Instruct

PIDP 3230

EVALUATION OF LEARNING

Author: Yuri Tricys, Date: June 19th, 2025

KAI RATIONALE REPORT

vcc Build Hello

Introduction

The following Rational Report accompanies the Knowledge Assessment Instrument created for PIDP 3230 Evaluation of Learning, June 2025.

Validity and Reliability

In terms of validity and reliability, the knowledge assessment instrument, hereafter referred to as the 'exam', was modelled on a published Stanford web development exam from 2019. None of the question from the Stanford exam were used. Instead, the exam content was modified by the author to test against the course outcomes defined in the course profile document created for PIDP 3210, Curriculum Development.

An reproduction of the Stanford test is available here:

https://xyz-abc-show-and-tell.github.io/xyz-abc-show-and-tell.github.io/pidp/3230_evaluation_of_learning/the_stanford_test.html

The course profile is available here:

https://xyz-abc-show-and-tell.github.io/pidp/3210_curriculum_development/pidp_3210_course_profile_with_cover_3_compressed.pdf

The outcomes from the Web Development 1 course profile are as follows:

Concepts/Theories/Problems Scenarios

- Beginning level understanding of HTML
- Beginning level understanding of CSS
- Beginning level understanding of JavaScript
- Beginning level understanding of Web Design
- Beginning level understanding of GIT (or subversion, if the student already uses that)
- Beginning level understanding of Responsive Web Develoment
- Beginning level understanding of the package manager NPM
- Beginning level understanding of basic web security practices

Skills / Practical / Hands on

- 1. Select and operate a text editor to write code
- 2. Correctly install, configure and use XAMPP, LAMPP, or WAMPP, or an npm package server (https-localhost, or) and run a local server
- 3. Back up code using versioning software (GIT)
- 4. Develop a plan to create a one or more page website
- 5. Create wireframe or prototype of a one or more page website
- 6. Create a valid HTML document that includes the mandatory HTML tags and attributes as specified in the course syllabu
- 7. Create a valid CSS document that adequately styles their webpage(s) in accordance with the styling specified in their wireframe or prototype
- 8. Create an HTML contact form from scratch (connected to email using a free service such as Zapier, or Google forms)
- 9. Create drop-down menu that opens and closes when a button is clicked using either vanilla JavaScript or a plug-in (if tl are using a framework such as WordPress)
- 10. Run a lighthouse audit using Google Chrome to determine the performance efficiency of their website
- 11. Create CSS media queries to ensure their website is visible and functional on mobile devices, tablets, and desktop devices

- 12. Load and use a custom font on their website
- 13. Deploy a website to a production server (Github)

Course Outcomes

- Demonstrate a beginning level understanding of HTML and CSS
- Demonstrate a beginning level understanding of design
- Demonstrate JavaScript proficiency at the beginners level
- Demonstrate Responsive Design
- Demonstrate proficiency of the version control system GIT, at the beginners level
- Build A Simple Web Application
- Publish A Simple Web Application
- Implement Basic Web Security Practices

Intention

The intention of the exam is to ensure students study relevant material. However, the test is also intended to measure the student's success in achieving the desired outcomes and understanding the intended concepts. However, it is expected the exam questions themselves, and especially the mix of question would change when the lectures for the course are written (there is currently only a course outline).

Moreover, it is understood that should the exam be representative of the program of a given academic administration, that administration would impart some specifications, which would again see the exam questions rewritten to ensure validity and reliability in the context of the institutions overall mandate. The questions in the Stanford exam, for example, were deemed by the author as too challenging with a higher frequency of higher mental process questions. Stanford likely has an agenda that includes ensuring only students of a predetermined caliber are rewarded a given bracket of grades.

The author's goal, in Web Development Level 1, on the other hand, is to expose students to as much material as possible, and imbue each student with the practical skill to engineer websites. As such, the question selection is quite easy and the level of testing stays for the most part at 'remembering' and 'understanding and application', with only 2 questions in the 'higher mental process' category.

Best Practices Knowledge Assessment Instrument Design

The Stanford exam was quite robust, in terms of best practices, which made it easy to apply best practices to the exam.

Information and instructions are provided clearly at the beginning of the exam, so that examinees will be aware of the time allocated for the entire exam, and the time they should spend on any given question.

There is a scoring guide at the beginning of the exam that examinees can use to determine in what order they'd like to answer the questions. The scoring guide can be used to grade the exam.

There is also a problem type key provided, since problem types were abbreviated in the question format.

The exam includes an embedded answer key. A copy without the embedded key needs to be prepared to adminster the exam.

In addition, if used 'in the wild', so the speak, the exam should be amended so that the multiple choice questions are all grouped in one section of the exam.

The directions for each question are quite clear, and there is sufficient white space between items so that the students may write notes to work on any problems.

A booklet with additional paper should be supplied with the exam.

Best Practices Problem Types

The problem types for the exam were chosen to ensure the exam is easy enough. A lot of material will be covered in the course. Student's will then drill down deeper into specific concepts in later courses.
However, unlike the Stanford exam, which allows for students to bring their own notes, the exam must be completed entirely from memory. The questions then were designed to be illustrative enough of the subject material, such that students can remember material taught in the course.
Each question is given a classification heading, to help students map knowledge during encoding to a higher level framework.

It was necessary to apply new problem types, as the problem types supplied in PIDP 3230 Evaluation of Learning did not include computer science related problem types.

The following problem types included in the course resources for PIDP 3230 did not include computer science problem types, they've been added below.

Here are the problem types included in PIDP 3230

- AR: Alternate Response - answering a question with one of two possible options (e.g., true/false, yes/no).
- SA: Short Answer - providing a concise written response to a question.
- MC: Multiple Choice - selecting the correct answer from a set of given options.
- RE: Restricted Essay - writing a limited essay response within a specific scope or length.
- EE: Extended Essay - writing a more comprehensive and detailed essay response.
- GI: Graphic Item - interpreting or analyzing a visual representation of information, such as a chart, graph, or diagram.

Here are the problem types added

- CT: Code Tracing - tracing the output or behavior of a given code snippet.
- CD: Code Debugging - identifying and fixing errors in a given code snippet.
- CC: Code Completion - completing a partially written code snippet to achieve a specific task.
- CW: Code Writing - writing a complete code snippet to solve a specific problem or achieve a specific task.
- MC-C: Multiple Choice - Code - multiple-choice questions related to code snippets, such as identifying the output or the purpose of a code snippet.
- FI: Fill-in-the-blank Code - filling in the blanks in a code snippet to complete a specific task.
- PS: Problem Solving - Algorithmic - solving a problem by designing and describing an algorithm, potentially with pseudocode.
- SE: Simulation/Execution - simulating or executing a given algorithm or code snippet to understand its behavior.
- EQ: Equivalence Checking - determining whether two or more code snippets or algorithms are equivalent in terms of thei output or behavior.
- CTE: Code Trade-offs/Explain - explaining trade-offs or design decisions in a given code snippet or algorithm.
- MAT: Matching - matching code snippets or algorithmic concepts with their corresponding descriptions or outputs.

These problem types assess various skills, such as:

- Code comprehension and analysis;
- Code writing and debugging;
- Algorithm design and problem-solving;
- Understanding of programming concepts and principles.

Some of them are used in the table of specifications included with this report.

Best practices for each problem type were considered during the design of the problems. Matching problems, for example, have the premise on the left and the response on the right. There are more responses than premises. While the value for each match is not explicitly specified, the total value of the problem is divisible by the number of matches, such that the

student can infer the value of any given match. Also, there is one matching problem in which the same response is used twice, and there is no note of that for the student, which was advised against in the best practices. While that was done to underscore the importance of the 'undefined' keyword, if student's should approach the instructor with complaints, the problem can be amended.

Language use in the alternate response questions was clear, and positive when possible. The alternate response questions are not trivial.

Many of the multiple choice questions are 'select all that apply' questions, which are a subcategory of multiple choice questions, but could be considered in some cases short answer formats. However, the stems are clear, the items are stated in simple clear language, the alternatives when used are plausible, and there are many distractors.

Short answer problems are stated in such a way that only a single brief answer is necessary. Words supplied refer to content taught in the course. Direct questions are used instead of incomplete statements.

The code completion problem set is used in a way that provides a clear template and clear instructions for the student. Moreover, code completion problems include a grading guide at the problem site, which can help students understand what code they need to write. Space is provided for the students to write code. While the code completion problems are relatively simple, if problems arise in the language use, they can be amended prior to exam deployment.

Target Student Population

While the target student population isn't currently known, when it is known the exam can be amended. For example, either or questions, wherein the examinees choose to answer a question from a group of questions, could be used to test classes with high diversity in skill level. Also, coding challenges could be presented in Typescript, instead of vanilla JavaScript for programs that emphasize 'corporate professionalism' in JavaScript coding practices.

Preparing the Learner

The material in the exam is intended to be altered to reflect the content in the lectures. It is likely a future iteration of the exam will contain a more fitting selection of problems. It will be up to the instructor to ensure students are aware of the material that will be tested, or unaware depending on the discretion of the instructor and the culture of the institution.

Feedback on the exam is of course an key component in ensuring students are both motivated to continue on their respective professional journeys and well informed regarding the relevancy and importance of additional efforts required to master subject matter. The onus to ensure feedback is appropriately provided is again left to the instructor and the administration.

Table of Specifications

While the table of specifications did require creating the AI equivalent of pivot tables (AI prompting), which were used to group and categorize counts and classifications, which did require effort, the resulting table (see below) was useful in visualizing the composition of the exam, in terms of problem types, lower to higher level emphasis, and Bloom's taxonomy.

The points for the exam were modified to total to 100, so that students can easily grasp the value of each problem, and the exam was given a total of 90 minutes (the same time allocated to the Stanford exam).

The total time student's are advised to spend on each question totals 85 minutes, leaving 5 minutes for students to read the instructions.

Because the exam was built from a template, the Table of Specifications was completed afterward, however, once the lectures are written for the course, it is the table of specifications that will be amended before the exam is amended. It is quite useful.

Table of Specifications for **Web Development 1** Final Exam — Course Value: 20%

Course Outcomes	Cognitive Level			Exam	
	Remembering	Understanding and Application	Higher Mental Processes	# of items	% of exam points
GIT Understanding	**LOW** MC 1	**MEDIUM** MAT 1 SA 1		3	8 %
CSS Understanding	**LOW** MC 1	**HIGH** MC 1		2	8 %
Develop Website Plan		**LOW** MC 1		1	2 %
Web Design Understanding		**MEDIUM** MC 1		1	8 %
JavaScript Understanding	**LOW** SA 2	**MEDIUM** MC 1 **HIGH** MC 2 MAT 1	**HIGH** CC 2	8	47 %
NPM Understanding		**MEDIUM** SA 1		1	3 %
Basic Web Security		**LOW** AR 1		1	2 %
Responsive Web Development		**MEDIUM** MC 1		1	4 %
Lighthouse Performance Audit		**MEDIUM** GI 1		1	3 %
SEO Understanding	**LOW** AR 1 MC 1	**LOW** MC 2 **MEDIUM** SA 1		5	12 %
Deploy to Github	**LOW** MC 1			1	2 %
HTML Understanding		**LOW** MC 2		2	4 %

Table of Specifications Key

Instructional emphasis or time spent on the topic: High, Medium, Low

Exam Key

AR Alternate Response, **SA** Short Answer, **MC:** Multiple Choice, **MAT:** Matching Exercise, **CC:** Code Completion, **GI:** Graphic Item

Conclusion

In conclusion, the knowledge assessment instrument for Web Development 1 demonstrates robustness in its design, adhering to best practices in knowledge assessment and incorporating a range of problem types to comprehensively evaluate student understanding. The exam's structure, including clear instructions, a scoring guide, and varied question formats, ensures a reliable assessment of students' knowledge and skills. The Table of Specifications provides a clear overview of the exam's composition, aligning with course outcomes and cognitive levels. Overall, the assessment instrument is well-suited to evaluate student learning outcomes in Web Development 1.

AI Models Used in Report

The ideas, structure, writing, and editing in this report, and the knowledge assessment instrument, were performed by the author. Various AI models were used in collecting data, verifying data, and formatting various arguments.

- Meta Llama 4 Maverick
- Qwen3 235b a22b
- Microsoft MAI DS R1
- Qwen 2.5 vl 72b Instruct
- Qwen 14B
- Meta Llama Llama 3.1 8B
- Mistral Mixtral 8x7B