PRIME RADIANT · REPOSITORY GUIDE

The Prime Radiant Codebase

Three passes over one repository: the map, the machine, and the nervous system that runs it — ending where the project goes next: a season that starts itself.

38 modules 379 tests · 100.00% coverage 15 mutants run & killed

4 surfaces live gate: make check

01 · THE MAP

Where everything lives

Everything interesting is one Python package plus four surfaces that consume it. The pipeline across the top row is the product; the row beneath it is the proof.

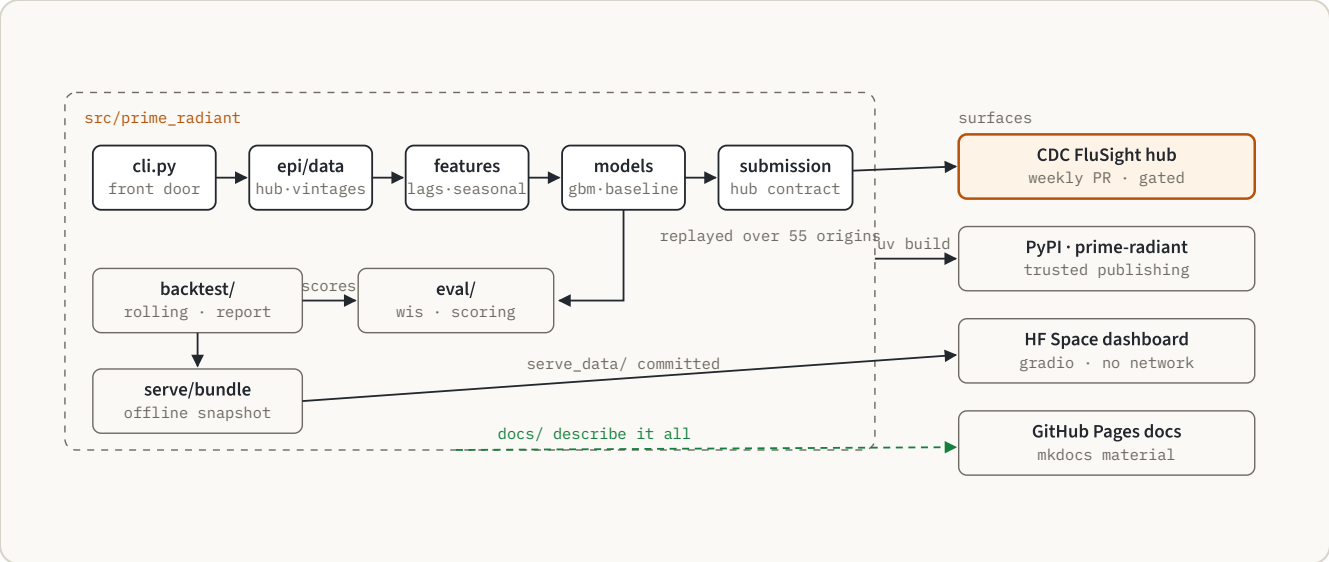


Figure 1. The package and its surfaces. Top row is the weekly product path; the middle row is the evidence machinery that replays the same code over history. Amber marks the one outward-facing, gated surface.

Reading order for a newcomer: `epi/cli.py` is the front door — three subcommands (`forecast`, `validate`, `bundle`), everything else hangs off them. `epi/data` gets and time-scopes the data, `features`/`models` do the machine learning, `submission` speaks the hub's contract, `backtest` + `eval` prove the model honest, and `serve` packages results for the dashboard. At the top level, `tests/` mirrors all of it at 100% coverage, `scripts/open_hub_pr.sh` is the one shell dependency, and `NOTES/` is the project's memory.

What one weekly forecast computes

The load-bearing idea is **vintage discipline**: the model must never see data dated after the forecast origin. So the data layer is built on git archaeology — checking out the truth file *as it existed* on a past date — rather than "download latest".

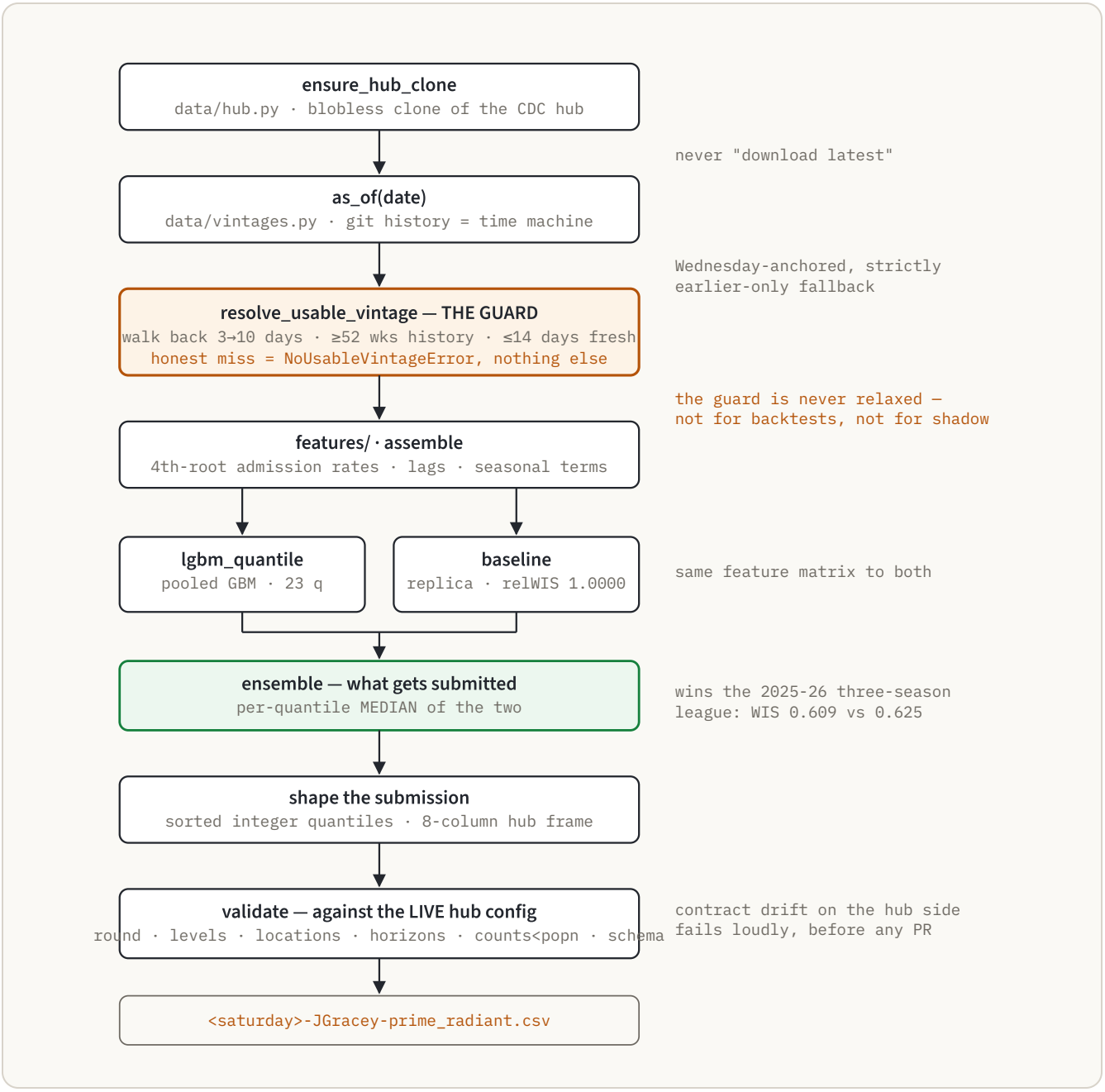


Figure 2. One weekly run, top to bottom. The amber guard is the mechanism the whole design leans on: it decides what data is honestly usable, and it fails with its own exception type so nothing else can masquerade as an off-season skip.

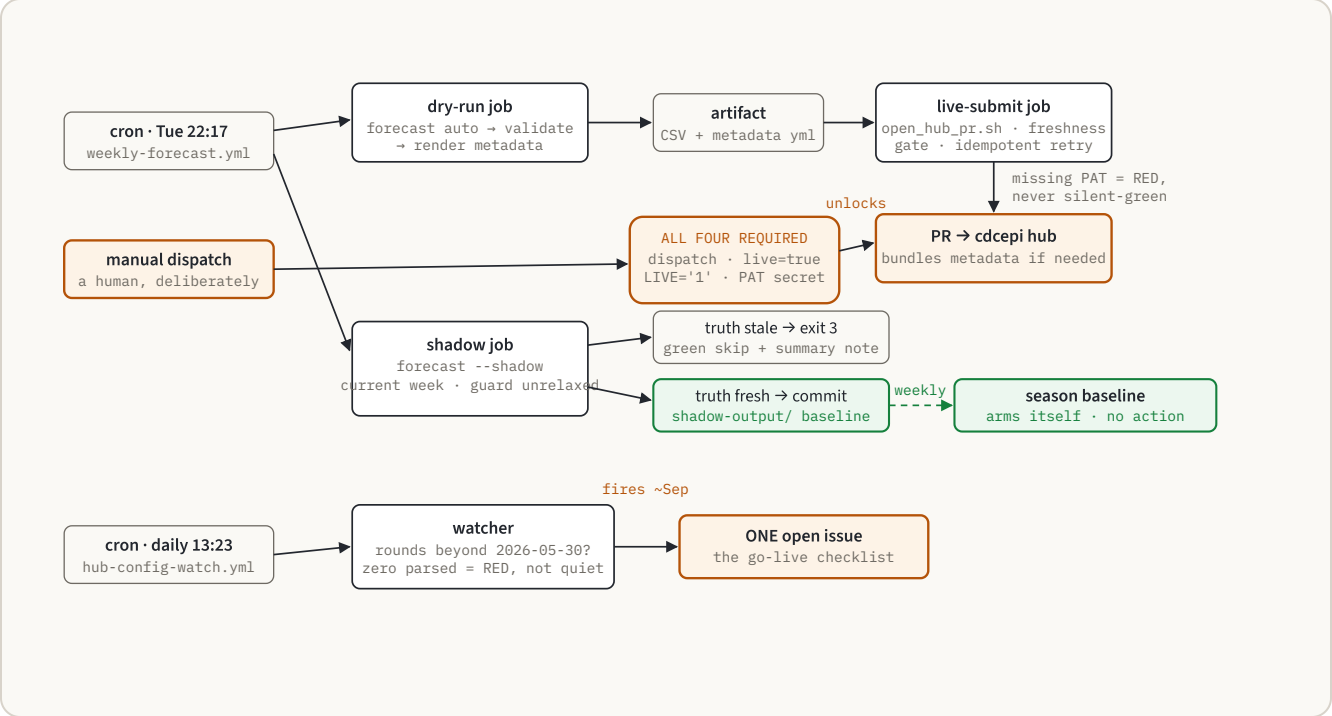
The same `run_origin` function drives live forecasts and the historical backtests: `backtest/rolling.py` replays it across 55 past origins, `eval/wis.py` scores each with the hub's own metric, and

`serve/bundle.py` snapshots the results into the offline bundle the dashboard serves. One code path — so a backtest win means something about the live path.

03 · THE NERVOUS SYSTEM

How it runs itself, and where the gates are

Two cron schedules do the routine work; a human dispatch — behind four simultaneous conditions — is the only way anything reaches the hub. The design rule throughout: **every gate is structure, not prose**, and each structural claim has a test that a mutant provably fails.



> **Figure 3.** The automation. Cron reaches only the honest paths; the hub PR sits behind four simultaneous conditions, and the one scary failure mode — quietly looking fine — is engineered out at each step.

This layer was adversarially reviewed by a four-refuter panel; every demonstrated defect was fixed with a regression test and a mutant proof:

DEMONSTRATED FINDING	STRUCTURAL FIX
Hub schema drift (a `KeyError`) masqueraded as an eternal "off-season skip"	✓ The guard's miss is its own exception type; drift now fails red
No freshness gate — a live dispatch would have submitted a months-stale round, green	✓ The PR script refuses any reference date outside 0–13 days out
A same-week retry after a partial failure dead-ended on the stranded fork branch	✓ Force-push to the disposable branch; skip create if the PR exists
A hub config restructure read as "no news", green, forever	✓ Zero parsed dates now fails the watcher red

04 · THE NEXT STEP

The season starts itself

Nothing can forecast the current week until the hub's own truth data resumes — dormant since July 9, and by three seasons of precedent it wakes in September, the same week the new season's config lands. Everything below the line is already running; everything above it waits on exactly one person.

Figure 4. The road to the first live submission. Green is autonomous and already in motion; amber requires a deliberate human dispatch through the four gates.

WHEN	WHAT FIRES	WHOSE ACTION
Now	Shadow job green-skips weekly; watcher checks the hub config daily	none
~Sep	Config lands → watcher opens its issue; truth resumes → shadow CSVs land in `shadow-output/`	none
The gap	PAT + `LIVE=1` set; dry-run dispatched against a real new-season round	operator
First window	First real submission — explicit go only	operator
~Nov 18	Guaranteed open: mass season start	operator
