Web Development

- L1 Standards - Web Development
- L2 Standards - Web Development

L1 Standards - Web Development

Benchmarks

#	Benchmark	What to look for
1	Functional Contribution	At least one merged PR to a YETI project (Juice, scouting site, tba-sdk, Basecamp, or equivalent) involving JavaScript/TypeScript logic. CSS-only or single-line changes do not count.
2	Code Authorship	Can explain what their contribution does and why it was structured that way. Walk through the code with them — if they cannot explain it, do not pass.
3	JavaScript Fundamentals	Can read and write variables, functions, loops, conditionals, and async/await without looking them up. Test live if there is any doubt.
4	Project Literacy	Understands the structure of the project they worked in — what the major folders and files are for and how the app fits together. Ask them to orient you.
5	Local Development	Can set up a local dev environment and run the project. Ask how they got it running the first time.
6	Code Consistency	Followed existing code style and patterns without repeated correction. Check the PR diff directly.
7	Guided Execution	Completed their contribution with mentor guidance. Independence is not required at L1 — verify that the scope was appropriate for a first contribution.

Assessment Process

1. Portfolio Review

Ask the student to share their portfolio before the conversation. At L1, this is a link to at least one merged PR or deployed feature in a YETI project.

Look for:

- Involves JavaScript/TypeScript logic (not just CSS or markup changes)
- Student is the primary author
- Scope appropriate for a guided first contribution — a feature, a bug fix, a UI change

If the contribution is CSS-only or a single-line change, it does not meet the bar — ask the student to ship something functional before scheduling the assessment.

For members — how to maintain your portfolio: Your portfolio at L1 is a link to your merged PR(s). Keep a note on what you built and what guidance you received. Before your assessment, be ready to walk an evaluator through what you shipped and why.

2. Structured Conversation (15–20 min)

Walk through their contribution together. The conversation serves as the competency check — if they can explain their own code, authorship is confirmed. Syntax recall is not the bar.

Sample prompts:

- "Tell me about the feature you shipped. What problem does it solve?"
- "Walk me through what this code does." (pick a specific section from their PR)
- "Why did you structure it this way? Did you consider anything else?"
- "What would happen if you removed this part?"
- "How did you set up and run the project locally?"
- "How did you know when the feature was working?"

If the student cannot explain code attributed to them, do not pass.

3. Reliability Check

Ask a mentor or student who worked with them directly this season:

- Did they follow through on tasks assigned to them?
- Did they communicate when they were stuck or something slipped?

Outcome

Pass: student meets all 7 benchmarks. Record in the shared assessment sheet (assessor, date, level, notes) and in Basecamp when that feature is available.

Not yet: name the specific gaps with concrete feedback. "You hit 6 of 7. Benchmark #1: the CSS change you submitted does not meet the functional scope bar. Ship a feature that involves JavaScript logic and come back."

L2 Standards - Web Development

Benchmarks

#	Benchmark	What to look for
1	Independent Contributions	Has shipped multiple features with minimal mentor guidance. Ask how much direction they needed and check PR history.
2	Code Review	Has reviewed multiple PRs with meaningful, specific feedback. Check the review comments — approvals without comment do not count.
3	Feature Scoping	Given a task description, can break it into steps and ship it. Ask them to walk through how they planned their most recent contribution.
4	Code Quality	Writes code others can read — consistent with project conventions, well-named, not needlessly complex. Check the diff directly.
5	Debugging	Has investigated and resolved a problem they did not introduce. Ask them to describe a specific debugging session in detail.
6	System Understanding	Can describe how their contributions fit into the larger codebase — what a feature touches and what depends on it. Test with specific questions about their own PRs.
7	Implementation Tradeoffs	Can explain what they built, what they considered and rejected, and why. Look for judgment, not just execution.
8	Portfolio Growth	L2 work is meaningfully more complex or more independent than L1 work. Compare the two directly — if the bar is ambiguous, it is not a pass.
9	JavaScript Fluency	Handles async patterns, module/component structure, and data flow without looking up fundamentals. Test live if there is any doubt.

Assessment Process

1. Portfolio Review

Ask the student to share their portfolio before the conversation. At L2, this should include at least two merged contributions showing independence and growth over L1 work, plus links to PRs where they left code review feedback.

Look for:

- At least two contributions, each more than minor changes
- At least one completed with minimal mentor involvement
- Multiple PRs reviewed as reviewer, each with substantive feedback
- Code quality consistent across contributions — not polished in one PR, messy in another

For members — how to maintain your portfolio: Include your merged PRs and links to PRs you reviewed. For each contribution, add a brief note on what it was and how much guidance you had. Your code review comments are part of your portfolio.

2. Structured Conversation (15–20 min)

Walk through their most complex contribution and their review history. The bar at L2 is explaining what their code does and the reasoning behind the choices they made.

Sample prompts:

- "Walk me through your most recent contribution. What guidance did you get, and what did you figure out on your own?"
- "You handled X this way — what did you consider and reject before landing here?"
- "Tell me about a PR you reviewed. What feedback did you leave and what happened?"
- "How does [their feature] fit into the project? What does it depend on? What depends on it?"
- "Describe a problem you debugged that you did not introduce. How did you approach it?"
- "What would break if you changed X?"

A student who cannot explain the reasoning behind their own structural choices is not at L2.

3. Peer Signal

Ask 2–3 teammates who have worked with this student directly:

- Has this person reviewed your code? Was the feedback useful?

- Have you worked through a problem together? Can they carry their own weight?
- Can you name something they helped you understand or improve?

Outcome

Pass: student meets all 9 benchmarks and peer signal is positive. Record in the shared assessment sheet (assessor, date, level, notes) and in Basecamp when that feature is available.

Not yet: name the specific gaps with actionable feedback. Common ones: code review participation (assign them to review an open PR now), independence (needs one more unguided contribution), system understanding (schedule an architecture walkthrough first).