2nd Annual

MidPoint Community Meetup

MidPilot – A Look Behind the Curtain

Tony Tkáčik - Backend Technical Leader

Martin Bielik – AI/ML Developer

Michal Zelenčík – AI/ML developer

Evolveum

Funded by the European Union
NextGenerationEU

RECOVERY AND RESILIENCE PLAN

Agenda

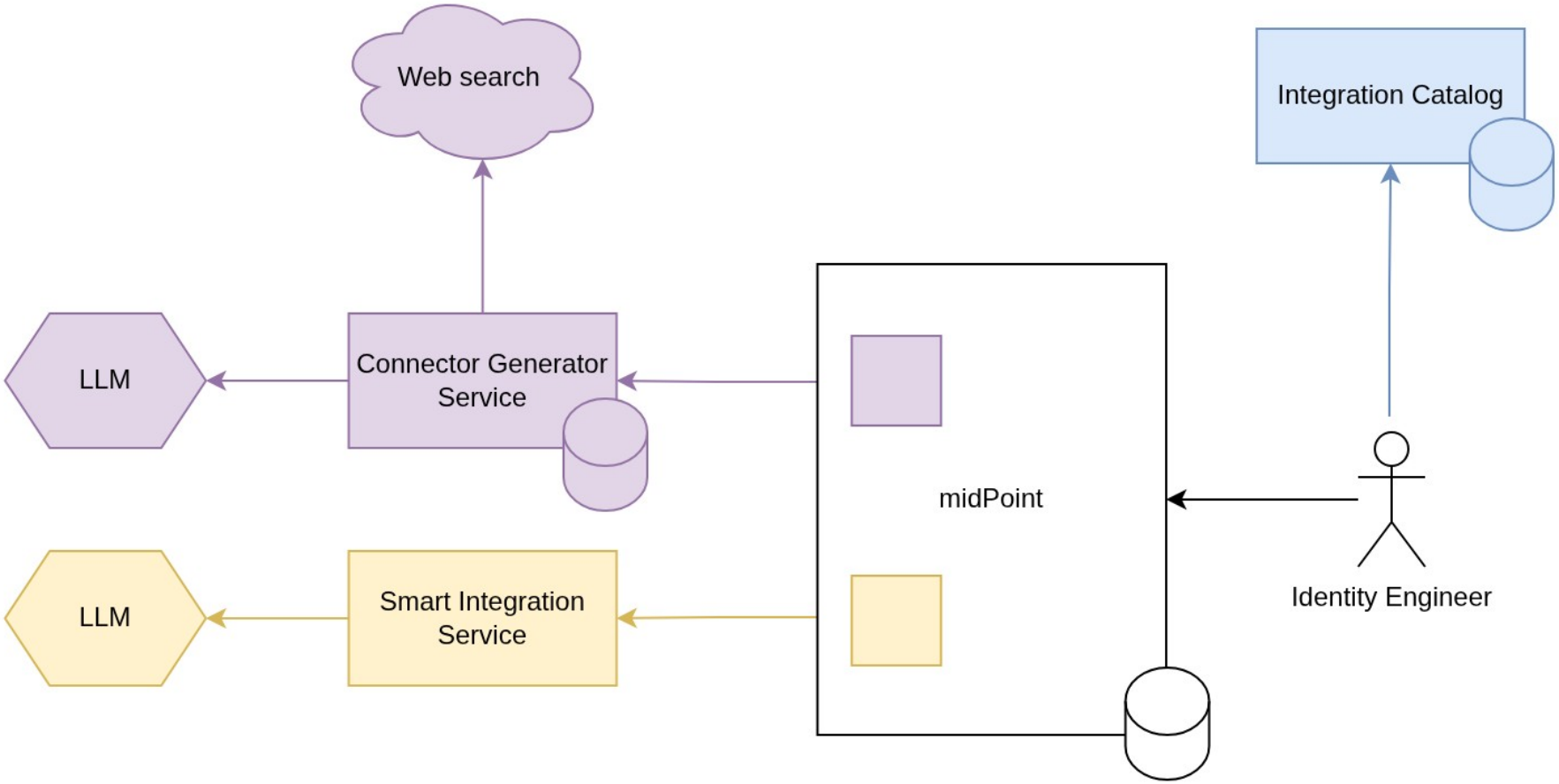
- MidPilot overview and architecture
- Insights from development & lessons learned

MidPilot project

MidPilot project

- AI-powered connector development
- Assisted classification, correlation and mappings

MidPilot project



AI-powered connector development

AI-powered connector development

create ConnId connectors easily

REST, SCIM, DB connectors (including ITSM systems)

wizard guide powered by AI

no-code/low-code config-only or simple code fragments

every execution is human-approved

integrated right into midPoint and MidPoint Studio

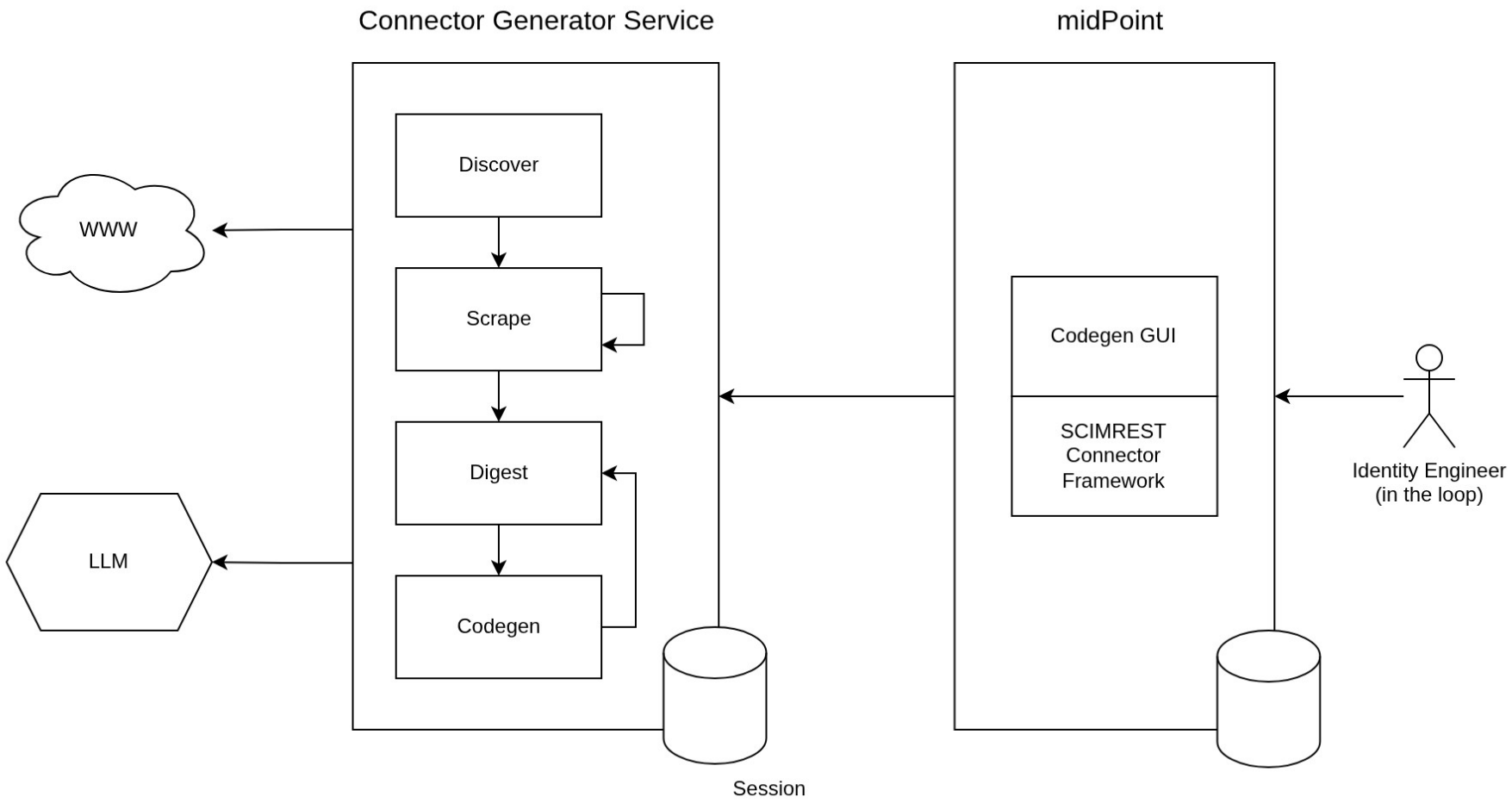
AI-powered connector development

Easier code development, review, and maintenance

Safe to use

Usable even for less technically experienced people

AI-powered connector development



Powered by

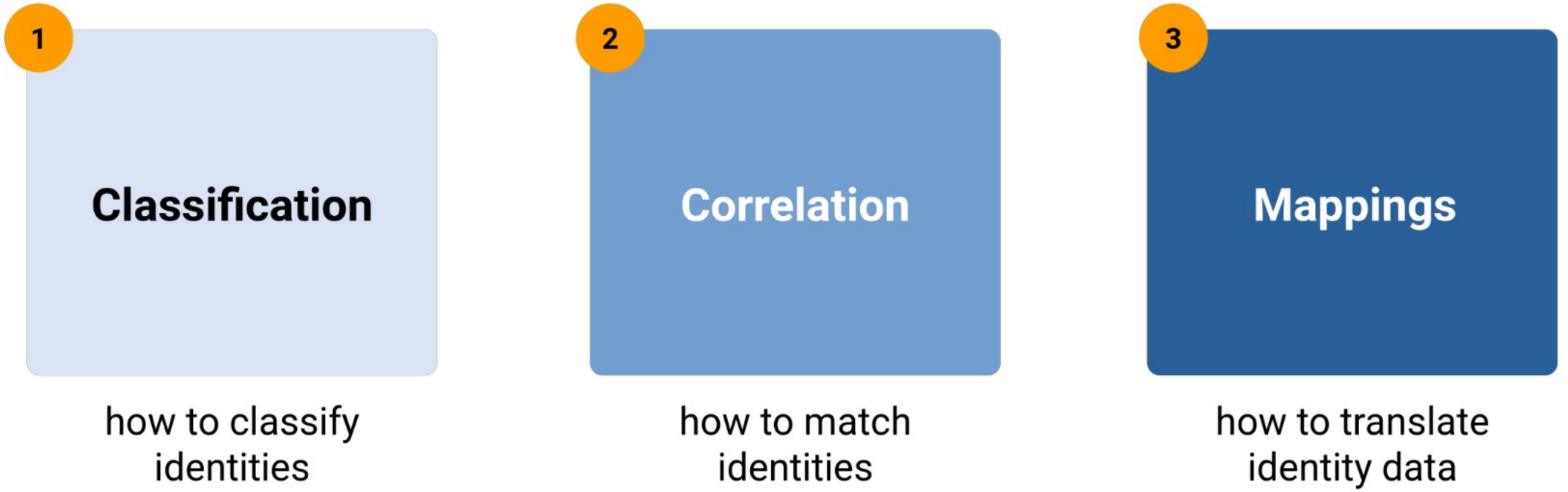
MidPilot project

Source code

- https://github.com/Evolveum/midpoint
- https://github.com/Evolveum/midpilot-connector-gen
- https://github.com/Evolveum/connector-scimrest

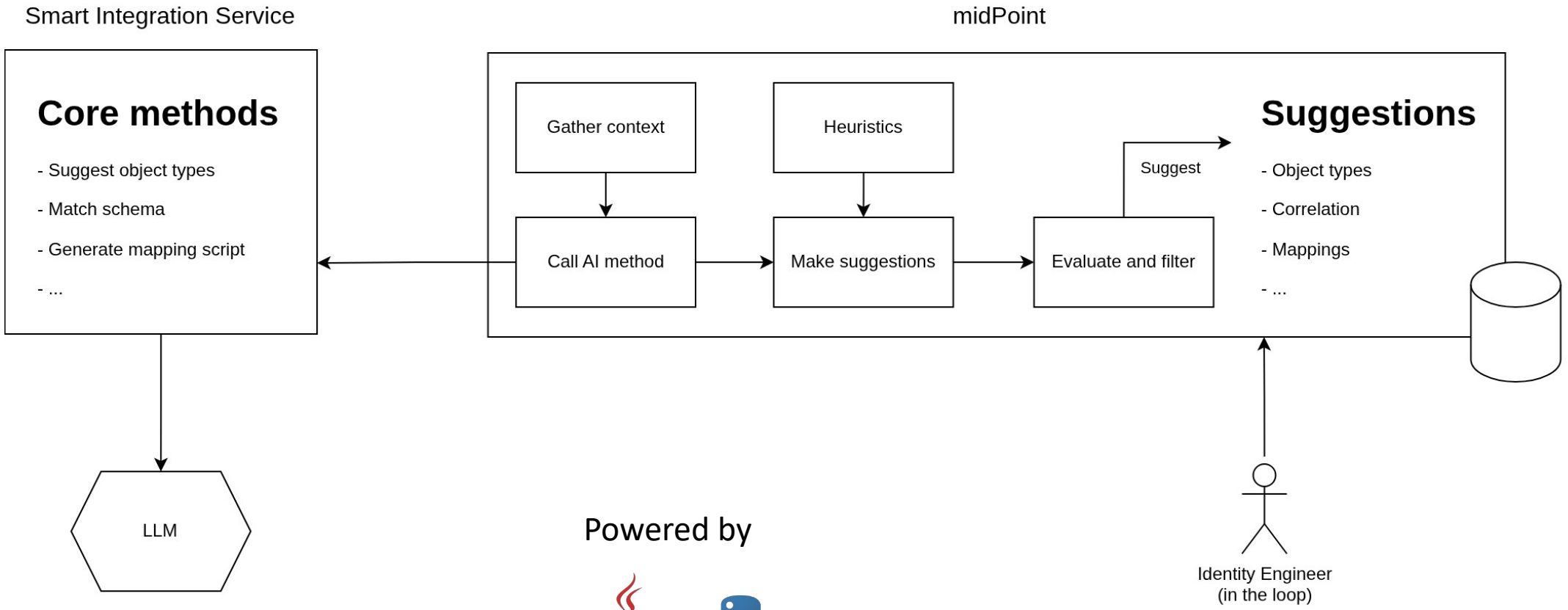
Assisted classification, correlation and mappings

Assisted classification, correlation and mappings



AI + Heuristics + UX

Assisted classification, correlation and mappings



Assisted classification, correlation and mappings

Source code

- https://github.com/Evolveum/midpoint
- https://github.com/Evolveum/midpilot-smart-integration

Large Language Models (LLMs)

Large Language Models (LLMs)

Many possibilities these days

- GPT-5, Gemini, Claude Sonnet, Mistral, Deepseek, Qwen, ...

Different criteria

- Benchmarks
- Price
- Open vs. closed models
- On premise vs. as a service
- Security
- Legislative and geo-political considerations, ...

Large Language Models (LLMs)

Our criteria: OSS model with good reputation and reasonable size

gpt-oss models family

- Open weight models by OpenAI
  - gpt-oss-120b
  - gpt-oss-20b for smaller deployments
- Those relatively small models hit almost SOTA quality
- Evaluated and tested by Evolveum

AI Observability

AI Observability

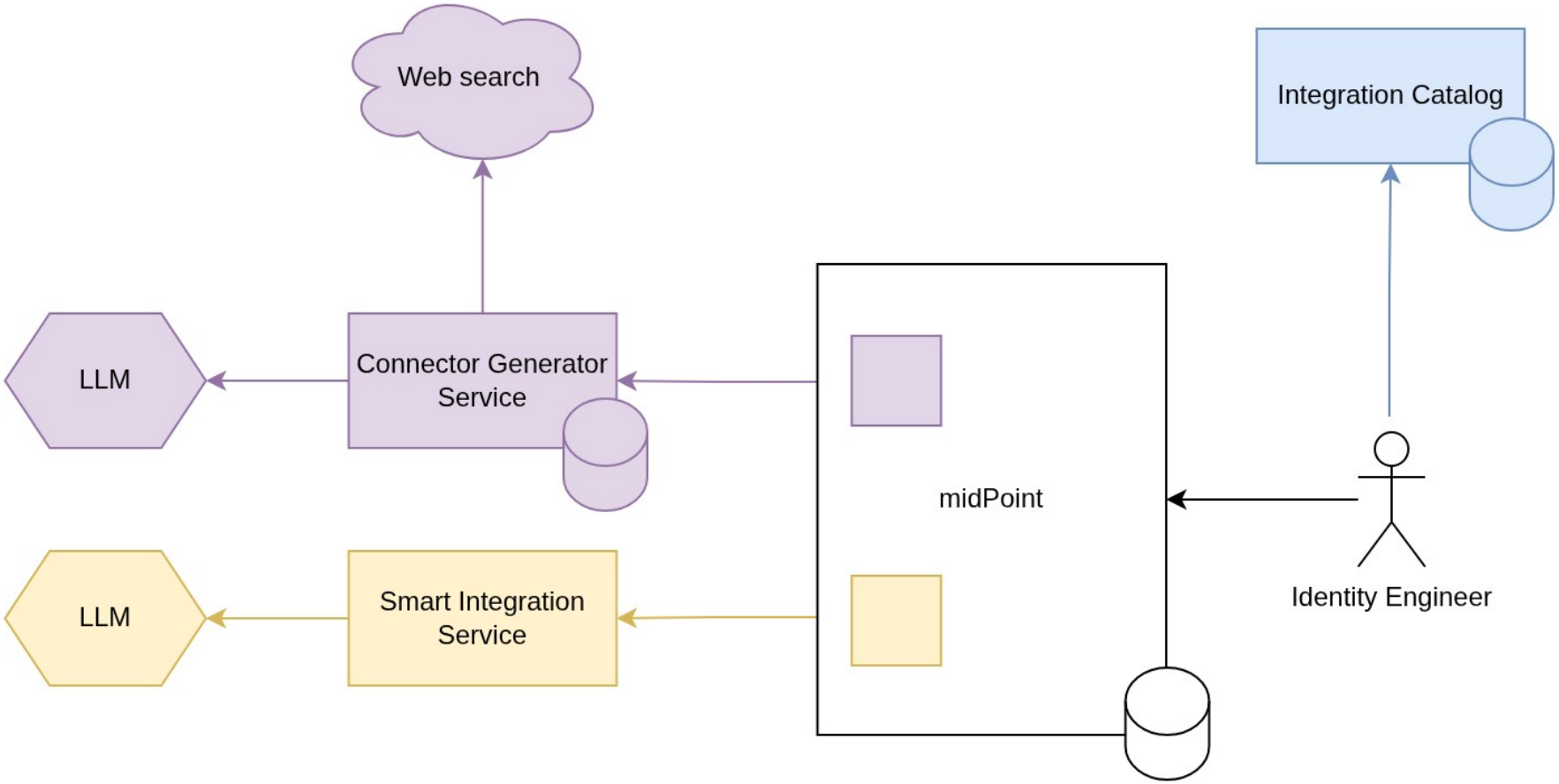
Visibility to the AI application is important

- During development for tuning and debugging
- Monitoring existing application

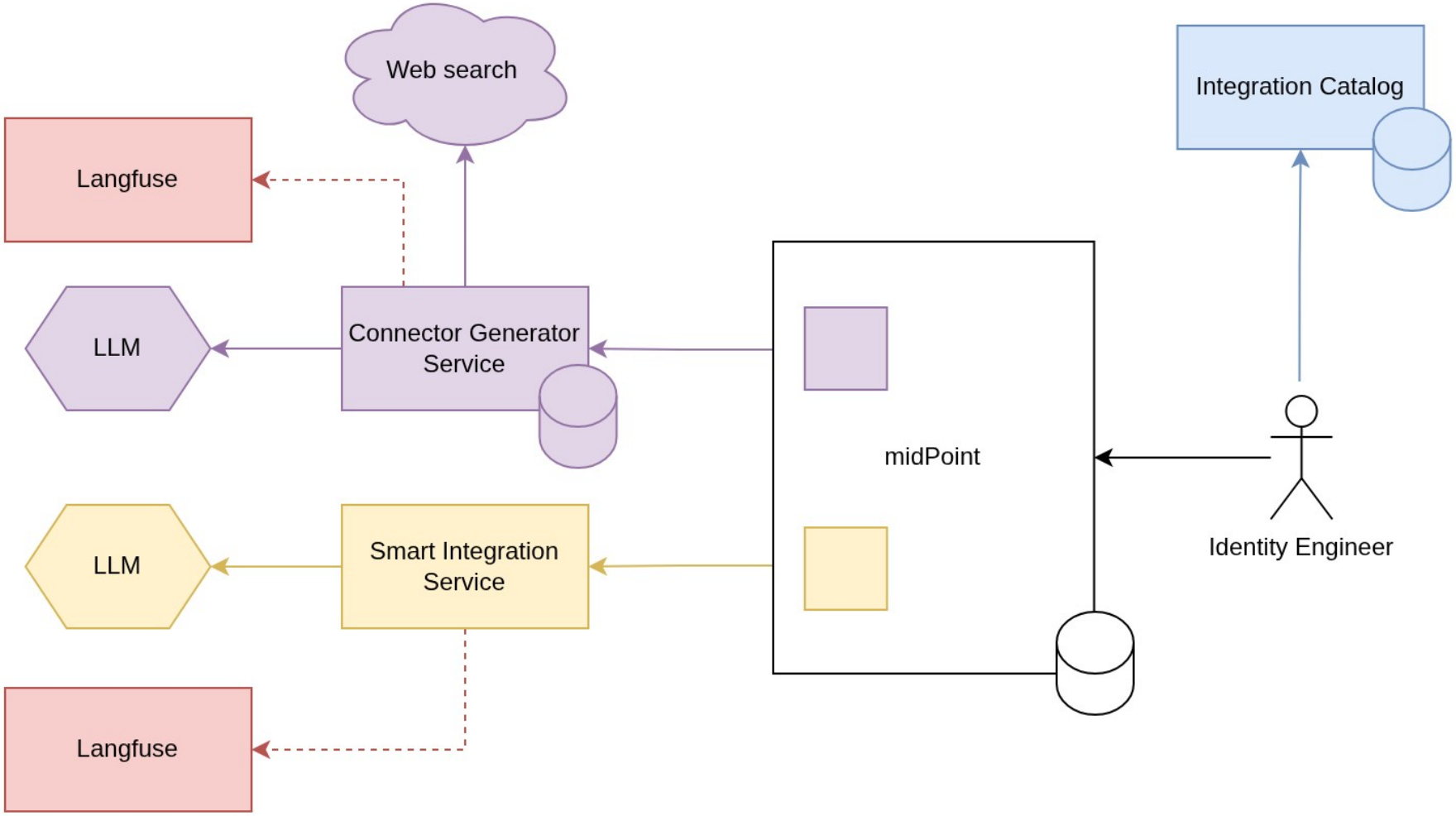
Langfuse

- Open-source LLM engineering platform
- Features
  - **LLM Application Observability**
  - Prompt Management
  - Evaluations
  - …

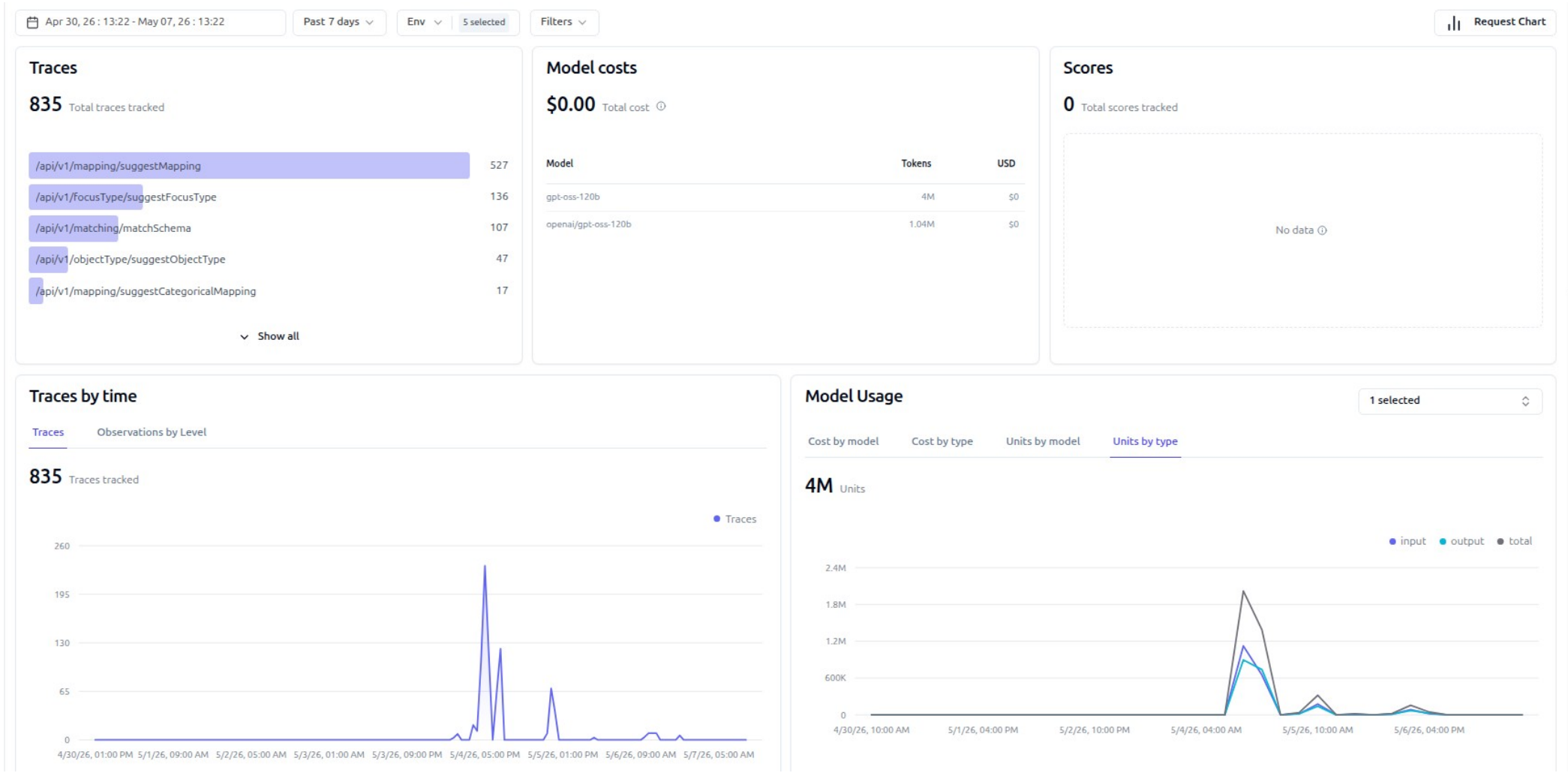
AI Observability



AI Observability



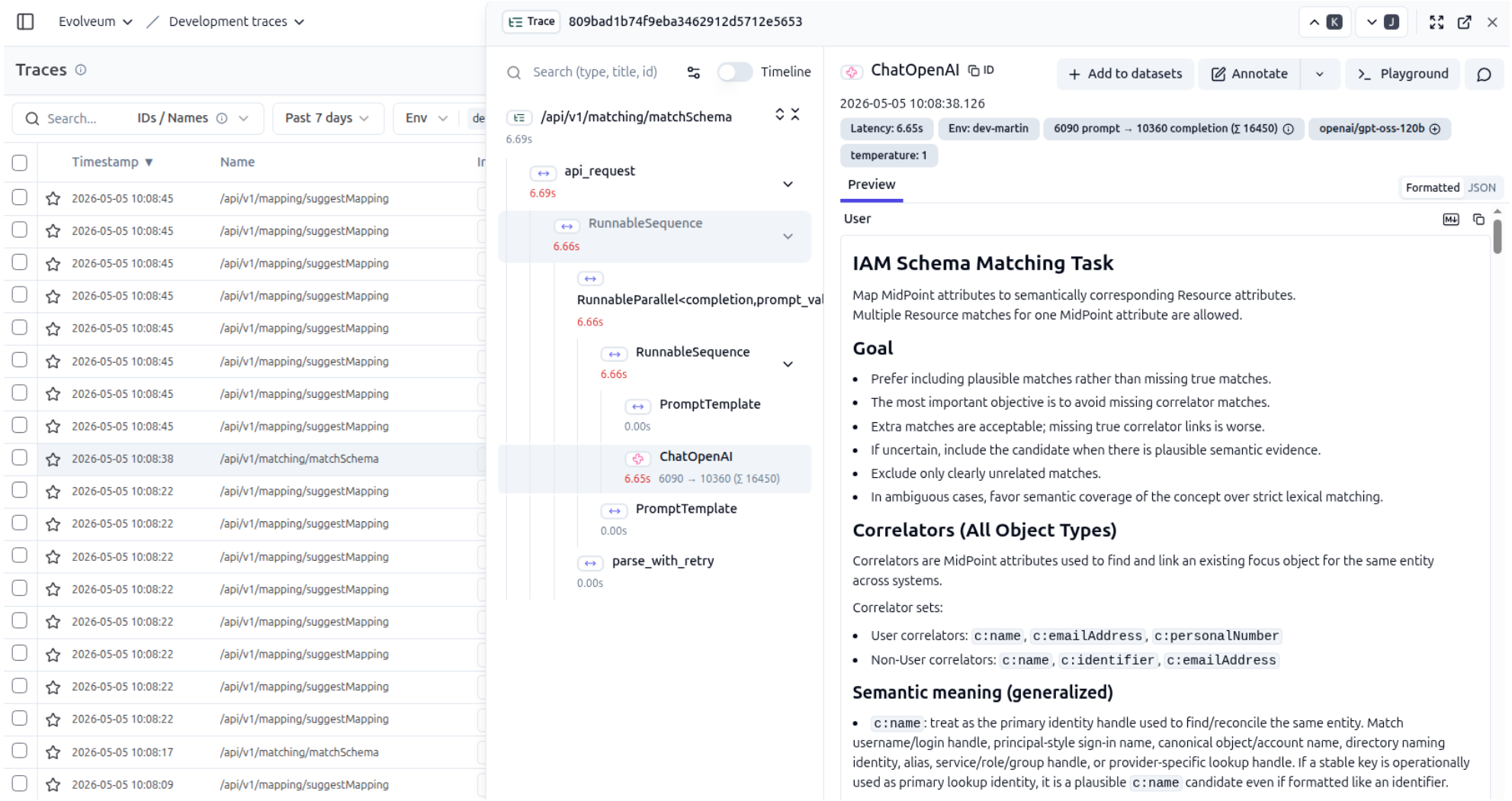
AI Observability



Evolveum

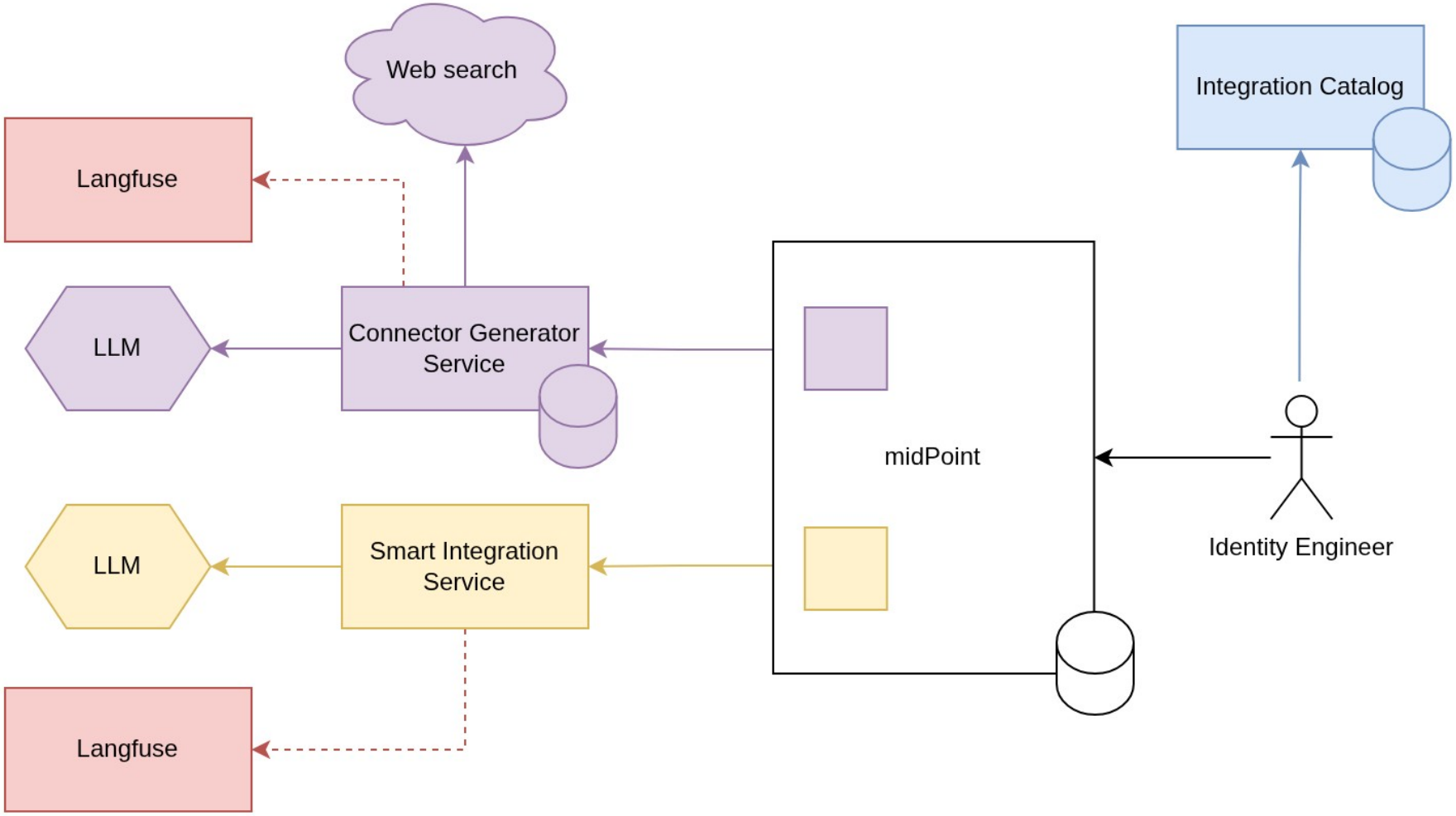
2nd Annual
MidPoint Community Meetup

AI Observability



MidPilot Deployment

MidPilot Deployment



MidPilot Deployment

```
LLM__OPENAI_API_BASE=https://api.openai.com/v1
LLM__OPENAI_API_KEY=my_api_key
LLM__MODEL_NAME=openai/gpt-5.5
```

OpenAI

```
LLM__OPENAI_API_BASE=https://bedrock-runtime.us-west-2.amazonaws.com/openai/v1
LLM__OPENAI_API_KEY=my_api_key
LLM__MODEL_NAME=openai.gpt-oss-20b-1:0
```

aws

```
LLM__OPENAI_API_BASE=http://localhost:11434/v1
LLM__OPENAI_API_KEY=ollama
LLM__MODEL_NAME=mistral-small3.2
```

MidPilot Deployment

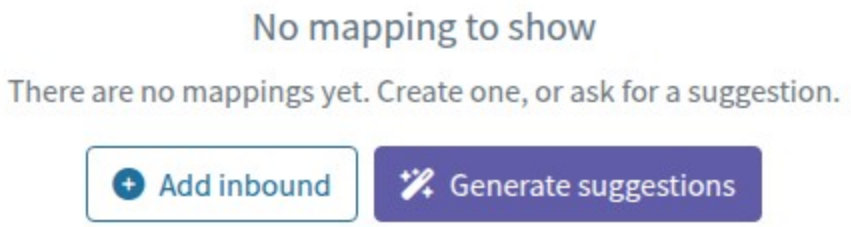
- Self-hosted services
  - Self-hosted LLM models
  - External LLM providers and models
- Evolveum hosted services and LLM models

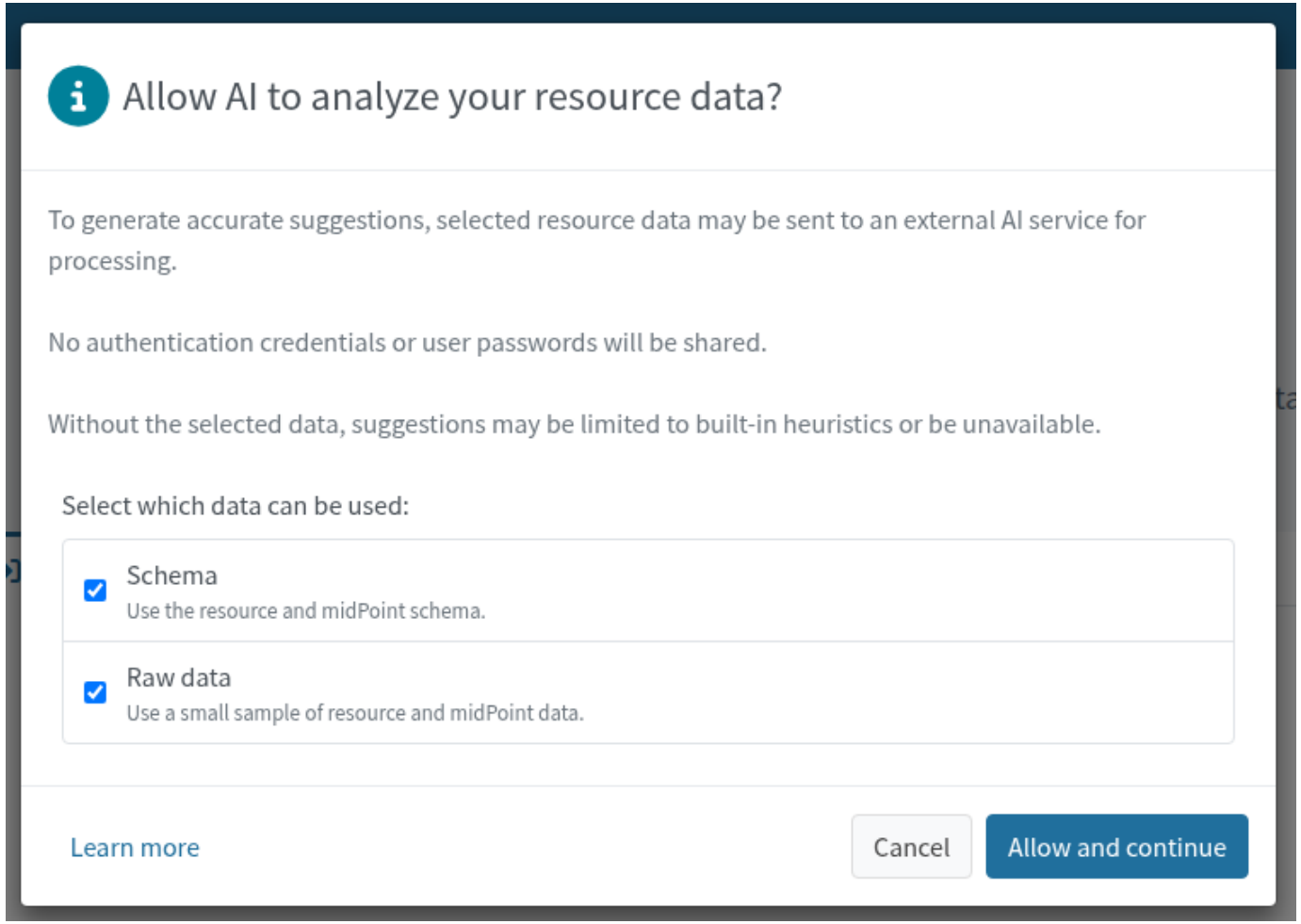
Notes on using AI in midPoint

Notes on using AI in midPoint

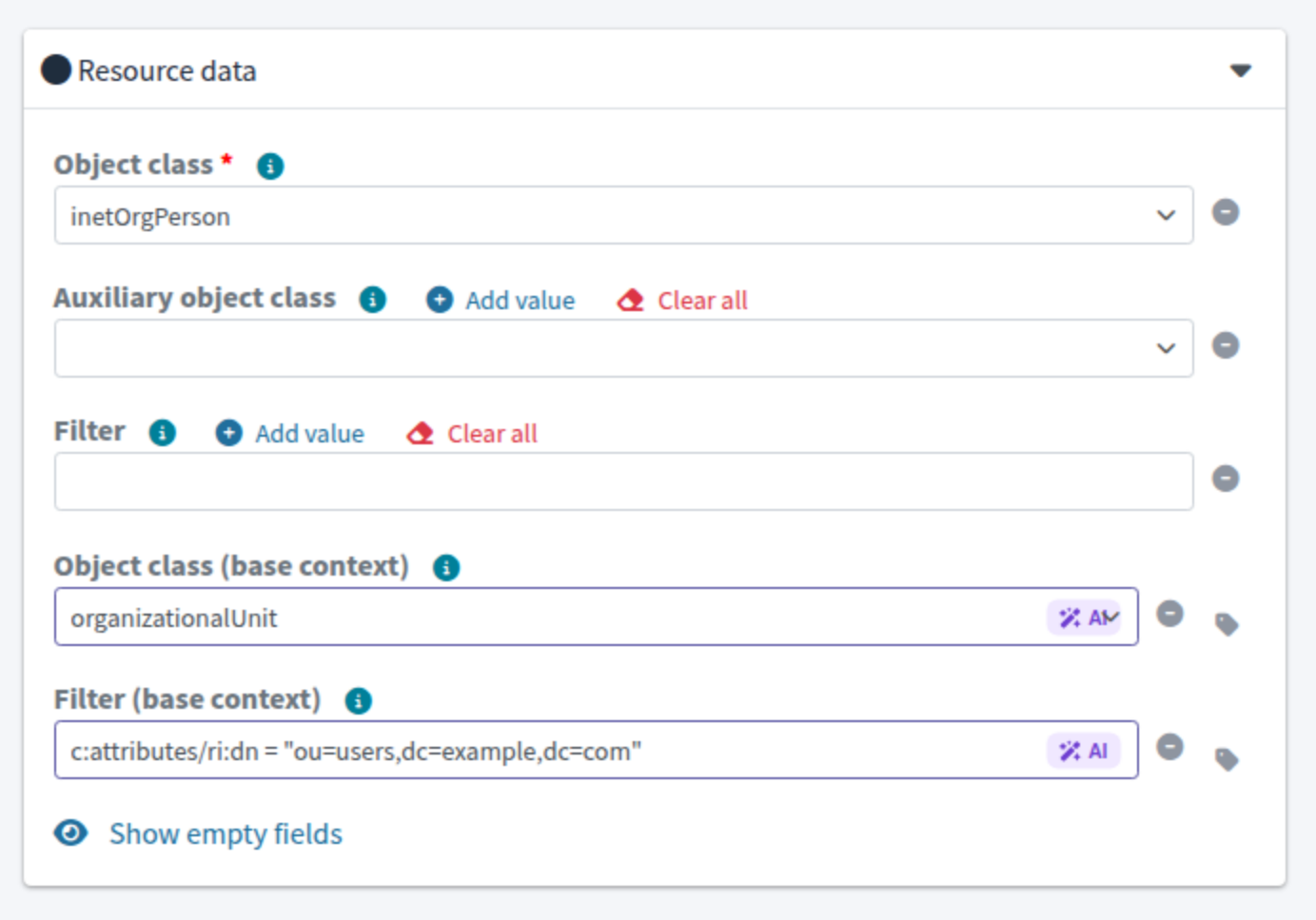
- AI features in midPoint are optional
- Always know when working with AI
- Always in control of your data

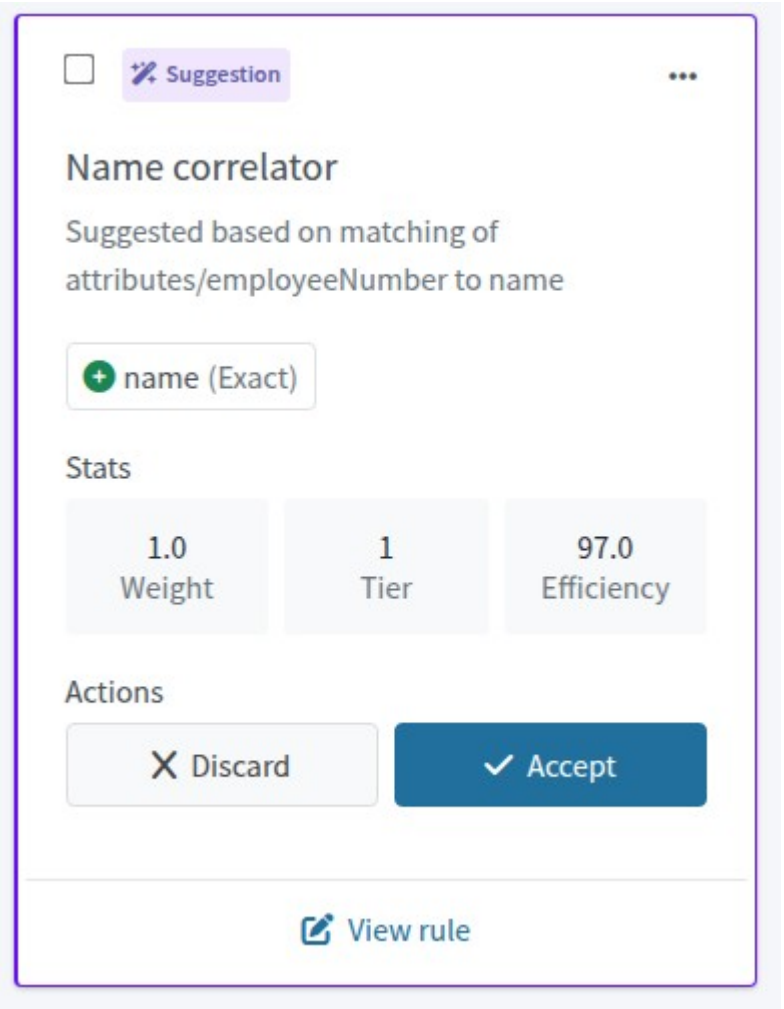
Notes on using AI in midPoint





Notes on using AI in midPoint





AI-powered connector development

Insights & Lessons learned

MidPoint Connector Generator: Agenda

- Expectation vs. reality
- Where AI brings value
- Where AI fails
- What worked in practice
- Lessons learned and bottlenecks

Why lessons learned matter

- Proofs of concept are easy; reliable production workflows are much harder
- In identity systems, a confident wrong answer can create security and operational risk
- The key question is whether AI output can be trusted, validated, and operated safely

Expectations vs. reality

Initial assumption	Practical lesson
One generic prompt can handle any vendor	Small, task-specific prompts produce more reliable outputs
AI can fully automate identity mapping	AI provides a fast first draft; engineers remain accountable
A larger model will understand every document	Context must be selected, structured, and filtered first
Fluent output is probably correct	Validation, dry-runs, and expert review are required

Where AI adds value

1. Discovery
Summarize large documentation sets and identify likely entities, operations, and endpoints.

2. Drafting
Prepare first-pass schema matches, mappings, connector logic, and test cases.

3. Review
Explain unfamiliar code, surface assumptions, and suggest the next engineering question.

4. Assistance
Support documentation, internal knowledge sharing, and repeated explanations.

Concrete delivery value

Large documentation	→	prioritized endpoints, entities, and gaps for review
Empty mapping	→	initial schema matches and connector scaffolding
New integration work	→	draft documentation, comments, and repetitive test cases

Where AI creates risk

Hallucinated fields

The model may invent attributes, endpoints, or operations that do not exist in the target system.

Incomplete logic

Generated code can cover the happy path while missing exceptions, permissions, or lifecycle details.

Vendor-specific details

API behavior, pagination, filtering, and error handling often differ from generic examples.

Confident wording

A polished explanation can hide uncertainty. The output still needs evidence and validation.

PRACTICAL RULE: VALIDATE EVERY GENERATED ATTRIBUTE, OPERATION, AND MAPPING AGAINST REAL SCHEMA AND RUNTIME BEHAVIOR.

Operational risks

1. Non-determinism

Similar prompts can return materially different outputs. This complicates versioning, CI checks, and approval workflows.

2. Retry cost

Long context windows and repeated attempts can turn simple experiments into real production cost.

3. Latency

Unpredictable response times make synchronous designs fragile. Many workflows need asynchronous processing.

What worked in practice

1. Rules for deterministic work

Use deterministic rules for normalization, fixed mappings, naming patterns, and simple transforms.

2. Validation before trust

Validate generated outputs with schema checks, linting, tests, preview mode, and dry-runs.

3. Human approval

Keep expert approval for security-sensitive mappings, business logic, exceptions, and user-facing changes.

Production-ready AI is hybrid, observable, and intentionally narrow.

Defensive engineering patterns

- **Use heuristics first**
  - Use normal code for deterministic cleanup whenever it is safer and more predictable
- **Validate strictly**
  - Treat model output as untrusted until validation, linting, and dry-runs pass
- **Provide coded fallbacks**
  - Keep non-LLM fallbacks for timeouts, common defaults, and stable operations
- **Preview before commit**
  - Use preview mode so AI output can be inspected before it changes groovy code

Guardrails are part of the product; the model is only one component.

Prompting and review patterns

Prefer narrow prompts

Split large requests into small, operation-specific prompts with explicit output formats.

Allow refusal

Prefer “I do not know” over guessing. Reject unknown fields and uncertain mappings.

Keep human review

Require engineering approval for sensitive mappings, exceptions, and final commits.

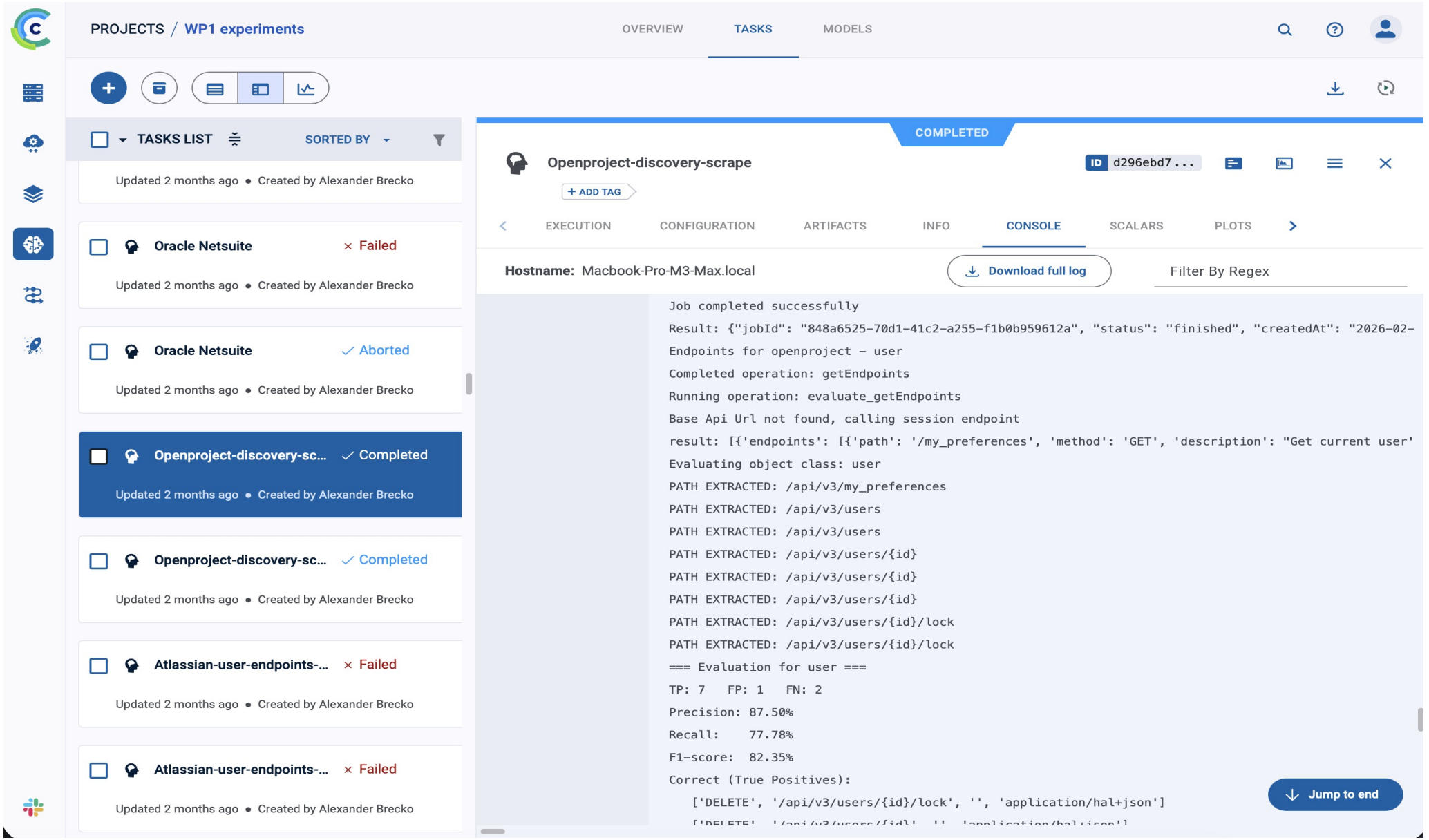
Make outputs observable

The team must be able to log, reproduce, inspect, and test each output.

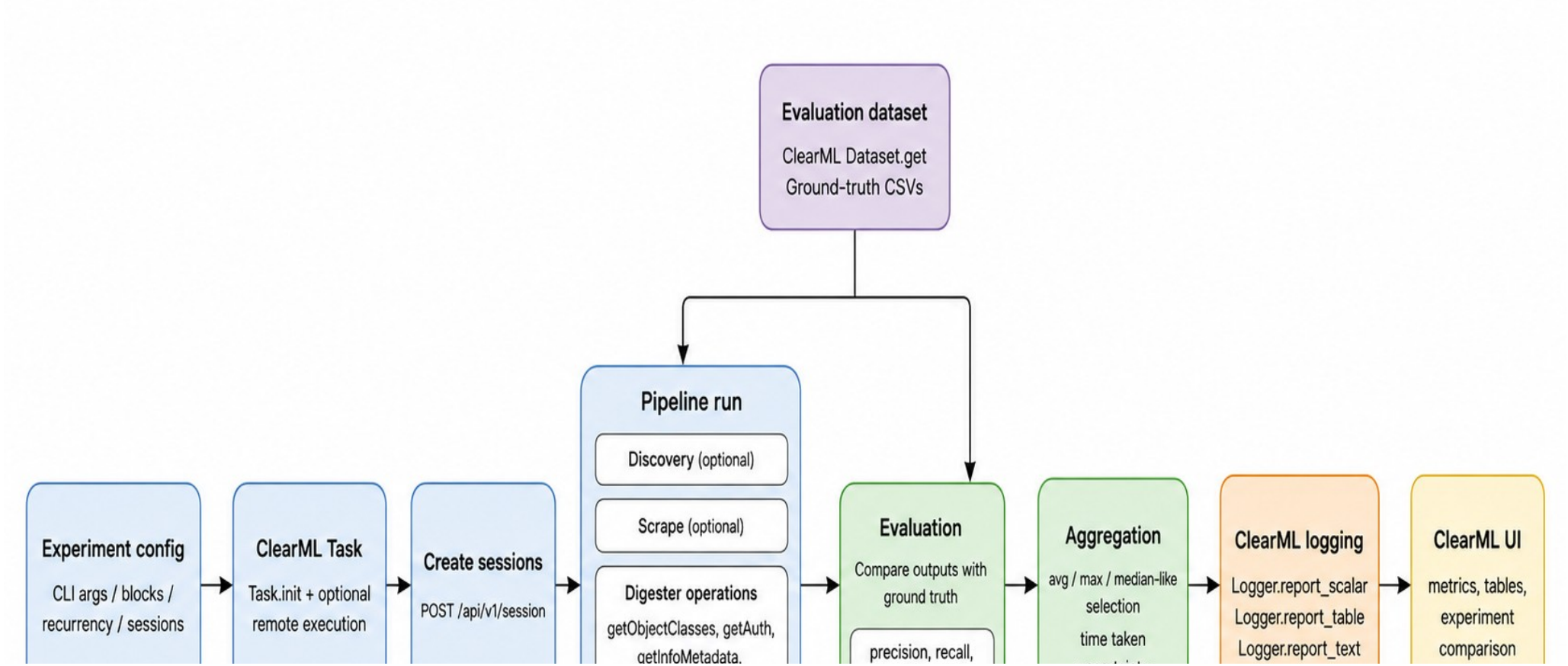
Reliable AI systems are inspectable, reproducible, and reviewable.

ClearML: Making AI Experiments Measurable

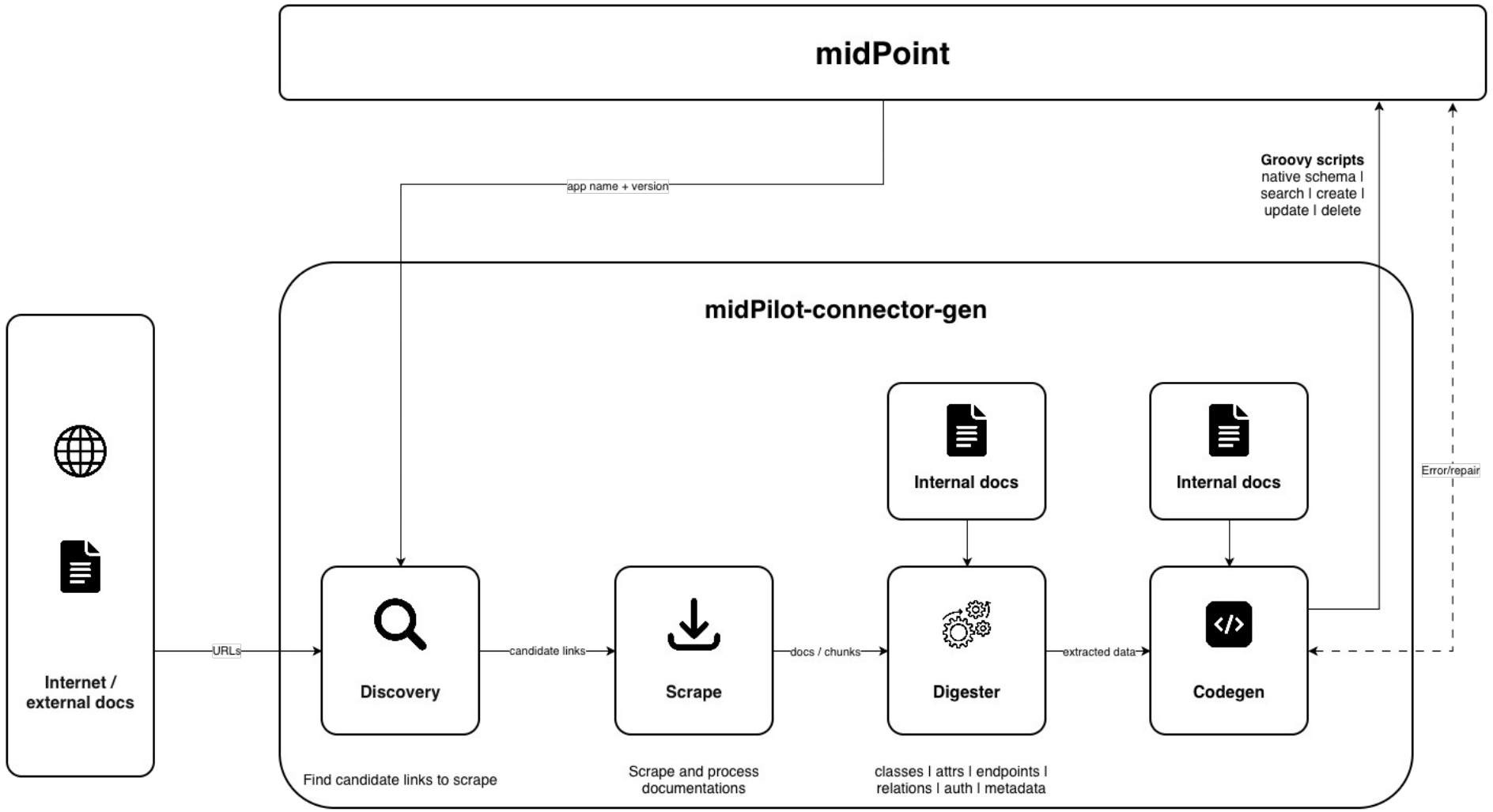
- To use ClearML for repeatable experiment evaluation
- To track selected operations, prompt/model variants, processing time, and evaluation metrics
- To compare outputs against prepared ground-truth datasets
- To measure precision, recall, F1, false positives, false negatives, and result uncertainty
- To replace subjective judgments like “this prompt looks better” with evidence-based decisions.


PROJECTS / WP1 experiments
OVERVIEW
TASKS
MODELS
TASKS LIST
SORTED BY
Updated 2 months ago • Created by Alexander Brecko
Oracle Netsuite
× Failed
Updated 2 months ago • Created by Alexander Brecko
Oracle Netsuite
✓ Aborted
Updated 2 months ago • Created by Alexander Brecko
Openproject-discovery-sc...
✓ Completed
Updated 2 months ago • Created by Alexander Brecko
Openproject-discovery-sc...
✓ Completed
Updated 2 months ago • Created by Alexander Brecko
Atlassian-user-endpoints-...
× Failed
Updated 2 months ago • Created by Alexander Brecko
Atlassian-user-endpoints-...
× Failed
Updated 2 months ago • Created by Alexander Brecko
COMPLETED
Openproject-discovery-scrape
ID d296ebd7...
+ ADD TAG
EXECUTION
CONFIGURATION
ARTIFACTS
INFO
CONSOLE
SCALARS
PLOTS
Hostname: Macbook-Pro-M3-Max.local
Download full log
Filter By Regex
Job completed successfully
Result: {"jobId": "848a6525-70d1-41c2-a255-f1b0b959612a", "status": "finished", "createdAt": "2026-02-
Endpoints for openproject - user
Completed operation: getEndpoints
Running operation: evaluate_getEndpoints
Base Api Url not found, calling session endpoint
result: [{'endpoints': [{'path': '/my_preferences', 'method': 'GET', 'description': "Get current user'
Evaluating object class: user
PATH EXTRACTED: /api/v3/my_preferences
PATH EXTRACTED: /api/v3/users
PATH EXTRACTED: /api/v3/users
PATH EXTRACTED: /api/v3/users/{id}
PATH EXTRACTED: /api/v3/users/{id}
PATH EXTRACTED: /api/v3/users/{id}
PATH EXTRACTED: /api/v3/users/{id}/lock
PATH EXTRACTED: /api/v3/users/{id}/lock
=== Evaluation for user ===
TP: 7 FP: 1 FN: 2
Precision: 87.50%
Recall: 77.78%
F1-score: 82.35%
Correct (True Positives):
['DELETE', '/api/v3/users/{id}/lock', '', 'application/hal+json']
Jump to end

ClearML Pipeline



MidPilot-connector-gen pipeline



Precision over recall

Typical LLM behavior

High recall: the model suggests many plausible options, including some that may be fabricated.

→

Identity requirement

High precision: only verified attributes and operations are acceptable. Guessing is unacceptable.

Engineering response

Design prompts and validation to prefer “I do not know” over confident false fields.

Key lessons 1–5

1	**Use AI for drafts, not decisions**	AI can prepare the first draft, but engineers remain accountable for the final result.
2	**Fluency is not accuracy**	Trust validation, linting, and tests rather than polished wording.
3	**Avoid the magic prompt**	Chained, narrow prompts are more reliable than one broad prompt.
4	**Use heuristics where possible**	Predictable tasks should remain in deterministic code whenever possible.
5	**Limit the context**	Filter and digest context before it reaches the model.

Key lessons 6–10

6	**Build pipelines, not isolated calls**	The workflow around the model matters more than any single API call.
7	**Prioritize precision over recall**	Require the model to reject guesses when the domain requires certainty.
8	**Monitor telemetry**	Latency, retries, and API spend can become architecture issues quickly.
9	**Design for non-determinism**	CI and review workflows must be designed for output variance.
10	**Keep human review**	Human review is often the final security layer and should be designed explicitly.

Practical outcome

- Faster discovery, drafting, and repetitive engineering work
- Senior engineers spend more time on review, exceptior and decisions
- Identity-critical delivery still requires expert validation a approval
- The bottleneck remains context quality, validation, and accountable handoff

AI accelerates engineering work, but it does not replace engineering accountability.

Assisted classification, correlation and mappings

Insights & Lessons learned

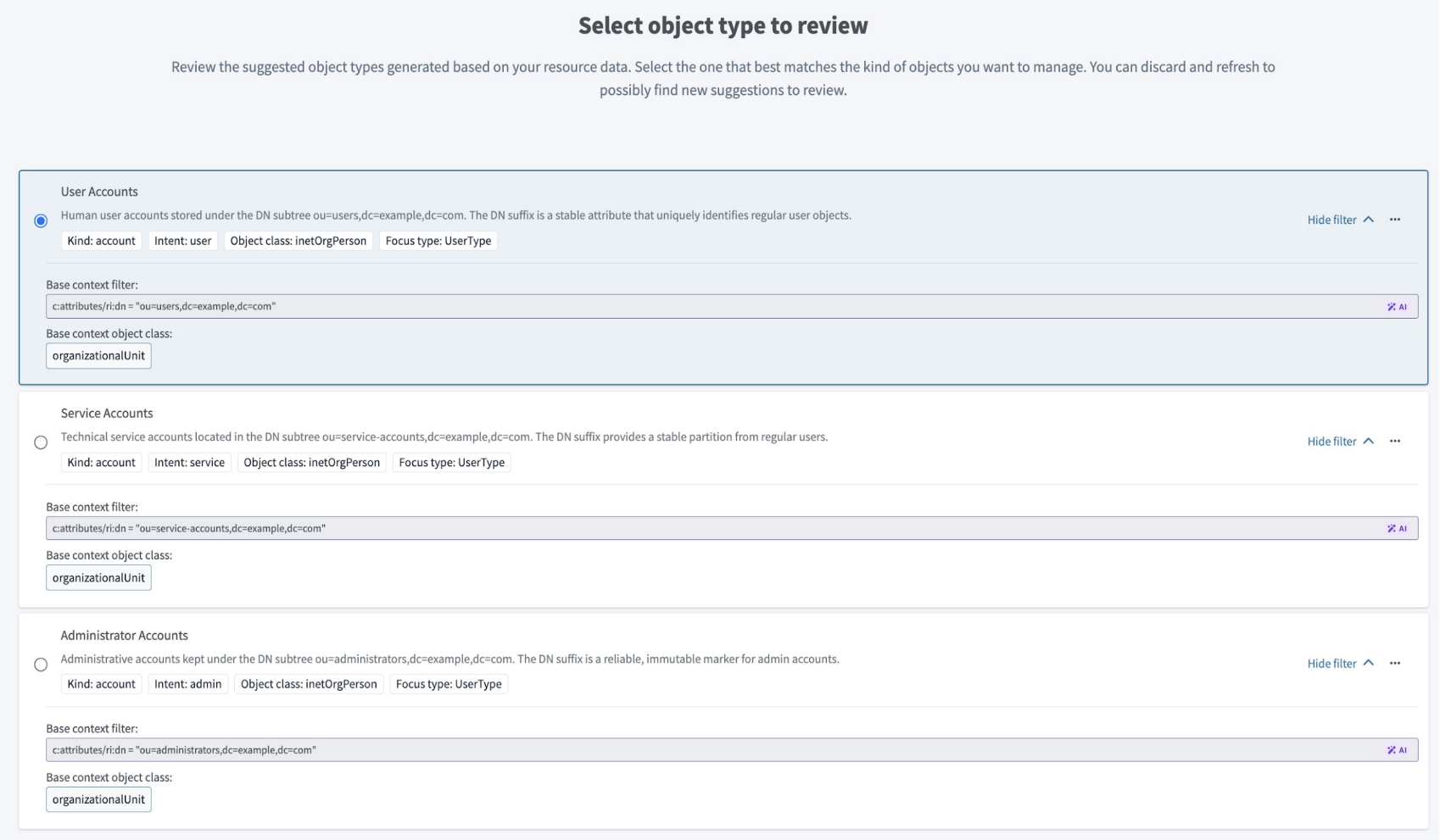
MidPoint Smart Integration: Agenda

- Object type suggestion
- Schema matching
- Correlation suggestion
- Schema mapping

Object Type Suggestions

Delineations

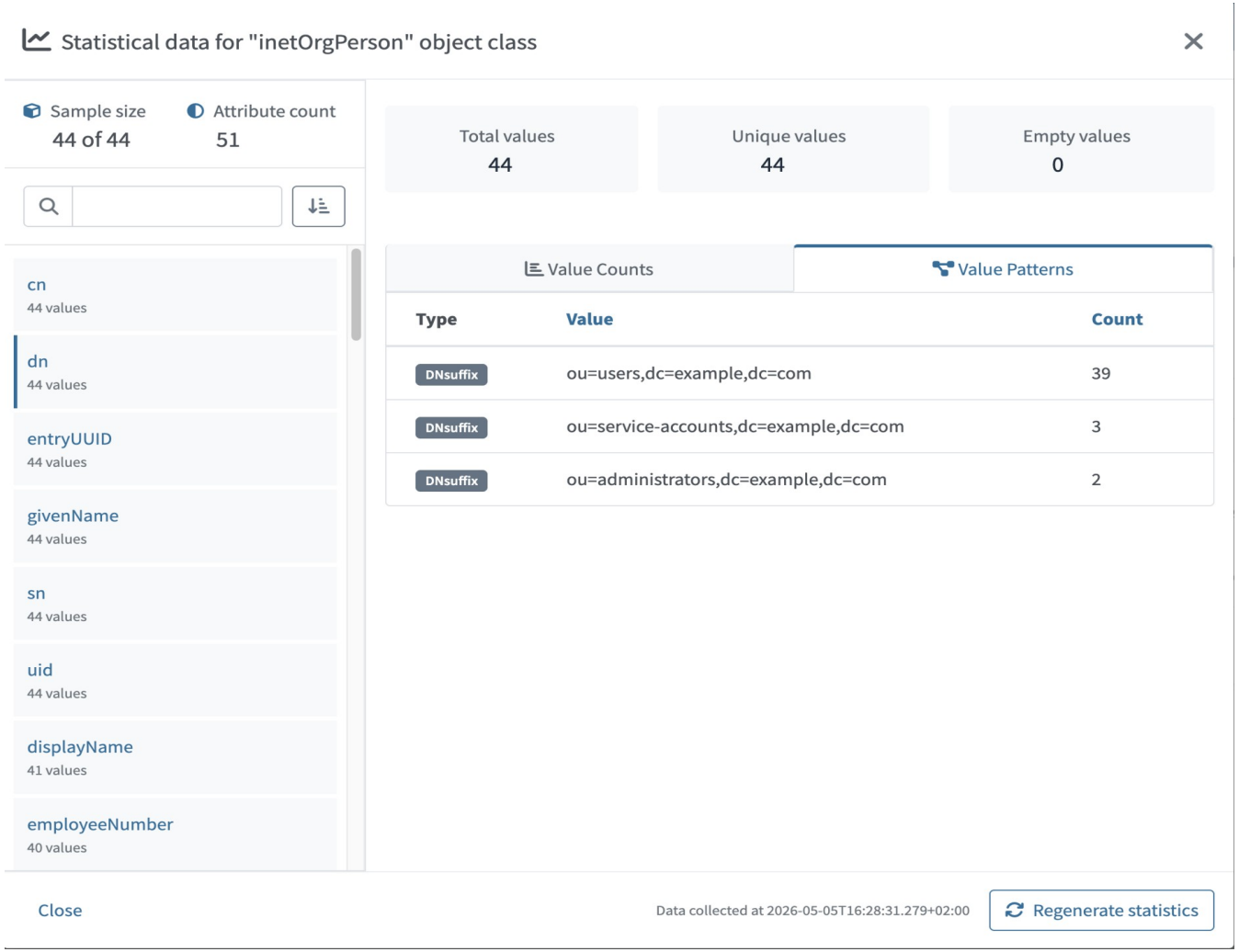
Delineations: Defining Object Type Boundaries



Delineations are used to split objects from a resource into **logical groups**, so MidPoint knows how each object type should be classified, synchronized, and processed.

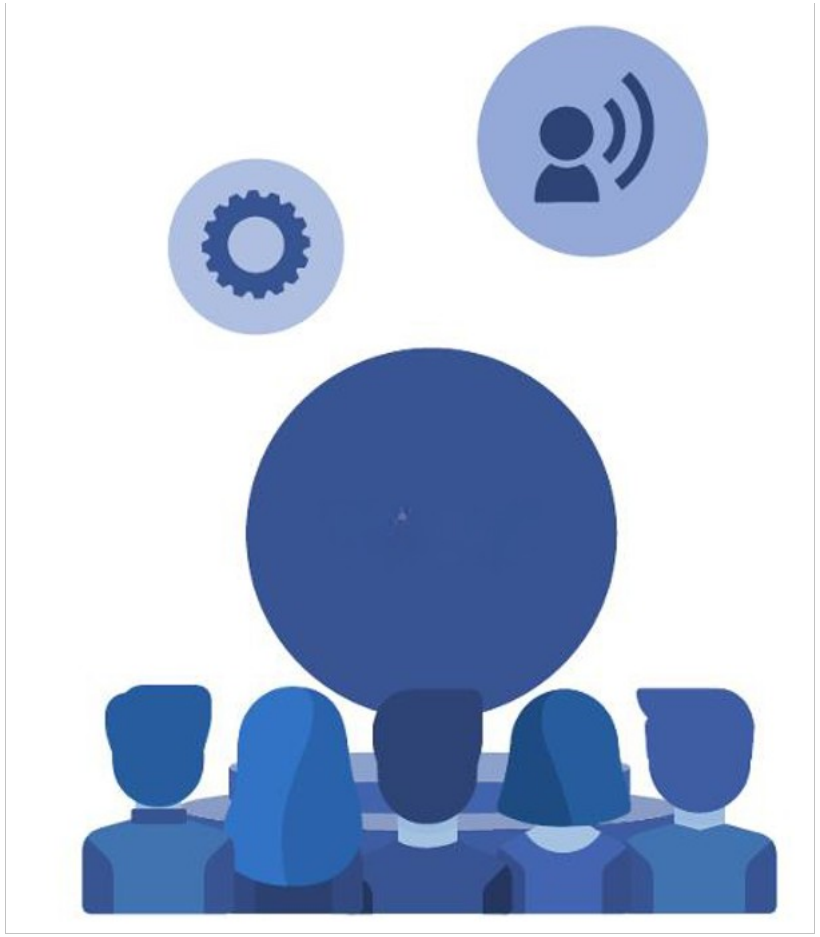
How We Use AI for Delineations

We mirrored domain experts' reasoning into measurable statistics and provided those signals to the LLM with explicit decision logic in the prompt.



Lessons Learned

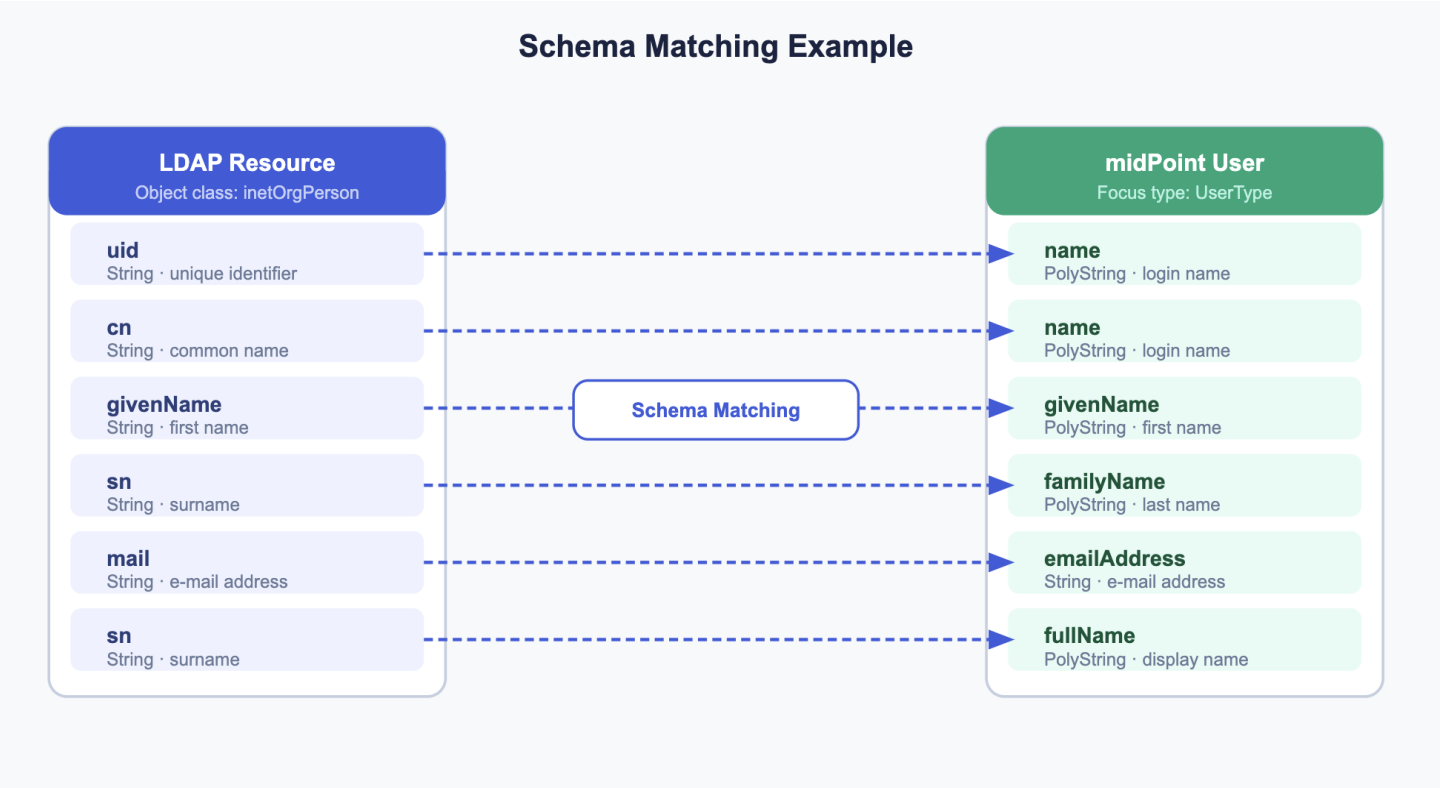
- **Domain experts** became an inseparable part of the system design
- **Human-in-the-loop** is needed also during evaluation
- The LLM worked well as a **junior domain assistant**
- **More** context **did not** mean better answers
- The biggest gains came from **prepared inputs**



Schema Matching

The Hidden Helper

Schema Matching: The Hidden Helper



Schema matching helps us understand **which attributes** from an external resource **correspond to attributes** in MidPoint.

user_first_nm, **givenname**, **fname**, **firstName**, and so on.

Schema Matching: From Heuristics to AI Suggestions

- Heuristics run first
- LLM receives only structured schemas
- LLM is asked to find semantic matches
- MidPoint merges both results
- Hallucinated matches are removed

Schema Matching: Results

MODEL	PRECISION	RECALL	F1	HALLUCINATIONS
openai/gpt-oss-20b	0.510	0.822	0.600	6
openai/gpt-oss-120b	0.441	0.868	0.563	3
google/gemini-3-flash-preview	0.410	0.920	0.543	7
nvidia/nemotron-3-nano-30b-a3b	0.516	0.598	0.528	66
openai/gpt-5.4	0.310	0.962	0.456	0

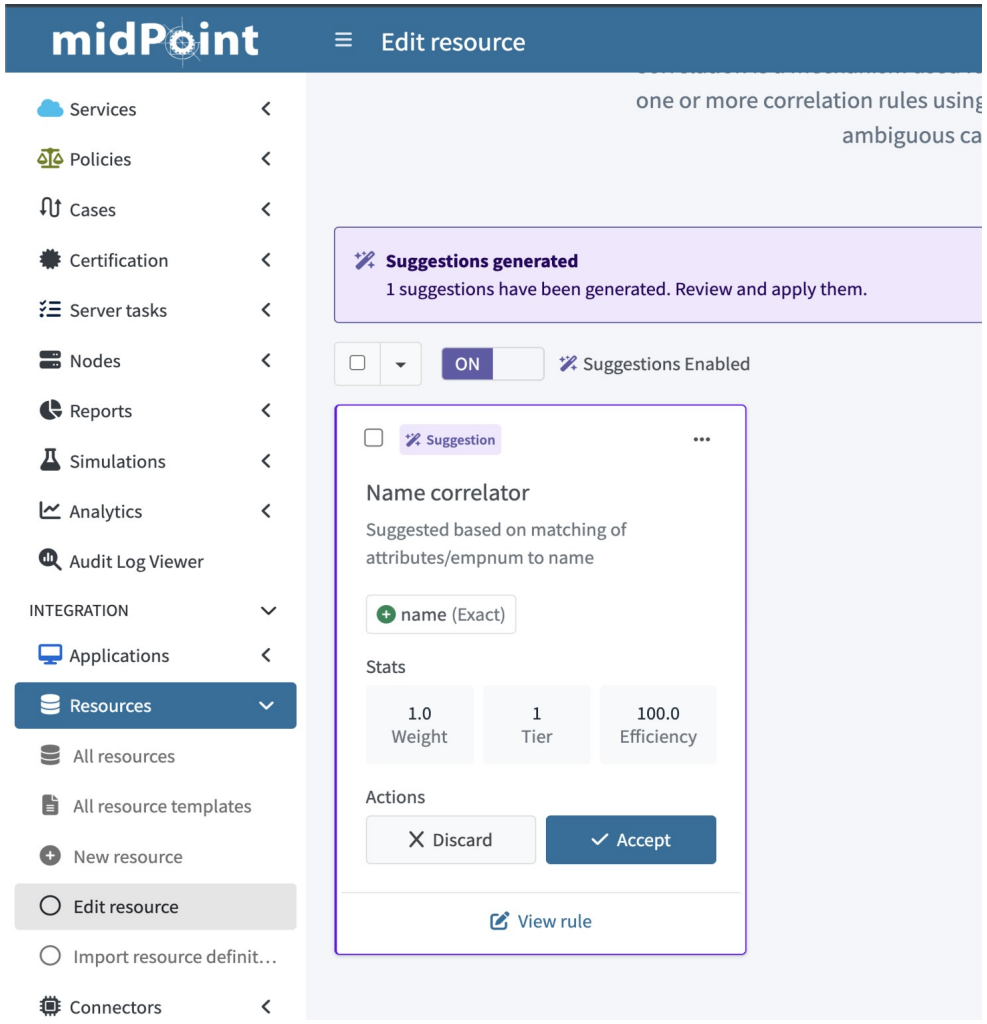
Suggest Correlation

From Schema Match to Identity Match

Correlation: From Schema Match to Identity Match

Once **schema matching** tells us which attributes correspond to MidPoint properties, **we score predefined candidates and select the best one.**

Predefined correlator candidates:
User: name, personalNumber, emailAddress
Non-user: name, identifier, emailAddress



Schema Mapping

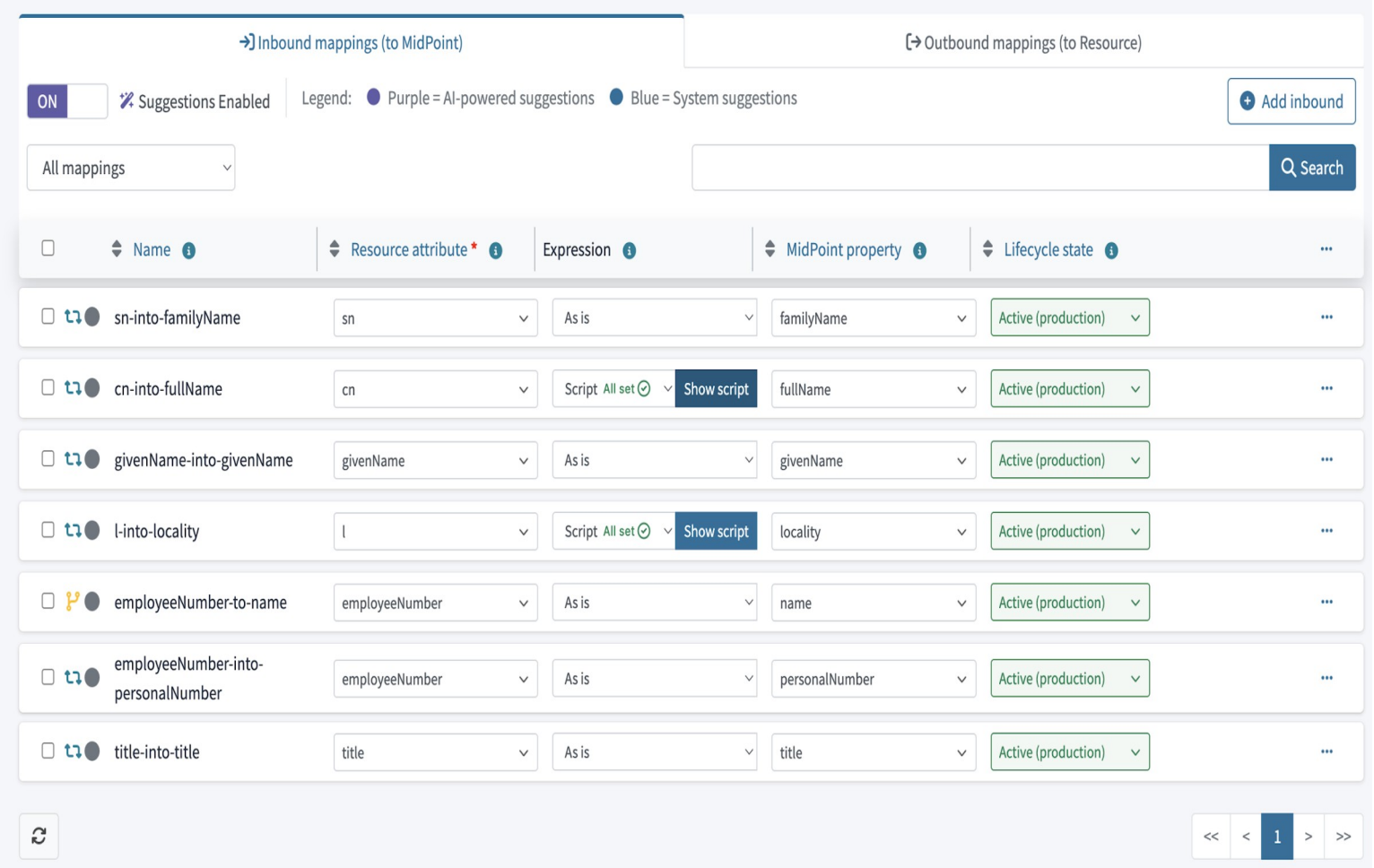
How Data Moves Between Systems

From Schema Matching to Schema Mapping

A **mapping** defines **how** a value is **copied**, **transformed**, or **generated** between a resource object and a midPoint object.

Simple idea:
Schema **matching** tells us what belongs together. **Mapping** tells us how the value should move or change.

Mappings: How Data Moves Between Systems



- The resource stores names in **uppercase** (JOHN DOE), **M**idPoint expects **title case**
- A date comes in as **20250101**, midPoint expects **ISO format 2025-01-01**
- An HR system uses **internal codes** (EMP, CONT, SVC) while **M**idPoint's **lifecycleState** has its own vocabulary (active, archived, …)

Mapping Suggestions: Layered Decision Pipeline

- System mappings first
- As-is evaluation
- Heuristic rules
- LLM only when needed
- Categorical mappings
- Parallel processing

Lessons Learned & Wrap-Up

Lessons Learned & Wrap-Up

- **Prepared inputs** matter more than model choice
- **Prompt engineering** is a first-class engineering discipline
- **Thoughtful UX** is what makes AI integration actually usable
- **Validate** before trusting
- **Human-in-the-loop** remains essential
- **Score** your outputs – and show those scores to the user
- **Model choice is always a time/quality/stability trade-off** – and the parameters matter...


Evolveum
Thank you for your attention
Feel free to ask your questions now!
Funded by the European Union
NextGenerationEU
RECOVERY AND RESILIENCE PLAN
2nd Annual
MidPoint Community Meetup