How this course is tested

In PHP

A Laravel course that ships code samples needs more than green unit tests on the LMS. The examples themselves have to compile. This site treats the Rust fences the same way: CI runs them.

In Rust

Four jobs watch the course:

1. **snippets** — `scripts/check-snippets.py` walks every lesson markdown file. Fences marked `rust` with a `fn main` must build and run. Fences marked `compile_fail=E0xxx` must fail with that error code. Unmarked fragments are listed in a report so they can be labelled over time.
2. **crates** — the companion repo runs `cargo fmt`, `clippy -D warnings`, and `cargo test --workspace`, with Postgres 16 for the shop-api sqlx tests.
3. **wasm** — `shop-pricing` and `shop-front` build for `wasm32-unknown-unknown`, plus their native unit tests.
4. **site** — the Laravel app installs, builds the frontend, and dry-runs `course:refresh` so internal course links stay valid.

You do not need to run those jobs to learn. When something in a lesson disagrees with `rustc`, open an issue: the fence should fail CI until it matches the compiler.

Key takeaways

- Runnable fences and intentional `compile_fail` fences are checked in CI.
- The companion workspace and the WASM crates are tested on every push.
- The site build plus a course dry-run catches broken internal links.