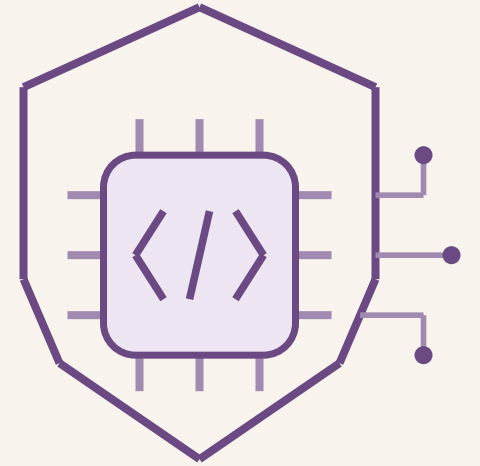


Before the Agent Writes Code: Policy-Aware Engines for AI Coding



Nnenna Ndukwe

AI Developer Relations Lead



[Resources + about me](#)

Quality starts before the first implementation

- 01 Why standards belong before coding
Engineering decisions → explicit expectations
- 02 How MCP makes standards available
Policy ownership → tools → retrieval
- 03 Follow a real GitHub task
Ticket → cited policy → proposed plan
- 04 Test adherence in the generated code
Rule → expected behavior → adherence check
- 05 Connect it to a team workflow
Codebase context → standards → independent review

Authentication logs need useful context without secrets

Make authentication failure logs safe without losing useful diagnostics

Preserve responses

Keep existing auth behavior

Useful diagnostics

Keep safe failure categories

Exclude secrets

Check both logging sinks

Teams decide which policies reach the coding agent

26 starter policies

BE-LOG-001 · Safe Structured Logging

```
observability/  
backend-logging-standard.md
```

Required Rules

```
LOG-1  Exclude secrets, tokens,  
authorization headers, session cookies...
```

Sources and Change Notice

```
OWASP Logging Cheat Sheet  
OpenTelemetry Logs Data Model
```

Bring team policies

```
team-policies/ + pack.toml
```



Select a pinned profile

Baseline + team packs

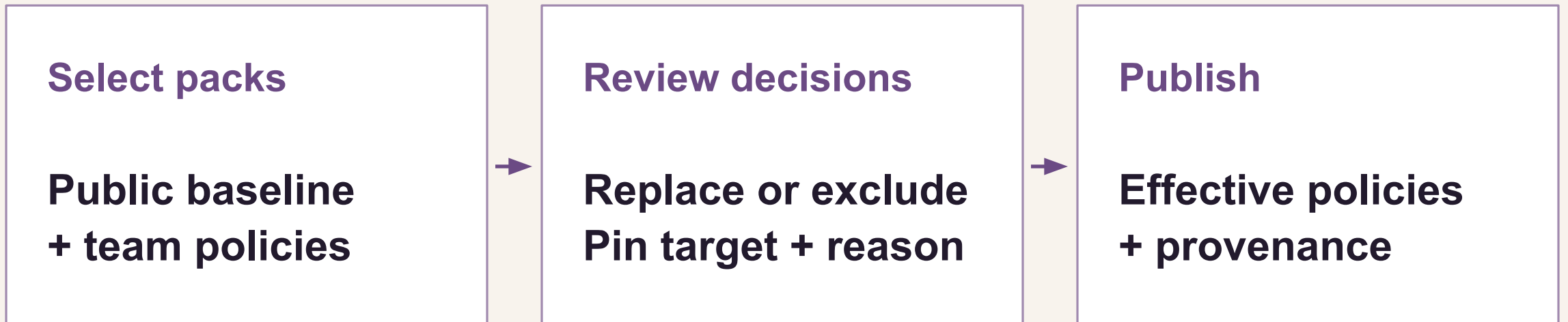


Publish the selected library

```
Validate → version → index  
Custom packs: local or self-hosted
```

The team decides which rules are active

Local or self-hosted policy profiles



A changed target needs a fresh review.

A policy update changes new work

One validated policy generation

Source text

Catalog

Search index

Runtime rules

In-flight work → held version

New work → activated version

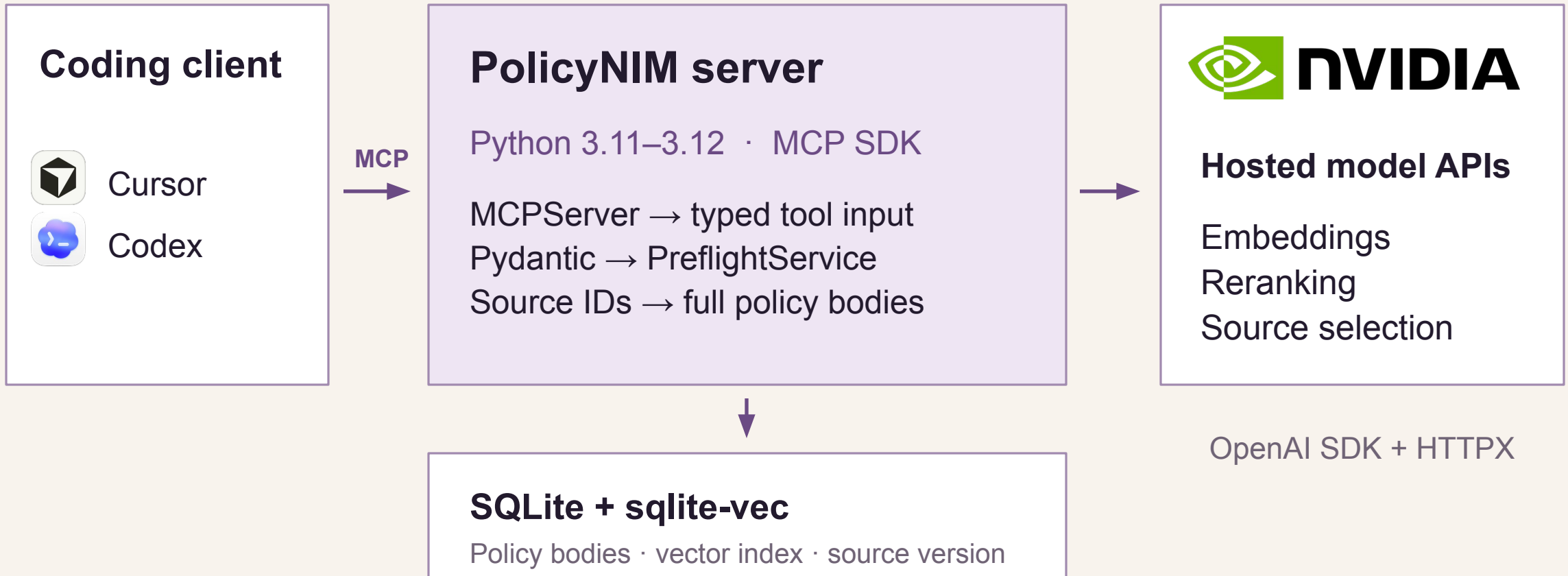
A matching rule changes what can execute

Runtime wrapper · shell commands / file writes / HTTP requests



Evidence: decision, confirmation, outcome, generation

PolicyNIM serves engineering rules through MCP



Four MCP tools expose policy sources before coding

policy_search

Explore retrieval

Ranked passages

Scores + source citations

policy_compile

Build a structured policy packet

Full selected policies

Evidence + citations + generation

policy_preflight

Prepare a coding task

Task-relevant policy sources

Full bodies + source citations

policy_preflight_evidence

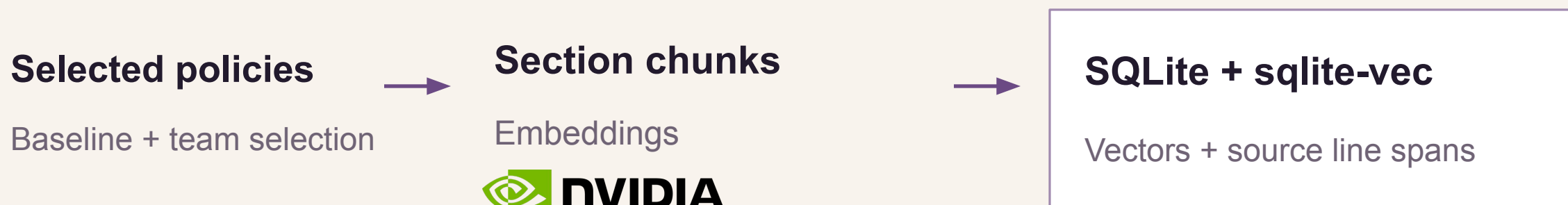
Inspect a preflight run

Preflight + same-run evidence trace

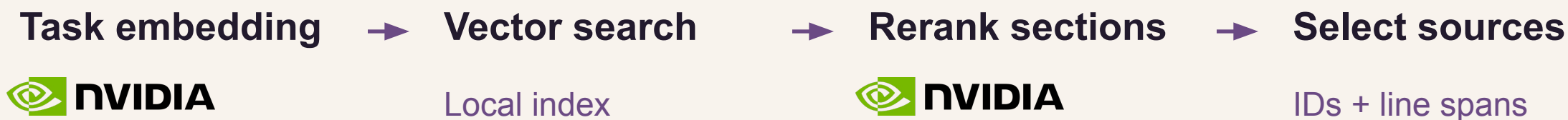
Sources + chunks + run/packet IDs

How PolicyNIM **rank**s engineering rules a coding task needs

BUILD THE INDEX

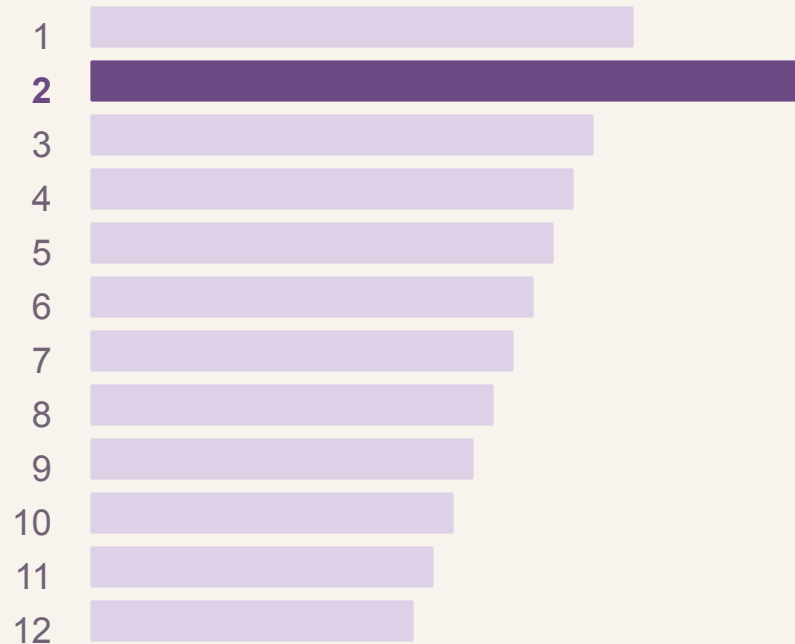


SERVE THE CODING TASK

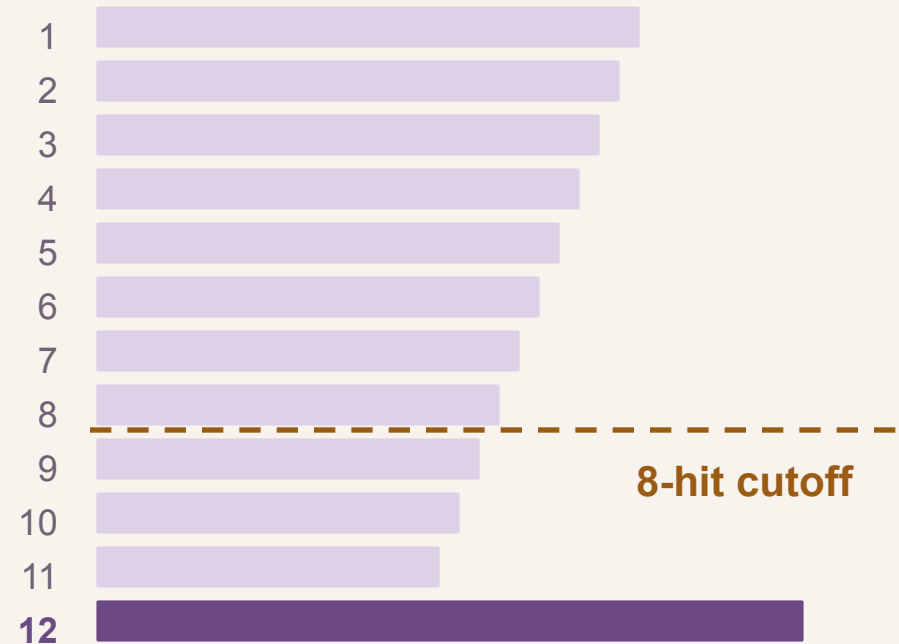


Reranking dropped the cleanup job's required rules

Dense retrieval



After reranking



Required-rule section: rank 2 → rank 12

Full policy sources keep requirements together

Selected policy

Purpose

Required rules

Conditions

Exceptions



Complete source
within the budget

If it won't fit, narrow the request.

A real citation can still hallucinate an eng standard/rule

Real citation / source

A real policy ID
and source location

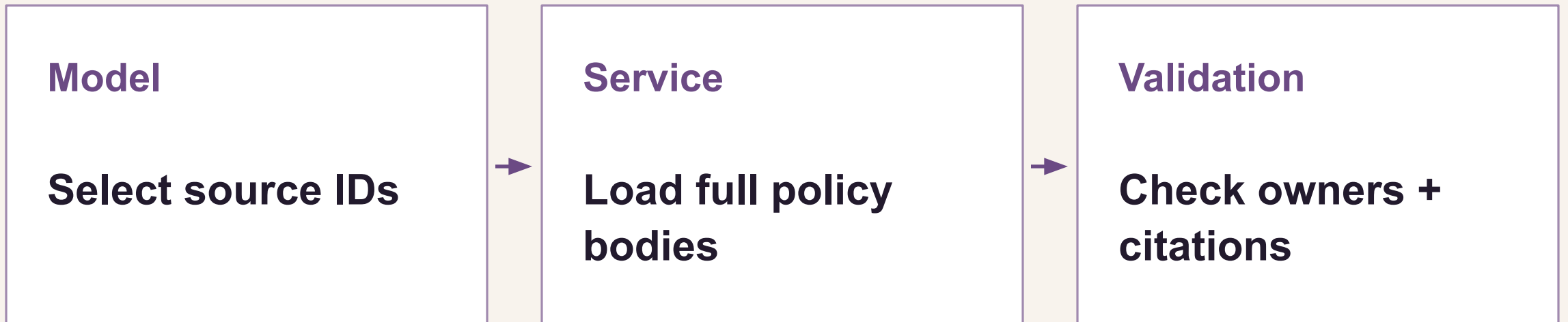
Hallucinated details

Batch sizes

API choices

Token lifecycle

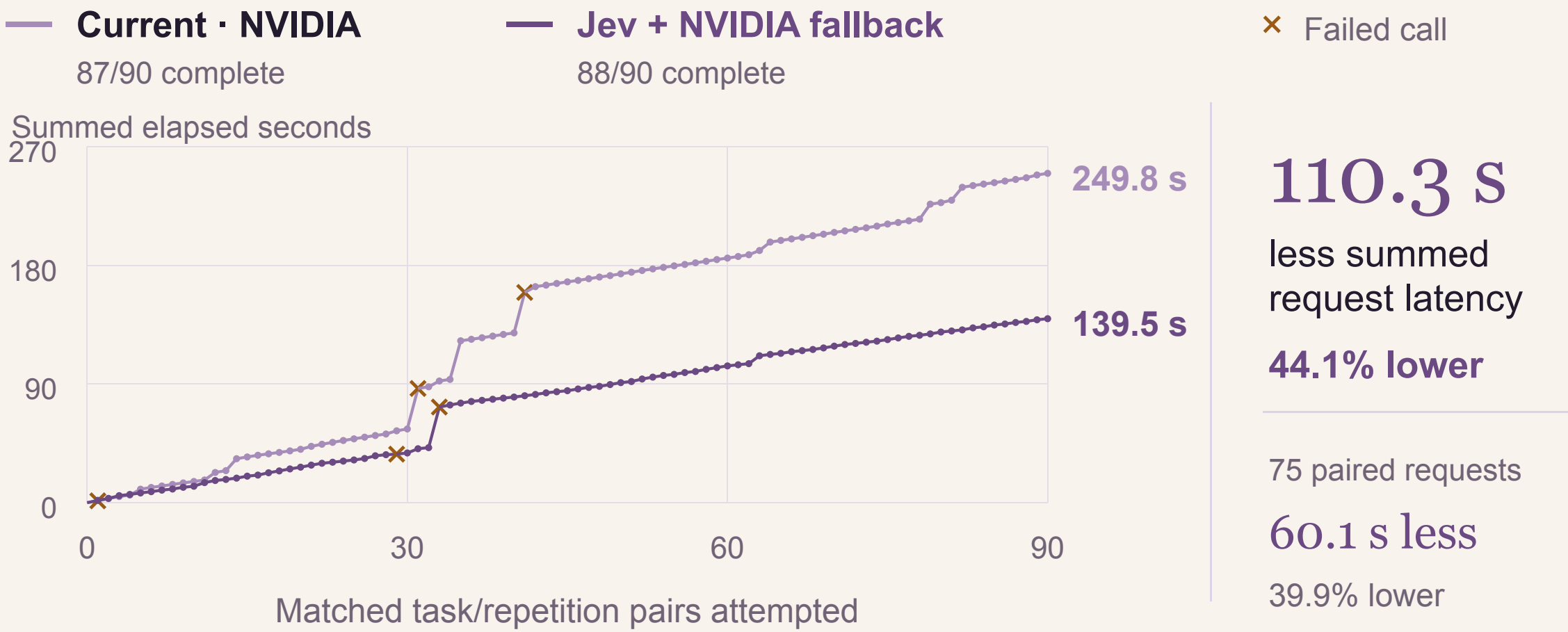
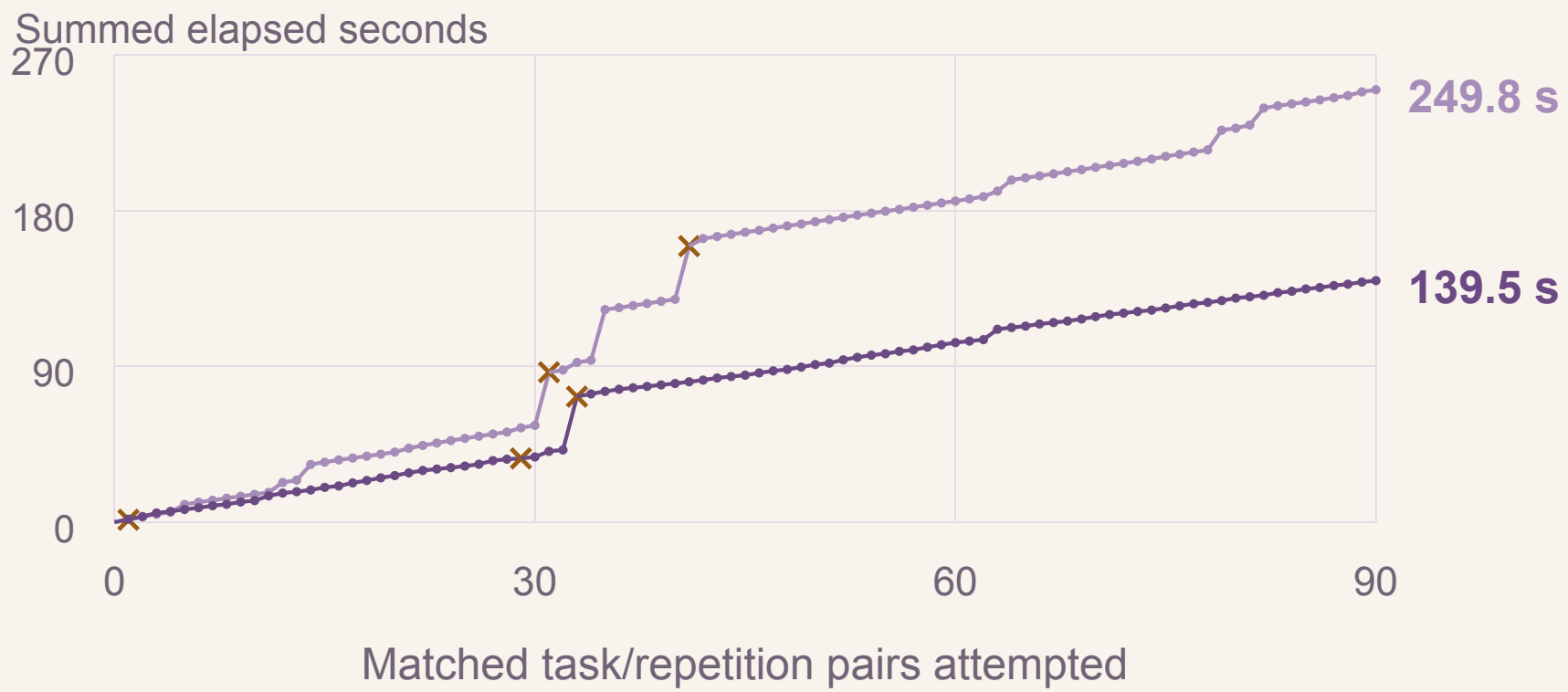
PolicyNIM returns the policy text the model selects



The response retains the source's rules, conditions, and exceptions.

Across 90 attempts, 110 seconds less latency with Jev

MCP pilot · 30 tasks, 3 passes · 90 reqs per variant



One policy version keeps text and citations aligned

One immutable generation

Source text

Catalog

Search index

Runtime rules

Each operation holds the source version it reads.

One task, three context conditions, twelve first drafts

Same model -> same task

Task only

Task + public interfaces

8 ranked chunks

Task + retrieved sections

Full policy sources

Task + PolicyNIM preflight

4 repeats

4 repeats

4 repeats

14 checks: cancellation · retries · duplicate effects · safe logging

All twelve first drafts still needed review

Task only

0 / 4

complete passes

27 / 56 checks passed

8 ranked chunks

0 / 4

complete passes

22 / 56 checks passed

Full policy sources

0 / 4

complete passes

25 / 56 checks passed

Measure the implementation against the rules it received.

The workflow starts with an authentication-logging issue

GitHub issue #2

Make authentication failure logs safe without losing useful diagnostics.

Preserve existing authentication responses and add regression coverage.

Trace both diagnostic paths

Application log + security event

app/auth/jwt.py

```
error_str = str(e)

logger.warning(
    f"JWT error: {error_str}"
)
```

Policy evidence turns the task into testable expectations

`policy_preflight_evidence`

BE-LOG-001 · LOG-1

Exclude secrets, tokens,
authorization headers,
session cookies...

Proposed plan

Replace exception text with
safe failure categories.

Keep both diagnostic paths useful.

Carry the rule into verification

Capture both sinks.

Check that secret markers stay out.

Code adherence means testing the rules the agent received

Logging example · rule → expected behavior → check

Rule supplied

LOG-1
Exclude sensitive values

Source: BE-LOG-001

Behavior expected

No token marker in
either diagnostic sink

Acceptance criterion

Adherence check

Inject marker → capture
diagnostics → assert
absence

Observable behavior

Use the same rule to **shape** the plan and **judge** the code.

An adherence test challenges the logging boundary

Given: a secret-bearing error

```
JWTErrror("SYNTHETIC_SECRET")  
  
verify_token(token, auth_error)  
  
Capture app_logs + events.
```

Illustrative assertions

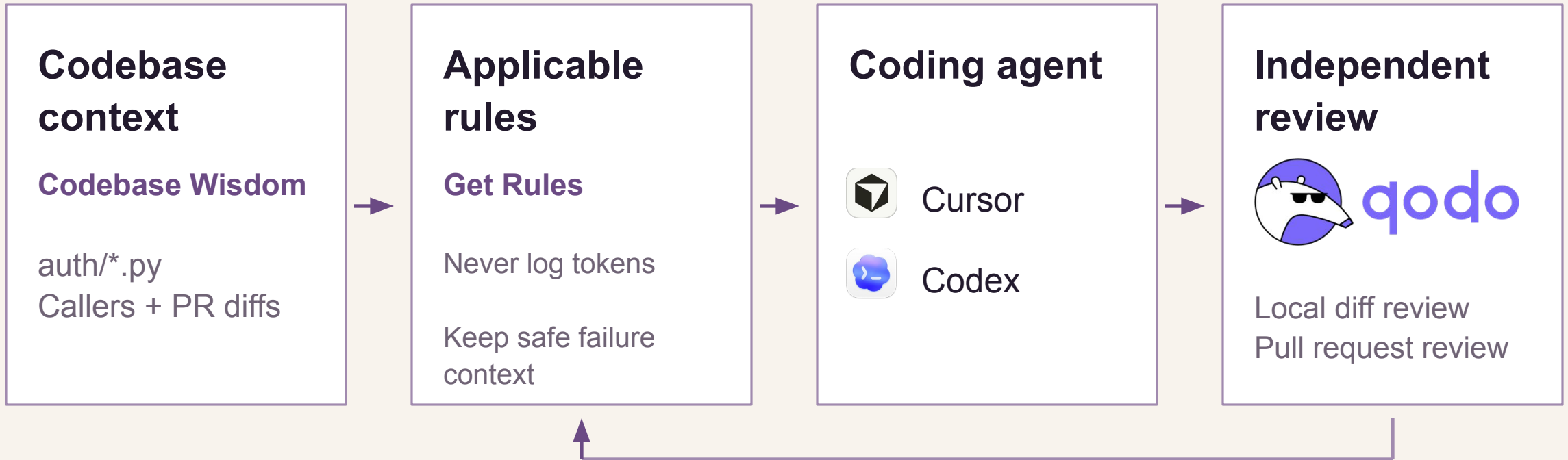
```
assert marker not in app_logs  
assert marker not in events
```

Preserve the auth contract

```
assert raised is auth_error  
assert reason == "jwt_error"
```

Qodo connects standards with code and review

Qodo Agentic Toolbox · MCP or CLI



Teams own the rules · scope by organization and repository · update from review

Bring engineering standards
into planning
for quality-driven agentic
engineering.

Keep exploring. Thank you!

The code. The evidence.
The open standards.

PolicyNIM · Qodo · MCP · AAIF

nnennandukwe.com



Scan for all resources