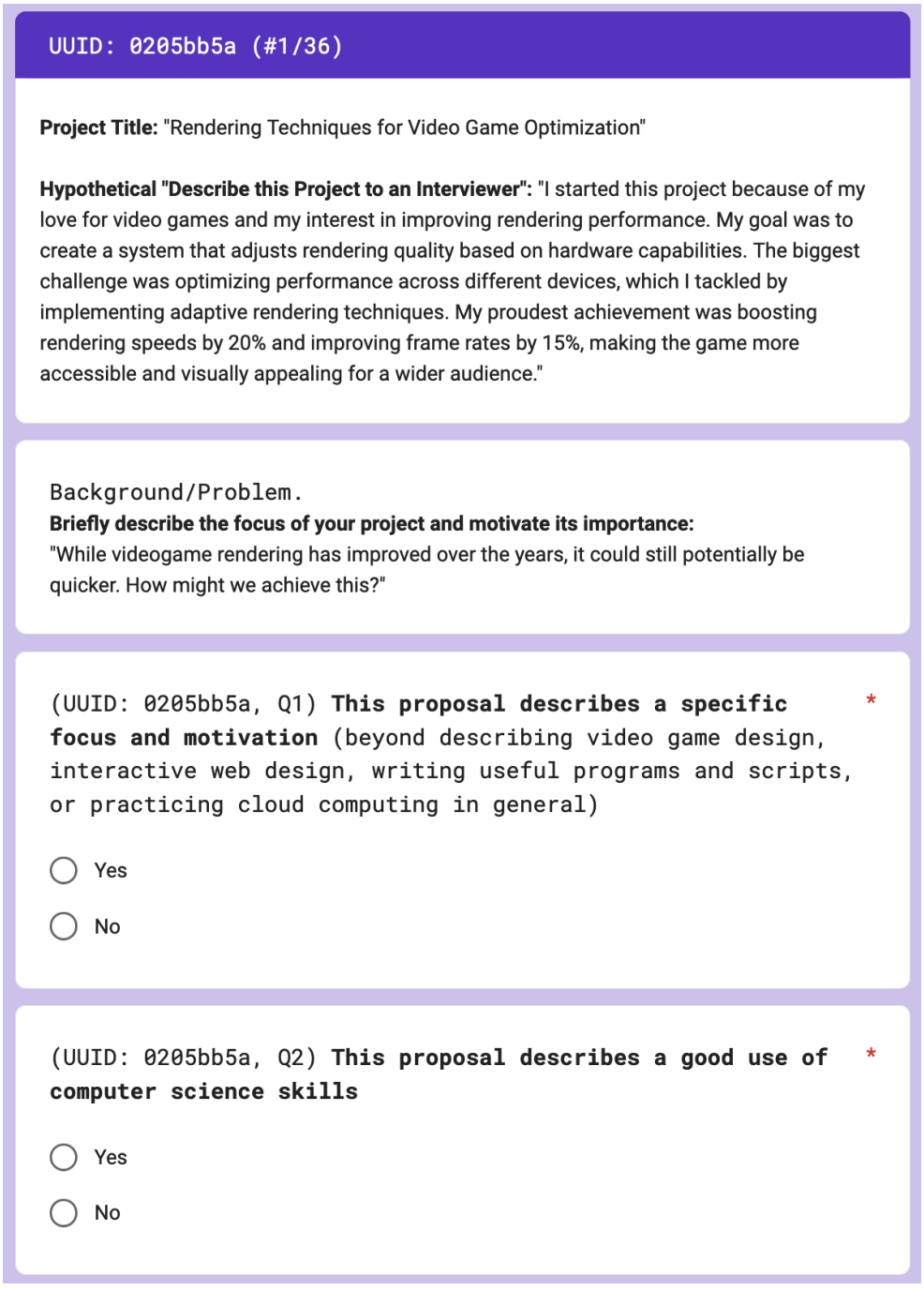

Figure 18: Items for evaluating a student's project proposal in terms of background and problem.

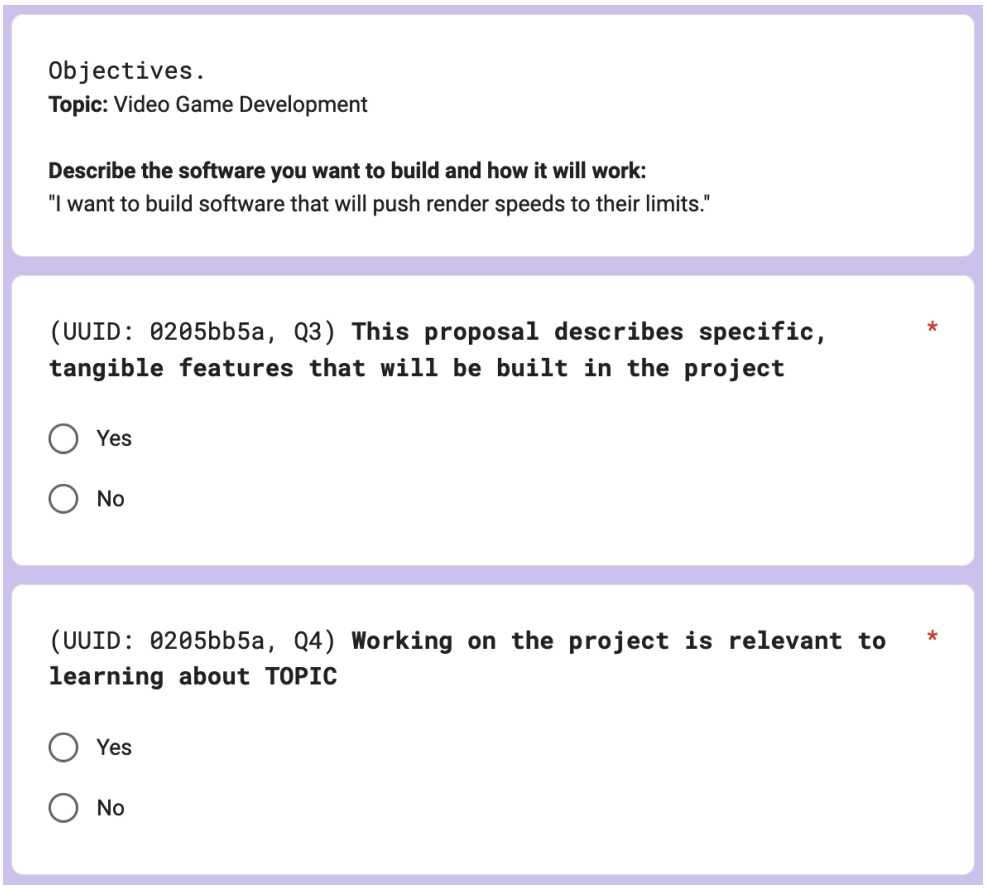

Figure 19: Items for evaluating a student's project proposal in terms of objectives.

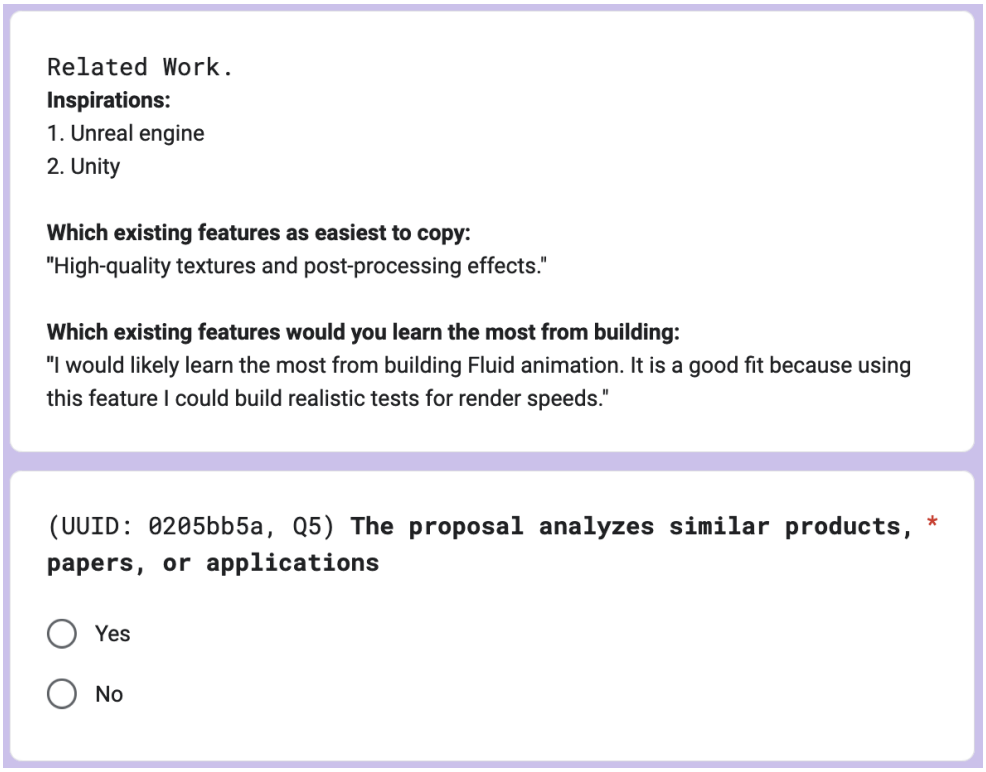

Figure 20: Items for evaluating a student's project proposal in terms of related work.

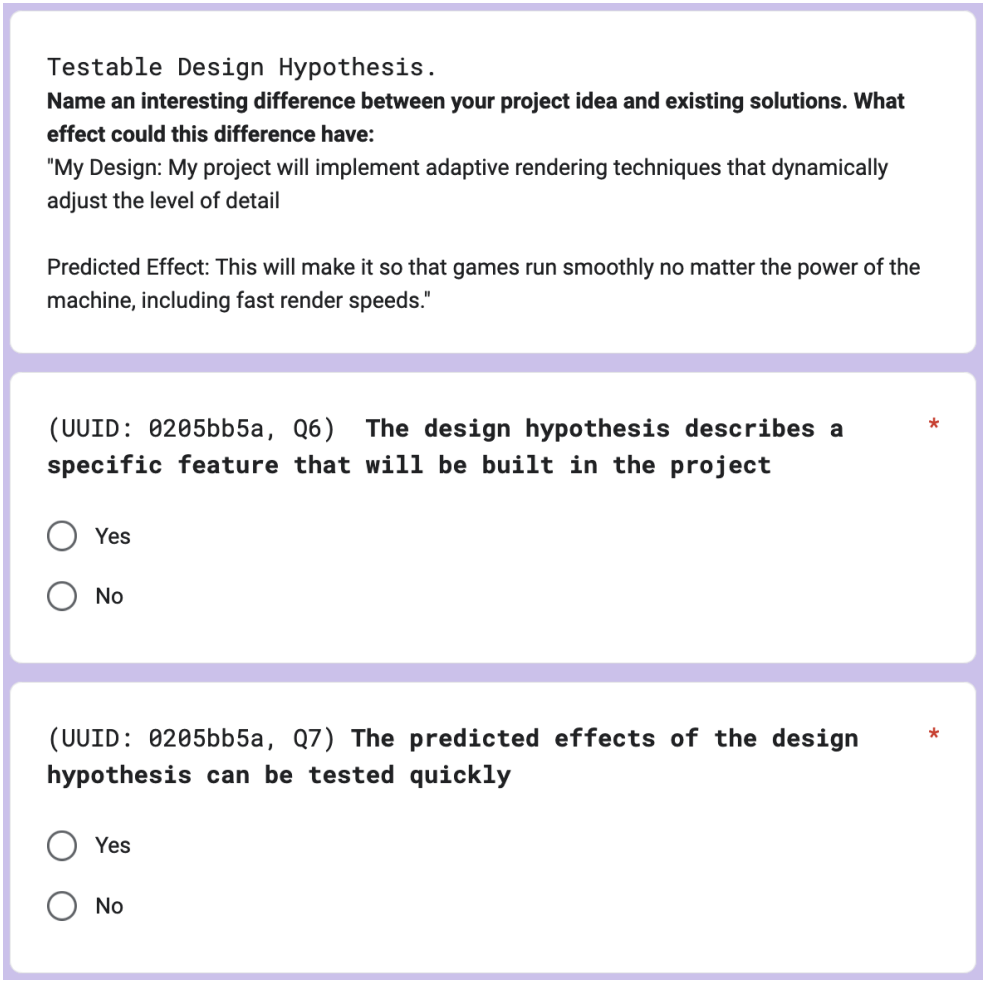

Figure 21: Items for evaluating a student's project proposal in terms of design hypothesis.

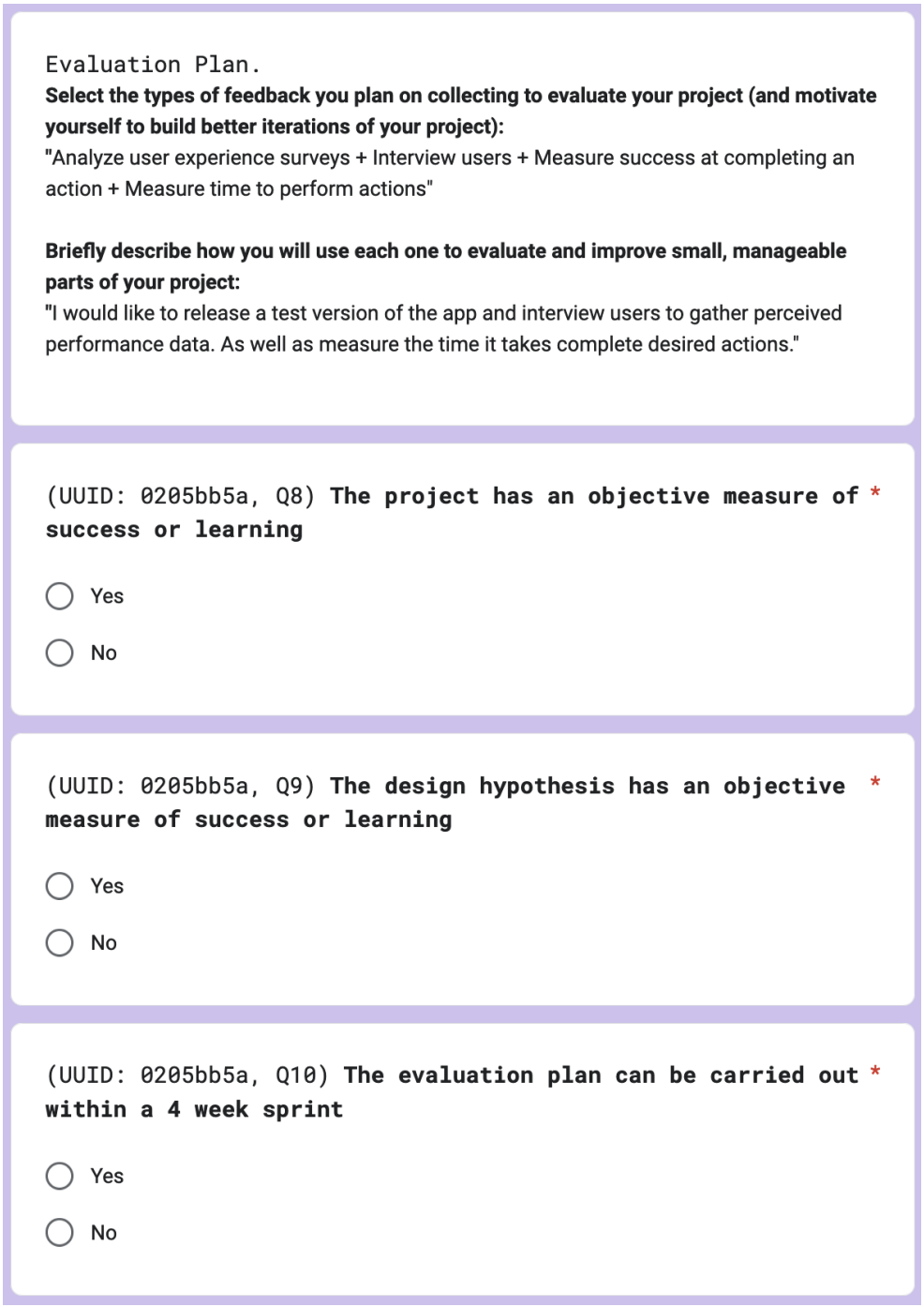

Figure 22: Items for evaluating a student's project proposal in terms of evaluation plan.

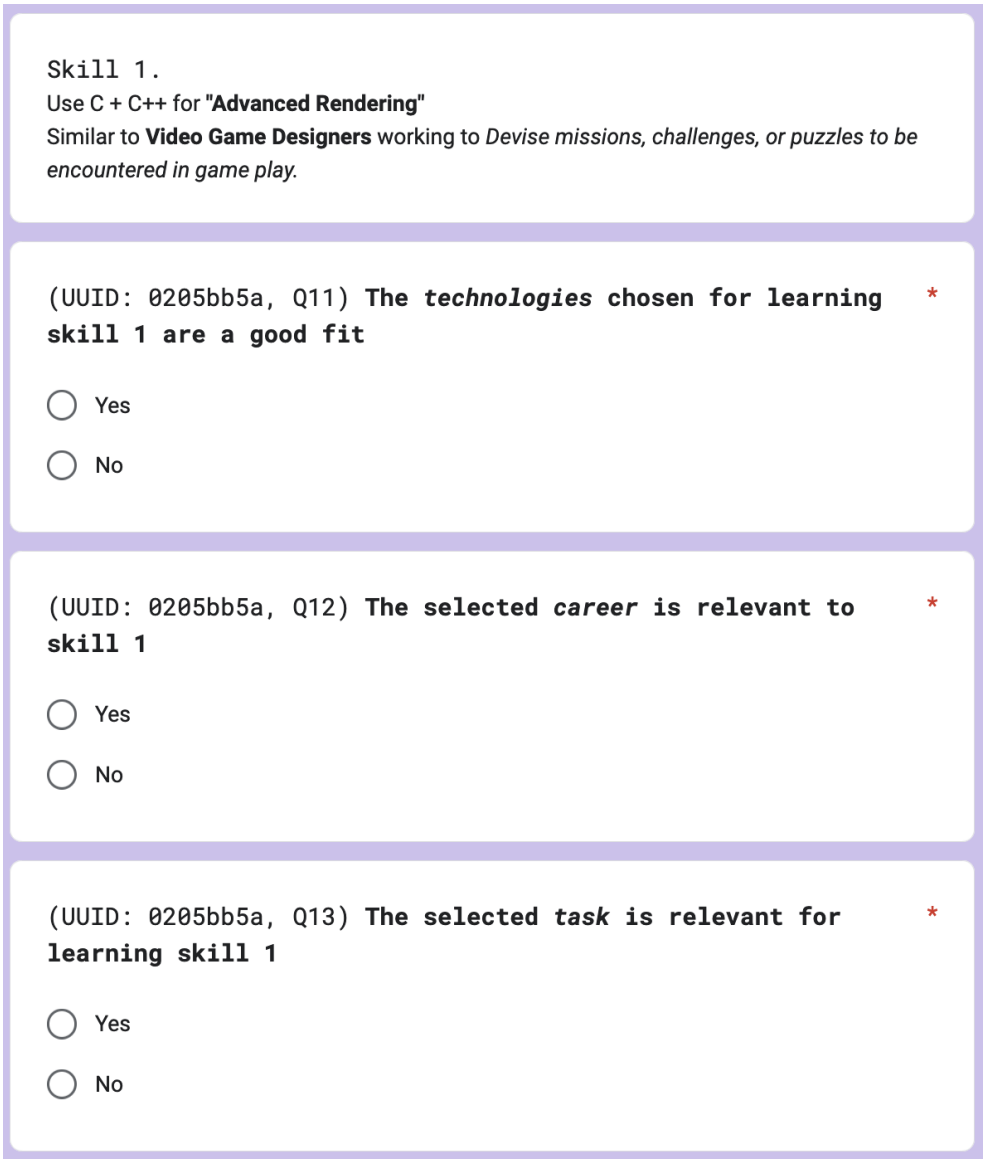

Figure 23: Items for evaluating whether student has matched their skill #1 to appropriate technologies, computer science career, and career task.

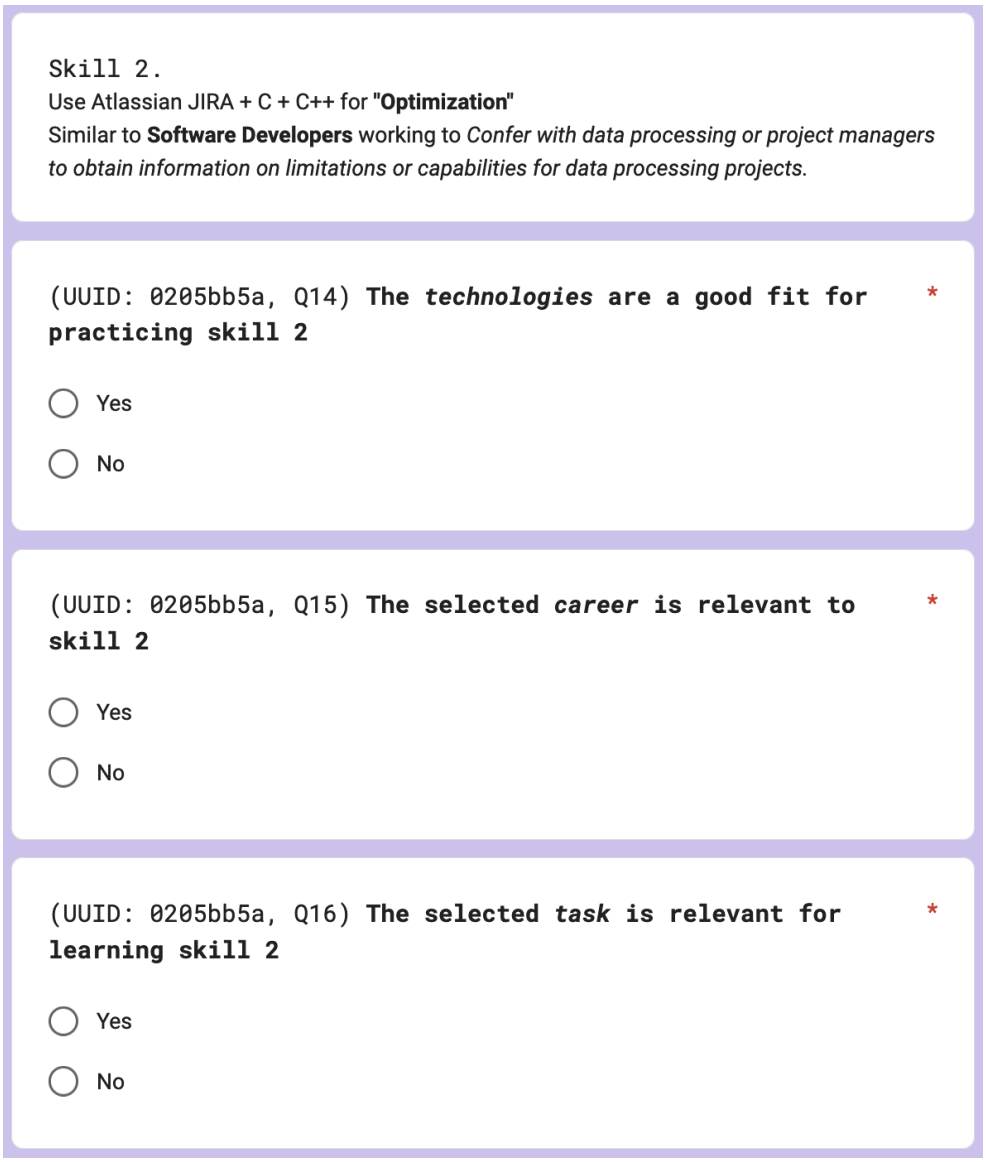

Figure 24: Items for evaluating whether student has matched their skill #2 to appropriate technologies, computer science career, and career task.

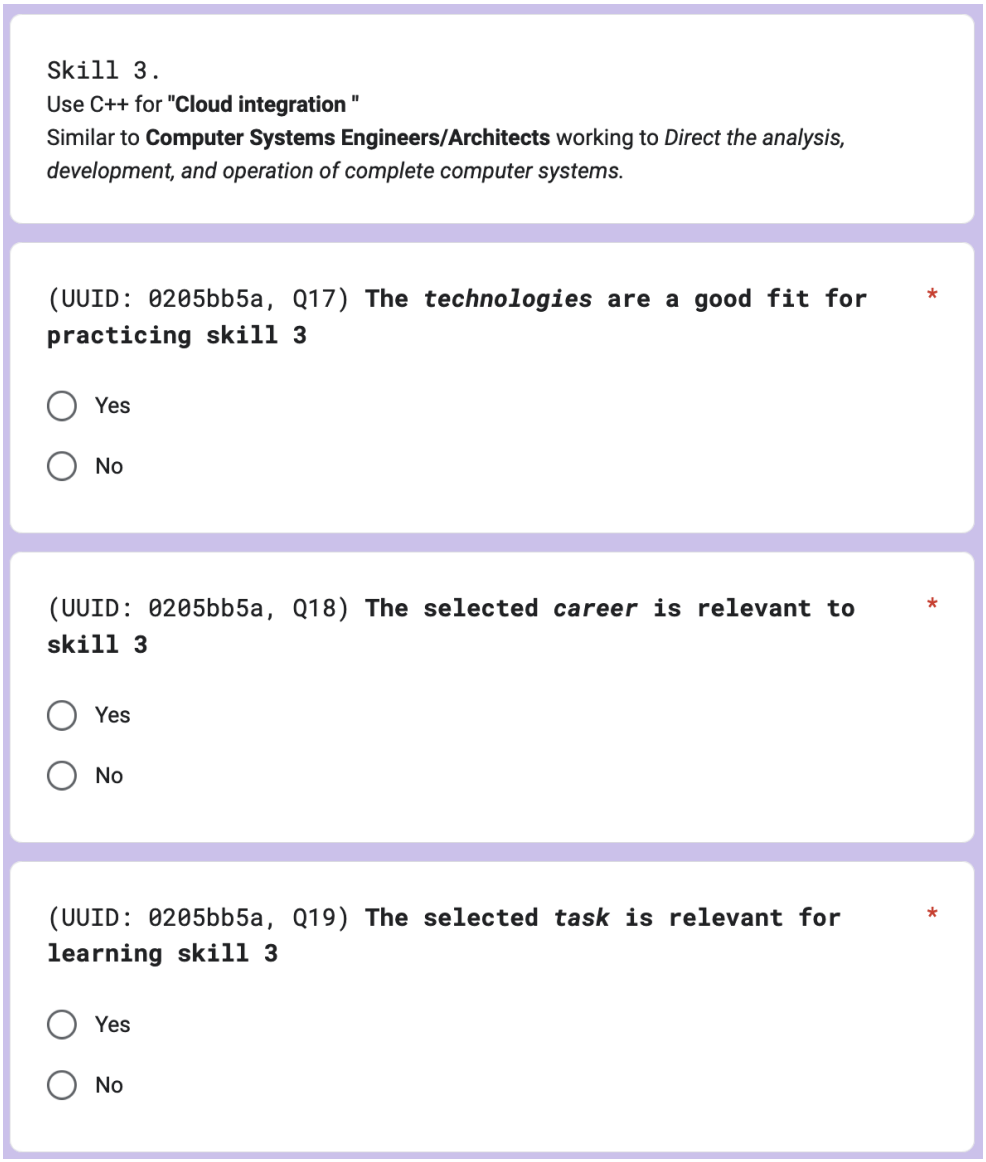

Figure 25: Items for evaluating whether student has matched their skill #3 to appropriate technologies, computer science career, and career task.

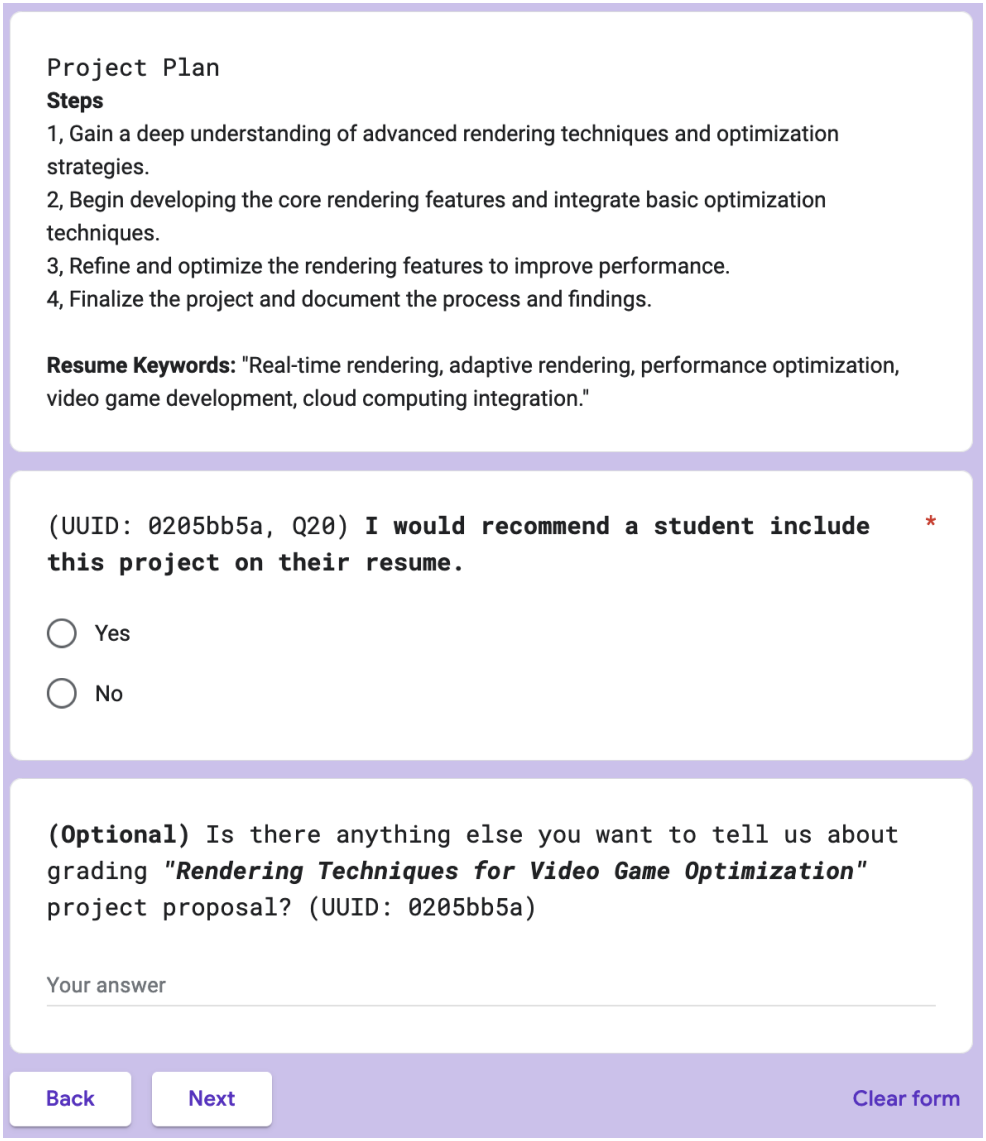

Figure 26: Item for evaluating overall quality and open-ended comments on the project proposal.

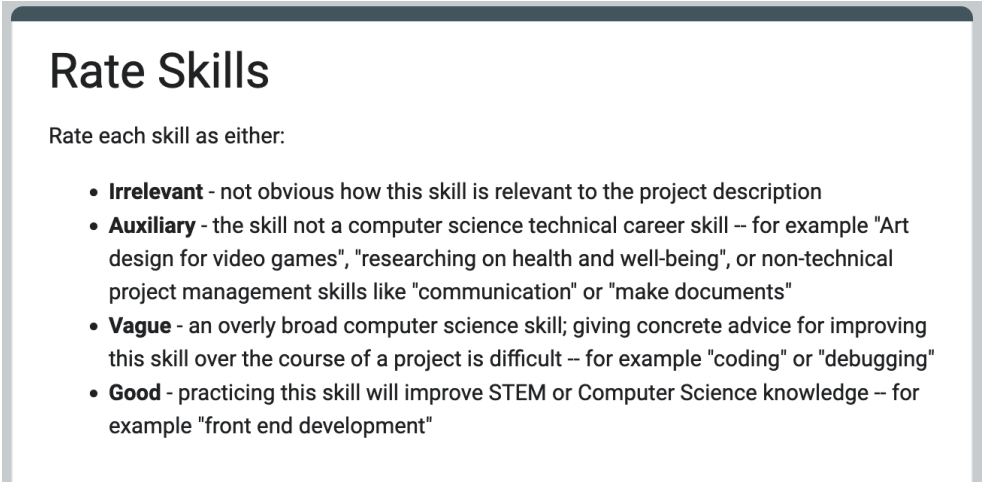

Figure 27: Instructions for classifying the skills written by students.

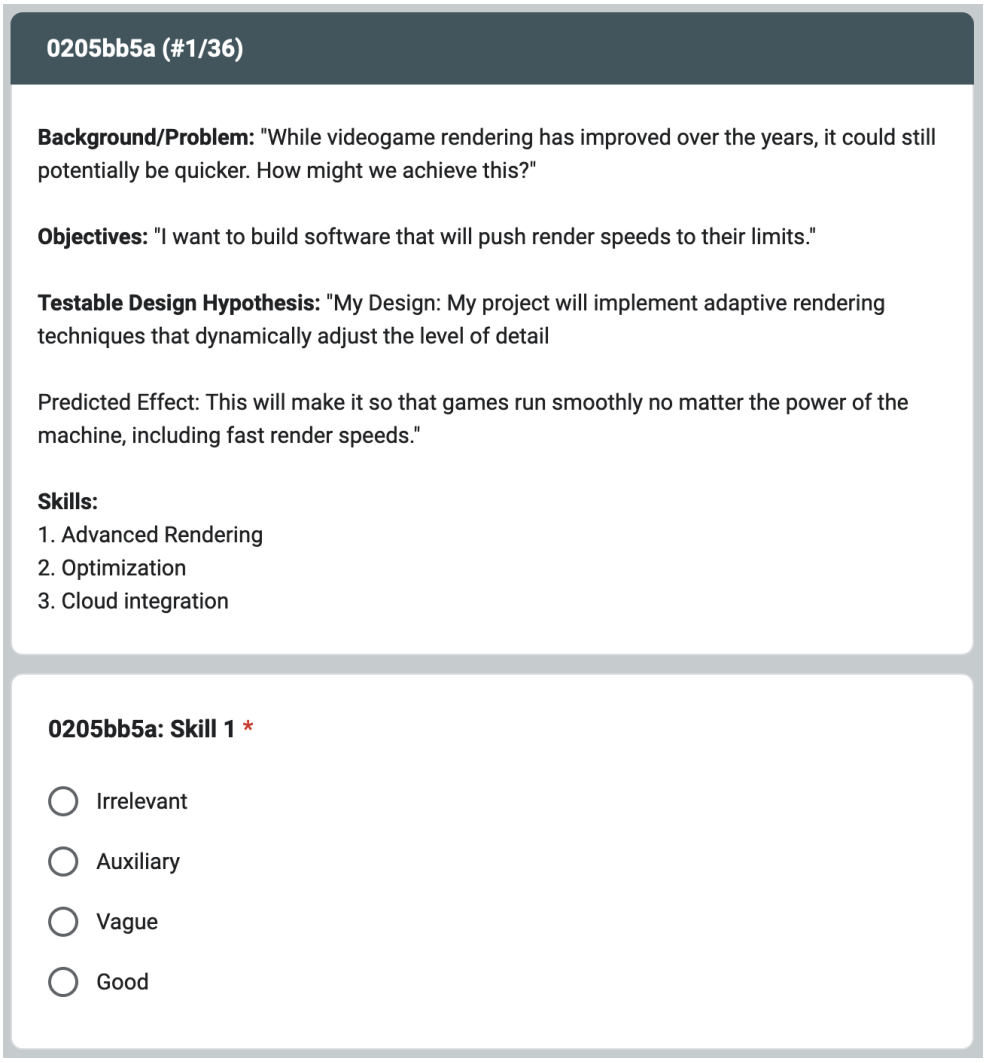

Figure 28: Project proposal and skill #1 classification question.

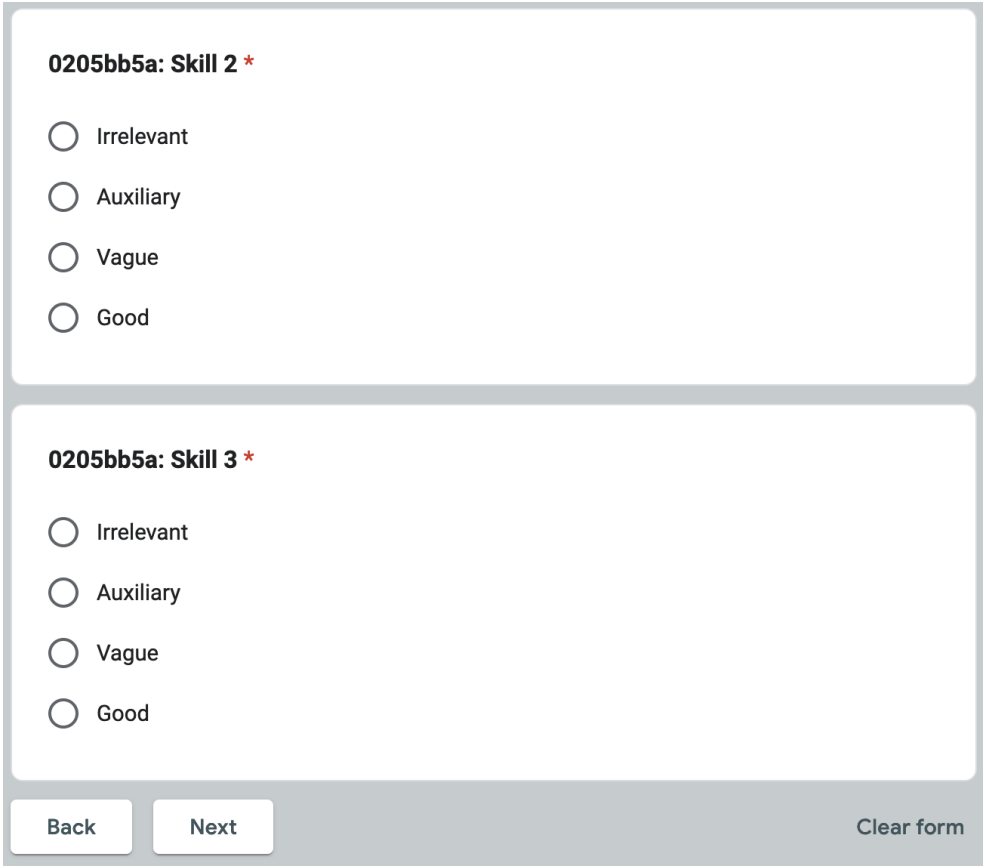

Figure 29: The skill #1 and skill #2 classification questions.

Appendix C. LLM-as-a-Judge Prompt GPT-4o

To produce ratings from GPT-4o, we call OpenAI API with the following prompt templates, populated with each project proposal's text. The rubric items are the same as the items used in the human expert grader's rubric given in Appendix B.

System Prompt - Quality Checklist & Resume Project
You are a teaching assistant. Your job is to check project proposals for quality and higher order thinking skills in Bloom's taxonomy (evaluation, synthesis). For each item in the checklist (Q1-Q11), respond with Yes or No.

User Prompt - Quality Checklist & Resume Project

Project Title: ```project_title```
Background/Problem.
Briefly describe the focus of your project and motivate its importance: ```q2_problem```
* Q1. This proposal describes a specific focus and motivation (beyond describing video game design, interactive web design, writing useful programs and scripts, or practicing cloud computing in general)
* Q2. This proposal describes a good use of computer science skills

Objectives:
Topic: ```q1_topic```
Describe the software you want to build and how it will work: ```q3_objective```
* Q3. This proposal describes specific, tangible features that will be built in the project
* Q4. Working on the project is relevant to learning about q1_topic

Related Work.
Inspirations: ```q4_inspirations```
Which existing features are easiest to copy: ```q5_analysis_copy```
Which existing features would you learn the most from building: ```q6_analysis_learn```
* Q5. The proposal analyzes similar products, papers, or applications

Testable Design Hypothesis.
Name an interesting difference between your project idea and existing solutions. What effect could this difference have: ```q7_your_design```
* Q6. The design hypothesis describes a specific feature that will be built in the project
* Q7. The predicted effects of the design hypothesis can be tested quickly

Evaluation Plan.
Select the types of feedback you plan on collecting to evaluate your project (and motivate yourself to build better iterations of your project): ```q8_feedback```
Briefly describe how you will use each one to evaluate and improve small, manageable parts of your project: ```q9_evaluation_plan```
* Q8. The project has an objective measure of success or learning
* Q9. The design hypothesis has an objective measure of success or learning
* Q10. The evaluation plan can be carried out within a 4 week sprint

Project Plan.
Steps: ```steps```
* Q11. I would recommend a student include this project on their resume.

Desired format:
* QX. {Yes | No}

System Prompt - Skill Pairing Classification

You are a teaching assistant. Your job is to check whether the skill students want to learn is appropriate for the listed career and career-specific task. The skill students want to learn is in double quotes. The mentor and the task are listed on the following line. For each item in the checklist (Q1-Q3), respond with Yes or No.

User Prompt - Skill Pairing Classification

Background/Problem: ```q2_problem```
Objectives: ```q3_objective```
Testable Design Hypothesis: ```q7_your_design```
Skill: ```skill```
* Q1. The technologies chosen for learning the skill are a good fit.
* Q2. The selected career is relevant to the skill.
* Q3. The selected career-specific task is relevant for learning the skill.
Desired format: * QX. {Yes | No}

System Prompt - Skill Classification

You are a teaching assistant. Rate each skill as either:
* Irrelevant - not obvious how this skill is relevant to the project description
* Auxiliary - the skill not a computer science technical career skill – for example "Art design for video games", "researching on health and well-being", or non-technical project management skills like "communication" or "make documents"
* Vague - an overly broad computer science skill; giving concrete advice for improving this skill over the course of a project is difficult – for example "coding" or "debugging"
* Good - practicing this skill will improve STEM or Computer Science knowledge – for example "front end development"
Respond with Irrelevant, Auxiliary, Vague, or Good.

User Prompt - Skill Classification

Background/Problem: ```q2_problem```
Objectives: ```q3_objective```
Testable Design Hypothesis: ```q7_your_design```
Skill: ```skill```
Desired format:
{Irrelevant | Auxiliary | Vague | Good}

Appendix D. Recruitment Call Posted on Prolific

Figure 30: Recruitment call displayed on Prolific crowd worker platform.