LESSON 87

Testing and Documentation

160 minutes (4 periods) | Applied Learning Tasks (ALTs) | Outcome 1.11 | Outcome 1.15 | Outcome 1.23 | Outcome 2.19 | Outcome 2.20 | Outcome 2.21 | Outcome 2.22

WHAT THIS LESSON DOES

In this lesson, you'll explore how to test and evaluate your project while documenting your progress. Follow step-by-step guidance on usability testing, peer reviews, and creating a structured report. Build a digital portfolio and complete a self-assessment rubric.

CURRICULUM OUTCOMES

Outcome	What the specification asks for
1.11	discuss the complex relationship between computing technologies and society including issues of ethics
1.15	consider the quality of the user experience when interacting with computers and list the principles of universal design, including the role of a user interface and the factors that contribute to its usability
1.23	reflect and communicate on the design and development process
2.19	test solutions and decisions to determine their short-term and long-term outcomes
2.20	identify and fix/debug warnings and errors in computer code and modify as required
2.21	critically reflect on and identify limitations in completed code and suggest possible improvements
2.22	explain the different stages in software testing

WHAT STUDENTS SHOULD BE ABLE TO DO

- Understand the importance of usability testing in creating user-friendly and functional interactive systems.
- Develop skills in gathering and applying feedback through peer reviews to enhance project quality.
- Master the structure and content of a comprehensive project report, including ethical considerations.
- Acquire the ability to compile and present work effectively in a digital portfolio.
- Gain proficiency in self-assessment to evaluate and reflect on personal project outcomes.

BEFORE THE LESSON

- Confirm each team's system actually runs end to end before the session, database to frontend. Testing a broken build wastes the class.
- Decide and share where the digital portfolio lives (school drive or platform) and check students can upload to it.
- Prepare a short peer review checklist so feedback is structured, not just 'looks nice'.
- Have the self-assessment rubric visible or copied so students can score against the five categories.

Testing and Documentation

HOW THE LESSON RUNS

- Introduction
- Usability Testing
- Peer Review Activity
- Report Structure Guidance
- Digital Portfolio Upload
- Self-Assessment Rubric

TEACHING MOVES

- 2. Usability Testing. Insist testers are watched silently, not coached. Tell students to note where a peer hesitates or clicks the wrong thing, and to fix confusing navigation before slow loading. The point is real users, not friends being polite.
- 3. Peer Review Activity. Give reviewers one thing to hunt for each: navigation, an ethics concern like data privacy, and one strength. Structured feedback beats vague praise. Have the author record what they will change, not just what was said.
- 4. Report Structure Guidance. Model one section briefly, ideally the ethics part, since students skip it. Remind them the report shows problem-solving and the design journey, so 'what went wrong and how I fixed it' earns marks, not just the finished screens.
- 5. Digital Portfolio Upload. Check the portfolio tells the whole journey: planning, testing evidence, ethics, iterations, not just final files. Have students name files clearly so an evaluator can navigate without them present.
- 6. Self-Assessment Rubric. Ask students to justify each score in one sentence, not just circle a number. The reflection on areas for improvement is the learning, so push for one concrete next action per weak category.

WHAT TO WATCH FOR

- Students assume a system that looks right and works for them is therefore usable and correct. Have them watch a peer use it cold. The gap between 'I know where to click' and a first-time user's confusion is the lesson.
- Treating the report as a description of the finished product rather than evidence of process and ethics. Point out that marks come from design decisions, testing findings and ethical reasoning. A screenshot with no explanation scores little.

DIFFERENTIATION

- Emerging: If a student's system is not running, pair them to test a working one first so they still practise usability testing and review while they fix their own. Give a sentence-starter template for the report so writing does not block the documentation task.
- Developing: Have them explain one change they made purely because of peer feedback, and why, before moving on. Ask them to identify which rubric category is weakest and name one specific fix.
- Proficient: Push them to document a genuine ethical trade-off (data privacy versus functionality) as portfolio evidence they could reuse in an Applied Learning Task. Have them run a second test round to show iterative improvement, not just a single pass.