Where Mullet goes next

Mullet 2.5.0 is out. What I plan next: a guard that checks a step fits a model's window, a price ceiling instead of a free switch, more settings and fewer defaults, and a separate machine for models. Plans, not features.



Published
1 November 2026

Reading time
4 minutes

About
Mullet

Tags
AI, Python

More on: Mullet

Mullet 2.5.0 is out. This post is what I plan to do next, with the evidence for each item. Apart from two settings that 2.5.0 already added, none of it is built.

A guard that checks a step fits

A local model has a fixed window, and a step can need more than it has. On 1 November a request of 65,620 tokens exceeded a 64k window, and the worker aborted with "request exceeds the context window". The sum is not hard: add up the tokens of the files the step will attach, the brief and the harness prompt (about 6k). My plan is to compare that with the usable window, which is the context minus the output limit in the opencode config (8,192 tokens), and to skip a local rung whose usable window is less than twice the sum.

The guard would do that before a step goes to a local or free rung, and show the reason when it skips one. At the moment I do the sum by hand, after a failure.

A price ceiling instead of free

Mullet's free rung is decided by a `:free` suffix on a model id. The listed prices I looked at on 1 November say a cheap paid step can cost cents, so I would rather define it by price. The user would set a ceiling per million tokens, Mullet would read current prices from OpenRouter's public model list, and zero would mean `:free` only. The setting would be called `ladder.cheap_max`.

By my arithmetic, a step of about 1.14 million tokens in and 30 thousand out, which is what one Strata run of mine used, would cost roughly $0.06 to $0.19 on GLM-5.3-Flash at the listed prices of the cheapest fp8 host and of Z.AI. That comes from listed prices, not a bill.

Settings, not defaults

Budgets and hardware differ, so each decision like this should be a documented per-user setting. 2.5.0 already did that twice: `hud.five_hour` for plans that have no five-hour limit, and `ladder.local_providers` so a local server such as Strata is priced as free. The ceiling and the guard would follow the same pattern.

Three ideas from a game framework's prompts

I read the role prompts of VibeGame, a framework that turns prompts into 2D web games using several roles (https://github.com/tettethu/VibeGame (https://github.com/tettethu/VibeGame), Apache-2.0), as text. I did not run the framework. Three ideas for coding jobs came out of it. They are my adaptations, not things the prompts say in these words, and none is built or tested:

- The prompts have the architect stop and report when something it needs is missing. For the worker, the first message would give the path of the brief, read-only, say to read it and never ask for it, and tell the worker to answer BLOCKED otherwise. I am aiming at a failure I saw on 1 November, where opencode compacted a local model's conversation and the model asked for the brief it had lost.
- The reviewer prompt says to "stop the moment you find any reject-worthy issue". Mullet's reviewer would report the first blocking defect and stop, so the next attempt has one clear thing to fix. That reason is my guess.
- The reviewer prompt lists counter-examples that do not count as evidence, such as "Frames are advancing". I would add the same to the planner's guidance on acceptance checks, with examples from coding jobs: an import-only check, a grep, and a `pytest -k` that selects nothing. None of those shows that the worker did what the step asked (pytest exits with code 5 when nothing is selected).

Mullet's gates already cover part of this on the review side. These would be changes to prompts and guidance, not new gates.

A machine that only runs models

My laptop runs Strata and my own work, with 62 GB of RAM and a card with 12 GB of VRAM. I would like a machine that only runs models and servers, with my laptop reaching it over the network.

The hardware thoughts in this section are unverified thinking, not tested. A Raspberry Pi running a VPN at home could let me reach that machine when I am away. Basalt, the Strata fork (https://github.com/jesdga95/basalt (https://github.com/jesdga95/basalt)), needs an NVIDIA Blackwell card, which I checked in its source. A secondary article, not a source I checked, reported about 153 tokens a second on a 16 GB RTX 5080, and I have not verified that figure. A card with more VRAM might also help Strata, because it could hold a bigger expert cache. I have not tested any of this.

Where it stands

Mullet 2.5.0 is tagged in the public repo, https://github.com/tomgould/mullet-public (https://github.com/tomgould/mullet-public), and its suite has about 18,700 tests. Planned items stay planned until they have gates and tests of their own. The earlier posts cover the design: Mullet 2, coding jobs you can leave running and Gates, not a model's word.