The Brilliant Employee · 03

Recommended 89% of the time. Ran 0.2% of the time.

The Gate sizes every task before a model sees it. Across the 3,717 decisions logged up to 4 August 22:48Z it recommended the cheap tier 89% of the time. Then I checked what actually ran.

sgnk.ai/brilliant ↗

Sagnik Mitra

Why this exists

sgnk is a one-person studio. A language model writes a large share of the production code. You do not need your own system to get answers out of a model. You need one to find out when the answers were wrong, because wrong work reads exactly like right work and no prompt fixes that.

Three things from that week, none visible from inside the tool: two fixes I had written up as done that were never made, a detector whose 1,112 catches were mostly one test string of my own, and a backup fleet dead for weeks in silence.

THE SYSTEM, AS OF 5 AUGUST 2026, 00:39Z	
rules written	11 June, 55 days ago
ledger running	27 June, 39 days
tasks logged	2,959, across 3,397 rows
controls on tool calls	**6 registered, 3 of them armed**

You can rent the model. You cannot rent the part that tells you it was wrong.

Five parts, and where today sits

Five parts sit around the model. What changes from one piece of writing to the next is which of them is under the microscope.

the Gate	sizes a job before a model is picked
the Gambler	recommends which model gets the job
the Rails	refuse the irreversible, before it runs
the Judge	grades finished work pass or fail
the Ledger	at least one append-only line per task

Listed in the order they act on a task, not the order they were built.

Today is about the Gate, marked above. The other four are the machinery around it, and this post does not assume you know any of them.

This page is the map. Nothing here needs another post to make sense.

Every task meets a bouncer before it meets a model

Nothing in this workplace goes straight to a model. Every request first passes the Gate, the component that sizes the job: is this trivial, does it need the expensive brain, or is the scope genuinely unenumerable. Assignment happens only after triage.

> **Plain words: THE GATE.** a pre-flight triage step that runs before any model call. It sizes the task, recommends a worker tier, and flags whether the job needs verification or human approval. It records all of that. It enforces none of it.

The standing rule behind it is blunt: the best model call is the one you never make. Trivial work skips the whole orchestration stack.

Triage before assignment. Always.

Three chutes out of the Gate

1. Cheap and single-shot: the floor model does the job alone. 89% of recommendations land here, and 95% of everything is recommended as a single shot.
2. Strong model: reserved for architecture, multi-file surgery, and deep investigation. 11% of recommendations.
3. Full workflow: multiple coordinated agents, only for scope you cannot enumerate up front. 5%.

The split is not thrift. It is the shape the work actually has. Most tasks are small, and pretending otherwise burns tokens on ceremony while teaching you nothing about which tasks were ever hard. Note the word recommendations. It is doing more work in that sentence than I realised when I wrote it.

Cheap by default. Expensive by exception.

What the Gate recommended, 3,717 decisions

THE GATE'S ADVICE, PINNED TO 4 AUGUST 22:48Z	
recommended floor (cheap)	3,311 = 89%
recommended strong	406 = 11%
recommended single shot	3,547 = 95%
recommended workflow	170 = 5%
floor AND guarded	**861 = 23%**
every row is an intention. Not one of them is an outcome.	

Every number on that card is something the Gate wanted. I read this table as though it described what my system did. It describes what my system advised.

A recommendation is not an outcome.

What the guard actually is

602

2,064

of the first 2,064 decisions, the ones the Gate flagged as needing verification or explicit approval, counted across every tier. Those rows end on 28 July

Guarded means the Gate appended a sentence to the worker's instructions: run the verify ladder before calling this done, or name this operation and get approval before running it. That is the whole mechanism. It writes the line and exits zero.

I am being fussy about that because the word guard oversells it. Nothing is blocked at this step. Enforcement is a different component. What the Gate contributes is that the instruction rides along with the task automatically instead of depending on me remembering. A cheap worker with a checker beats an expensive worker with a reputation, but only while the checker runs.

Name what your guard actually does, not what you hoped it would.

The target I retired

OLD TARGET, RETIRED 28 JULY

metric
% blocked from the heavy path

target
30 to 50%

problem
measures cost, not safety

status
not comparable, do not revive

CURRENT TARGET

default
floor tier, single shot

strong tier
architecture, multi-file, deep digs

workflow
unenumerable scope only

watched number
the guarded fraction

The old target died because the Gate stopped emitting the thing it measured. The replacement watches the number that rots quietly.

Watch the guardrail metric, not the cost metric.

Then I checked what actually ran

I went looking for the model field, to confirm the 89% held in practice. There is no model field. The Gate has only ever recorded what it advised.

```
$ jq -r --arg c "$CUT" 'select(
    .gate_tier=="floor" and
    .timestamp<=$c).model' \
    traces/*.jsonl | sort | uniq -c
   698 claude-opus-5
    98 claude-opus-4-8
    10 unknown
     8 claude-fable-5
     2 claude-sonnet-5
```

2 of 816

ledger rows carrying a floor-tier gate verdict ran on the cheap tier. The flagship ran 796 of them, and 10 name no model at all. Only 915 of the ledger's 3,357 rows carry a gate verdict, which is its own gap.

A log that records intent can never tell you it is being ignored.

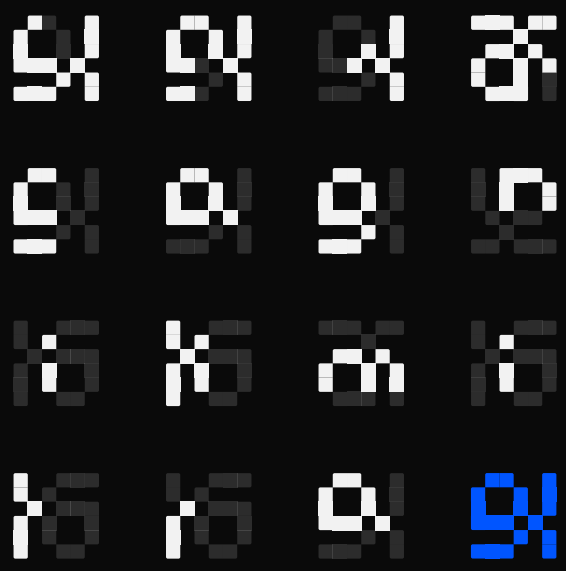
Why not just use the best model for everything

> **REJECTED.** A router that sends every task to the flagship model. It feels safe and measures nothing: you pay for reputation, collect no verification record, and never learn which tasks were trivial all along. I rejected this design in June, built the Gate to avoid it, and then ran it anyway, unnoticed, across every one of those 3,717 decisions.

That is the part worth taking. The design was right. The measurement was pointed at the wrong file. A component that advises and a component that executes are two different things, and if only one of them writes to a log, the log will agree with you forever.

Reliability lives in the workplace, not the hire. So does self-deception, if you let the workplace grade its own homework.

Log the outcome, or you are logging your own opinion.



Cheap by default was the plan. I never checked what shipped.

sgnk.ai/brilliant ↗

Sagnik Mitra

I build AI systems and the machinery that keeps them honest: gates, ledgers, judges, and rails. Product design for years, engineering the whole time. This series is what the building actually taught me.

`sgnk.ai · in/sagnikmitra · github.com/sagnikmitra`