

COELHO NEXUS

A KUBERNETES-NATIVE MULTI-SERVICE AI AGENT
PLATFORM

*DOCS DISTILLER
YOUTUBE CONTENT SEARCH
RESEARCH RADAR*

*FASTAPI
FASTHTML
FASTMCP*

RAFAEL COELHO

BUILT SOLO. PRODUCTION-READY.

3

MICROSERVICES

FastAPI · FastHTML · FastMCP

6

DATA STORES

PostgreSQL · Neo4j · Qdrant
Elasticsearch · MinIO · Redis

18

TERRAFORM MODULES

K3d · ArgoCD · Helm
Full GitOps IaC

3

AI FEATURES

Docs Distiller
YouTube Search
Research Radar

100%

FREE-TIER LLMS

20+ models · Bandit routing
BYOK encryption

1

ENGINEER

Code → Infra → GitOps
End-to-end ownership

ABOUT THE PROJECT

Production-grade, Kubernetes-native AI Agent Platform — built solo, shipped end-to-end.

- 3 microservices (FastAPI, FastHTML, FastMCP) fused into a unified intelligence layer
- 100% free-tier LLMs — 20+ models, bandit routing, BYOK (Bring Your Own Key) credential management
- Full OpenTelemetry + LangFuse observability — every LLM call traced, every token accounted for



PLATFORM ARCHITECTURE

01.

THREE MICROSERVICES & THREE FEATURES

FastAPI + Celery orchestrate six isolated task queues, one per AI feature pipeline.

FastHTML delivers an HTMX-in-SSR (Hypertext eXtensions + Server-Side Rendering) reactive UI with Cytoscape graph visualizations and SSE (Server-Sent Events) live event streaming.

FastMCP exposes four AI agent discovery tools with rate-limited token buckets and BYOK (Bring Your Own Key) credential injection.

02.

INFRASTRUCTURE AS CODE & SOLO OWNERSHIP

18 Terraform modules provision the full data layer — PostgreSQL, Neo4j, Qdrant, Elasticsearch, MinIO, Redis — declaratively via Terragrunt on K3d.

ArgoCD automates the full GitOps delivery pipeline; OpenTelemetry dual export feeds LangFuse trace visualization and Grafana dashboards.

THE PLATFORM IN ACTION

A visual walkthrough of the COELHO Nexus platform — three production-grade AI features (Docs Distiller · YouTube Content Search · Research Radar) built on FastAPI, FastHTML, and FastMCP, orchestrated on Kubernetes with full LangFuse and OpenTelemetry observability.

COELHO
NEXUS



DOCS DISTILLER

Catalog
Ingestion
Pipeline
Study

COELHO
NEXUS

DOCS DISTILLER

The full depth of 400+ pages of official documentation — distilled from weeks of reading into hours of study.

1.

Pick any framework and it downloads the complete official documentation — every page, automatically cleaned into Markdown

2.

Planner reads the entire corpus and organizes it into a textbook outline — the structure a human expert would design

3.

Synth writes each chapter from the real source material — code examples preserved, checked against the docs for accuracy

Why it matters:

- The same depth as reading every page yourself — at a fraction of the time, because the work is already done.
- Zero cost: Every LLM call — reading, planning, writing — automatically routes to the best free-tier model available. No API bills, ever.

DOCS DISTILLER - CATALOG

COELHO Nexus

HomeDocs DistillerYouTube Content SearchResearch Radar

Docs DistillerCatalogIngestionPipelineStudy

Search 107 frameworks...107 of 107

CATEGORY: All

ARUNDO
ADTK
PYTHON

ALIBI EXPLAIN
Alibi Explain
PYTHON

A

Ansible
—

Apache Airflow

Apache Airflow
PYTHON

Argo CD

—

Python

✓ INGESTED

Asyncio
PYTHON

BASH
Bash
PROGRAMMING LANGUAGE

Browser Use

✓ INGESTED

Browser Use
PYTHON

CatBoost

CatBoost
PYTHON

Celery (Python)

✓ INGESTED

Celery (Python)
PYTHON

Claude Code Docs

✓ INGESTED

Claude Code
—

Crawl4AI

Crawl4AI
PYTHON

dagster docs

Dagster
PYTHON

dask

Dask
PYTHON

dbt Labs

dbt
—

DELTA LAKE

Delta Lake (Python)
PYTHON

docker

Docker
—

DuckDB

DuckDB (Python)
PYTHON

elasticsearch

ElasticSearch (Python)
PYTHON

EVIDENTLY AI

Evidently
PYTHON

FastAPI

FastAPI
PYTHON

Qdrant

FastEmbed
PYTHON

FastHTML

✓ INGESTED

FastHTML
PYTHON

FASTMCP

FastMCP
PYTHON

git

GO

GO

HELM

K3D

Selected: DuckDB (Python)

Start Ingestion

1 Search field

Real-time search across 100+ catalogued documentation sources. Partial-match filtering instantly narrows the catalog by framework name or technology domain.

2 Category filter

Filter the catalog by technology domain — AI/ML, databases, infrastructure, DevOps, monitoring. Groups semantically related frameworks for focused multi-source ingestion runs.

3 Ingested option

Marks frameworks already ingested into the knowledge base. The badge identifies gaps and lets you trigger re-ingestion when documentation is updated.

4 Ingestion button

Triggers the 4-tier ingestion pipeline for selected sources — lms_full.txt → lms_txt → sitemap → httpx-crawl-with-Playwright-fallback — with real-time progress streamed via SSE to the Pipeline view.

DOCS DISTILLER - INGESTION

COELHO Nexus

HomeDocs DistillerYouTube Content SearchResearch Radar

Docs Distiller

CatalogIngestionPipelineStudy

Scaffold · 80 pages · 562.2 KB · sitemap · 1s ago

Search pages... (/)

Installing Scaffold | Scaffold · sitemap · 6.5 KB

▼ LIFECYCLE-HOOKS · 1

Lifecycle Hooks | Scaffold · sitemap · 6.7 KB

▼ LOG-TAILING · 1

Log Tailing | Scaffold · sitemap · 2.7 KB

▼ PIPELINE-STAGES · 1

Scaffold Pipeline Stages | ... · sitemap · 2.1 KB

▼ PORT-FORWARDING · 1

Port Forwarding | Scaffold · sitemap · 4.8 KB

▼ QUICKSTART · 1

Quickstart | Scaffold · sitemap · 9.8 KB

▼ REFERENCES · 9

API | Scaffold · sitemap · 733 B

API V2 | Scaffold · sitemap · 748 B

CLI | Scaffold · sitemap · 83.3 KB

80 of 80 pages

Scaffold > references > CLI | Scaffold

sitemap · 83.3 KB > Open in drawer

CLI

Scaffold command-line interface provides the following commands:

End-to-end pipelines:

• [scaffold run](#) - to build & deploy once

• [scaffold dev](#) - to trigger the watch loop build & deploy workflow with cleanup on exit

• [scaffold debug](#) - to run a pipeline in debug mode

Pipeline building blocks for CI/CD:

• [scaffold build](#) - to just build and tag your image(s)

• [scaffold deploy](#) - to deploy the given image(s)

• [scaffold delete](#) - to cleanup the deployed artifacts

• [scaffold render](#) - build and tag images, and output templated Kubernetes manifests

• [scaffold apply](#) - to apply hydrated manifests to a cluster

Getting started with a new project:

• [scaffold init](#) - to bootstrap Scaffold config

• [scaffold fix](#) - to upgrade from older scaffold.yaml schema version to newer scaffold.yaml schema version

Other Commands

1 Search pages

Search within the ingested documentation pages of the selected framework. Instantly locates specific sections, commands, or API references within the full fetched documentation tree.

2 Fetched docs pages

Complete list of documentation pages fetched from the source URL. Each entry shows the page title and ingestion tier used — llms_full.txt, llms_txt, sitemap, httpx-crawl-with-Playwright-fallback.

3 Ingested frameworks dropdown

Switch between all frameworks currently in the knowledge base. Selecting a framework updates the page list and main view to show its complete documentation structure.

4 Main page visualization

Full rendered view of the selected documentation page as captured and processed. Shows the raw content before it enters the Planner → Synth pipeline for study guide generation.

COELHO
NEXUS

DOCS DISTILLER - INGESTION

COELHO Nexus

HomeDocs DistillerYouTube Content SearchResearch Radar

Docs Distiller

CatalogIngestionPipelineStudy

Scaffold · 80 pages · 562.2 KB · sitemap · 1s ago

Search pages... (/)

(ROOT)2

Scaffold · sitemap · 2.9 KB

Scaffold 2.0 Documentatio... · sitemap · 5.5 KB

BUILDERS12

Bazel | Scaffold · sitemap · 2.5 KB

Build | Scaffold · sitemap · 2.0 KB

Build environments | Scaffold · sitemap · 489 B

Builder types | Scaffold · sitemap · 528 B

Cloud Native Buildpacks | ... · sitemap · 4.5 KB

Cross-platform and multi-... · sitemap · 2.9 KB

Custom Build Script | Skaff... · sitemap · 9.4 KB

Docker Build | Scaffold · sitemap · 16.8 KB

Google Cloud Build | Skaff... · sitemap · 6.1 KB

In cluster build | Scaffold · sitemap · 3.2 KB

Jib Build | Scaffold · sitemap · 9.2 KB

80 of 80 pages

Scaffold > references > CLI | Scaffold

CLI

Scaffold command-line interface provides the following commands:

End-to-end pipelines:

• skaffold run - to build & deploy once

• skaffold dev - to trigger the watch loop build & deploy workflow with cleanup on ex

• skaffold debug - to run a pipeline in debug mode

Pipeline building blocks for CI/CD:

• skaffold build - to just build and tag your image(s)

• skaffold deploy - to deploy the given image(s)

• skaffold delete - to cleanup the deployed artifacts

• skaffold render - build and tag images, and output templated Kubernetes manifest

• skaffold apply - to apply hydrated manifests to a cluster

Getting started with a new project:

• skaffold init - to bootstrap Skaffold config

• skaffold fix - to upgrade from older skaffold.yaml schema version to newer skaffold.yaml schema version

Other Commands

2

Search ingested frameworks...

1

Scaffold

80 pages · 15m ago

3

4

Celery (Python)

493 pages · 6d ago

3

4

FastHTML

14 pages · 6d ago

3

4

Browser Use

45 pages · 6d ago

3

4

Pydantic

308 pages · 7d ago

3

4

Asyncio

77 pages · 8d ago

3

4

Claude Code

195 pages · 8d ago

3

4

- 1

Ingested frameworks dropdown options

Lists all documentation sources currently in the knowledge base, each showing its framework name, and total page count for context planning.
- 2

Search ingested frameworks options

Real-time search within the framework picker. Filter across all ingested documentation sources by name to quickly locate and switch to any framework in the knowledge base.
- 3

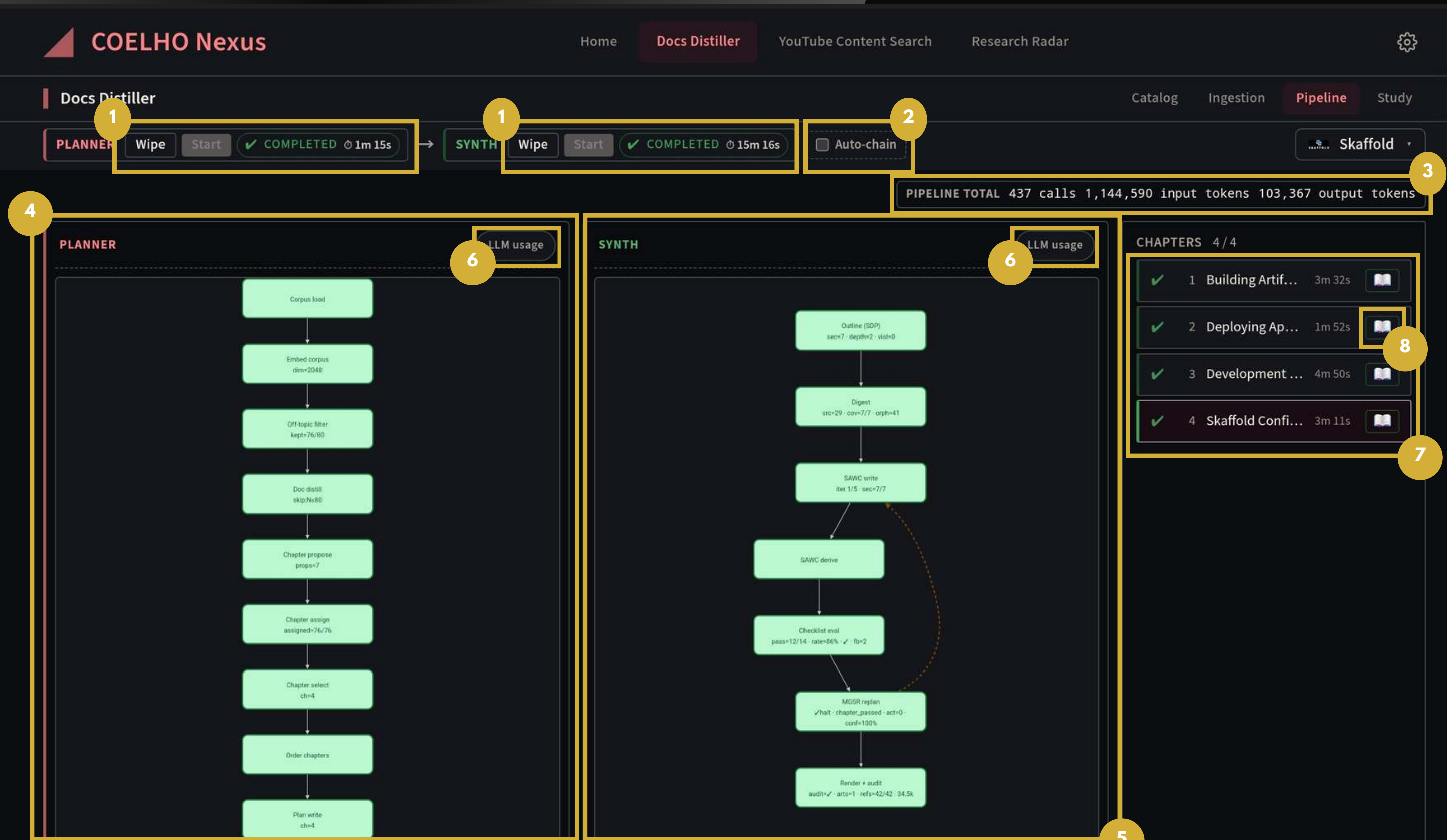
Refresh framework docs ingestion

Re-runs the full 4-tier ingestion pipeline for the selected framework. Use when upstream documentation has been updated to sync the knowledge base with the latest content.
- 4

Delete framework docs ingestion

Removes all ingested pages for the selected framework from the knowledge base — clearing Qdrant vectors, chunk metadata, and tier records — enabling a clean re-ingestion from scratch.

DOCS DISTILLER - PIPELINE



1 Pipeline control bar

Controls for starting, stopping, and wiping the Planner and Synth pipelines. The status bar shows real-time pipeline state (RUNNING / COMPLETED / IDLE) and elapsed time per stage.

2 Auto-chain option

When enabled, automatically triggers the Synth pipeline immediately after Planner completes — running the full Planner → Synth sequence end-to-end without manual intervention.

3 Run completion stats

Aggregate run statistics: total documents processed, chapters proposed, chapters assigned, and synthesis iterations completed across the full pipeline execution.

4 LangGraph Planner

LangGraph planning pipeline that distills ingested documents into a structured chapter outline — LLM proposes chapters from parallel summaries, assigns each doc to its best fit, then selects for maximum corpus coverage.

5 LangGraph Synth

LangGraph synthesis pipeline that expands the Planner outline into full study guide chapters via SAWC (Section-Aware Writer-Critic) — iterative drafting with LLM critic until quality gates pass.

6 LLM telemetry per stage

Per-node LLM call telemetry: model used, tokens consumed, latency, and bandit arm selection. Routes through the FGTS-VA (Fine-Grained Thompson Sampling with Variance Awareness) bandit rotator across 20+ free-tier models.

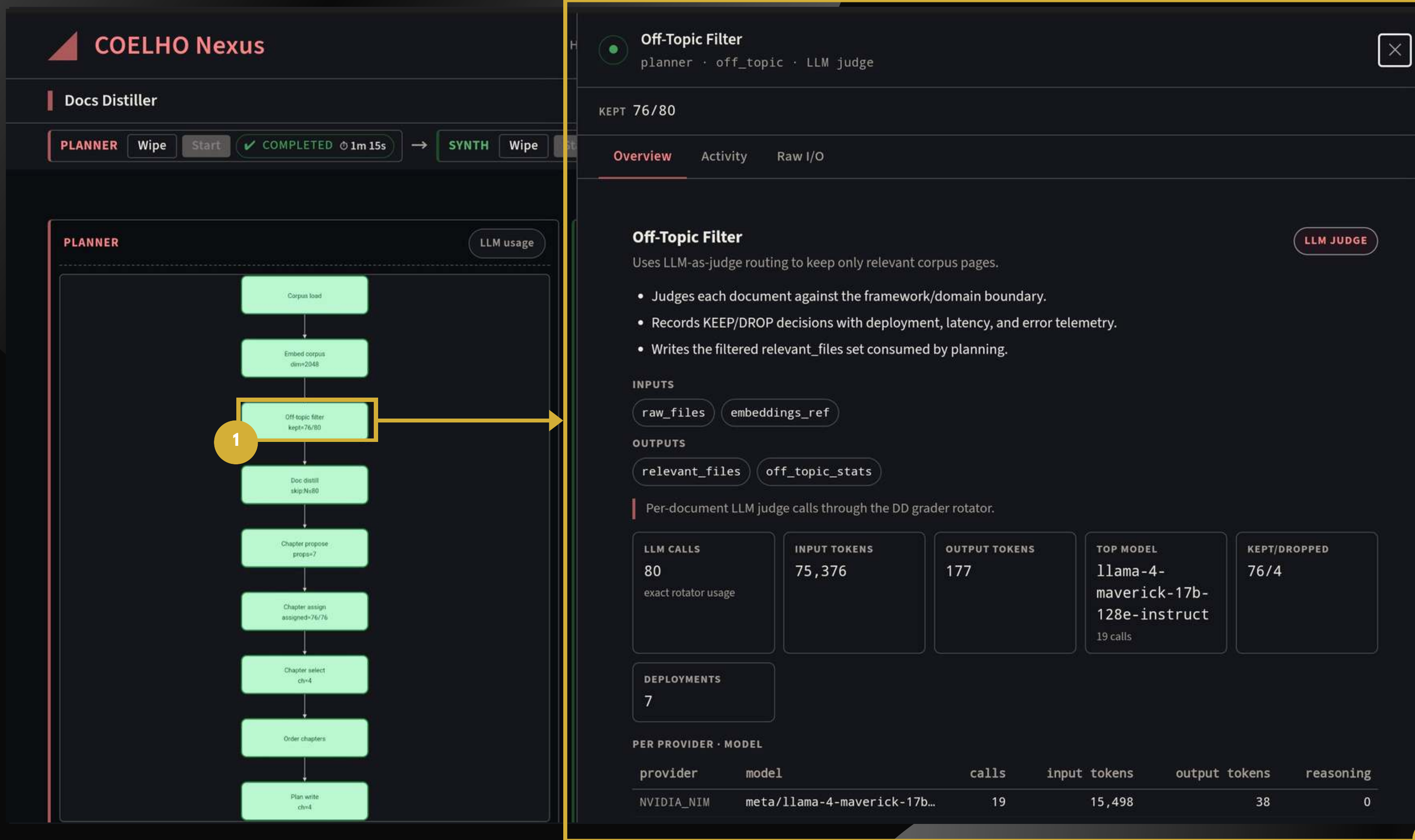
7 Study chapters synthesized

Running count of completed study guide chapters. Each chapter is independently synthesized and streamed to the Study view as it finishes — no need to wait for the full run.

8 Chapter reading button

Opens the completed study guide chapter directly in the Study view. Available as each chapter finishes synthesis, enabling progressive reading before the full run completes.

DOCS DISTILLER - PIPELINE



1 Clickable LangGraph node

Click any node in the LangGraph Planner/Synth flow to inspect its execution — input/output documents, filtering decisions, LLM calls, and token usage per node.

2 Node execution telemetry

Real-time execution detail for the selected Planner/Synth node — judge configuration, input/output document counts, total tokens consumed, latency distribution, and per-document LLM call breakdown.

DOCS DISTILLER - PIPELINE

COELHO Nexus

Home

Docs Distiller

PLANNER

Wipe

Start

COMPLETED 1m 15s

→

SYNTH

Wipe

Start

PLANNER

LLM usage

Corpus load

Embed corpus
dim=2048

Off-topic filter
kept=76/80

Doc distill
skip=N=80

Chapter propose
props=7

Chapter assign
assigned=76/76

Chapter select
chs=4

Order chapters

Plan write
ch=4

Planner LLM usage

Latest planner run, grouped by node and provider/model.

PLANNER LLM USAGE

Latest planner run

Updated from graph checkpoints

160

CALLS

212,819

INPUT TOKENS

12,458

OUTPUT TOKENS

2,376

REASONING

OFF_TOPIC 80

CHAPTER_ASSIGN 76

ORDER_CHAPTERS 3

CHAPTER_PROPO... 1

provider	model	calls	input tokens	output tokens	reasoning
NVIDIA_NIM	meta/llama-4-maverick-...	42	74,993	2,924	0
GROQ	meta-llama/llama-4-sco...	36	45,151	2,712	0
MISTRAL	mistral-medium-3	27	29,055	1,241	0
NVIDIA_NIM	qwen/qwen3-next-80b-a3...	25	26,138	1,600	0
GROQ	llama-3.3-70b-versatile	11	15,254	792	0
NVIDIA_NIM	moonshotai/kimi-k2.6	8	7,076	125	0
NVIDIA_NIM	qwen/qwen3.5-397b-a17b	4	3,786	8	0
GROQ	openai/gpt-oss-120b	3	5,320	1,428	1,143
GROQ	openai/gpt-oss-20b	3	4,666	1,518	1,233
NVIDIA_NIM	nvidia/llama-3.3-nemot...	1	1,380	110	0

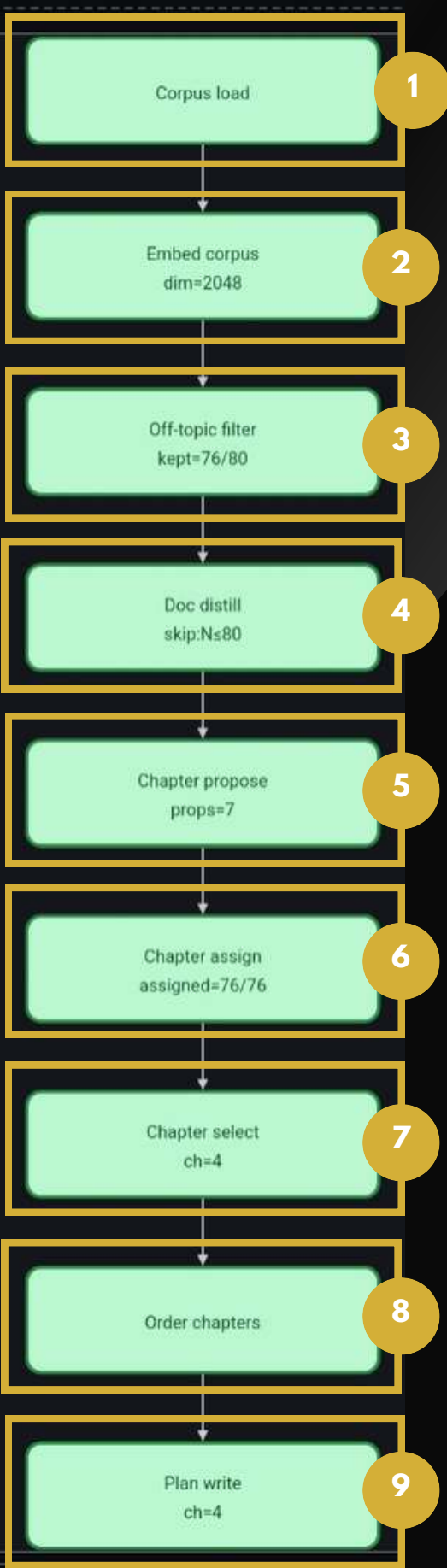
- 1

Planner/Synth LLM usage panel

Opens the LLM token summary for the last Planner/Synth run — total calls, input/output/reasoning tokens, and per-node model breakdown grouped by bandit algorithm arm.
- 2

Bandit algorithm routing telemetry

Per-node, per-provider token accounting — shows which model the FGTS-VA (Fine-Grained Thompson Sampling with Variance Awareness) bandit routed each call to, with input/output token counts. Full traceability across 20+ free-tier models.



1 **Corpus Inventory**

Reads the finalized MinIO ingestion manifest, resolves per-page object keys, and computes byte-size stats — the source of truth that seeds all downstream nodes.

2 **Semantic Embedding**

Token-aware BPE (Byte-Pair Encoding) chunking aligned to the NVIDIA NIM embedding model → L2-normalized dense vectors packed as .npz in MinIO. Hash-skipped on re-runs.

3 **Off-Topic Filter**

Bandit-routed LLM judge (KEEP / DROP) with cosine anchor similarity and concurrent per-doc evaluation. Defaults to KEEP on failure — no content silently dropped.

4 **Document Distillation**

Parallel LLM calls produce structured distillates (summary + key terms) per doc. Repair loop + fallback ensure full corpus coverage under rate-limits or parse failures.

5 **Chapter Proposal**

Long-context LLM samples N chapter structures from all distillates + H2 structural seeds. USC (Universal Self-Consistency) voting aggregates proposals; optimal stopping halts early when consensus converges.

6 **Document Assignment**

Per-doc LLM confidence scoring across all proposed chapters. Lexical fallback + rescue floor guarantee every content doc gets assigned — no silent drops.

7 **Coverage Selection**

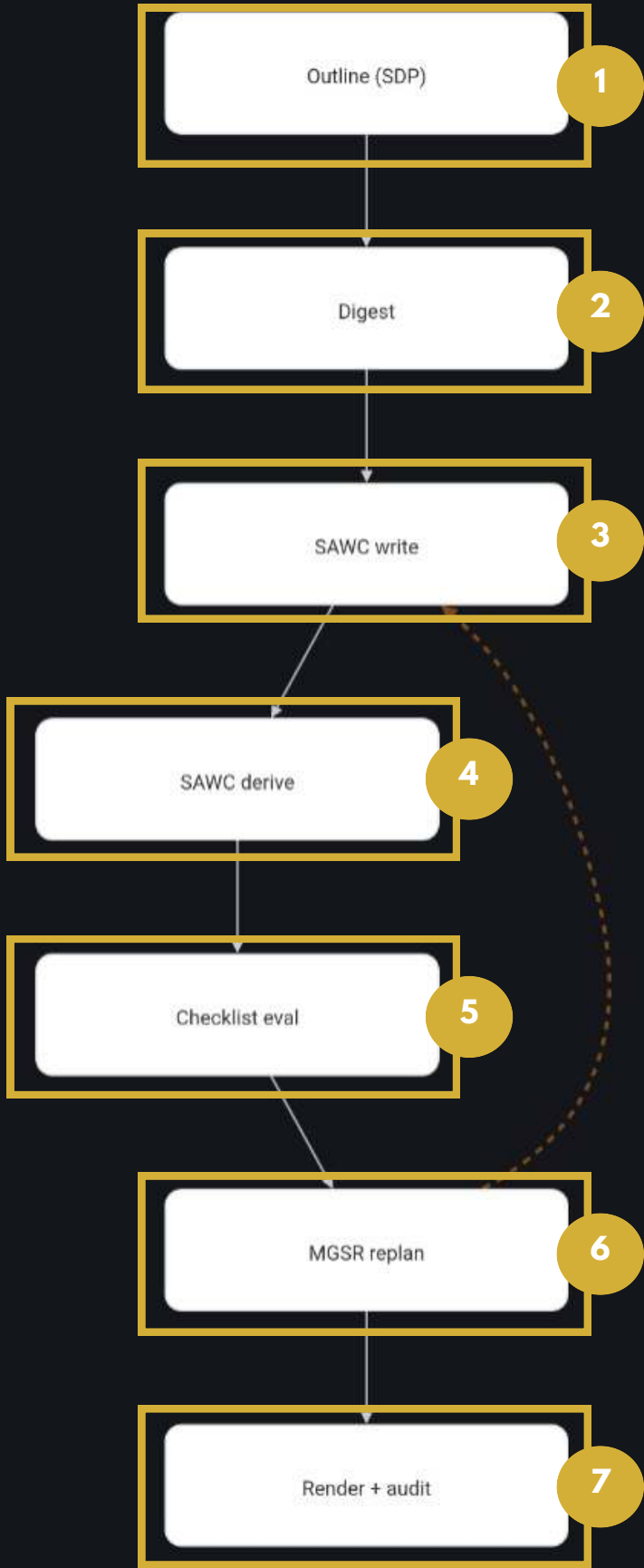
Greedy algorithm selects the minimal chapter set covering all docs above the minimum threshold. Writes versioned + latest plan blobs to MinIO for resume-safety.

8 **Pedagogical Ordering**

N bandit-routed LLM samples rank chapter sequences independently. Borda aggregation + foundational-prefix rule ensure foundational concepts always precede advanced material.

9 **Plan Assembly**

LLM-free final step: sanitizes chapter metadata, inlines provenance refs, and persists the versioned + latest plan blob. Output feeds directly into the Synth pipeline.



1 Outline Generation

LLM generates the chapter outline as a dependency DAG (Directed Acyclic Graph) — H2 sections with prerequisite links, adaptive section count, and fuzzy-dedup headings. The structured skeleton all downstream nodes build from.

2 Source Digest

24-concurrent bandit-routed LLM call per source doc — extracts key facts, section contributions, and relevance scores. Content-addressed cache avoids redundant calls on re-runs.

3 SAWC Writer-Critic

Best-of-N LLM drafts per section + critic picker (MAMM-Refine, arXiv:2503.15272). Structured output: heading, intro, subtopics with code refs and citations. 2-attempt repair loop enforces SAWC (Section-Aware Writer-Critic) alignment.

4 Derived Code Enrichment

Analogical Prompting (arXiv:2310.01714) detects thin vault blocks and generates derived business code examples via MPSC (Multi-Path Self-Consistency) sampling — ensures every section has implementable code.

5 Quality Checklist

Multi-criteria pass/fail gating: density, code ref coverage, citation density, LLM-judge criteria, CoCoA (Code Comprehension then Auditing, arXiv:2410.03131) alignment, and faithfulness scoring. Generates per-criterion feedback for the replan loop.

6 MGSR (Meta-Grounded Self-Refinement)

Reads checklist failures and issues structured replan actions. Trivial-pass fast-path at ≥80% skips LLM entirely. On failure: bandit-routed replan → conditional edge loops back to SAWC (Section-Aware Writer-Critic) write.

7 Render & Audit

Resolves vault sentinel hashes to byte-exact code blocks, runs cached LLM normalization, deduplicates sections, computes audit metrics, and persists the final chapter Markdown artifact.

DOCS DISTILLER - STUDY

COELHO Nexus

HomeDocs DistillerYouTube Content SearchResearch Radar

Docs Distiller

CatalogIngestionPipelineStudy

Search

Scaffold

CHAPTERS

4 / 4 synthesized15m 16s

Building Artifacts3m 32s

Deploying Applications1m 52s

Development Workflow4m 50s

Scaffold Configuration3m 11s

Building Artifacts

Audit ✓5 sections20 sources23.5k chars

Building Artifacts

Builder Types and Execution Environments

Scaffold supports six builder types (Dockerfile, Jib, Buildpacks, Bazel, ko, Custom) across three execution environments (local, in-cluster, Google Cloud Build). This section maps those builders to their available environments and shows how to configure them.

Local Build Default Configuration

The `local` key under the `build` section explicitly sets the execution environment to local, which is Scaffold's default context using locally-installed tools like Docker or Bazel.

```
build:
  local: {}
```

Custom Builder Script and Dependencies

The `custom` builder type uses a `buildCommand` (here `./build.sh`) to execute builds. The `dependencies` block with `paths` and `ignore` tells Scaffold which files to watch for changes.

ON THIS PAGE

Builder Types and Execution Environments

Local Build Default Configuration

Custom Builder Script and Dependencies

Targeting amd64 Platform Locally

Targeting arm64 Platform Remotely

Docker-Based Builds: Local, Kaniko, and Cloud Build

Local Docker Build Configuration

Docker Cache Configuration

Kaniko Build with Dependencies

Kaniko Pull Secret Configuration

Google Cloud Build Configuration

Cluster Pull Secret Configuration

Jib, Bazel, and Buildpacks for Non-Docker Builds

Buildpacks builder image selection

Buildpacks file dependencies and ignore patterns

Bazel artifact target for Docker load

Bazel cross-platform builds via platforms list

Jib multi-module build with artifact dependencies

COELHO
NEXUS

YOUTUBE CONTENT SEARCH

Source
Ingestion
Ask
Query

COELHO
NEXUS

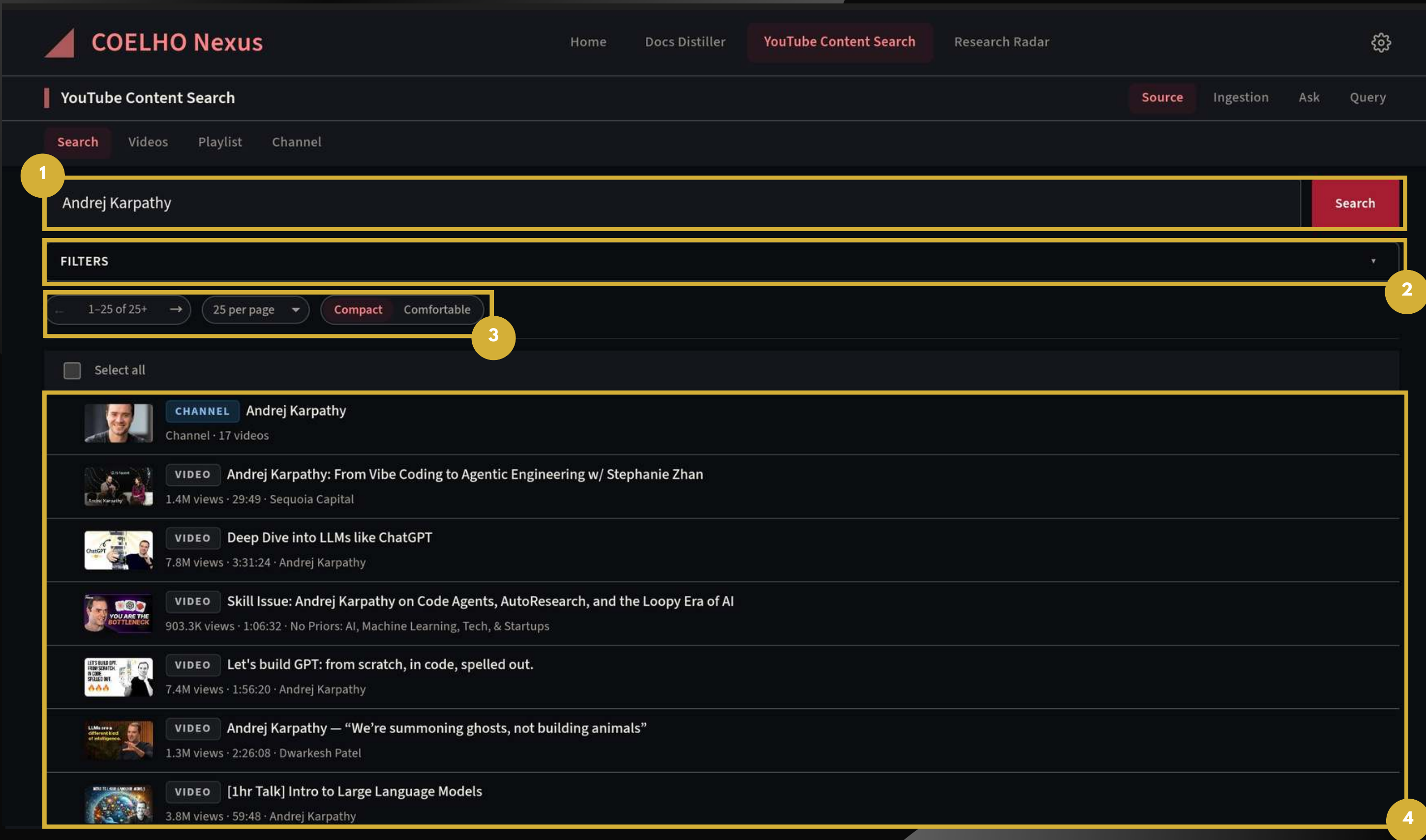
YOUTUBE CONTENT SEARCH

Information locked inside YouTube videos — invisible to Google, unreachable by ChatGPT or Claude — finally searchable.

1. Point it at any channel and it extracts full transcripts straight from the video page — the same content Google can't index
2. Every transcript is indexed three ways — semantic, graph, and full-text (Qdrant + Neo4j + Elasticsearch) — connections surface automatically
3. Ask a question and get an answer synthesized across every matching video — with the exact source clips cited, not a generic summary

Why it matters: Cross-video insight that Google and general AI assistants can't give you — they can't see inside YouTube's transcripts.

YOUTUBE CONTENT SEARCH - SOURCE



1 yt-dlp Live Search

Quota-free YouTube search via yt-dlp — returns videos, channels, and playlists with title, channel, view count, duration, likes, and upload date in a single call. Sub-2-second results piped directly into the multi-row ingestion selector.

2 Native HTML Filter Grid

Nine search parameters — duration range, content kind, upload date window, view and like thresholds, title keyword, and channel name — applied server-side via native HTML form controls on every yt-dlp call. No JavaScript event handlers, no client-side filter state.

3 Results Controls

Pagination with configurable page size (10–200 per query), prev/next controls with a live count display, and Compact/Comfortable density toggle for scanning large result sets fast.

4 Video Result Cards

Each card surfaces thumbnail, channel name, view count, duration, like count, and upload date. Per-row checkbox plus master Select All feeds the Start Ingestion pipeline with exactly the videos you choose.

COELHO NEXUS

YOUTUBE CONTENT SEARCH - SOURCE

COELHO Nexus

Home

Docs Distiller

YouTube Content Search

Research Radar

YouTube Content Search

SourceIngestionAskQuery

Search

Videos

Playlist

Channel

Andrej Karpathy

Search

FILTERS

DURATION

Any duration

KIND

All kinds

UPLOADED AFTER

UPLOADED BEFORE

MIN VIEWS

e.g. 10000

MAX VIEWS

e.g. 1000000

MIN LIKES

e.g. 100

TITLE CONTAINS

Plain text or *=op

CHANNEL NAME

Creator or channel

☐ Sort by newest

☐ Exclude shorts

1-25 of 25+ →

25 per page ▾

Compact

Comfortable

☐ Select all



CHANNEL

Andrej Karpathy

Channel · 17 videos



VIDEO

Andrej Karpathy: From Vibe Coding to Agentic Engineering w/ Stephanie Zhan

1.4M views · 29:49 · Sequoia Capital



VIDEO

Deep Dive into LLMs like ChatGPT

7.8M views · 3:31:24 · Andrej Karpathy



VIDEO

Skill Issue: Andrej Karpathy on Code Agents, AutoResearch, and the Loopy Era of AI

1 Parametric Search Controls

Nine real-time search parameters — duration range, content kind (video/channel/playlist), upload date bounds, view range and like minimum, title keyword, and channel filter — applied on every yt-dlp query via native HTML form controls.

YOUTUBE CONTENT SEARCH - SOURCE

COELHO Nexus

HomeDocs DistillerYouTube Content SearchResearch Radar

YouTube Content Search

SourceIngestionAskQuery

SearchVideosPlaylistChannel

1

96jN2OCofLs
LCEmiRjPEtQ
qh0FkdgieS8
l2ZK3ngNvvl
au3vwA6lca

Fetch videos

Pasted videos

5 videos
5 of 5 loaded

☒ Select all loaded

Filter by title or channel...

☒ Andrej Karpathy: From Vibe Coding to Agentic Engineering w/ Stephanie Zhan
1.4M views · 2026-04-2929:49

☒ Andrej Karpathy: Software Is Changing (Again)
2.5M views · 2025-06-1939:31

☒ The Most Talented Man in AI
355.2K views · 2026-05-319:20

☒ Advice for machine learning beginners | Andrej Karpathy and Lex Fridman
1.2M views · 2022-10-315:47

☒ Heroes of Deep Learning: Andrew Ng interviews Andrej Karpathy
199.3K views · 2017-08-0815:11

☒ Fetch transcripts

Languages

en, pt, es

Loaded 5 of 5 videos.

2

3

Ingest all 5

Start Ingestion (5 videos)

1 Batch URL Input

Paste YouTube video IDs, short URLs, or full watch URLs — one per line, 50+ entries per batch. Click Fetch Videos to pull title, channel, view count, duration, likes, and upload date via yt-dlp with zero API quota.

2 Fetched Video Cards

Each card shows thumbnail, title, channel, view count, duration, and upload date. Per-row checkbox and master Select All let you pick exactly which videos to ingest before starting the pipeline.

3 One-Click Pipeline Trigger

Ingest All selects every fetched video. Start Ingestion dispatches the Celery chain: yt-dlp extract → Qdrant vector upsert → Neo4j entity graph → cache invalidation. Count badge updates in real time as videos are selected.

COELHO Nexus

HomeDocs DistillerYouTube Content SearchResearch Radar

YouTube Content Search

SourceIngestionAskQuery

Library · 4 videos

Search title / channel / descriptor

STATUS: All

CHANNELS: All

LANGUAGES: All

Clear

PIPELINE 0:00

StopRetryWipe cacheDismissVIDEOS 5 →

Phase 1 · ElasticSearch

Done100%

5 metadata · 5 transcripts in ES (5 new · 0 cached)

Phase 2 · Qdrant

Done100%

5 transcripts · 67 chunks · 67 points

Phase 3 · Neo4j

Done100%

310 nodes · 264 rels · 11 merged

Select all

#1 AI ENGINEER

The Most Talented Man in AI

Newsthink · 355.4K views · 9:20 · 9.9K likes · 2026-05-31

10.4K chars · en · 62 entities

Andrej Karpathy

Andrej Karpathy: From Vibe Coding to Agentic Engineering w/ Stephanie Zhan

Sequoia Capital · 1.4M views · 29:49 · 28.1K likes · 2026-04-29

33.1K chars · en · 106 entities

Software in the era of AI

Andrej Karpathy: Software Is Changing (Again)

Y Combinator · 2.5M views · 39:31 · 57.7K likes · 2025-06-19

43.0K chars · en · 100 entities

10,000 hours

Advice for machine learning beginners | Andrej Karpathy and Lex Fridman

Lex Clips · 1.2M views · 5:47 · 34.8K likes · 2022-10-31

6.1K chars · en · 31 entities

Each card shows thumbnail, title, channel, view count, duration, and live ingestion status. Failed videos expose the broken phase inline so you can retry a single video without re-running the full batch.

YOUTUBE CONTENT SEARCH - ASK

The screenshot shows the COELHO Nexus web application. The top navigation bar includes 'Home', 'Docs Distiller', 'YouTube Content Search' (highlighted), and 'Research Radar'. The main header for 'YouTube Content Search' contains a mode selector (1) with 'Auto', 'Fast', 'Standard', and 'Deep' options; a scope selector (2) set to 'All channels'; and a thread selector (3) with '+ New thread' and 'THREAD: New'. The main content area shows a user query: 'What are the main themes across all videos?'. Below this, the 'ASSISTANT' section (4) is in 'DEEP' mode, displaying a 'THINKING' status and a 'RESEARCH PLAN' with four sub-questions: 'What are the most frequently mentioned keywords and phrases across all video transcripts?', 'How do speakers' tones and language usage vary across different topics and themes?', 'What are the common narratives, storylines, or ideas that emerge from the videos?', and 'Are there any recurring emotional or sentimental themes that are expressed by the speakers?'. At the bottom, there is a text input field (5) with the placeholder 'Ask anything about the indexed videos...', a 'Mode: DEEP' indicator, and a 'Stop' button.

1 Query Mode Selector

Four depth modes — Auto, Fast, Standard, and Deep — balance speed against answer quality. Auto picks the best mode per question; Deep runs full multi-step RAG with sub-question decomposition and source verification.

2 Channel Scope

Scope any query to all indexed channels or a specific one. Narrows vector retrieval and BM25 search to a focused corpus — ideal for drilling into a single creator's content or comparing themes across channels.

3 Conversation Threads

Each thread maintains its own conversation history and context window. Create a new thread to start fresh or switch between saved threads to resume a prior research session without losing context.

4 Deep RAG Pipeline

Deep mode decomposes the question into sub-questions, runs RETRIEVE → GRADE → GENERATE → VERIFY across indexed transcripts, and streams a grounded answer with inline sources. The research plan and step status update in real time.

5 Natural Language Input

Ask anything about the indexed video library in plain language. The active query mode is shown inline; Stop halts a running generation mid-stream without losing the partial answer.

YOUTUBE CONTENT SEARCH - ASK

COELHO Nexus

HomeDocs DistillerYouTube Content SearchResearch Radar

YouTube Content Search

SourceIngestionAskQuery

AutoFastStandardDeepSCOPE: All channels

+ New threadTHREAD: dc80ccd92200

YOU

What are the main themes across all videos?

ASSISTANT DEEP in 9m 51s

THINKING

TL;DR: The videos chart a transition to AI-driven software—“vibe coding” and agentic engineering—while stressing new documentation practices, the democratizing promise of LLMs, and the need to manage their limits, all delivered in a mix of conversational, visionary, and pragmatic tones.

AI-driven software is the core narrative

The speakers frame LLMs and AI agents as the next layer of software, replacing line-by-line coding with “vibe coding” and “agentic engineering” that let humans guide AI in English. They illustrate this with tools like Cursor and OpenClaw, and propose LLM-friendly docs such as markdown and an `llms.txt` file to bypass HTML parsing. The shift is positioned as “software 3.0,” where prompting becomes computation and the barrier to entry drops dramatically. [2] [1] [4]

Tone shifts across formats

- Interview:** Karpathy uses self-deprecating hedges (“kind of,” “you know”) and humor, reflecting on concrete experiments. [3]
- Keynote opening:** He adopts a declarative, evangelistic register, coining terms like “autonomy slider” and declaring the paradigm shift “extremely bullish.” [2]

Ask anything about the indexed videos...

Done.

Send

SOURCES4

[1]The Most Talented Man in AI
NewthinkVECTOR

[2]Andrej Karpathy: Software Is Changing...
Y CombinatorVECTOR

[3]Heroes of Deep Learning: Andrew Ng...
Preserve Kno... VECTOR

[4]Andrej Karpathy: From Vibe Coding to Agenti...
Sequoia Capital VECTOR

1

2

1 Grounded Answer with Citations

Deep mode delivers a structured answer with TL;DR, section headings, and inline numeric citations linking each claim to its source video. The assistant's thinking time and total generation duration are shown above the response.

2 Source Attribution Panel

Each cited source surfaces as a card with thumbnail, title, channel, and retrieval method (VECTOR = Qdrant dense search). Citations in the answer body map directly to the corresponding source card — full traceability from claim to transcript.

YOUTUBE CONTENT SEARCH - QUERY

COELHO Nexus

HomeDocs DistillerYouTube Content SearchResearch Radar

YouTube Content Search

SourceIngestionAskQuery

ElasticsearchQdrantNeo4j

1

BACKEND Neo4j · YCS · entities + videos

Done · 264 msHistoryRun

1MATCH (v:Video)-[b:BELONGS_TO]->(c:Channel)2WITH v, c, b LIMIT 83OPTIONAL MATCH (d:Document)-[m:MENTIONS]->(e:__Entity__)4WHERE d.video_id = v.id5WITH v, c, b, d, m, e LIMIT 3006RETURN v, c, b, d, m, e

ASK AI Describe what you need; the result fills the editor above.

e.g. List the 10 most-cited papers about agentic RAG...

okMODEL nvidia_nim/z-ai/glm-5.1Generate

3

Results 300 / 300 rows in 264 ms

GraphTableJSON

4

VideoChannelDocumentActivityTechnologyConceptProductFieldPersonCourseOrganizationCharacteristicQuoteWebsiteFeatureSecurity RiskLanguageEventDateCountryPlatformNodeTitleCityProgramAgeMoneyStatement

282 nodes · 304 relationships · cose fallback

FitUnlockRelayout

5

1 Query Backend Selector

Switch between three query backends: Elasticsearch (full-text BM25), Qdrant (dense vector search), and Neo4j (Cypher graph queries). Each backend exposes its own editor tuned to that store's query language.

2 Backend Query Editor

Query editor that adapts to the selected backend — Cypher for Neo4j, DSL for Elasticsearch, or filter syntax for Qdrant. Run queries directly against the chosen store; execution time and result count shown inline.

3 AI Query Generator

Describe what you need in plain language and the LLM writes the correct query for the active backend — Cypher, Elasticsearch DSL, or Qdrant filters — filling the editor above. Zero boilerplate, instant executable output.

4 Result View Modes

Switch results between Graph (force-directed visualization), Table (row/column), and JSON (raw response). All three views update instantly from the same query execution — no re-run needed.

5 Query Result Visualization

Results rendered according to the active backend and selected view — a force-directed knowledge graph for Neo4j, ranked document rows for Elasticsearch, or vector similarity results for Qdrant. Node legend, relationship count, and layout controls available for graph views.

RESEARCH RADAR

Pipeline
Digest

COELHO
NEXUS

RESEARCH RADAR

"Hundreds of papers published daily, most never read and fewer ever built — Research Radar finds the ones worth coding."

1.

Continuously scans arXiv, Semantic Scholar, HuggingFace Daily Papers, and Hacker News for what's actually worth reading

2.

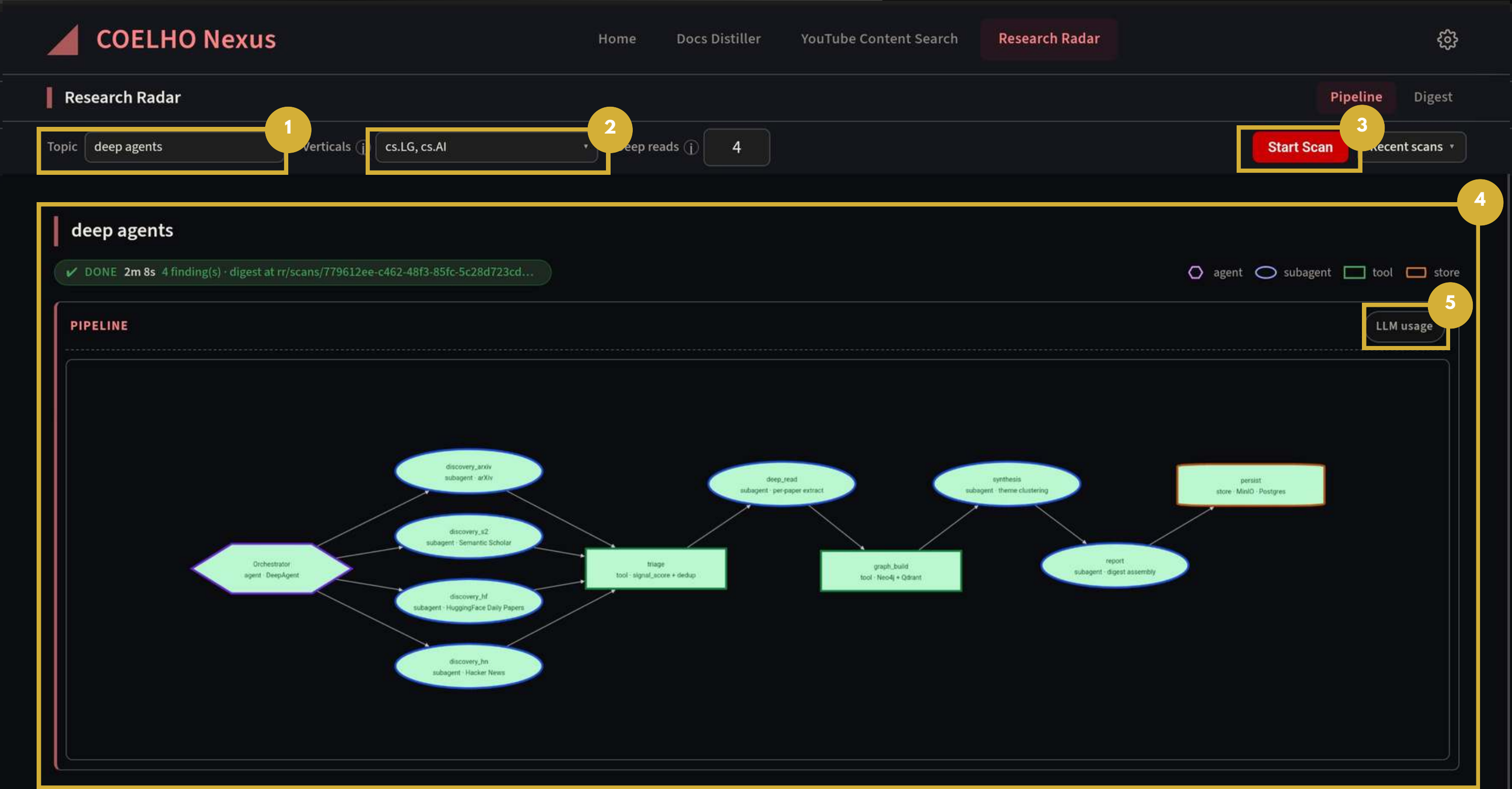
A DeepAgents orchestrator reads each paper and extracts the core idea — what it would actually take to build this in Python

3.

Every idea is ranked by real-world value — prioritizing what's implementable and profitable across LLMOps, agents, and quant finance

Why it matters: Turns a firehose of research nobody has time to read into a short list of ideas you could actually ship.

RESEARCH RADAR - PIPELINE



1 Research Topic Input

Define the research topic in plain language — the orchestrator uses it as the seed query across all discovery subagents. Supports any domain: AI, finance, math, security, or any intersection thereof.

2 Vertical Scope Selector

Pin the scan to specific academic verticals (e.g. cs.LG, cs.AI) to focus discovery on relevant arXiv categories. Narrows signal extraction to the exact subfields that matter for the topic.

3 Scan Controls

Start Scan launches the DeepAgents orchestrator; deep reads sets how many papers receive full per-paper extraction. Recent scans gives one-click access to prior runs by topic and timestamp.

4 DeepAgents Pipeline Graph

Live execution graph of the full Research Radar pipeline: Orchestrator → 4 parallel discovery subagents (arXiv, Semantic Scholar, HuggingFace Daily Papers, Hacker News) → triage → deep reads → synthesis → graph build → persist → digest assembly. Each node shows agent/subagent/tool type and real-time status.

5 LLM Usage Tracker

Per-run token consumption breakdown across all LLM calls in the pipeline — orchestrator planning, per-paper extraction, theme synthesis, and digest assembly. Full cost attribution across the DeepAgents execution.

RESEARCH RADAR - PIPELINE

COELHO Nexus

HomeDocs DistillerYouTube Content SearchResearch Radar

Research Radar

PipelineDigest

Topicdeep agents

Verticals*i*cs.LG, cs.AI

Deep reads*i*4

Start ScanRecent scans ▾

deep agents

DONE2m 8s4 finding(s) · digest at rr/scans/779612e

PIPELINE

Add custom code

Browse all (155)

e.g. eess.SP

+ Add

cs.LG

— ML

cs.AI

— AI

cs.CL

— NLP

cs.CV

— Vision

stat.ML

— ML theory

math.OC

— Optimization

math.PR

— Probability

q-fin.PR

— Quant pricing

q-fin.ST

— Quant statistics

Orchestrator
agent · DeepAgent

discovery_hf
subagent · HuggingFace Daily Papers

discovery_hn
subagent · Hacker News

graph_build
tool · Neo4j + Qdrant

synthesis
subagent · theme clustering

report
subagent · digest assembly

persist
store · MinIO · Postgres

agent

subagent

tool

store

LLM usage

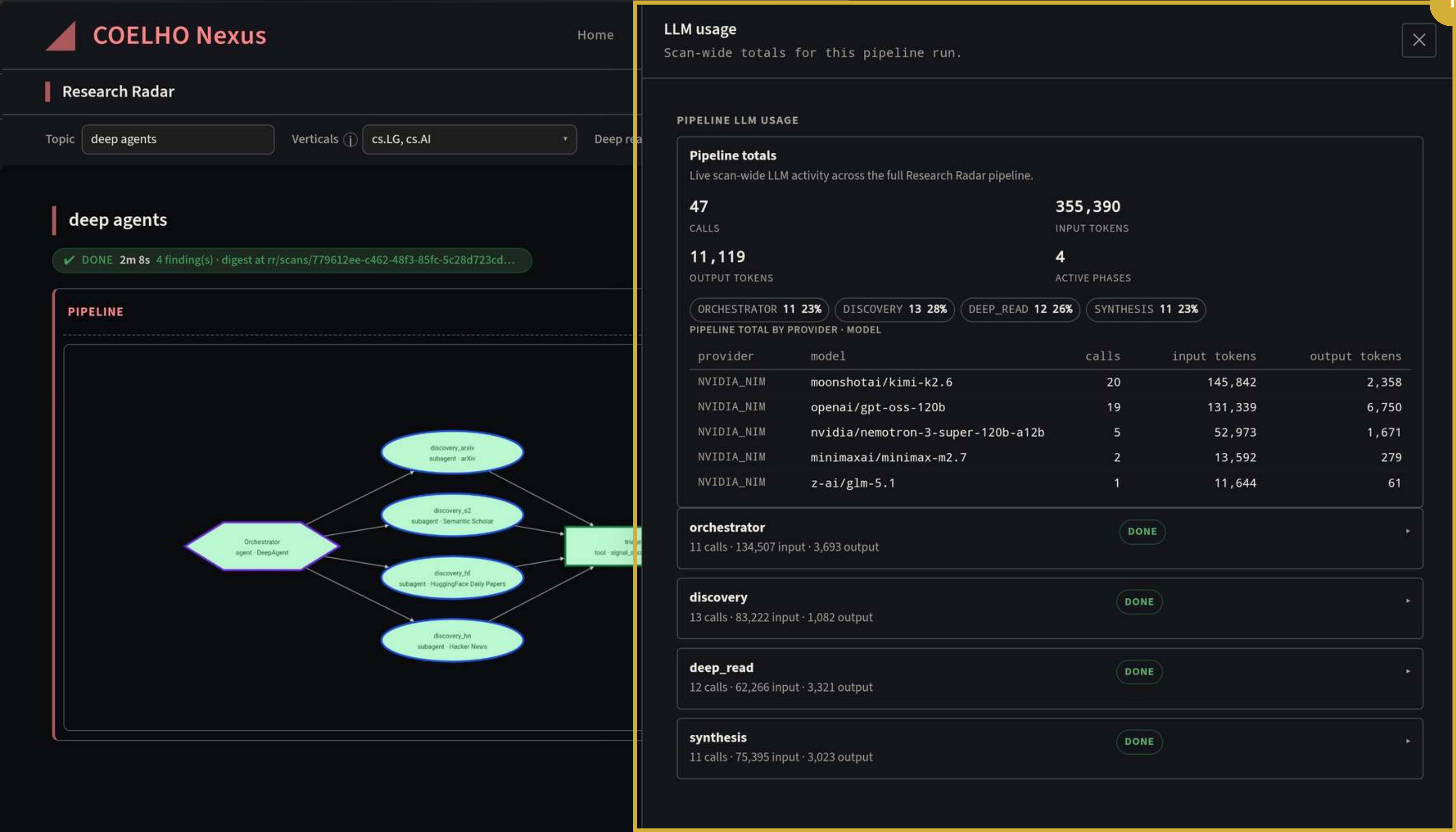
1

1 arXiv Vertical Picker

Multi-select dropdown covering all 155 arXiv categories — from cs.LG (ML) and cs.AI to math.OC (Optimization), q-fin.PR (Quant pricing), and beyond. Type any custom arXiv code to add it. Active verticals are checked in red and drive which papers the discovery subagents fetch.

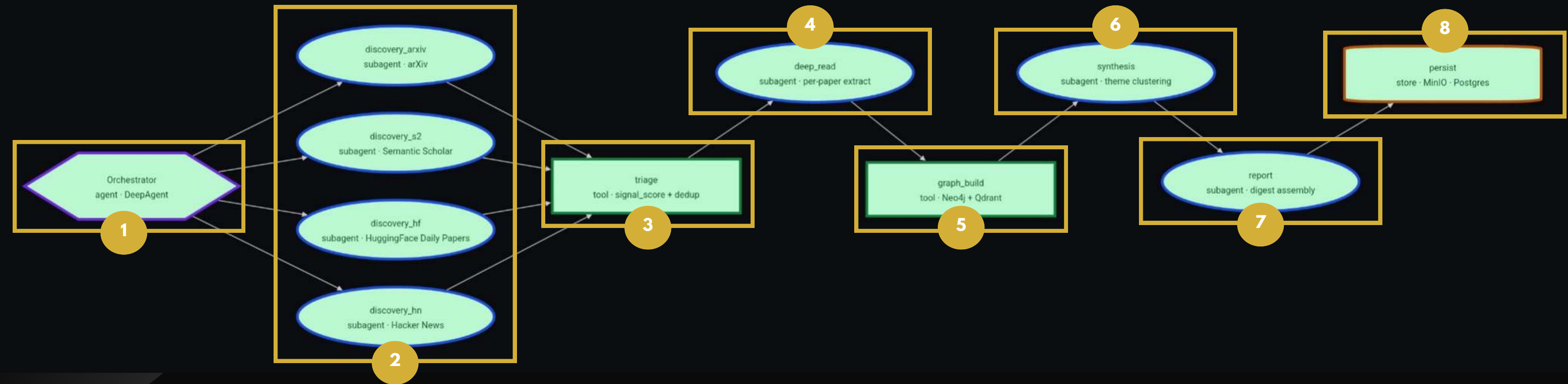
COELHO
NEXUS

RESEARCH RADAR - PIPELINE



1 LLM Usage Breakdown

Scan-wide token accounting across every LLM call in the pipeline, broken down by provider/model via the LLM rotator and by pipeline phase — orchestrator, discovery, deep_read, and synthesis. Shows call counts, input and output tokens per phase, giving full cost attribution for any run.



1 DeepAgent Orchestrator

FGTS-VA bandit-routed orchestrator built with `create_deep_agent`. Sequences the full pipeline: parallel discovery → triage → deep_read fan-out → graph_build → synthesis → digest. `PhaseEnforcerMiddleware` guarantees every fs artifact exists before the run closes; `PhaseEventsMiddleware` streams per-phase SSE to the UI.

2 4-Source Discovery Fan-Out

Four parallel DeepAgents subagents — arXiv, Semantic Scholar, HuggingFace Daily Papers, and Hacker News — each with isolated context and MCP-backed search tools. Skills (`arxiv_query_shaping`, `rotator_etiquette`) injected at build time. Results stashed to the virtual fs for triage.

3 Signal Triage & Dedup

Deterministic tool: `normalize` → cross-source dedup by `arxiv_id` → off-topic rerank gate (NIM `nemotron-rerank-1b`, drops bottom 50%) → source-diversity quota (min-1-per-source) → `signal_score` ranking → top-N written to fs. Idempotency guard prevents re-triage mid-scan

4 Deep Read Fan-Out

Parallel subagent fan-out — one DeepAgents instance per top-N paper. Each extracts: problem, method, math, how-to-build, money angle, and confidence score. Isolated context per paper; augmented with the `paper_extraction` skill. Results written to fs per `arxiv_id`.

5 Graph & Vector Persist

Deterministic tool: embeds each abstract via `llama-nemotron-embed-1b-v2` (NIM, 2048d), then upserts to Neo4j (paper metadata + relationships) and Qdrant (dense vectors). No LLM calls — pure I/O sink for both stores.

6 Theme Synthesis

Single subagent dispatched once after `deep_read` completes. Reads all extractions + triage ranking; produces 3–7 emerging themes spanning ≥2 papers, cross-paper convergence notes, and a 2–3 sentence executive summary. Augmented with the `cross_paper_synthesis` skill.

7 Digest Assembly

Reads synthesis themes, per-paper extractions, and triage ranking from the virtual fs. Assembles the final structured digest — findings, themes, and executive summary — as a JSON artifact ready for persistence.

8 Scan Persistence

Digest JSON and per-paper extraction artifacts persisted to MinIO under `rr/scans/{scan_id}/`. Scan metadata and finding records written to Postgres. Both stores bootstrapped idempotently at FastAPI lifespan startup.

RESEARCH RADAR - DIGEST

COELHO Nexus

Home

Docs Distiller

YouTube Content Search

Research Radar

Research Radar

Pipeline

Digest

Recent scans ▾

deep agents

THEMES

All

agent security vulnerabilities

modular architecture for agent robustness

long-horizon multi-turn planning

Security of deep agents is emerging as a critical concern, with multiple groups proposing evaluation framework...

#1

2602.07652

arxiv

Agent-Fence: Mapping Security Vulnerabilities Across Deep Research Agents

💡

AgentFence can be packaged as a security-testing plug-in for popular agent toolkits (LangChain, AutoGPT, LangGraph) to certify that deployed agents stay within their goal and authority envelope, reducing costly breaches such as unauthorized API calls or wallet draining. Enterprises can integrate it into CI/CD pipelines for LLM-ops, charging for compliance-as-a-service or licensing the evaluation suite to AI product teams.

agent security vulnerabilities

long-horizon multi-turn planning

Deep dive →

#2

2605.24245

hn

Deep-Research Agents Can Be Poisoned via User-Generated Content

💡

Security teams can offer a monitoring service that scans incoming user content for poisoning patterns before it reaches the agent, enabling safer deployment of LLM-powered assistants in customer support pipelines handling high query volumes.

agent security vulnerabilities

Deep dive →

#3

2504.03024

arxiv

Deep Reinforcement Learning via Object-Centric Attention

💡

Deep is attention module for some AI robotica sim to real, or autonomous vehicle perception stacks where rival level RL agents must generalize across lighting, weather, or background changes

1 Theme Navigator

Synthesis-extracted themes from the scan — click any chip to filter the digest to papers tagged under that theme. The executive summary below gives the LLM's 2–3 sentence cross-paper overview of the full scan.

2 Ranked Paper Cards

Each card shows the paper's signal rank, arXiv ID, source badge (arXiv, Hacker News, Semantic Scholar, HuggingFace Daily Papers), title, LLM-extracted money angle, and assigned themes. Deep dive opens the full per-paper extraction: problem, method, math formulation, how-to-build, and confidence score.

COELHO
NEXUS

RESEARCH RADAR - PIPELINE

COELHO Nexus

Home Docs Distilled

Research Radar

deep agents

THEMES

All agent security vulnerabilities modular architecture for agent robustness long-horizon multi-turn planning

Security of deep agents is emerging as a critical concern, with multiple groups proposing evaluation framework...

#1

2602.07652

Agent-Fence: Mapping Security Vulnerabilities Across Deep Research Agents

💡 AgentFence can be packaged as a security-testing plug-in for popular agent toolkits (LangChain, AutoGPT, LangGraph) to certify that deployed agents stay within their goal and authority envelope, reducing costly breaches such as unauthorized API calls or wallet draining. Enterprises can integrate it into CI/CD pipelines for LLM-ops, charging for compliance-as-a-service or licensing the evaluation suite to AI product teams.

agent security vulnerabilities long-horizon multi-turn planning

#2

2605.24245

Deep-Research Agents Can Be Poisoned via User-Generated Content

💡 Security teams can offer a monitoring service that scans incoming user content for poisoning patterns to support pipelines handling high query volumes.

agent security vulnerabilities

#3

2504.03024

Deep Reinforcement Learning via Object-Centric Attention

💡 Deep reinforcement learning module for game AI robot navigation to reach autonomous vehicle navigation stage.

#1 · signal 0.271

2602.07652

Agent-Fence: Mapping Security Vulnerabilities Across Deep Research Agents

Sai Puppala, Ismail Hossain, Md Jahangir Alam, Yoonpyo Lee, Jay Yoo, Tanzim Ahad, Syed Bahauddin Alam, Sajedul Talukder

Extracted Confidence: 90%

1

MONEY ANGLE

2

PROBLEM

3

METHOD

4

HOW TO BUILD

5

MATH

6

BUILD

AgentFence can be packaged as a security-testing plug-in for popular agent toolkits (LangChain, AutoGPT, LangGraph) to certify that deployed agents stay within their goal and authority envelope, reducing costly breaches such as unauthorized API calls or wallet draining. Enterprises can integrate it into CI/CD pipelines for LLM-ops, charging for compliance-as-a-service or licensing the evaluation suite to AI product teams.

COELHO
NEXUS

1

Money Angle

LLM-extracted commercial applicability — how the paper translates into a product, service, or portfolio edge. Written by the deep_read subagent from the abstract and extracted fields.

2

Problem Statement

2–3 sentence extraction of the real-world gap the paper closes and why existing approaches fall short. The fastest signal for deciding if the paper is relevant to your domain.

3

Method Summary

4–6 sentence extraction of the paper's core algorithm, architecture, or technique — enough depth to evaluate implementation feasibility without reading the full PDF.

4

How to Build

Implementation notes from the deep_read subagent — which components to wire together, which libraries apply, and what the critical implementation choices are. Bridges the gap between understanding the method and actually building it.

5

Math Formulation

Key formulas in LaTeX, each annotated with its role in the algorithm. Extracted by the deep_read subagent to surface the mathematical core without hunting through the paper.

6

On-Demand Code Synthesis

Generates complete, runnable Python from the paper's 5 extraction fields on click. FGTS-VA bandit routes to the strongest coder in the pool; one round of Self-Refine enforces no TODOs or stubs. Output cached in MinIO per paper and prompt version.

SETTINGS

COELHO
NEXUS

SETTINGS

COELHO
NEXUS

[Home](#)[Docs Distiller](#)[YouTube Content Search](#)[Research Radar](#)

Settings

Choose the AI providers and free models COELHO Nexus may use. Keys are encrypted and stored on the server — they're never sent back to your browser, and they survive restarts.

✓ Ready — required NVIDIA NIM key is set.

Enable all keyed providers

Test all

3 provider(s) keyed · 3 active in rotator

Groq

FREE

Custom key

Paste GROQ_API_KEY

Save

Test

Remove

▸ Models

NVIDIA NIM

FREE

REQUIRED

Custom key

Paste NVIDIA_API_KEY

Save

Test

Remove

▸ Models

Cerebras

FREE

No key

Paste CEREBRAS_API_KEY

Save

Test

Remove

1

Settings Gear

One click from anywhere in the app. Opens BYOK provider management — replacing what used to require editing Helm values and redeploying the cluster just to rotate a single API key.

2

Provider Key Rows

One row per provider — Groq, NVIDIA NIM, Cerebras, and more. Paste a free-tier key, save it, and test it live. The REQUIRED badge marks NVIDIA NIM as the canonical provider; the toggle takes any provider in or out of the rotator instantly. Keys are Fernet-encrypted at rest and never leave the server.

LLM ROTATOR — ZERO-COST ADAPTIVE INFERENCE

COELHO
NEXUS

24-DIM CONTEXT VECTOR

Every LLM call is encoded as a 24-dimension feature vector — task type, chapter position, vault size, thinking-budget flag, time-of-day, and live per-provider error rates — so the bandit can tell a Synth call on a huge vault from a quick off-topic check, and route each differently.

COMPOSITE REWARD SIGNAL

A single success/fail flag can't tell a fast, well-formed response from a slow one that barely passed — so the reward blends four signals: success, schema validity, latency, and recall, each weighted by production impact. Failures are penalized by severity.

FGTS-VA SELECTION (NEURIPS 2025)

Variance-aware Thompson sampling, published NeurIPS 2025: providers with noisy or thin reward history get sampled more optimistically, so one lucky response doesn't get over-trusted and one bad one doesn't get permanently written off.

\$0 PROVIDER POOL

Five free-tier providers — NVIDIA NIM, Groq, Cerebras, Mistral, and Gemini — spanning frontier open-weight models (Llama, Qwen, Nemotron, Kimi) to sub-second inference. Engineered for zero per-token cost and zero vendor lock-in — a deliberate architecture choice, not a budget constraint.

LLM ROTATOR — ZERO-COST ADAPTIVE INFERENCE

1. Initialization

$$A_a \leftarrow \frac{\lambda}{\max(0.1, p)} I_{24}$$

$$b_a \leftarrow A_a \cdot \frac{p}{24} \mathbf{1}_{24}$$

(p = benchmark_prior, λ = 1.0)

2. Context vector

$$\psi \in \mathbb{R}^{24}$$

3. Posterior mean (per arm)

$$\hat{\theta}_a = A_a^{-1} b_a$$

4. FGTS-VA variance estimate

$$\hat{\sigma}_a^2 = \max(\sigma_{\min}^2, \hat{\sigma}_{\text{EWMA}}^2)$$

5. Variance-aware Thompson sample

$$\theta_s \sim \mathcal{N}(\hat{\theta}_a, \hat{\sigma}_a^2 A_a^{-1})$$

6. Score with feel-good bonus

$$s_a = \psi^\top \theta_s + \beta \sqrt{\psi^\top A_a^{-1} \psi}$$

7. Arm selection

$$a^* = \arg \max_{a \in \mathcal{A}} s_a(\psi)$$

8. Reward signal

$$r = \begin{cases} \text{pen}(\text{class}), & \text{call fails} \\ 0.30 + 0.25 \mathbf{1}_{\text{schema}} + 0.10 \cdot \text{clip}(1 - \ell, -2, 2) + 0.25 \cdot \text{clip}(\rho_{\text{hash}}, 0, 1), & \text{call succeeds} \end{cases}$$

(ℓ = latency_s / expected_latency_s)

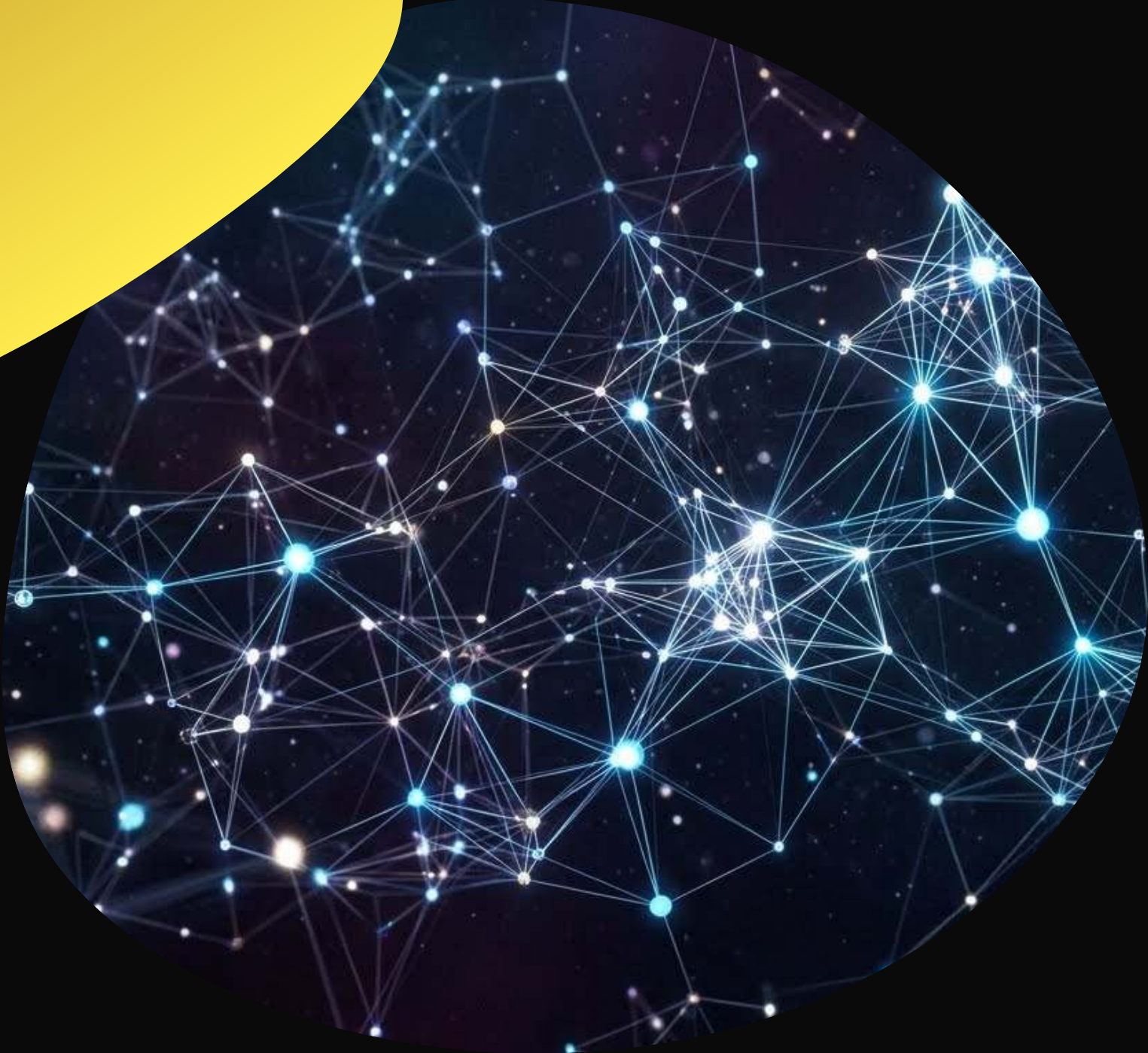
$$\text{pen}(\text{class}) \in \{-0.10, -0.20, -0.40, -0.40, -0.50, -0.60, -0.80\}$$

rate_limit -0.10 · content_filter -0.20 · schema_invalid -0.40 ·
unknown -0.40 · server_error -0.50 · timeout -0.60 · auth_error -0.80

9. Forgetting update

$$\hat{\sigma}_{\text{EWMA}}^2 \leftarrow (1 - \alpha) \hat{\sigma}_{\text{EWMA}}^2 + \alpha \left(r - \psi^\top \hat{\theta}_a \right)^2$$

$$A_a \leftarrow (1 - \gamma) A_a + \psi \psi^\top$$
$$b_a \leftarrow (1 - \gamma) b_a + r \psi$$



01.

THREE MICROSERVICE ARCHITECTURE

Three purpose-built services compose into one coherent AI platform, sharing a single LLM rotator, six data stores, and a full OpenTelemetry observability pipeline.

FastAPI + Celery owns the compute layer — six isolated task queues route Docs Distiller ingestion, YouTube Content Search indexing, and Research Radar orchestration concurrently without contention.

FastHTML delivers a zero-JavaScript server-side frontend via HTMX-in-SSR, with Cytoscape.js pipeline graph visualization and SSE streaming for real-time lifecycle events.

FastMCP completes the surface layer, exposing four research discovery tools as MCP-native endpoints with per-tool token budgets, rate limiting, and Fernet-encrypted BYOK credential injection.

MAIN APPLICATIONS



Docs Distiller

Multi-tier document ingestion pipeline supporting Playwright CDP, sitemap, HTTPX, fallback, and manual modes.

Transforms raw web content into structured knowledge through planner, resolver, and synth stages with full LangFuse observability.



YouTube Content Search

Full semantic search over YouTube transcripts with 21 subpackages covering caching, chunking, content extraction, embeddings, Elasticsearch indexing, Neo4j knowledge graphs, Qdrant vector search, RAG, and conversation management.



Research Radar

Autonomous research agent powered by DeepAgents framework. Scans arXiv, Semantic Scholar, Hacker News, and HuggingFace daily papers through an 8-phase orchestrator: Init, Discovery, Triage, Deep Read, Graph Build, Synthesis, Digest, and Persist.

MAIN APPLICATIONS

COELHO
NEXUS



DOCS DISTILLER

4-Tier Ingestion Pipeline: `llms_full.txt`, `llms_txt`, `sitemap`, and `httpx-crawl-with-Playwright-fallback` modes. Planner-Resolver-Synth Workflow: index-graph chapter-outline planning, domain resolving, and multi-stage synthesis. LangFuse Session Tracking: every `study_id` tracked as a LangFuse session with full trace observability. MinIO Artifact Storage: page bodies, planner outputs, and synthesized content persisted in S3-compatible storage.



YOUTUBE CONTENT SEARCH

Full Semantic Search: complete pipeline from YouTube transcripts to dense vector embeddings via NVIDIA NIM. Neo4j Knowledge Graphs: extracts entities and relationships from video content into queryable graph structures. Qdrant Vector Search: 2048-dimensional embeddings for semantic similarity over processed transcripts. Elasticsearch Full-Text: hybrid search combining dense vectors with traditional full-text indexing.



RESEARCH RADAR

Autonomous DeepAgents Orchestrator: 8-phase pipeline — Init, Discovery, Triage, Deep Read, Graph Build, Synthesis, Digest, and Persist. 4 Discovery Sources: arXiv (Atom XML), Semantic Scholar (tldr/citations), HuggingFace daily papers, and Hacker News (Algolia). PhaseEnforcer Middleware: ensures all phases complete before termination; PhaseEvents Middleware streams per-node SSE events separately. Signal Scoring: an 8-term weighted composite — relevance, recency, citation velocity, influential-citation ratio, vertical fit, cross-tier buzz, code presence, and arXiv-ID presence.

CORE PLATFORM COMPONENTS



LLM Router & BYOK

FGTS-VA bandit (NeurIPS 2025) routes across ~20 free-tier model deployments — NVIDIA NIM, Groq, Mistral, Gemini, and Cerebras. Per-arm variance-aware Thompson sampling with forgetting factor adapts routing in real time. Cascading fallbacks on rate limits and errors. Embeddings and reranking pinned to NVIDIA NIM. BYOK keys Fernet-encrypted in MinIO — never sent to the browser, survive restarts.



6 Data Stores

PostgreSQL: relational backbone with shared connection pool.
Neo4j: knowledge graphs for entity and relationship extraction across all three features.
Qdrant: 2048-dimensional vector search via NVIDIA NIM embeddings.
Elasticsearch: full-text and hybrid search over transcripts and document pages.
MinIO: S3-compatible artifact store for page bodies, planner outputs, and encrypted credentials.
Redis: caching, pub/sub, Celery broker, and bandit state persistence.

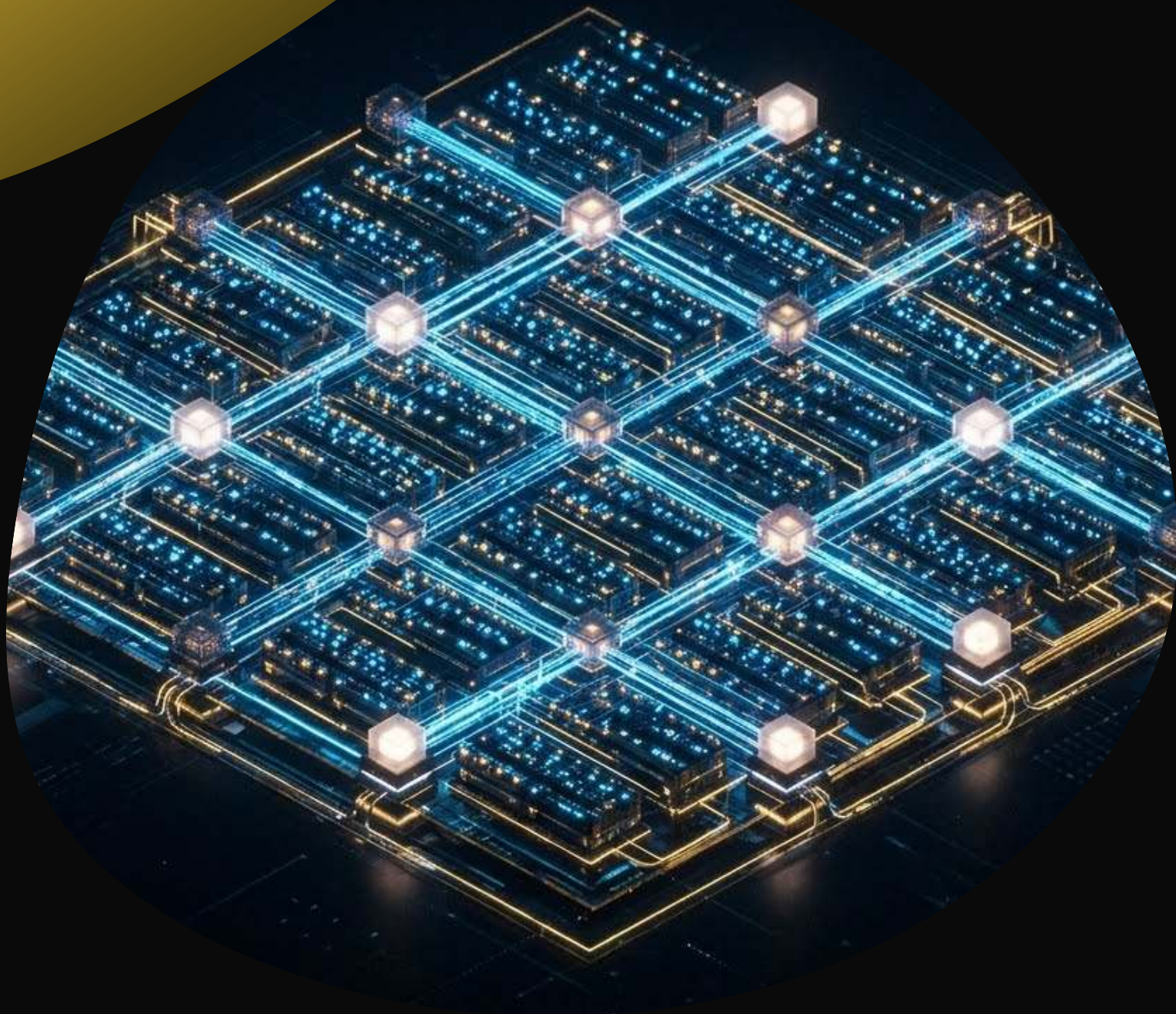


OpenTelemetry Observability

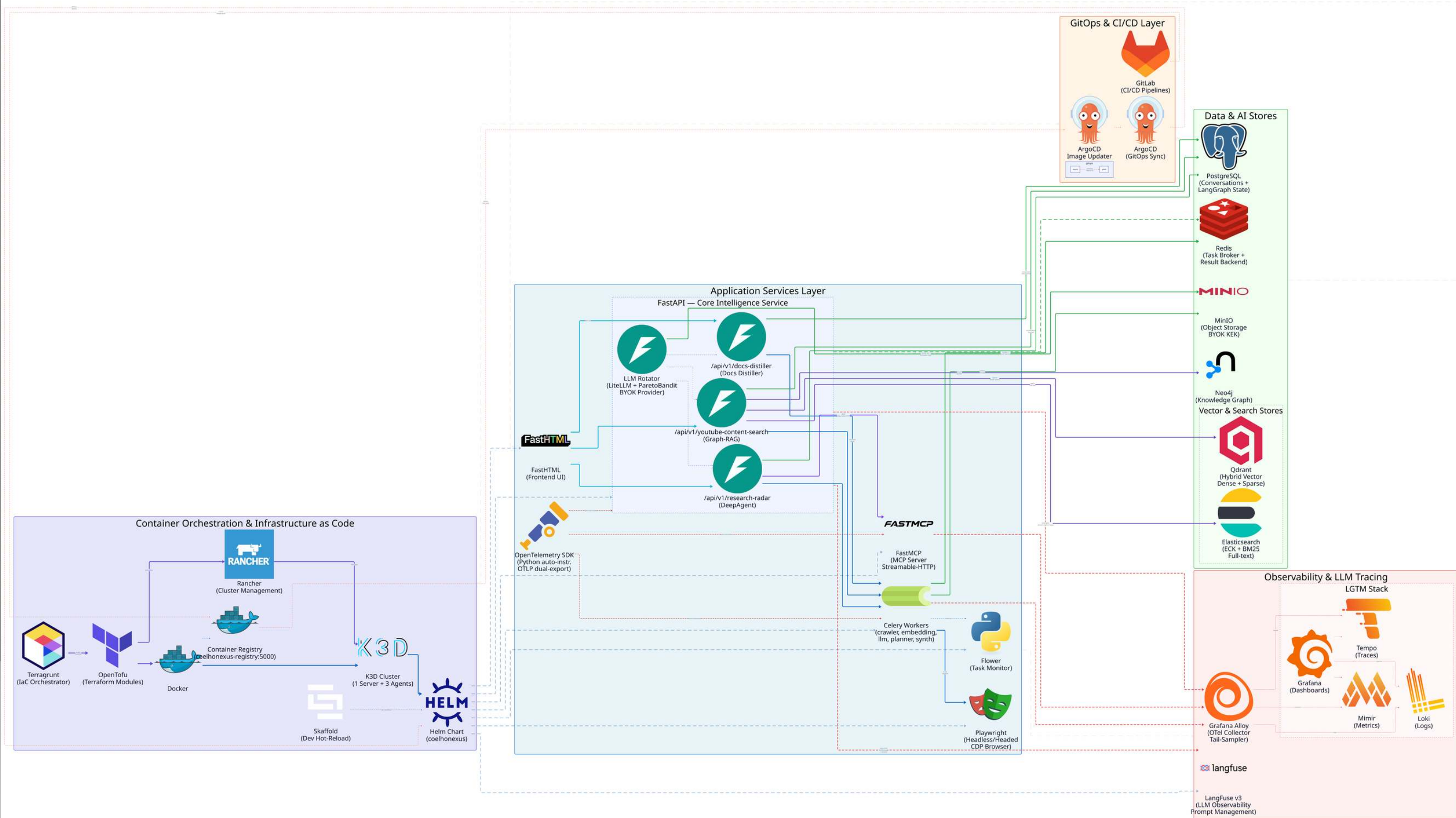
Dual-export pipeline: Alloy gRPC for the LGTM stack and LangFuse HTTP for LLM trace correlation.
BatchSpanProcessor tuned to q=6144, b=256, d=10s.
LangFuseSpanGate triple-gate filter drops non-domain spans.
Nine auto-instrumentations including httpx, redis, psycopg, and aio-pika.
Per-feature distinct trace patterns for Docs Distiller, YouTube Content Search, and Research Radar.

INFRASTRUCTURE AS CODE & MICROSERVICES

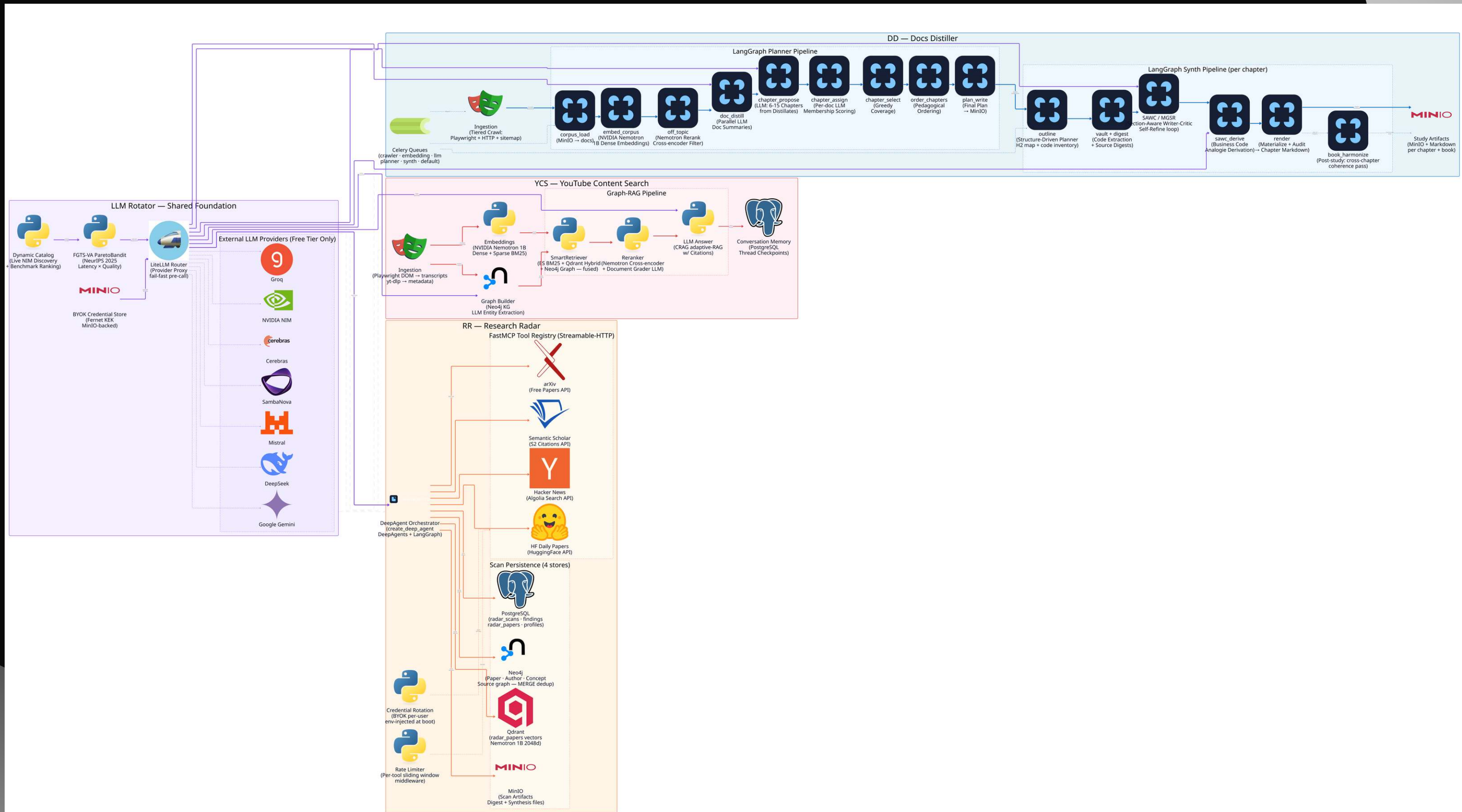
- 1 - CONTAINER ORCHESTRATION LAYER
- 2 - GITOPS & CI/CD LAYER
- 3 - DATA & STORAGE LAYER
- 4 - APPLICATION SERVICES LAYER
- 5 - OBSERVABILITY & MONITORING LAYER



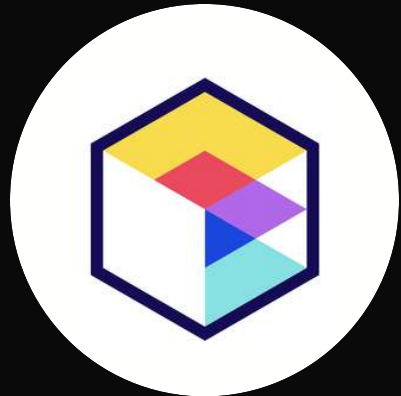
PLATFORM ARCHITECTURE



AI PIPELINE ARCHITECTURE

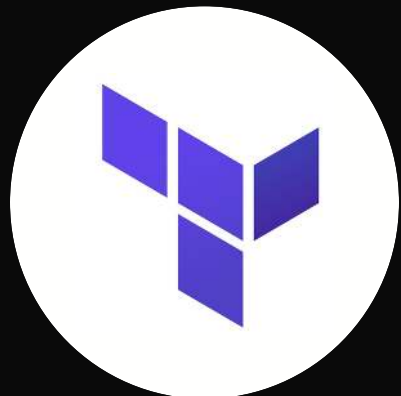


1 - CONTAINER ORCHESTRATION LAYER



TERRAGRUNT

- Orchestrates 18 OpenTofu modules across 5 dependency layers with DRY terragrunt.hcl configs
- Enforces provisioning order: k3d/Kubernetes cluster bootstrap → Rancher → ArgoCD → data stores → observability → apps
- Isolates remote state per module with automatic backend configuration, enabling safe parallel applies
- **Why Terragrunt:** The composable IaC orchestration layer shared across COELHO Nexus and all future COELHO projects



OPENTOFU / TERRAFORM

- Provisions the k3d cluster with 1 server + 2 agent nodes via local-exec wrapping the k3d CLI
- Manages Kubernetes, Helm, Rancher, and ArgoCD providers across 18 isolated state files
- Configures host volume mounts for persistent storage and a local container registry on port 5000
- **Why OpenTofu:** Open-source Terraform fork that eliminates vendor lock-in while maintaining full HCL compatibility — the execution engine underneath Terragrunt



KUBERNETES

- Orchestrates 20+ containerized services via declarative manifests — scheduling, networking, and lifecycle management
- Handles service discovery via ClusterIP and external access via NodePort, connecting frontend to backend seamlessly
- Enforces liveness/readiness probes, resource quotas, and automatic pod restart for self-healing operations
- **Why Kubernetes:** The industry-standard orchestration platform that provides the reliability, scalability, and operational patterns required for a production-grade microservice architecture



DOCKER

- Containerizes all custom services (FastAPI, FastHTML, FastMCP) with optimized single-stage Dockerfiles
- Provides BuildKit-accelerated layer caching for rapid image rebuilds during Scaffold hot-reload development
- Serves as the runtime foundation for k3d's Kubernetes nodes, running the entire cluster in containers
- **Why Docker:** Standardizes every service into portable, reproducible containers — a prerequisite for Kubernetes orchestration and consistent behavior across development and production

2 - GITOPS & CI/CD LAYER



K3D

- Runs lightweight K3s distributions inside Docker containers, replicating production Kubernetes locally
 - Provides built-in container registry (coelhonexus-registry:5000) for local image distribution during CI/CD
 - Supports 1 server + 2 agent nodes with host volume mounts and auto-restart via Docker unless-stopped policy
 - **Why K3D:** Enables a production-identical Kubernetes environment on a single development machine with minimal resource overhead
-



HELM

- Packages FastAPI, FastHTML, FastMCP, Celery, and Flower into a single versioned chart with no external subchart dependencies
 - values.yaml centralizes all service configuration: OTel endpoints, data store URLs, embedding model, and feature flags
 - Layered values files (values.yaml + values-coelhonexus.yaml) enable cluster-specific overrides without duplicating config
 - **Why Helm:** The standard Kubernetes package manager that turns the entire platform into a single, versioned, reproducible deployment unit
-



ARGOCD

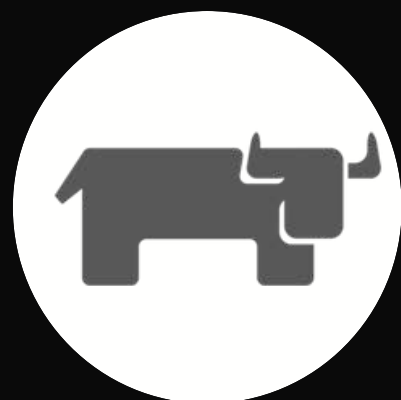
- Syncs the coelhonexus Helm chart from GitHub master to the coelhonexus namespace with automated pruning and self-healing
 - ArgoCD Image Updater polls coelhonexus-registry:5000 every 2 minutes via digest strategy for zero-downtime rollouts
 - Two deployment targets: coelhonexus-dev (Skaffold, ports 23020–23039) and coelhonexus prod (ports 23000–23019)
 - **Why ArgoCD:** GitOps-native deployment pipeline that makes Git the single source of truth for every running service
-

2 - GITOPS & CI/CD LAYER



SKAFFOLD

- Orchestrates local development with hot-reload Python file sync directly into running pods — no container rebuild
- Manages port forwarding for FastAPI, FastHTML, FastMCP, and Flower simultaneously on localhost
- sha256 content-based tag policy guarantees every build reflects current source — no stale image cache
- **Why Scaffold:** Eliminates the rebuild-redeploy cycle, enabling sub-second code changes in production-identical container environments



RANCHER

- Rancher Server v2.14.1 — single dashboard to view and manage every pod, service, and log across the cluster
- The daily operational front door — deployment health and failure debugging across 18 infrastructure modules, no kubectl required
- Complements Grafana/LGTM, not duplicates it: Rancher shows raw cluster state, LGTM shows application telemetry
- **Why Rancher:** Mission control for the cluster — what makes solo-operating a multi-service Kubernetes platform practical

3 - DATA & STORAGE LAYER



NEO4J

- Neo4j Community Edition + APOC Core powers dual knowledge graphs: YCS entities and RR paper networks
 - LangChain LLMGraphTransformer populates both graphs via Bolt protocol (neo4j.neo4j.svc.cluster.local:7687)
 - G1GC-tuned JVM with lazy heap commit, string deduplication, and ExitOnOutOfMemoryError for homelab stability
 - **Why Neo4j:** The only store that captures relationships between entities – enabling graph-aware queries across video content and research papers that flat indexes cannot answer
-



QDRANT

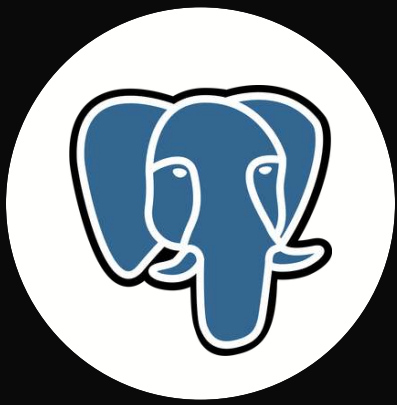
- Vector store for 2048-dimensional embeddings generated by NVIDIA NIM's llama-nemotron-embed-1b-v2 model
 - Powers semantic similarity search for YCS transcript chunks and Research Radar paper deduplication across sources
 - Chunk-level storage with payload metadata (video_id, channel, timestamps) enables filtered semantic search across YCS transcript collections
 - **Why Qdrant:** Native vector search with API key authentication and ServiceMonitor telemetry – fastest path from embedding to nearest-neighbor retrieval
-



ELASTICSEARCH

- ECK Operator deploys Elasticsearch 8.18 with two YCS indexes: coelhonexus-youtube-metadata and coelhonexus-youtube-transcriptions
 - File-realm authentication with ECK auto-generated TLS; ML and transform roles disabled for resource efficiency
 - Kibana deployed via ECK Stack – index health, query debugging, and transcript search in one dashboard
 - **Why Elasticsearch:** Enables hybrid search combining dense vector similarity with traditional full-text indexing – the query engine behind YCS Ask
-

3 - DATA & STORAGE LAYER



POSTGRES SQL

- PostgreSQL 18 standalone serving LangFuse, YCS conversation history, Research Radar stores, and LangGraph AsyncPostgresSaver
 - AsyncPostgresSaver persists Planner and Synth LangGraph state — enables mid-pipeline recovery without re-running completed nodes
 - ServiceMonitor enabled — PostgreSQL query latency and connection pool metrics scraped by Alloy and exported to Grafana Mimir
 - **Why PostgreSQL:** Relational backbone shared across LLM pipeline checkpointing, LangFuse observability, and application conversation history — one durable store for all stateful workloads
-



MINIO

- S3-compatible object store (standalone) with API on port 9000 and web console on port 9001
 - Stores page bodies, planner outputs, and synthesized chapter artifacts in the coelhonex bucket
 - Hosts Fernet-encrypted BYOK LLM credentials — central object store across all pipeline stages
 - **Why MinIO:** Production-grade S3 locally — eliminates external cloud storage dependencies while maintaining full AWS S3 API compatibility
-



REDIS

- Celery broker powering async task chains for DD ingestion, Planner, Synth, and YCS pipeline stages
 - Persists FGTS-VA bandit CellState per provider/process and enforces single-flight pipeline locks (dd:planner:lock:{slug})
 - Tracks ingestion progress counters and LLM rotator settings generation for cross-worker cache invalidation
 - **Why Redis:** In-memory speed for sub-millisecond state reads across Celery workers, bandit updates, and progress polling
-

4 - APPLICATION SERVICES LAYER



FASTAPI

- Domain-driven async backend (DD, LLM, YCS, RR) with OpenTelemetry trace_id/span_id injected on every log line
 - Provisions all 6 stores at lifespan: AsyncPostgresSaver, Redis, Neo4j, Elasticsearch, Qdrant, and MinIO bucket
 - Celery task dispatcher for DD ingestion, Planner, Synth, and YCS pipeline chains
 - **Why FastAPI:** Async-first, domain-isolated backend that co-provisions every data store and the full OTel pipeline at boot
-



FASTHTML

- Server-side BFF rendering full-feature UIs for Docs Distiller, YCS, Research Radar, and Settings BYOK
 - Reverse-proxies all /api/ calls to FastAPI – zero CORS configuration needed on the frontend
 - Static assets served with Cache-Control: no-cache + ETag revalidation for instant Scaffold redeploy reflection
 - **Why FastHTML:** Pure Python server-side rendering eliminates a JS framework while keeping the client snappy via HTMX + ES modules
-



FASTMCP

- Streamable-HTTP ASGI MCP server with 4 registered tools: arXiv, Semantic Scholar, HuggingFace Daily Papers, Hacker News
 - RateLimitMiddleware + TelemetryMiddleware gate every tool call with OTel spans exported to Alloy
 - Resources and prompts registered alongside tools – full MCP capability surface for Research Radar orchestration
 - **Why FastMCP:** Exposes academic discovery tools as first-class MCP primitives consumable by any MCP-compatible AI client
-

4 - APPLICATION SERVICES LAYER



CELERY

- Distributed task queue executing DD ingestion, Planner, Synth, and YCS pipeline chains as async Celery chains
 - OTel-instrumented via `worker_process_init` — every task emits spans with `trace_id/span_id` to Alloy
 - Flower deployed alongside on port 23022 for real-time task monitoring, worker inspection, and queue visibility
 - **Why Celery:** Decouples long-running LLM pipeline execution from FastAPI — workers scale independently of the API layer
-



PLAYWRIGHT

- Two separate browser pools run in the cluster — a full visible Chromium instance, and a lightweight invisible one
 - The visible pool scrapes YouTube video transcripts straight from the page — YouTube blocks the same request when headless
 - The invisible pool is Docs Distiller's last-resort tier, rendering JavaScript-only documentation sites simple fetching can't parse
 - **Why Playwright:** Two of the three platform features depend on it for content with no public API — YouTube transcripts and JS-heavy docs
-

5 - OBSERVABILITY & MONITORING LAYER



GRAFANA ALLOY

- OTel-native collector receiving OTLP gRPC (:4317) and HTTP (:4318) from FastAPI and Celery workers across all 3 domains
 - Routes all 3 telemetry signals from a single River config: traces → Tempo, logs → Loki, metrics → Mimir remote_write
 - ServiceMonitor + PodMonitor discovery and namespace-scoped pod log tailing eliminate a separate Prometheus agent and log shipper
 - **Why Grafana Alloy:** Single OTel-native pipeline unifying metrics, logs, and traces — replaces three separate collectors with one River config
-



GRAFANA

- External PostgreSQL backend for persistent dashboard and user storage — no bundled SQLite, survives pod restarts
 - Datasource sidecar auto-imports Loki/Tempo/Mimir via ConfigMaps labeled grafana_datasource: 1 at startup
 - Dashboard-as-code via Helm provisioning — all dashboards version-controlled alongside infrastructure
 - **Why Grafana:** Single pane of glass unifying infrastructure metrics, distributed traces (including LangFuse LLM spans), and pod logs from the full LGTM stack
-



LANGFUSE

- Self-hosted LLM observability (sign-up disabled, single-org) tracking every LangGraph session across Docs Distiller, YCS, and Research Radar
 - OTel integration correlates LLM spans (model, latency, token counts) with infrastructure traces in Grafana Tempo
 - Redis-backed session queue with tuned socket timeouts for long-running Celery pipeline traces — no dropped spans on multi-hour runs
 - **Why LangFuse:** The only store where LLM cost, quality, and latency are visible per-session — closes the observability gap that Mimir metrics alone cannot cover
-

5 - OBSERVABILITY & MONITORING LAYER



GRAFANA LOKI

- Monolithic SingleBinary (v3.7.1) with schema v13 + TSDB store — replaces deprecated BoltDB-shipper for index efficiency
 - S3 backend on MinIO (loki-chunks + loki-ruler buckets) — pod logs persisted beyond pod lifetime with compactor-driven retention
 - Datasource ConfigMap labeled grafana_datasource: 1 auto-wired to Grafana's sidecar on startup
 - **Why Grafana Loki:** LogQL-queryable log store backed by the same MinIO already serving the platform — zero additional storage infrastructure
-



GRAFANA TEMPO

- Monolithic single-binary StatefulSet (v2.10.1) running ingester, querier, query-frontend, and compactor in one pod
 - Receives OTLP traces from Alloy and correlates them with LangFuse LLM spans via shared trace_id across all 3 domains
 - Trace blocks stored in MinIO with configurable retention and compactor-driven cleanup
 - **Why Grafana Tempo:** End-to-end trace backend connecting FastAPI request spans to Celery task execution and LangFuse LLM calls
-



GRAFANA MIMIR

- Distributed mode (v3.0.4) with ingestion limits tuned to 100K series/s rate and 200K burst — handles cluster restart metric cascades
 - Central MinIO backend with bundled MinIO + Memcached sub-charts disabled — single storage instance, leaner footprint
 - Single-tenant, PromQL-compatible — all Grafana dashboards and alerting rules work without modification
 - **Why Grafana Mimir:** Long-term metrics store with PromQL compatibility — replaces ephemeral Prometheus TSDB with durable MinIO-backed object storage
-

PYTHON ECOSYSTEM

LLM & AGENT ORCHESTRATION



LANGCHAIN



LANGGRAPH



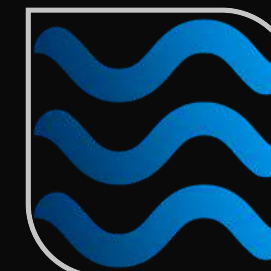
DEEPAGENTS



LITELLM



LANGFUSE



RIVER

DATA & RETRIEVAL CLIENTS



QDRANT



NEO4J



ELASTICSEARCH



PSYCO PG
(POSTGRES SQL)



BOTO3
(MINIO/S3)

INGESTION & PARSING



CRAWL4AI



PLAYWRIGHT



YT-DLP



BEAUTIFULSOUP4



LXML



MARKDOWN-IT-PY

WEB FRAMEWORK & SERVICES



FASTAPI



FASTHTML



FASTMCP



CELERY



REDIS

CORE & OBSERVABILITY



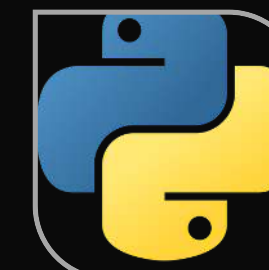
PYDANTIC



NUMPY



CRYPTOGRAPHY



HTTPX



TENACITY



OPENTELEMETRY

THE MICROSERVICES ENGINE

A closer look at the services powering COELHO Nexus — the 4-service application layer (FastAPI, FastHTML, FastMCP, Celery), 6 data stores (PostgreSQL, Neo4j, Qdrant, Elasticsearch, MinIO, Redis), and the observability pipeline (OpenTelemetry → Grafana Alloy → Loki/Tempo/Mimir → Grafana, plus LangFuse for LLM traces).



 COELHO Nexus

Home

Docs Distiller

YouTube Content Search

Research Radar



COELHO Nexus

Built for the operator who needs to **search videos**

One control plane for three knowledge surfaces: framework documentation, YouTube transcripts, and academic research feeds. Every LLM call routed through an adaptive bandit, every pipeline checkpointed for surgical replay.

Open Docs Distiller →

[Browse all surfaces](#)

7

FRAMEWORKS INGESTED

Docs Distiller

4

VIDEOS PROCESSED

YouTube Content Search

13

RADAR SCANS LAUNCHED

Research Radar

8.5 MB

CORPUS SIZE

across all surfaces

Docs Distiller

LIVE

YouTube Content Search

LIVE

Research Radar

LIVE

The Actual Homepage

Live numbers pulled from the running system, not placeholder stats. One control plane for all three features, with the LLM Rotator's settings one click away.



MICROSERVICES - FASTAPI

COELHO Nexus - FastAPI

/openapi.json

COELHO Nexus - FastAPI

LLM Rotator

GET	/api/v1/llm/health	Llm Health	⌵
GET	/api/v1/llm/settings/providers	List Providers	⌵
GET	/api/v1/llm/settings/readiness	Readiness	⌵
GET	/api/v1/llm/settings/providers/health	Providers Health	⌵
GET	/api/v1/llm/settings/providers/{pid}/models	Provider Models	⌵
POST	/api/v1/llm/settings/providers/{pid}/models	Set Provider Models	⌵
POST	/api/v1/llm/settings/providers/{pid}/key	Set Provider Key	⌵
DELETE	/api/v1/llm/settings/providers/{pid}/key	Delete Provider Key	⌵
POST	/api/v1/llm/settings/providers/{pid}/test	Test Provider Key	⌵

Auto-Generated API Contract

Every domain — Docs Distiller, YCS, Research Radar, LLM Rotator — ships a live OpenAPI 3.1 schema at /openapi.json. 124 REST endpoints across 4 domains, always in sync with the code because it's generated from it, not hand-written.

LLM Rotator Settings API

The exact endpoints behind the BYOK Settings page — list providers, check readiness, set or delete a key, test it live. The entire free-tier routing pool is reconfigurable without touching a deployment.



MICROSERVICES - CELERY (FLOWER)

Flower Workers Tasks Broker Documentation

Show 15 workers Search:

Worker	Status	Active	Processed	Failed	Succeeded	Retried	Load Average
worker@coelhonexus-celery-ff8b655c6-jp5js	Online	0	0	0	0	0	5.42, 5.26, 5.57
Total		0	0	0	0	0	

Showing 1 to 1 of 1 workers

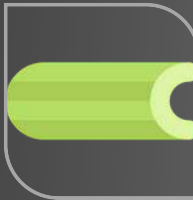
Previous1Next

6 Domain-Isolated Task Queues

Every domain — ingestion, planner, synth, YCS, Research Radar — runs on its own Celery queue, environment-suffixed so dev and prod share one Redis broker without stealing each other's jobs. Research Radar's long DeepAgents runs get a dedicated queue so they never block the rest of the pipeline.

1 Live Worker Health, Zero Extra Code

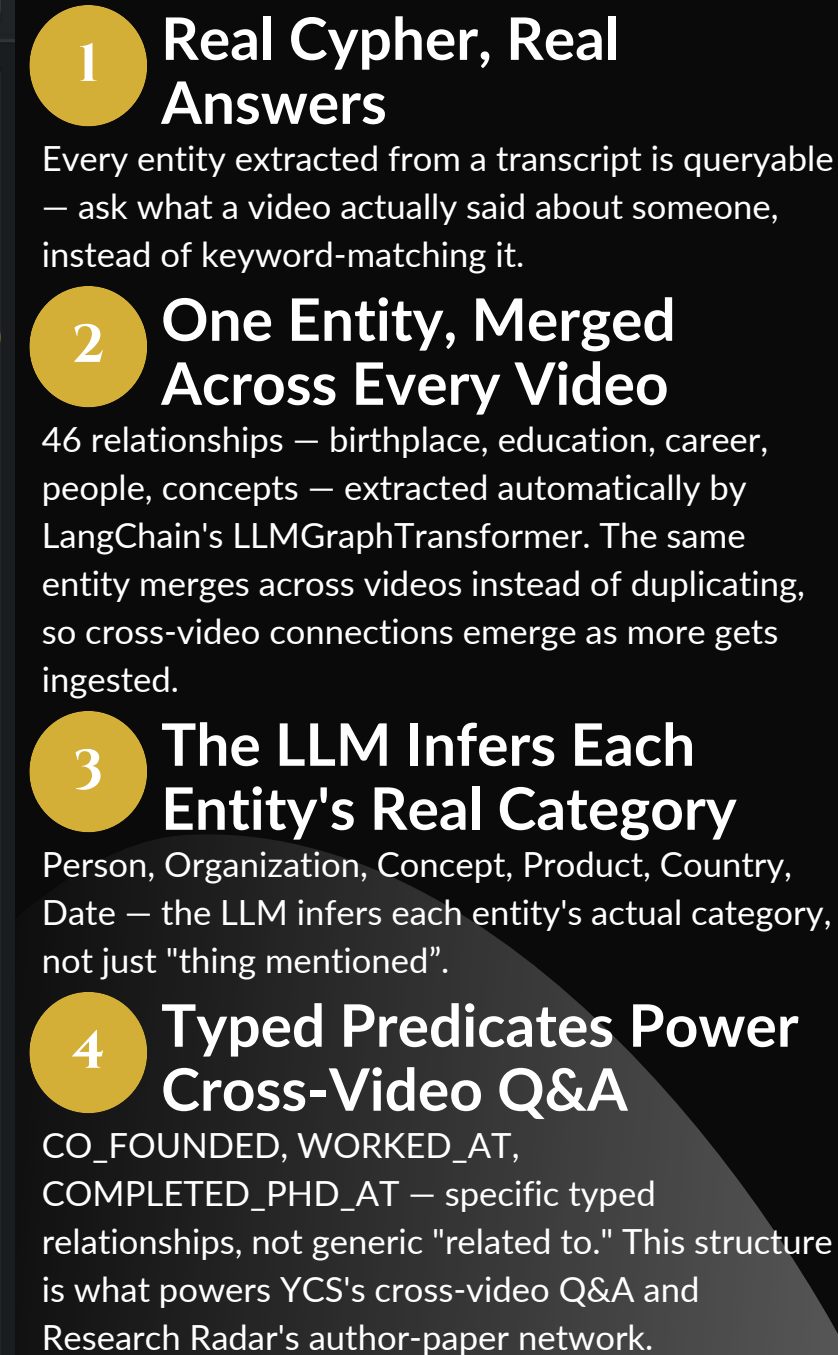
The worker name is the actual running Kubernetes pod (coelhonexus-celery-ff8b655c6-jp5js), auto-discovered by Flower over the Redis broker. Active, Processed, Failed, Succeeded, Retried, and load average update in real time — full task-level observability Celery ships with out of the box.



COELHO NEXUS

http://localhost:23001

Password: **neo4j-demo-password**



MICROSERVICES - QDRANT



Welcome

Console

Collections

Tutorial

Datasets

Access Tokens

Qdrant v1.17.1

Qdrant

🔑 🔒 🌙 ☀️

Collections

+ Create Collection

📄 Upload Snapshot

🔍 Search Collection

1

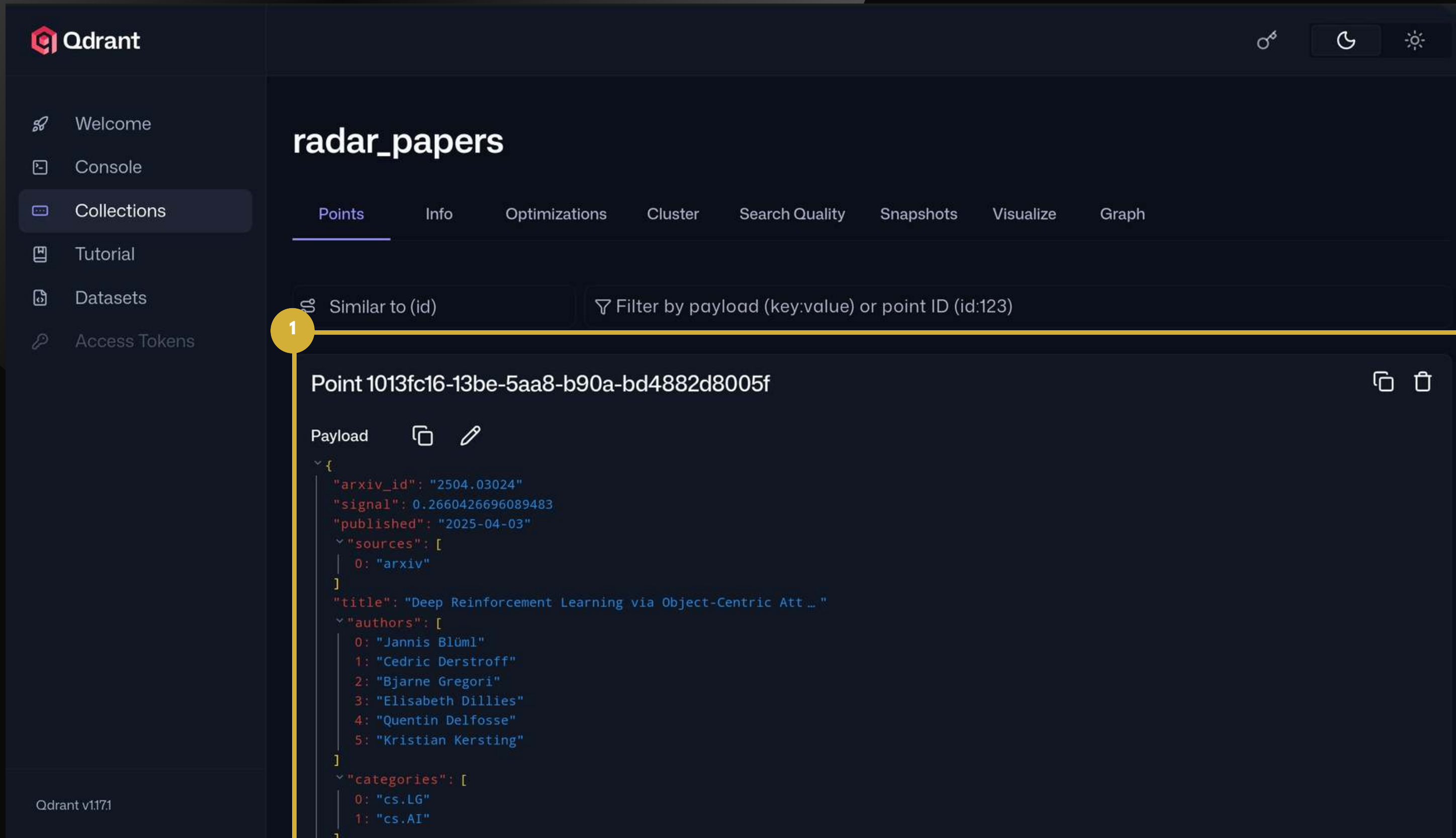
NAME	STATUS	POINTS (APPROX)	SEGMENTS	SHARDS	VECTORS CONFIG	ACTIONS
radar_papers	● GREEN	19	2	1	Default 2048 Cosine	⋮
youtube-transcripts	● GREEN	67	4	1	dense 2048 Cosine sparse Sparse	⋮

1 Two Collections, Two Features, Live

radar_papers backs Research Radar's paper dedup; youtube-transcripts backs YCS's semantic search — and it's hybrid, pairing dense 2048-dim embeddings with sparse vectors in one collection for better recall than similarity search alone.



MICROSERVICES - QDRANT



Qdrant

Welcome
Console
Collections
Tutorial
Datasets
Access Tokens

radar_papers

Points Info Optimizations Cluster Search Quality Snapshots Visualize Graph

Similar to (id) Filter by payload (key:value) or point ID (id:123)

1

Point 1013fc16-13be-5aa8-b90a-bd4882d8005f

Payload

```
{
  "arxiv_id": "2504.03024"
  "signal": 0.2660426696089483
  "published": "2025-04-03"
  "sources": [
    0: "arxiv"
  ]
  "title": "Deep Reinforcement Learning via Object-Centric Att..."
  "authors": [
    0: "Jannis Blüml"
    1: "Cedric Derstroff"
    2: "Bjarne Gregori"
    3: "Elisabeth Dillies"
    4: "Quentin Delfosse"
    5: "Kristian Kersting"
  ]
  "categories": [
    0: "cs.LG"
    1: "cs.AI"
  ]
}
```

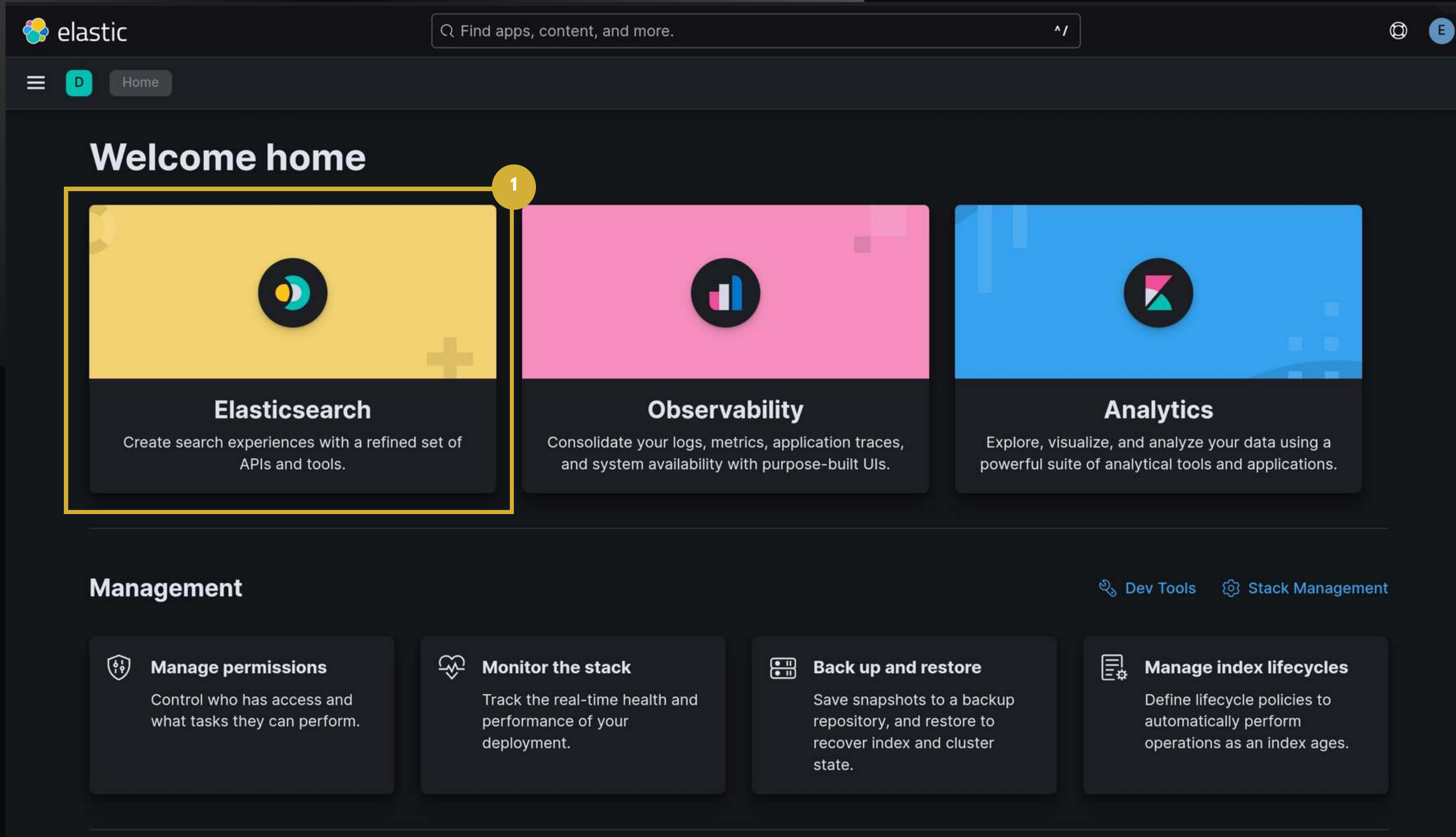
Qdrant v1.17.1

1 Every Payload Carries Its Own Relevance Score

Each point in the collection carries its own relevance score, calculated based on the real arXiv metadata — title, authors, categories, publish date — plus signal, Research Radar's own computed relevance score that ranks papers before they ever reach the digest. The embedding is just one field in a much richer record.



MICROSERVICES - ELASTICSEARCH (KIBANA)



1 Accessing Elasticsearch Data

This tile is the entry point into Elasticsearch from Kibana — click it to browse, search, and manage the youtube-metadata and youtube-transcriptions indexes directly, no separate tool needed.

Kibana

<https://localhost:23014>

Username: **elastic**

Password: **coelhonexus-demo-password**

ElasticSearch (API)

<https://localhost:23013>

Username: **elastic**

Password: **coelhonexus-demo-password**



MICROSERVICES - ELASTICSEARCH (KIBANA)

elastic

Find apps, content, and more.

^/

E

≡

D

Elasticsearch

Content

Elasticsearch indices

Endpoints & API keys

Elasticsearch

Home

Content

Indices

Connectors

Web Crawlers

Build

Search Applications

Behavioral Analytics

Relevance

Synonyms

Getting started

Elasticsearch

Vector Search

Semantic Search

AI Search

Elasticsearch indices

Default settings

Create a new index

⚠ Enterprise Search has not been configured

The Elastic web crawler is not available without Enterprise Search.

Review setup guide

Available indices

Show hidden indices

Only show crawler indices

Filter Elasticsearch indices

Index name	Index health	Docs count	Ingestion name	Ingestion method	Ingestion status	Actions
coelhonexus-youtube-metadata	● green	5		API	Connected	
coelhonexus-youtube-transcriptions	● green	5		API	Connected	

Console

Notebooks

- 1

Navigating to Elasticsearch Indices

Click Indices under Content in the left sidebar to see every index Elasticsearch is currently managing.
- 2

The Two Indices Powering YCS

coelhonexus-youtube-metadata and coelhonexus-youtube-transcriptions — both green, both API-connected, both actively ingesting. This is the real data YCS Ask searches against.



MICROSERVICES - ELASTICSEARCH (KIBANA)

elastic

Find apps, content, and more.

^/

E

Endpoints & API keys

D

Elasticsearch

Content

Elasticsearch indices

coelhonexus-youtube-transcriptions

Endpoints & API keys

Elasticsearch

Home

Content

Indices

COELHONEXUS-YOUTUB...

Overview

Documents

Index mappings

Pipelines

Connectors

Web Crawlers

Build

Search Applications

Behavioral Analytics

Relevance

Synonyms

Getting started

Elasticsearch

Vector Search

coelhonexus-youtube-transcriptions

View in Playground

Overview

Documents

Mappings

Pipelines

Search documents in this index

< 1 >

Showing 5 of 5. Search results maxed at 10,000 documents.

Document id: 96jN20C0fLs_en

Field	Contents
id	"96jN20C0fLs_en"
video_id	"96jN20C0fLs"
lang	"en"
	"We're so excited for our very first special guest. He has helped build modern AI, then explain modern AI, and then occasionally rename modern AI. He actually helped co-ound open AAI right inside of this office. Was the one who actually got Autopilot working at Tesla back in the day, and he has a great gift of making the most complex technical shifts for both search and..."

Console

Notebooks

- 1

Click Documents to See What's Actually Stored

The Documents tab lists every record in this index – real ingested transcript chunks, not just index metadata.
- 2

This Is What YCS Ask Actually Searches Against

id, video_id, lang, and the actual spoken text – this is what YCS Ask searches against: real transcript content, chunked and indexed per video.



MICROSERVICES - MINIO

MINIO

OBJECT STORE

AGPL LICENSE

User

Object Browser

Access Keys

Documentation

Administrator

Buckets

Policies

Identity

Monitoring

Events

Configuration

License

Object Browser

Filter Buckets

Name	Objects	Size	Access
backups	194,874	1.4 GiB	R/W
coelhonexus	5,389	61.2 MiB	R/W
loki-chunks	4,068	58.5 MiB	R/W
loki-ruler	0	0.0 B	R/W
mimir-alertmanager	0	0.0 B	R/W
mimir-blocks	76	3.1 GiB	R/W
mimir-ruler	0	0.0 B	R/W
tempo-traces	742	120.2 MiB	R/W

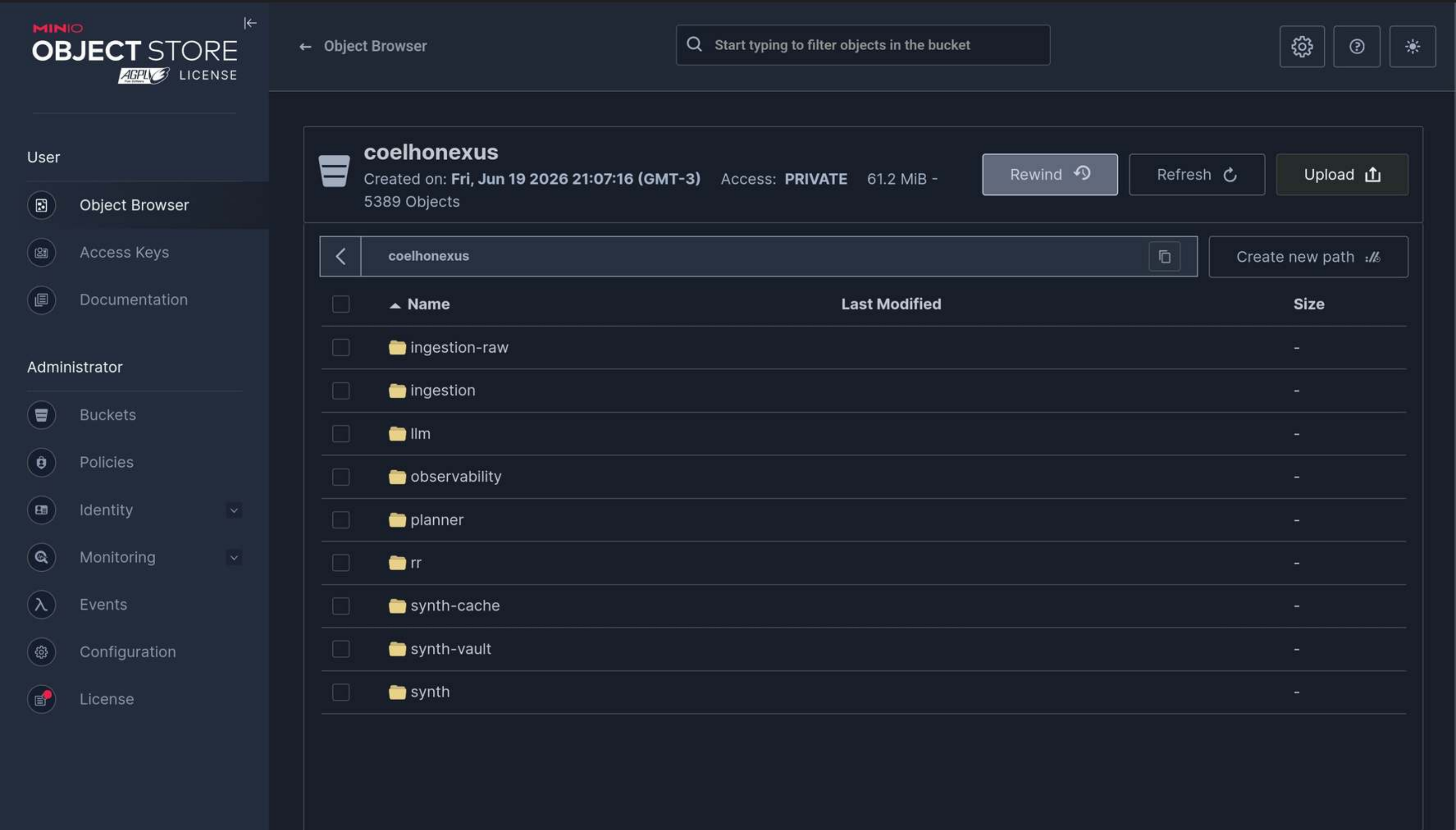
1

One Object Store, Every Backend's Data

Loki, Tempo, and Mimir don't use traditional databases — they chunk logs, traces, and metrics into S3-compatible objects, decoupling storage from compute so each scales independently. The same MinIO instance also holds coelhonexus — the application's own working data, covered on the next slide — proving one object-storage abstraction serves both the observability stack and the product itself.



MICROSERVICES - MINIO



1 One Bucket, Organized by Feature

ingestion, planner, and synth trace the Docs Distiller pipeline start to finish; rr holds Research Radar's artifacts; llm is the encrypted BYOK credential store powering the LLM Rotator. Nothing dumped flat into one namespace — every feature gets its own prefix, with synth-cache and synth-vault separating ephemeral working data from what's meant to persist.

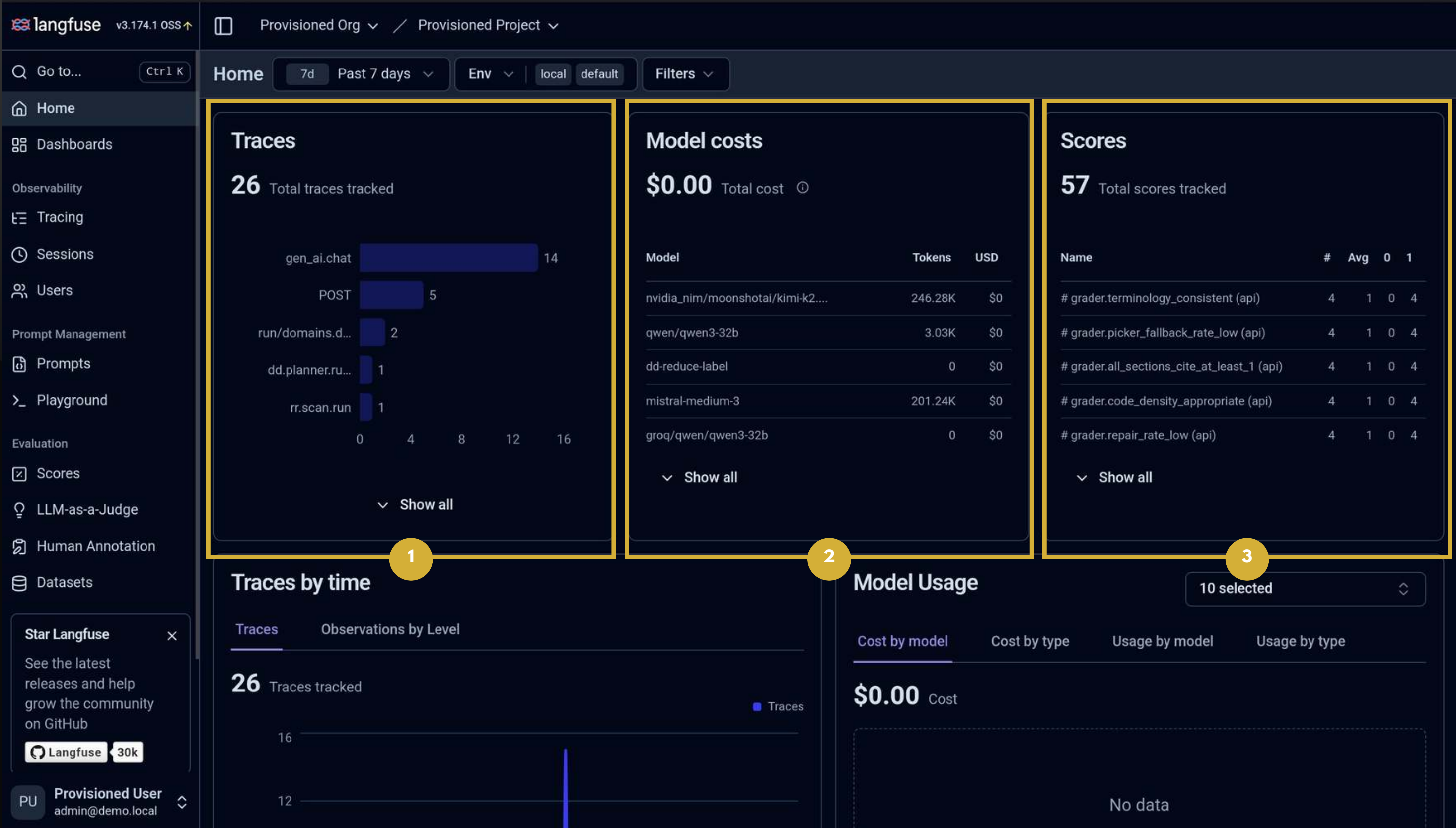
COELHO
NEXUS

MinIO (Console UI)
<http://localhost:23016>
Username: minioadmin
Password: minioadmin

MinIO (S3 API)
<http://localhost:23015>
Username: minioadmin
Password: minioadmin



MICROSERVICES - LANGFUSE



1 Every Signal Source, One Timeline

Auto-instrumented OTEL spans (gen_ai.chat) sit next to hand-written ones (dd.planner.run, rr.scan.run) in the same view. Every layer of the stack reports here, not just raw LLM calls.

2 Real Usage, Zero Dollars

Hundreds of thousands of tokens processed across multiple models, and every single one reads \$0.00 — this is the free-tier rotator's actual bill, not a theoretical claim.

3 Every Generation Gets Graded, Automatically

Named graders — terminology consistency, citation coverage, code density — score every generation automatically. Synth output gets judged the same way every time, not spot-checked by hand.



MICROSERVICES - LANGFUSE

langfuse

v3.174.1 OSS

Provisioned Org

Provisioned Project

Go to...

Ctrl K

Home

Dashboards

Observability

Tracing

Sessions

Users

Prompt Management

Prompts

Playground

Evaluation

Scores

LLM-as-a-Judge

Human Annotation

Datasets

Star Langfuse

See the latest releases and help grow the community on GitHub

Langfuse 30k

Provisioned User

admin@demo.local

Tracing

Traces Observations

Hide filters

Views 0

Search...

IDs / Names

7d Past 7 days

Off

Columns 29/41

Filters

Environment

Mode

SELECT TEXT

local

default

Trace Name

Mode

SELECT TEXT

gen_ai.chat 14

POST 5

run/domains.d... 2

(empty) 1

dd.planner.run 1

rr.scan.run 1

yca.ask.stream... 1

yca.ingest.neo... 1

dd.synth.study... 1

Trace ID

User ID

Session ID

Metadata

Timestamp	Name	Input	Output
2026-06-28 19:55:53	rr.scan.run	{ "topic": "deep agents", "verticals": ["cs.LG", "cs.AI"], "top_n": 4, "scan_id": "779612..." }	{ "status": "done", "n_findings": 4, "total_candidates": 10 }
2026-06-28 17:53:55	gen_ai.chat	"You are a senior Neo4j engineer. \nGenerate ONE Cypher query that answers the following question: 'What are the main themes across all videos?'"	MATCH (v:Video) WHERE toLower(v.title) CONTAINS 'deep agents'
2026-06-28 17:53:54	gen_ai.chat	"You are a senior Neo4j engineer. \nGenerate ONE Cypher query that answers the following question: 'What are the main themes across all videos?'"	MATCH (v:Video) WHERE toLower(v.title) CONTAINS 'deep agents'
2026-06-28 17:53:40	gen_ai.chat	"You are a senior Neo4j engineer. \nGenerate ONE Cypher query that answers the following question: 'What are the main themes across all videos?'"	{ }
2026-06-28 17:53:39	gen_ai.chat	"You are a senior Neo4j engineer. \nGenerate ONE Cypher query that answers the following question: 'What are the main themes across all videos?'"	MATCH (d:Document) WHERE toLower(d.title) CONTAINS 'deep agents'
2026-06-28 17:52:41	gen_ai.chat	"You are a senior Neo4j engineer. \nGenerate ONE Cypher query that answers the following question: 'What are the main themes across all videos?'"	{ }
2026-06-28 17:52:32	gen_ai.chat	"You are a senior Neo4j engineer. \nGenerate ONE Cypher query that answers the following question: 'What are the main themes across all videos?'"	MATCH (d:Document) WHERE toLower(d.title) CONTAINS 'deep agents'
2026-06-28 17:52:19	gen_ai.chat	"You are a senior Neo4j engineer. \nGenerate ONE Cypher query that answers the following question: 'What are the main themes across all videos?'"	{ }
2026-06-28 17:51:40	gen_ai.chat	"You are a senior Neo4j engineer. \nGenerate ONE Cypher query that answers the following question: 'What are the main themes across all videos?'"	MATCH (v:Video) WHERE v.title CONTAINS 'deep agents'
2026-06-28 17:51:27	gen_ai.chat	"You are a senior Neo4j engineer. \nGenerate ONE Cypher query that answers the following question: 'What are the main themes across all videos?'"	MATCH (v:Video) WHERE v.title CONTAINS 'deep agents'
2026-06-28 17:51:02	gen_ai.chat	"You are a senior Neo4j engineer. \nGenerate ONE Cypher query that answers the following question: 'What are the main themes across all videos?'"	MATCH (v:Video) WHERE v.title CONTAINS 'deep agents'
2026-06-28 17:50:44	gen_ai.chat	"You are a senior Neo4j engineer. \nGenerate ONE Cypher query that answers the following question: 'What are the main themes across all videos?'"	MATCH (v:Video) WHERE v.title CONTAINS 'deep agents'
2026-06-28 17:46:41	gen_ai.chat	"You are a senior Elasticsearch engineer. \nGenerate ONE Elasticsearch _source for the following query: 'What are the main themes across all videos?'"	{ }
2026-06-28 17:46:39	gen_ai.chat	"You are a senior Elasticsearch engineer. \nGenerate ONE Elasticsearch _source for the following query: 'What are the main themes across all videos?'"	{ "query": { "multi_match": { "query": "Andrej", "fields": ["title", "description"] } } }
2026-06-28 17:46:34	gen_ai.chat	"You are a senior Elasticsearch engineer. \nGenerate ONE Elasticsearch _source for the following query: 'What are the main themes across all videos?'"	{ }
2026-06-28 17:09:57	yca.ask.stream.run	{ "question": "What are the main themes across all videos?", "route": "search_videos" }	{ "status": "done", "answer": "**TL;DR:** The video discusses the importance of deep agents in the current landscape of AI and machine learning." }
2026-06-28 15:36:11	yca.ingest.neo4j.run	{ "kind": "neo4j", "video_ids_preview": ["96jN20C0fLs", "LCemiRjPEtQ", "qh0Fkd..."] }	{ "status": "done", "kind": "neo4j", "task_id": "be2..." }
2026-06-27 22:38:45	POST		

Rows per page 50

Page 1 of 1

1 One Decorator, Every Domain's Traces

DD, YCS, and RR each wrap their pipeline nodes with the same @traced pattern – one decorator turns a function into a span that reports to both the infra observability stack and LangFuse at once. Research Radar nests phase-level spans as children of its parent scan. Nobody wired LangFuse into each domain separately; every feature reuses the same primitive.



MICROSERVICES - LANGFUSE

langfuse v3.174.1 OSS ↑

Go to... Ctrl K

Home

Dashboards

Observability

Tracing

Sessions

Users

Prompt Management

Prompts

Playground

Evaluation

Scores

LLM-as-a-Judge

Human Annotation

Datasets

Star Langfuse

See the latest releases and help grow the community on GitHub

Langfuse 30k

Provisioned User admin@demo.local

Provisioned Org

Provisioned Project

Tracing

Traces Observations

Hide filters

Views 0

Search...

Filters

Environment

Mode:

SELECT TEXT

local

default

Trace Name

Mode:

SELECT TEXT

gen_ai.chat 14

POST 5

run/domains.d... 2

(empty) 1

dd.planner.run 1

rr.scan.run 1

ycs.ask.stream... 1

ycs.ingest.neo... 1

dd.synth.study... 1

Trace ID

User ID

Session ID

Metadata

Trace ycs.ask.stream.run: b680ab373d30ef4310b650f779923458

Search

Timeline

ycs.ask.stream.run 9m 50s

ycs.ask.stream.run 9m 50s

ycs.node.rag.contextualize

ycs.node.rag.classify 20.69s

gen_ai.chat 20.68s 731 → 129 (Σ 860)

ycs.node.rag.plan 0.65s

gen_ai.chat 0.65s 116 → 104 (Σ 220)

ycs.node.rag.subagent 2m 25s

ycs.node.rag.retrieve 5.82s

ycs.retriever.smart_fanout 5.82s

gen_ai.chat 0.37s 301 → 18 (Σ 319)

gen_ai.rerank 2.89s

ycs.node.rag.grade 48.77s

gen_ai.chat 1.17s 816 → 221 (Σ 1,037)

gen_ai.chat 18.69s 592 → 9 (Σ 601)

gen_ai.chat 3.47s 672 → 127 (Σ 799)

gen_ai.chat 0.45s 863 → 12 (Σ 875)

gen_ai.chat 0.73s 600 → 11 (Σ 611)

gen_ai.chat 1.02s 807 → 357 (Σ 1,164)

gen_ai.chat 30.00s

ycs.ask.stream.run ID

Add to datasets Annotate Add comment

2026-06-28 17:09:57.132

Latency: 9m 50s Session: dc80ccd92200 User ID: default

Env: local Version: 1.0.0

54,058 prompt → 11,005 completion (Σ 65,063)

Preview Log View Scores

Formatted JSON

Input

Path Value

question "What are the main themes across all videos?"

route "search_stream"

force_mode ""

channel_ids empty list

thread_id "dc80ccd92200"

Output

Path Value

status "done"

answer ***TL;DR:** The videos chart a transition to AI-driven software—"vibe coding" and agentic engineering—while stressing new documentation practices, the democratizing promise of LLMs, and the need to manage their limits, all delivered in a mix of conversational, visionary, and pragmatic tones.

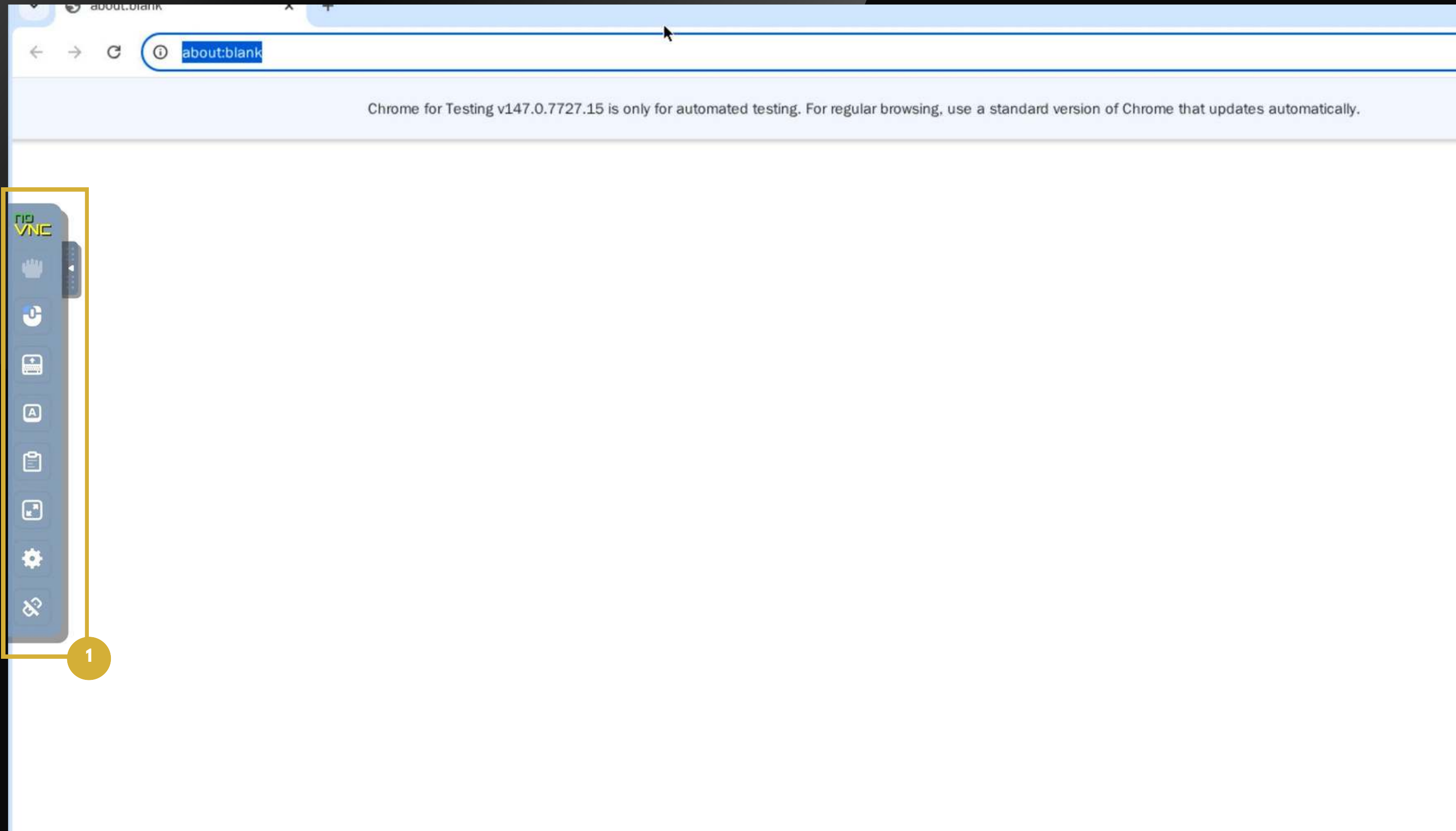
AI-driven software is the core narrative
The speakers frame LLMs and AI agents as the next layer of software, replacing line-by-line coding with "vibe coding" and "agentic engineering" that

1 What Gets Sent to LangFuse, Automatically

Every span here — the parent trace, nested sub-calls, each model call's own token count and latency — is data the instrumentation transmitted the moment @traced wrapped a function. No node writes its own logging code; the payload assembles itself at every nesting depth and rolls up into one session automatically.



MICROSERVICES - PLAYWRIGHT



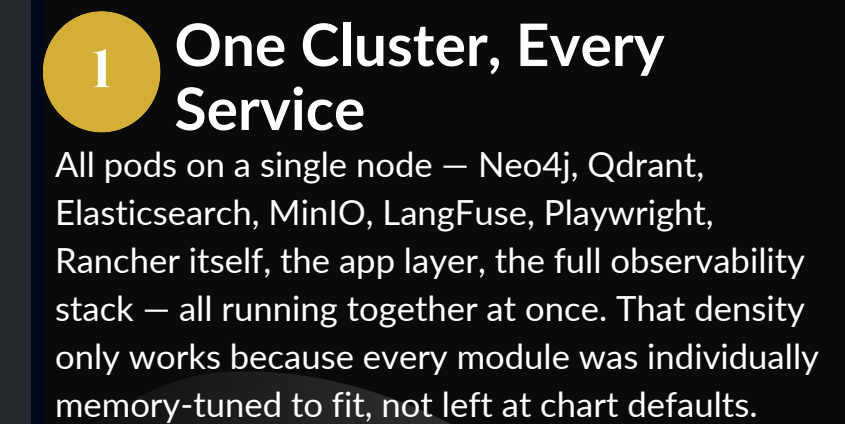
Headed/Headless Modes, One Real Browser

Headed mode renders with a full display — this noVNC view is watching it live — because YCS needs a real, visible Chromium to get past YouTube's anti-bot detection during transcript scraping. Headless mode drops the display entirely for Crawl4AI's bulk framework-doc crawls, where speed matters more than visibility.

1 A Live Session You Can Actually Take Over

Clipboard sync, a virtual keyboard, connection settings — this sidebar turns noVNC from a passive screen-share into an actual remote desktop. If a scrape stalls on an unexpected popup or verification step, a human can click into this exact browser session and clear it by hand, live, instead of just watching it fail.







- 1 One Click to Every Running Workload**

Workloads groups every execution primitive Kubernetes has — CronJobs, DaemonSets, Deployments, StatefulSets, Jobs, raw Pods — into one menu. Selecting Deployments here surfaces all 25 running on the cluster, not just this project's own.
- 2 Every Microservice, One Screen**

This is the screen that shows every namespace and every deployment on the cluster at once — every microservice this project depends on to actually run, all in one place, fully visible and usable from here.

Password: **rancher-demo-bootstrap**

COELHO NEXUS

MICROSERVICES - GRAFANA

 Grafana

Home

Bookmarks

Starred

Dashboards

Explore

Drilldown

Alerting

Connections

Administration

Dashboards

Search...

ctrl+k

+

?



Dashboards

Create and manage dashboards to visualize your data

Search for dashboards and folders

Filter by tag

Starred

Sort

Name

> AI

> CI-CD

> Databases

> Network

> Observability

> Storage

COELHO Nexus - DD Pipeline — Planner + Study Health

COELHO Nexus - Investigations

COELHO Nexus - LLM Rotator — Discovery + Benchmark Health

COELHO Nexus - Overview

COELHO Nexus - RR — Research Radar Scan Health

COELHO Nexus - Service Topology

COELHO Nexus - YCS Ask — Retrieval + Answer Health

Loki / Chunks

Tags

coelhonexus dd mimir otel planner study

coelhonexus investigation lgtm logs metrics traces

coelhonexus llm mimir otel rotator

coelhonexus dd lgtm otel overview rr ycs

coelhonexus mimir otel pipeline rr

coelhonexus lgtm mimir service-graph tempo topc

ask coelhonexus mimir otel retrieval ycs

loki

1

Every Signal, Cross-Linked

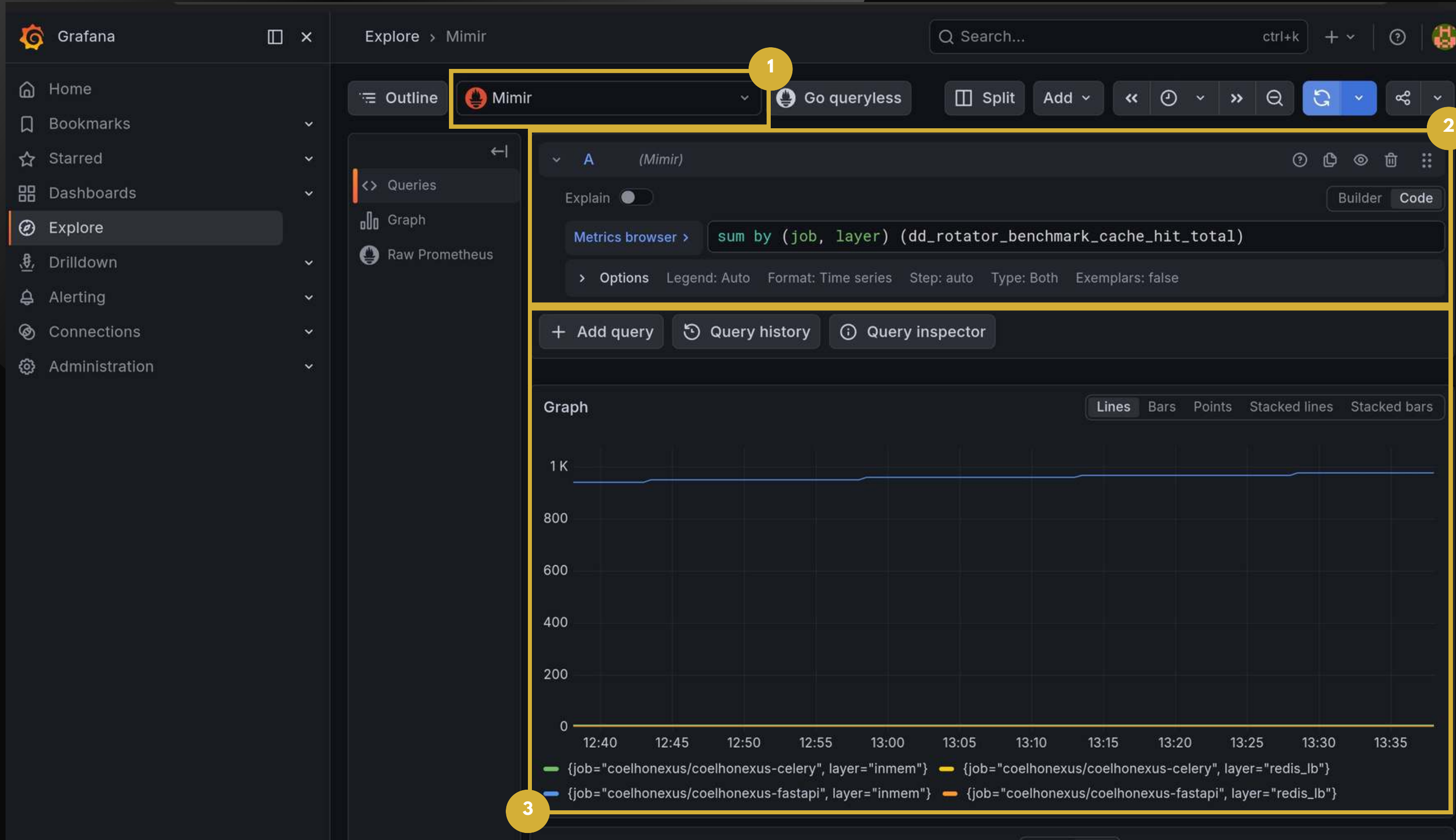
One dashboard per feature, plus a cross-signal Investigations view — but the real design choice is that they're wired together, not just co-located: a slow metric has an exemplar link straight to the exact trace that caused it, and every trace links onward to its own logs. These dashboards ship inside the application's own Helm chart, not a separate infra concern, so they version alongside the features they actually monitor.

COELHO
NEXUS

Grafana
http://localhost:23022
Username: admin
Password: admin



MICROSERVICES - GRAFANA



1 One Explorer, Three Backends

This dropdown is the only thing that changes between querying metrics, logs, or traces — Mimir, Loki, and Tempo all live behind the same Explore interface, same shortcuts, same UI, just a different datasource selected.

2 PromQL You Write, With Help Finding What Exists

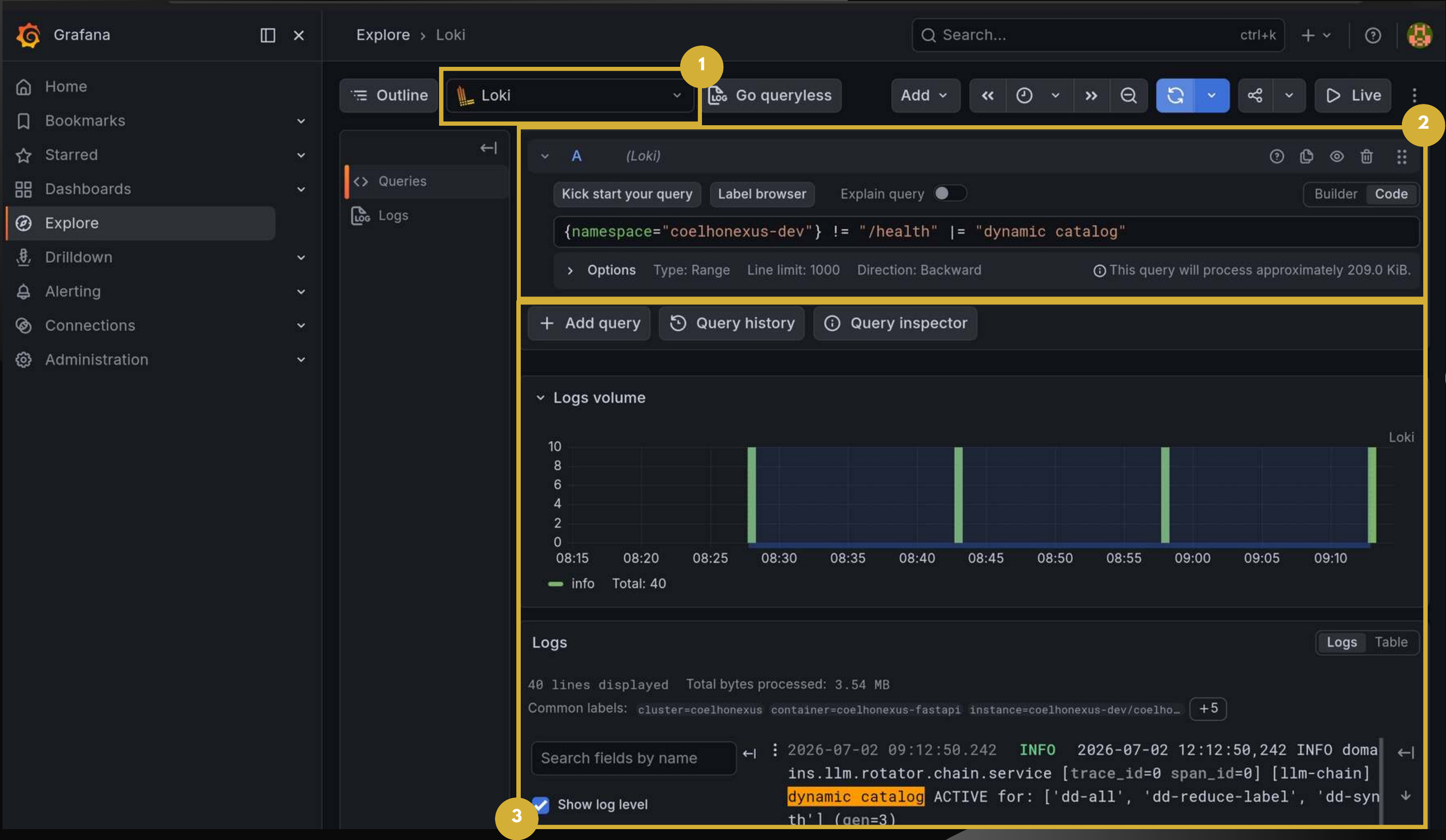
Metrics Browser helps you discover what's available without memorizing exact names, but the query itself is hand-written PromQL — `sum by (job, layer)` — full query-language control, not a simplified GUI-only builder.

3 Where the Query Actually Resolves

This panel renders whatever query sits above it — line graph, legend, time axis — built automatically the moment the query runs. Change the query and the whole panel redraws instantly, no dashboard to build first, no separate step to visualize the result.



MICROSERVICES - GRAFANA



1 A Different Signal, A Different Language

Metrics aggregate into numbers; logs stay raw text. Loki's query reflects that — label-matching to pick a stream, then a text filter, not a math expression. Same Explore shell as Mimir, a completely different way of asking a question.

2 Where You Write and Run the Query

This is the query editor — type LogQL here, run it, and the results below update immediately. Everything else in this box is just support tooling around that one core action.

3 Where the Results Show Up

This is where matching log lines appear — a volume histogram on top, the actual text underneath. Whatever the query above finds, it renders here, nothing to configure separately.

Grafana

<http://localhost:23022>

Username: admin

Password: admin

COELHO
NEXUS



Home

Bookmarks

Starred

Dashboards

Explore

Drilldown

Alerting

Connections

Administration

Explore > Tempo

Q Search...

ctrl+k

+

?

Outline

Tempo

Go queryless

Split

Add

<<

🕒

>>

🔍

🔄

🔗

Queries

Table

Query type

Search

TraceQL

Service Graph

Import trace

Service Name

=

Select value

Span Name

=

Select value

Status

=

Select value

Duration

span

>

e.g. 100ms, 1.2s

<

e.g. 100ms, 1.2s

Tags

span

Select tag

=

Select value

{ }

Edit in TraceQL

> Search Options

Limit: 20

Spans Limit: 3

Table Format: Traces

Streaming: Disabled

> Metrics Options

Step: auto

Type: Range

Streaming: Disabled

+ Add query

🕒 Query history

🔍 Query inspector

Table - Traces

	Trace ID	Start time	Service	Name	Duration
>	12bbaf462a63d342b6	2026-07-02 13:44:38	coelhonexus-fastmcp	GET /health	<1 ms
>	12066ccd96a600ba87	2026-07-02 13:44:33	coelhonexus-fastmcp	GET /health	1 ms
>	f36ad742b2ef4a023b9	2026-07-02 13:44:28	coelhonexus-fastmcp	GET /health	<1 ms

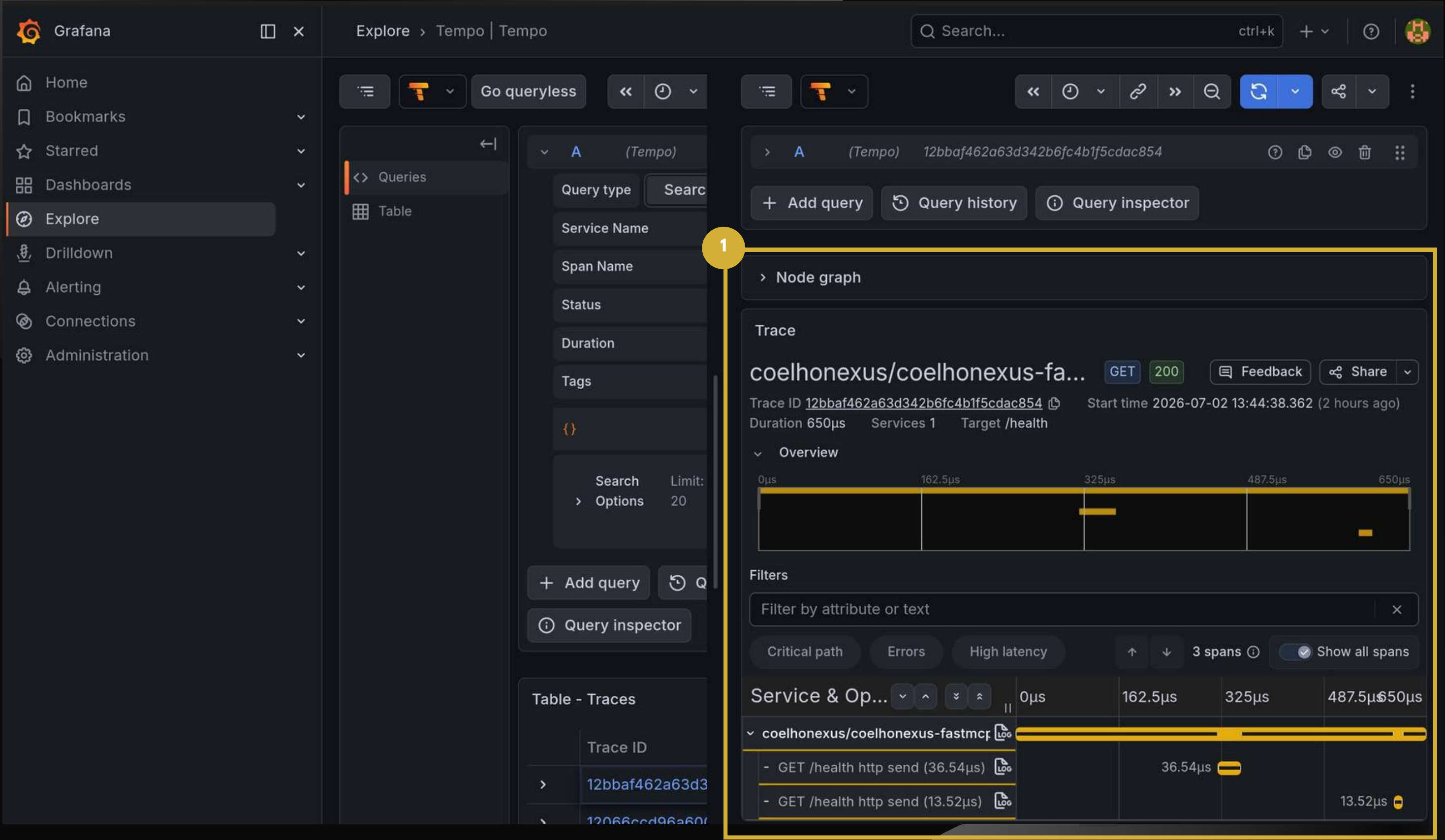
Traces aren't numbers or text lines — they're a hierarchy of spans, parent to child. Tempo's picker sits in the same Explore shell as Mimir and Loki, but what it queries is shaped completely differently.

Same core action as the other two — configure a query, run it, see results below — but Tempo defaults to a filter form instead of free-text code. "Edit in TraceQL" is right there if you want to drop into the query language directly.

This is where matching traces appear — one row per trace, service, name, and duration at a glance. Click any row to expand its full span waterfall.



MICROSERVICES - GRAFANA



1 One Trace, Every Span, Expanded

Clicking a row from the results table opens this – the parent request and every child span nested underneath it, each with its own exact duration. Same waterfall view for any trace in the system, a health check or a full pipeline run alike.



MICROSERVICES - ARGOCD



argo

v3.3.6

Applications

Settings

User Info

Documentation

Resource filters

NAME

NAME

KINDS

KINDS

SYNC STATUS

☐ Synced 4

☐ OutOfSync 0

HEALTH STATUS

☐ Progressing 0

☐ Suspended 0

☐ Healthy 5

Applications / coelhonexus

APPLICATION DETAILS NETWORK

DETAILS

DIFF

SYNC

SYNC STATUS

HISTORY AND ROLLBACK

DELETE

REFRESH

Log out

APP HEALTH  Healthy

SYNC STATUS  Synced to master (7b3d786) 
Auto sync is enabled.
Author: rafaelfcoelho1409 <rafaelfcoelho1409@hotmail...>
Comment: Excessive comments removed from codes, ...

LAST SYNC  Sync OK to 7b3d786 
Succeeded 19 minutes ago (Thu Jul 02 2026 18:01:26 GMT-0300)
Author: rafaelfcoelho1409 <rafaelfcoelho1409@hotmail...>
Comment: Excessive comments removed from codes, ...



172.18.0.2



172.18.0.3



172.18.0.4



coelhonexus-celery-59ccbb66...



9 minutes running 2/2 more

coelhonexus-fastmcp



20 minutes

coelhonexus-fastapi



20 minutes

coelhonexus-fasthtml



20 minutes

coelhonexus-flower



20 minutes

The Graph Proves What the Code Says

Three services trace back through Docker-network IPs to the cloud icon — those are the ones with a LoadBalancer path out. FastMCP sits apart with no such line, because its Service is deliberately ClusterIP-only: an internal gateway other pods call, never meant to be reached directly. The topology isn't decorative — it's the same internal/external split the Service manifests define, just drawn instead of read.

ArgoCD

<http://localhost:23023>

Username: **admin**

Password: **admin**

COELHO
NEXUS



LET'S WORK TOGETHER



<https://rafaelcoelho.pages.dev/>