

GUARDRAILS, NOT GATES

Scaling Juniors in the AI Era

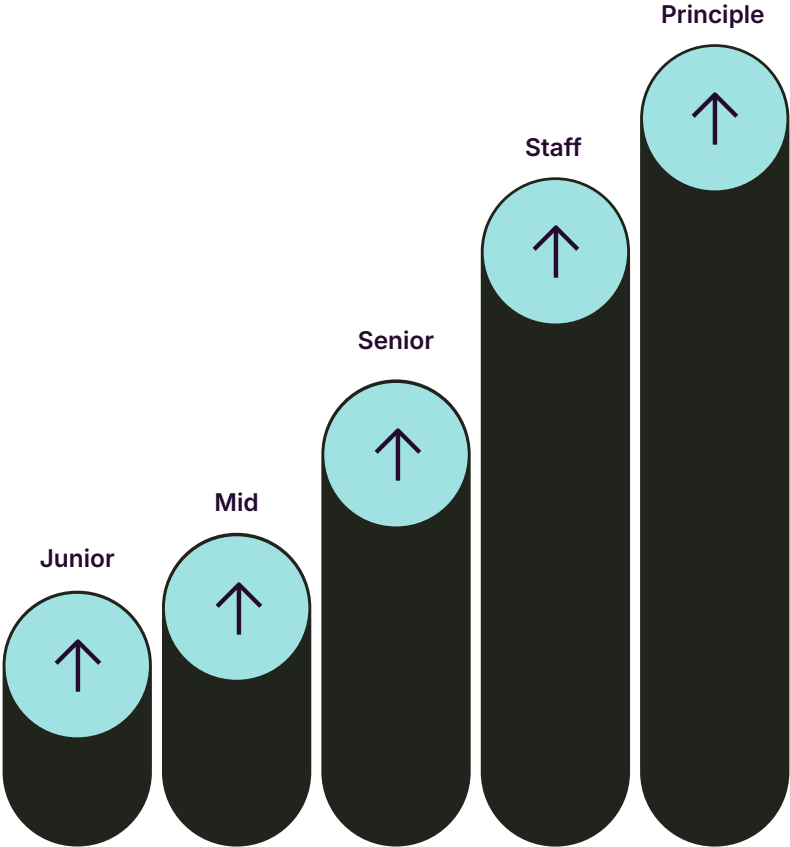




Rick Clegg

Developer Experience Lead







Class of 2026



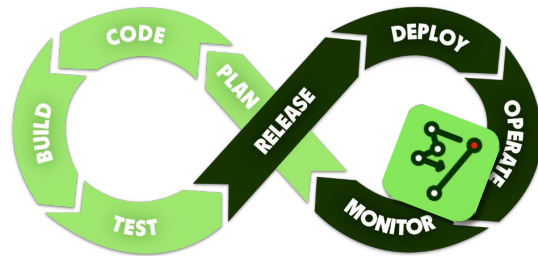
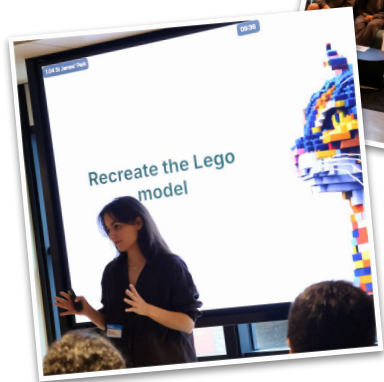
First day on the job

Educate

The master and the apprentice



Engineering Bootcamp

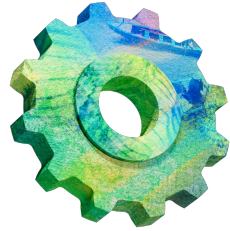


Engineering Academy



Working

Learning the best ways to work to have the biggest impact in your role.



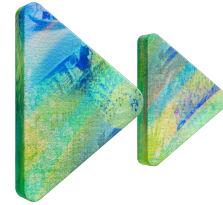
Coding

The core skills you need to be an engineer at Wise



Operating

How to we support running software to keep Wise working for our customers.

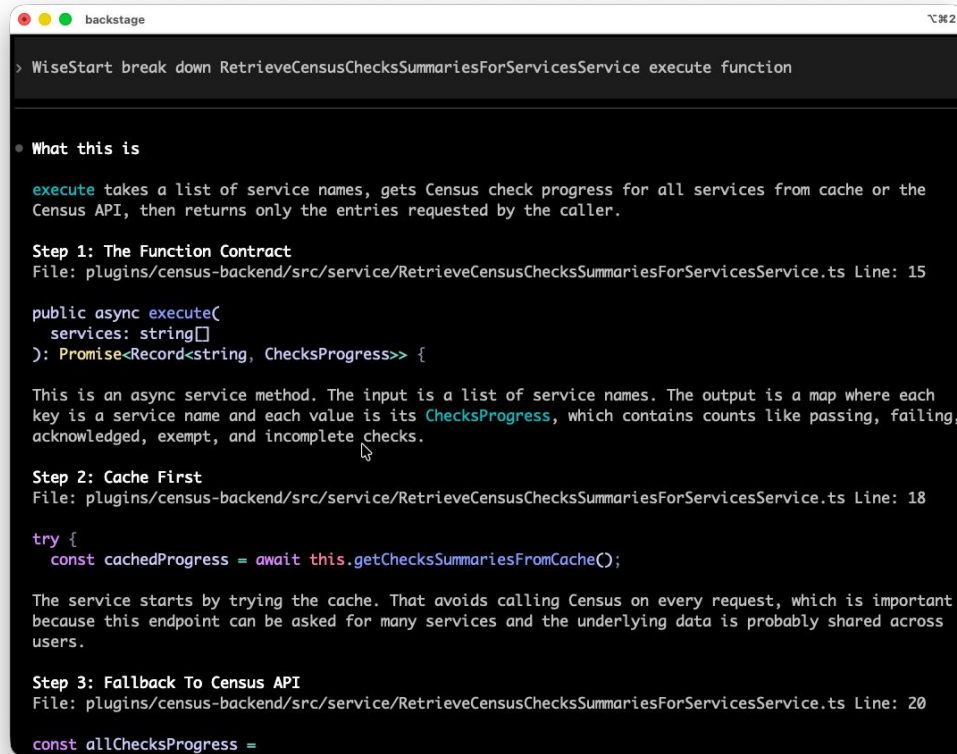


Advancing

Deeper dives into technical subjects to increase your impact.

Stop the slop

"WiseStart break down..."



The screenshot shows a window titled 'backstage' with a search bar containing the text '> WiseStart break down RetrieveCensusChecksSummariesForServicesService execute function'. Below the search bar, there is a section titled 'What this is' with a description of the 'execute' function. This is followed by 'Step 1: The Function Contract' which shows the function signature and a brief explanation of its parameters and return type. Next is 'Step 2: Cache First' which shows a code snippet for caching progress and an explanation of why this is important. Finally, 'Step 3: Fallback To Census API' is partially visible at the bottom.

```
> WiseStart break down RetrieveCensusChecksSummariesForServicesService execute function
```

• What this is

`execute` takes a list of service names, gets Census check progress for all services from cache or the Census API, then returns only the entries requested by the caller.

Step 1: The Function Contract
File: `plugins/census-backend/src/service/RetrieveCensusChecksSummariesForServicesService.ts` Line: 15

```
public async execute(  
  services: string[]  
) : Promise<Record<string, ChecksProgress>> {
```

This is an async service method. The input is a list of service names. The output is a map where each key is a service name and each value is its `ChecksProgress`, which contains counts like passing, failing, acknowledged, exempt, and incomplete checks.

Step 2: Cache First
File: `plugins/census-backend/src/service/RetrieveCensusChecksSummariesForServicesService.ts` Line: 18

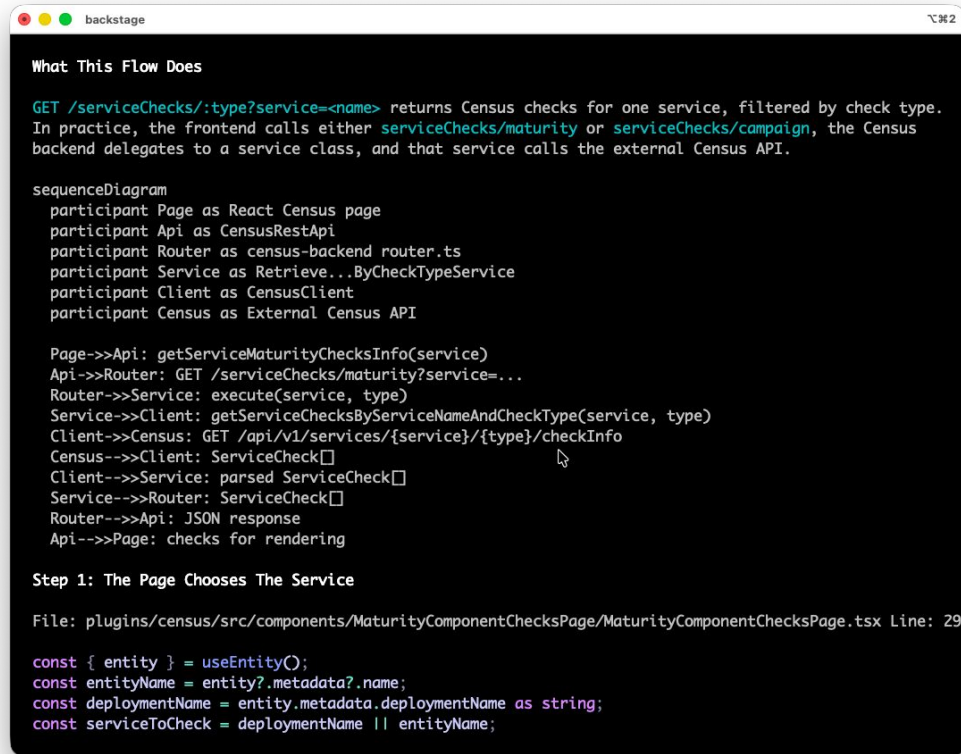
```
try {  
  const cachedProgress = await this.getChecksSummariesFromCache();
```

The service starts by trying the cache. That avoids calling Census on every request, which is important because this endpoint can be asked for many services and the underlying data is probably shared across users.

Step 3: Fallback To Census API
File: `plugins/census-backend/src/service/RetrieveCensusChecksSummariesForServicesService.ts` Line: 20

```
const allChecksProgress =
```

"WiseStart code flow..."



```
backstage ⌘ 2  
  
What This Flow Does  
  
GET /serviceChecks/:type?service=<name> returns Census checks for one service, filtered by check type.  
In practice, the frontend calls either serviceChecks/maturity or serviceChecks/campaign, the Census  
backend delegates to a service class, and that service calls the external Census API.  
  
sequenceDiagram  
    participant Page as React Census page  
    participant Api as CensusRestApi  
    participant Router as census-backend router.ts  
    participant Service as Retrieve...ByCheckTypeService  
    participant Client as CensusClient  
    participant Census as External Census API  
  
    Page->>Api: getServiceMaturityChecksInfo(service)  
    Api->>Router: GET /serviceChecks/maturity?service=...  
    Router->>Service: execute(service, type)  
    Service->>Client: getServiceChecksByServiceNameAndCheckType(service, type)  
    Client->>Census: GET /api/v1/services/{service}/{type}/checkInfo  
    Census-->>Client: ServiceCheck[]  
    Client-->>Service: parsed ServiceCheck[]  
    Service-->>Router: ServiceCheck[]  
    Router-->>Api: JSON response  
    Api-->>Page: checks for rendering  
  
Step 1: The Page Chooses The Service  
  
File: plugins/census/src/components/MaturityComponentChecksPage/MaturityComponentChecksPage.tsx Line: 29  
  
const { entity } = useEntity();  
const entityName = entity?.metadata?.name;  
const deploymentName = entity.metadata.deploymentName as string;  
const serviceToCheck = deploymentName || entityName;
```

"WiseStart review with me..."

```
backstage ⌘2
```

• What I'm Reviewing

GET `/serviceChecks/:type` so invalid check types are rejected before we call Census.

```
router.get("/serviceChecks/:type", async (request, response) => {
  const { type } = request.params;
  const queryService = request.query?.service;

+  if (!Object.values(CheckTypes).includes(type as CheckTypes)) {
+    response.status(400).send('Invalid type parameter: ${type}');
+    return;
+  }
+
  if (queryService !== undefined) {
    const checks =
      await retrieveCensusServiceChecksByServiceNameAndCheckTypeService.execute(
        queryService as string,
        type as CheckTypes
      );
  }
});
```

First Impression

This is the right instinct. The old code used `type as CheckTypes`, but a TypeScript cast is a paper badge, not a door lock. It makes the compiler nod, but it does not stop `/serviceChecks/banana?service=my-service` at runtime.

Decision 1: Runtime Validation

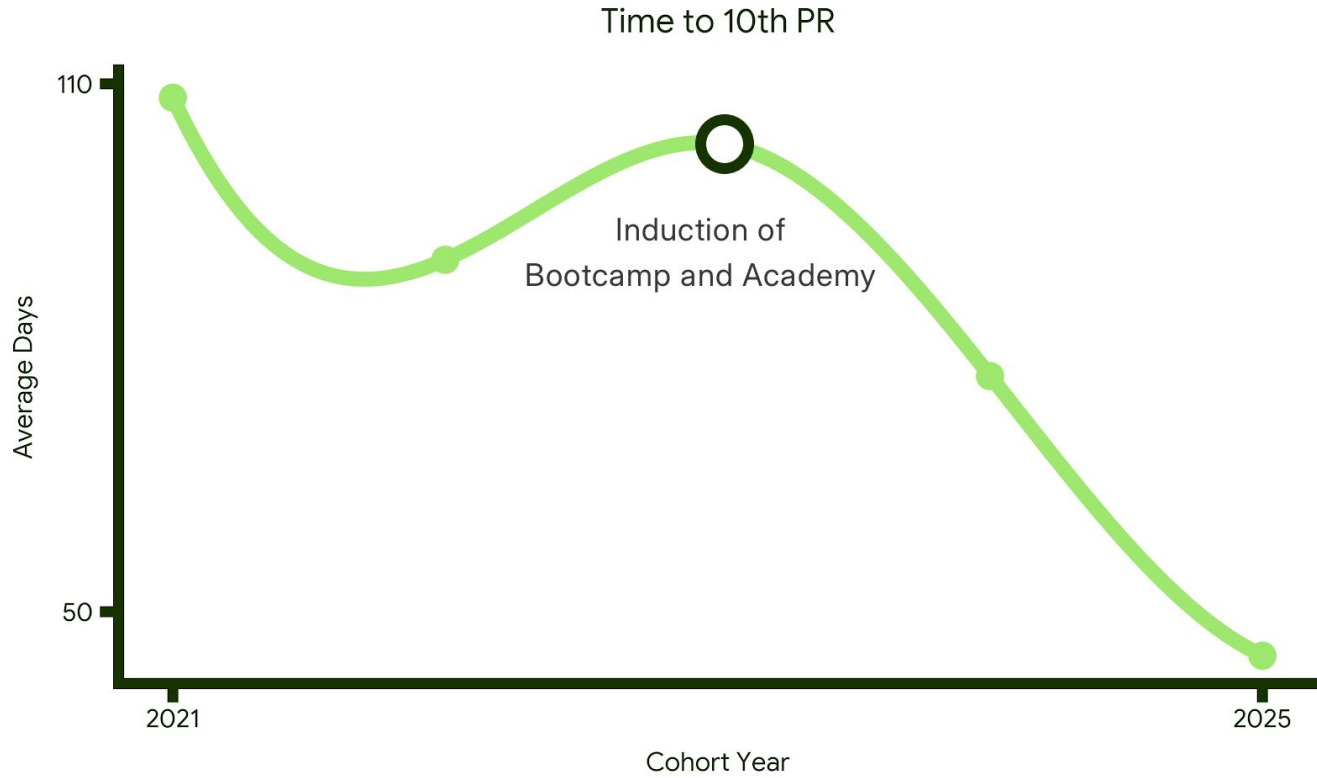
The route now checks the `type` before calling the service. That is good because the router is the border checkpoint for HTTP-shaped data.

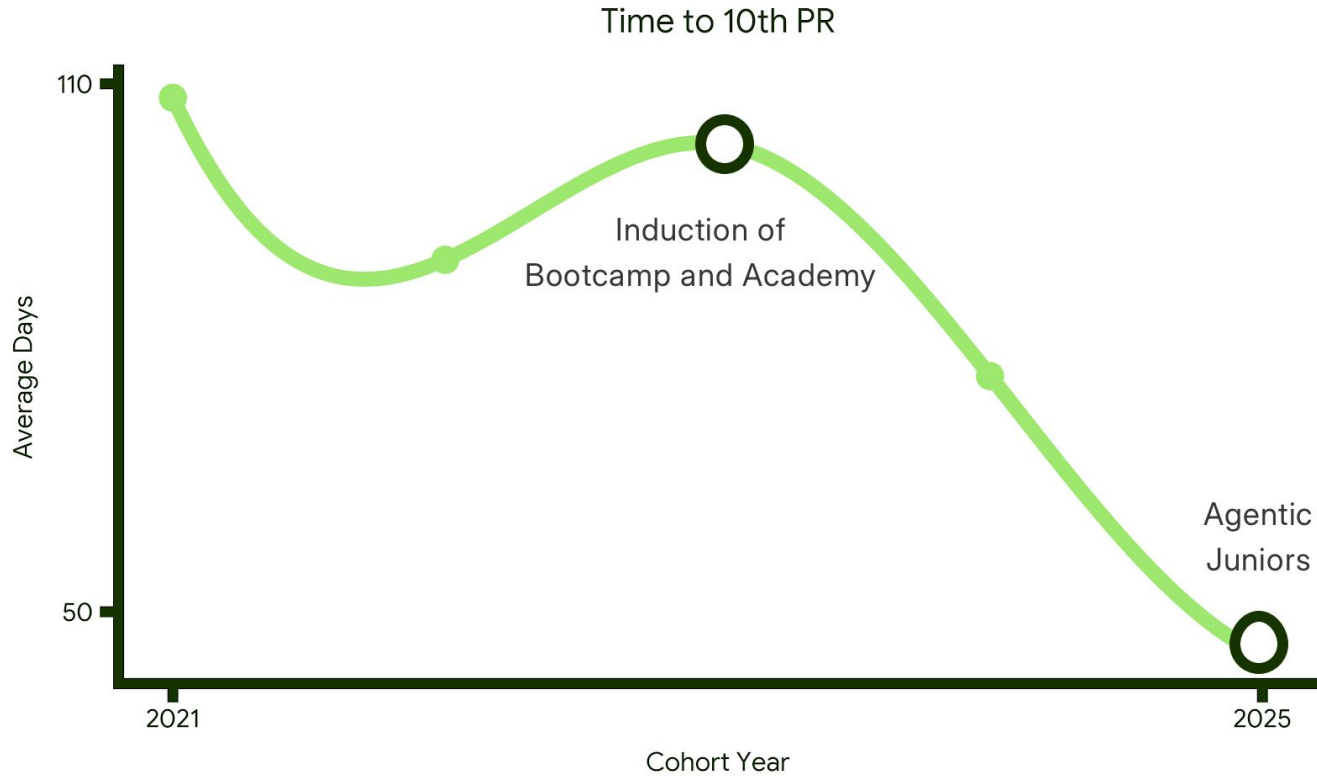
I'd probably wrap this in a tiny helper:

```
const isCheckType = (type: string): type is CheckTypes =>
```

The impact



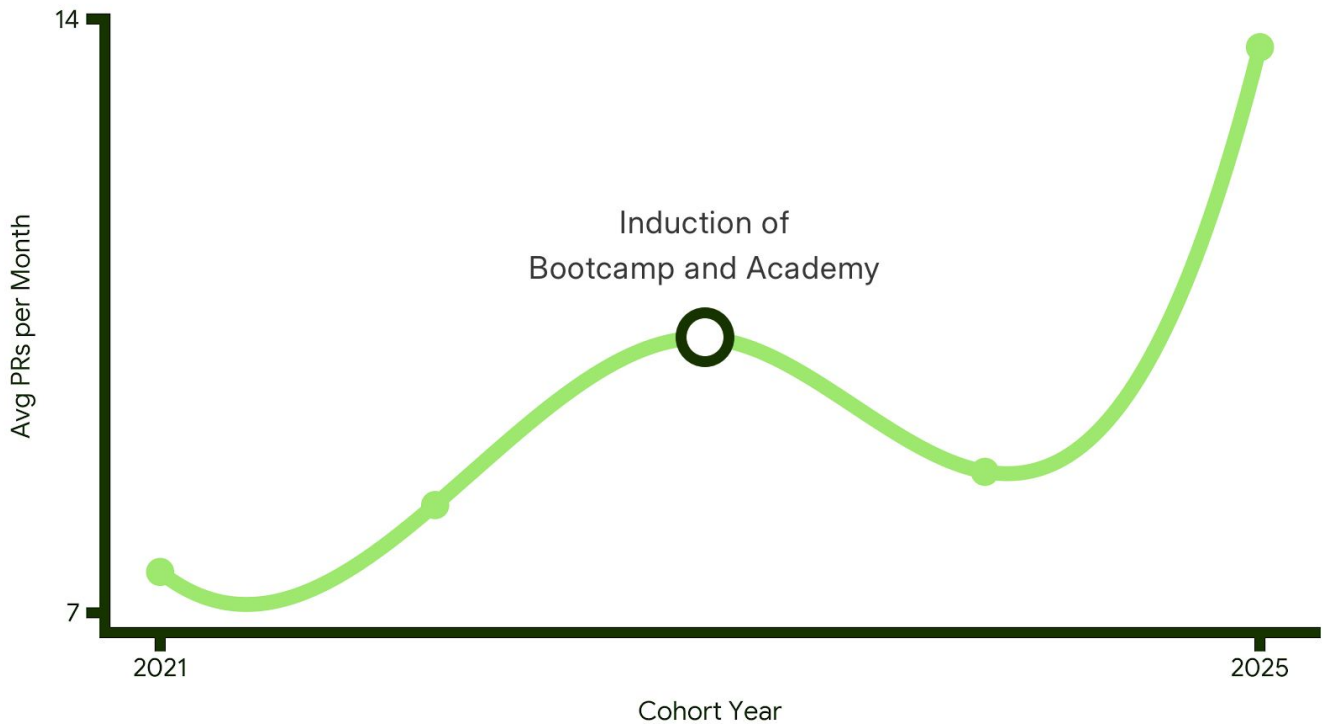




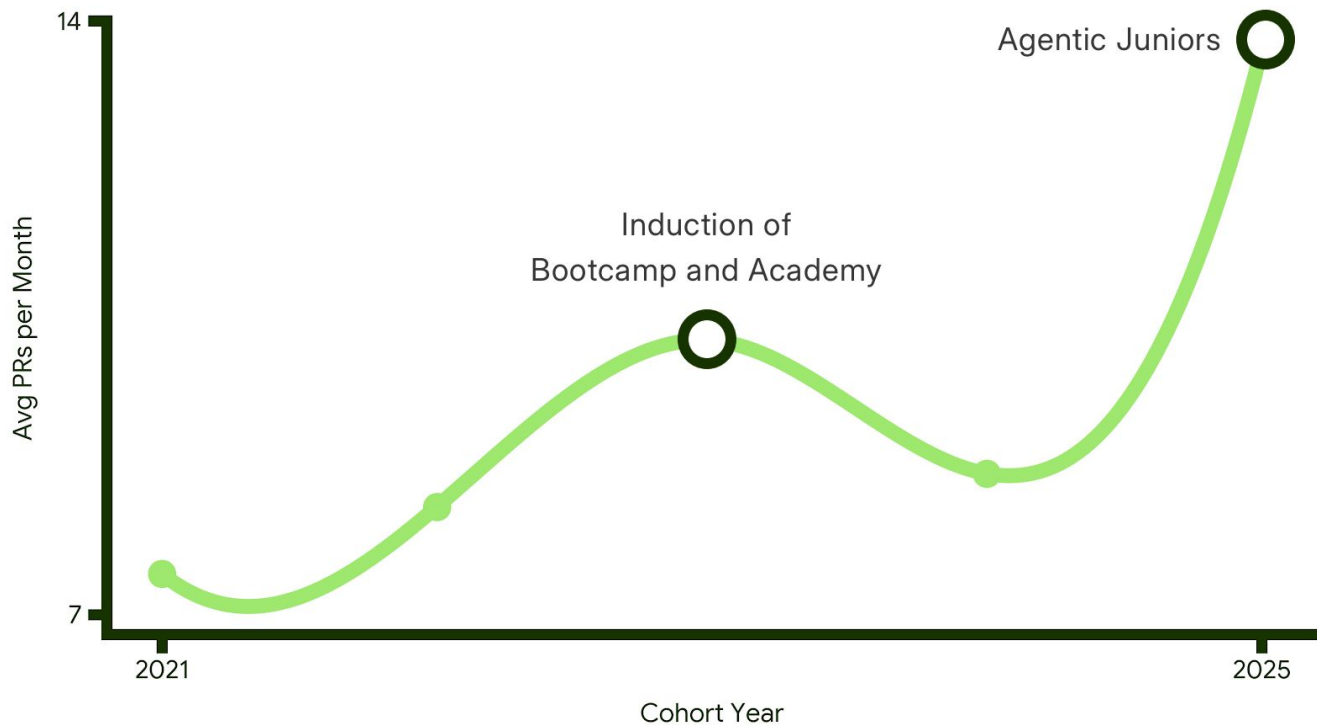
Average PRs per Engineer per Month



Average PRs per Engineer per Month



Average PRs per Engineer per Month



Reduce the burden



code-quality-agent Bot commented • edited ▾

Code Review Summary

Overview

This PR adds a well-structured opt-in feature (`trackReviewedPRs`) to surface PRs that have new commits since a team member's last review. The config threading through Kafka → domain → services → Slack rendering is clean and follows existing patterns.

Key Findings

Bug — Cross-repo PR number collision in awaiting-review exclusion set (high)

`buildAwaitingReviewSet` uses PR number alone as the map key, but PR numbers are repo-scoped. If different repos have the same PR number, a PR in one repo could be incorrectly excluded because another repo's PR with the same number is awaiting review.

Performance — API call fan-out (medium)

The service makes $O(\text{members} \times \text{repos})$ search API calls plus $O(\text{candidates} \times 2)$ detail calls. With 10 team members, 5 repos, and 20 candidate PRs, that's ~90 GitHub API calls per summary run. The PR description notes this is intentional ("1 call per team member per repo"), but the additional per-candidate calls for reviews and commits aren't bounded. Consider adding a candidate cap or batching strategies.

Positive aspects:

- Clean interface-based dependency injection with narrow interfaces (`teamMemberResolver` , `prSearcher` , etc.)
- Graceful degradation — errors are logged as warnings and the feature continues
- Good test coverage for the happy path and key exclusion scenarios
- `shouldSkipNotification` correctly accounts for reviewed PRs so that teams with only re-review PRs still get notified

Tip: To re-trigger a review on future pushes, comment `/wise-review` on this PR.

Note: Wise Review Agent is BETA. This review is generated by AI and mistakes could happen.

Help us improve Wise Review Agent:

- 🗨️ React or reply to inline comments to help us learn what's useful
- 📄 [Submit feedback](#) on this review via our feedback form
- 💡 Share general feedback in [#wise-review-agent-feedback](#)



Wise Agentic Platform

Now

Quality
Security

Testing

Cross Boundary

Next

CI Build

Documentation

Later

Run agentic tasks in the cloud

Wise Brain MCP

Engineering Knowledge Base

Confluence

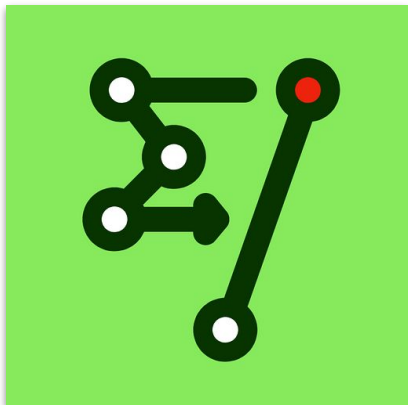
Software Catalogue


Team Data


Service Maturity Checks



Vulnerabilities





 **PR Bot** APP 10:40
📣 PR in `backstage` by `@mulugeta.tamiru`
Simplify Review Agent config UI to single-repo selection
<https://github.com/transferwise/backstage/pull/2004> · `+217/-184` · `4 files` · Bugfix
✅ 1 🔄 1 😊

 **PR Bot** APP 08:31
📣 PR in `platon` by `@oleg.zaiezdnyi`
add `pull_request_review` webhook handler
<https://github.com/transferwise/platon/pull/1423> · `+571/-33` · `20 files` · New feature, Test improvement
✅ 1 🔄 1 😊

 **PR Bot** APP 11:12
📣 PR in `ai-engineering-hub` by `@ricky.clegg`
add management level mapping details to `workday` context
<https://github.com/transferwise/ai-engineering-hub/pull/231> · `+1/-1` · `1 file` · Documentation update
✅ 1 🔄 1 😊
 **2 replies** Last reply 2 days ago

Guardrails

Canary

spinnaker.cd-wi.se/#applications/balance/canary/report

Wise EXP ALL Docs Sheets Slides Drive Credal Gemini NotebookLM All Bookmarks

SPINNAKER Search Applications Tenant Environment Search rickyclegg Help

Wise

- CLUSTERS
- BUILDS
- PIPELINES
- CANARY CONFIGS
- CANARY REPORTS
- TASKS
- CONFIG
- PROFILER

Showing 5 most recent canary intervals

FINAL SCORE	CANARY INTERVALS	APPLICATION	CONFIG
100	2026-05-15 09:02:56 UTC		Pipeline
100	2026-05-12 07:14:13 UTC		Pipeline
85	2026-05-11 09:39:51 UTC		Pipeline
93.33	2026-05-09 10:38:44 UTC		Pipeline
100	2026-05-08 21:55:41 UTC		Pipeline
96.67	2026-05-08 13:35:46 UTC		Pipeline
100	2026-05-08 11:32:32 UTC	balance	balance_canary Pipeline

platon | Maturity | Wise Dev F

COMPONENT — SERVICE
PLATON ☆

Owner Lifecycle

OVERVIEW API DEPENDENCIES RELEASES **MATURITY** CAMPAIGNS DOCS SECURITY

Engineering Maturity

Track and improve your service quality across different capabilities.

Current Level: **GOLD**

UNCERTIFIED > BRONZE > SILVER > GOLD

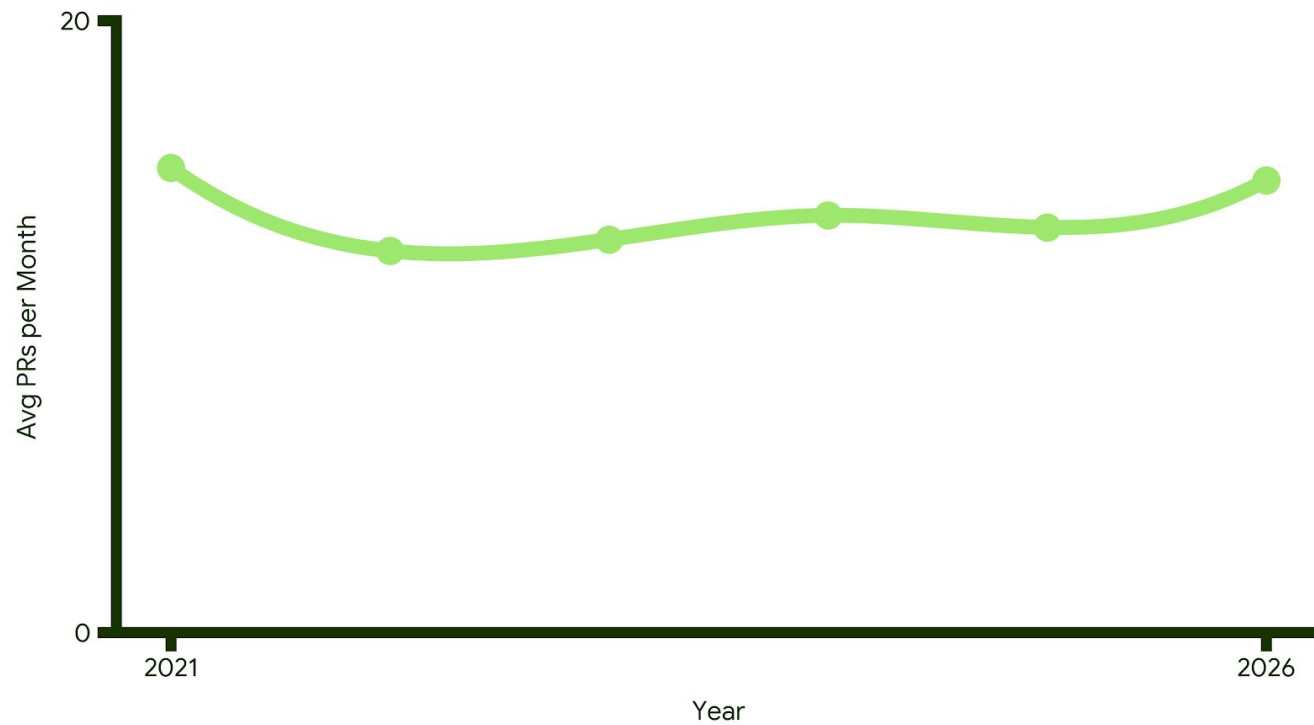
All Delivery Reliability Security Data

✖ **Bronze** (10/12)

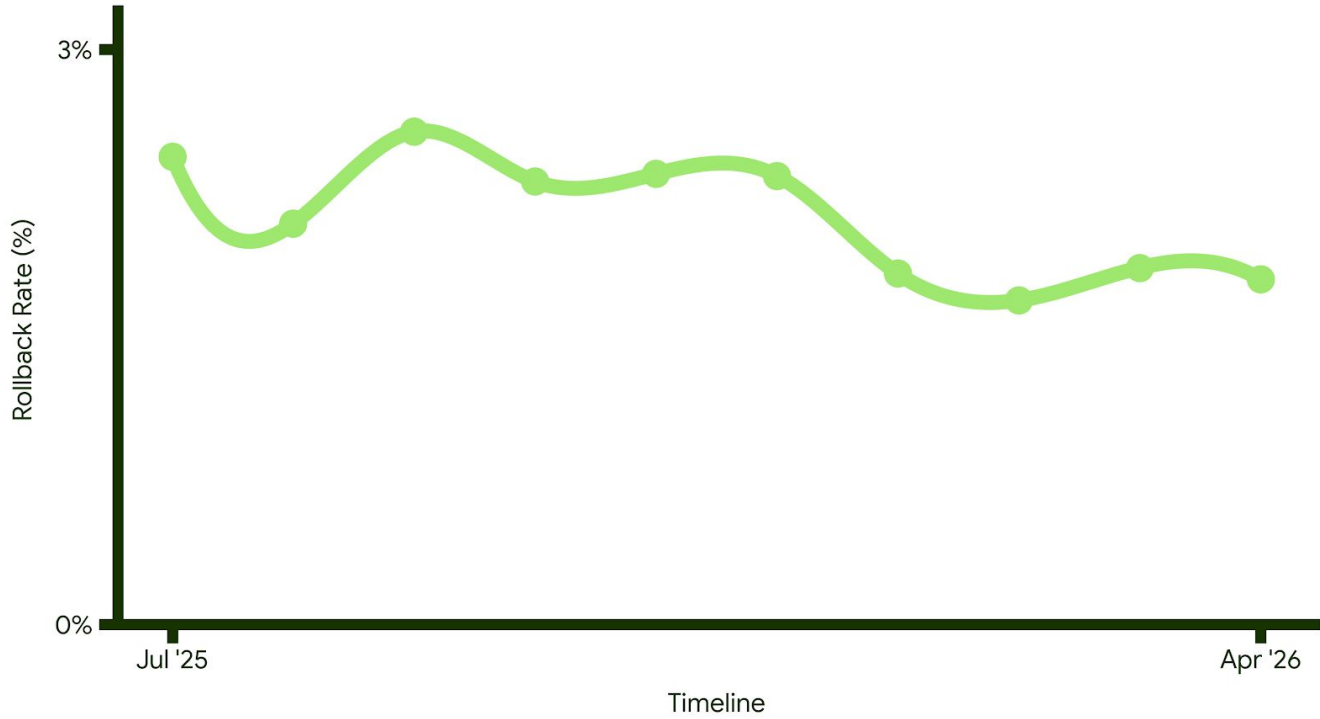
- ✓ All data assets are tagged Data 3 hours ago
- ✓ All unpartitioned database tables are under 1TB in size Data 3 hours ago
- ✓ Architectural decision is documented Reliability 3 hours ago
- ✓ Availability SLO threshold is met Reliability 3 hours ago
- ✓ DORA functions are mapped correctly Reliability 3 hours ago
- ✓ DORA functions have been reviewed within the last year Reliability 3 hours ago

A senior perspective

Average PRs per Month (Non-Juniors)



Change Fail Rate



Takeaways

- Find your best teachers to create a level playing field of skills
- Use MCPs, SKILLS, plugins to bake your engineering standards into your workflows
- Higher quality code, tightly scoped PRs reduces the burden on senior review
- Pair senior and juniors on initiatives working in parallel

