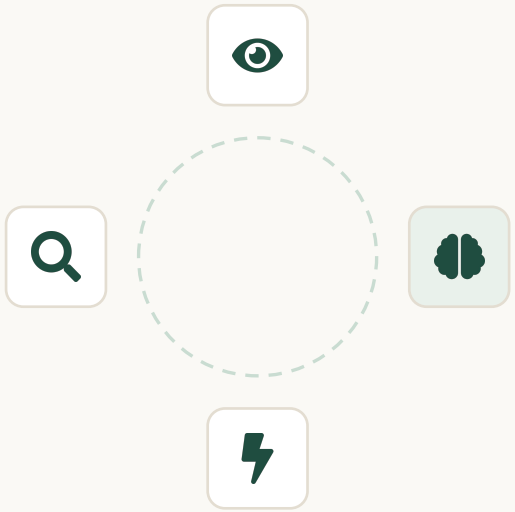
INDUSTRIAL AI — WEEK 1

Agent Engineering, done properly.



The industry moved to agents. This course starts where the industry is — and holds itself to the standard of the people who wrote the playbook: Anthropic and LangChain.

Lecture 1 of 24 · Industrial AI · August 2026 · Zoom + Arfa Tower

WHY YOU ARE HERE

Three things we told you in the webinar

You registered because of these three claims. Tonight — and this whole course — is built to make them true for you.

“Your first job has been taken by agents. Now you have to go above them.”

Entry-level intake collapsed. The way up is to be the person who BUILDS and runs the agents.

“Verification is the biggest bottleneck in AI.”

Anyone can wire an agent. Engineers prove it works. Evaluation is tonight’s centerpiece — and the course’s.

“LLM-assembled code survives the prototype — and falls apart in production.”

The gap between demo and production is engineering discipline. That discipline has a syllabus. This is it.

— said on stage at our own launch webinar, August 2026

OUR STANDARD

The bar for this course is the industry’s bar

Not a YouTube syllabus. The same sources, tools, and testing style the leading AI companies use.

The canon we read

Anthropic’s engineering posts and LangChain’s field reports — primary sources, quoted and cited on the slides, never second-hand summaries.

The stack we use

LangSmith, LangGraph, Claude & GPT APIs, vector databases — the same tools the 57%-in-production teams run every day.

The way we test

Certification-style checkpoints: a scenario, a vote, then the dissection — why the right answer is right AND why each wrong answer is a trap.

What “industry-ready” means here: by Week 24 you defend a working, evaluated, deployed system in front of reviewers — the certificate confirms it; the portfolio proves it.

Two numbers that are both true

THE GRAVEYARD

95%

of enterprise GenAI pilots deliver zero measurable ROI

MIT — State of AI in Business

why · no learning from feedback · no workflow fit · nobody measures

THE PAYOFF

$60M

Klarna's assistant profit impact · 2.3M chats/month · the work of ~850 people

Klarna, company reports

why · narrow scope · hard guardrails · instant human handoff · measured weekly

Same technology. The difference is engineering discipline — and discipline can be taught.

Meanwhile, the industry has already moved

LangChain surveyed 1,340 professionals in late 2025 — the freshest picture of agent engineering we have.

57%

of organizations already have agents in production

67%

among large enterprises (10,000+ employees)

89%

have implemented observability for their agents

33%

name QUALITY as the #1 barrier — ahead of cost and latency

Read those together: the jobs are in production agents, and the skill that separates candidates is not “can you build an agent” — it is “can you observe, evaluate, and improve one.” That is exactly where this course begins.

Source: LangChain, State of Agent Engineering 2026 (n = 1,340)

A CHANGE YOU DESERVE TO HEAR EXPLAINED

Same syllabus. New order. Here’s the rationale.

Every topic on the published outline is still here — we changed the sequence, not the promise.

What the outline promised

- PyTorch · Hugging Face · LLMs & RAG
- Computer vision · diffusion · VLMs
- Compression · deployment · monitoring
- Capstone that makes you industry-ready

What industry interviews for

- Agents & tool use — in production NOW (57%)
- RAG grounded in company data
- Evals, tracing, observability (89% adoption)
- Cost, gateways, and serving at scale

So we teach in this order

- Months 1–2 · agents, RAG, evals, gateways
- Month 3 · what’s under the hood (PyTorch, transformers)
- Month 4 · vision & multimodal — fully intact
- Months 5–6 · ship, operate, capstone

The principle is backwards design: start from the job you want, sequence the learning to reach it fastest, keep every promised topic.

THE ARC

Your six months, re-mapped

M1	**Agent Engineering Foundations**	Wks 1–4	landscape · tool use & MCP · RAG · first traced agent
M2	**Production LLM Systems**	Wks 5–8	evals & LLM-as-judge · memory & multi-agent · AI gateways · fine-tuning
M3	**Under the Hood**	Wks 9–12	PyTorch · deep nets · transformer internals · compression & edge
M4	**Vision & Multimodal**	Wks 13–16	detection · data curation (FiftyOne) · diffusion · VLMs & multimodal agents
M5	**Ship & Operate**	Wks 17–20	deployment · monitoring & drift · safety & governance · architecture at scale
M6	**Capstone**	Wks 21–24	design → build → evaluate → demo day & technical defense

Full week-by-week map at the end of tonight's deck — screenshot it.

TONIGHT

Two hours, five acts, three checkpoints

I	**The Landscape**	inside the model · chatbot → copilot → agent · the loop	25 min
II	**The Patterns**	the Anthropic playbook · workflows vs agents · LIVE DEMO: a traced agent	30 min
III	**Knowledge & Context**	RAG · vector search · context engineering	25 min
IV	**Verification**	the bottleneck · traces, datasets, LLM-as-judge · LIVE DEMO: LangSmith	30 min
V	**The Road & Your Part**	vision · gateways · the 24-week map · your assignment	10 min

Checkpoints run exam-style: you vote, then we dissect why every wrong answer is wrong.

I

The Landscape

Same class of model, three eras of software — and the loop that changed everything.

60 seconds inside the model

One mechanism explains both the magic and the failures: predict the next token, again and again.

append “Paris” · feed everything back in · predict again — that loop is all an LLM ever does

Why tonight cares: a next-token predictor has no goals, no memory between calls, and no idea what your company knows. Agents, RAG, tools, memory, evals — every single thing in the next two hours is engineering wrapped around that one limitation.

ACT I · FOUNDATIONS

Three facts to build on

Trained on the internet

It has read everything public — and nothing about your clients, your tickets, your codebase, your policies.

→ that gap is why RAG exists (Act III)

Context window = its desk

Whatever is not in the prompt does not exist for the model. The window is the desk, not the library.

→ that limit is why context engineering exists

It always answers

When it doesn’t know, it autocompletes anyway — fluently. Confidence and correctness are different things.

→ that risk is why evals exist (Act IV)

Hallucination is not lying. It is confident autocomplete — and it is an engineering problem, not a moral one.

Same model. Different system.

What changed between eras is the software around the model — not the intelligence inside it.

CHATBOT

2022 – early 2023

Text in, text out. Q&A and drafting. Knows only its training data.

NO LOOP · STATELESS

COPILOT

2023 – 2024

Retrieval plus suggestions. It drafts — a human approves every step.

HUMAN IN THE LOOP

AGENT

2024 → today

It does the work: plans, calls tools, observes results, iterates.

HUMAN ON THE LOOP

Plain-language version: a chatbot answers. A copilot drafts. An agent finishes the job.

The definitions the industry actually uses

From Anthropic's "Building Effective Agents" — the most-cited engineering post in this field. Learn these verbatim.

WORKFLOWS

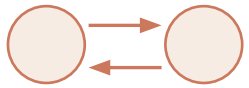
"systems where LLMs and tools are orchestrated through predefined code paths"

You write the path. The model fills in the steps. Predictable, debuggable, cheap.

AGENTS

"systems where LLMs dynamically direct their own processes and tool usage, maintaining control over how they accomplish tasks"

The model writes the path — at runtime. Flexible, powerful, and harder to trust.

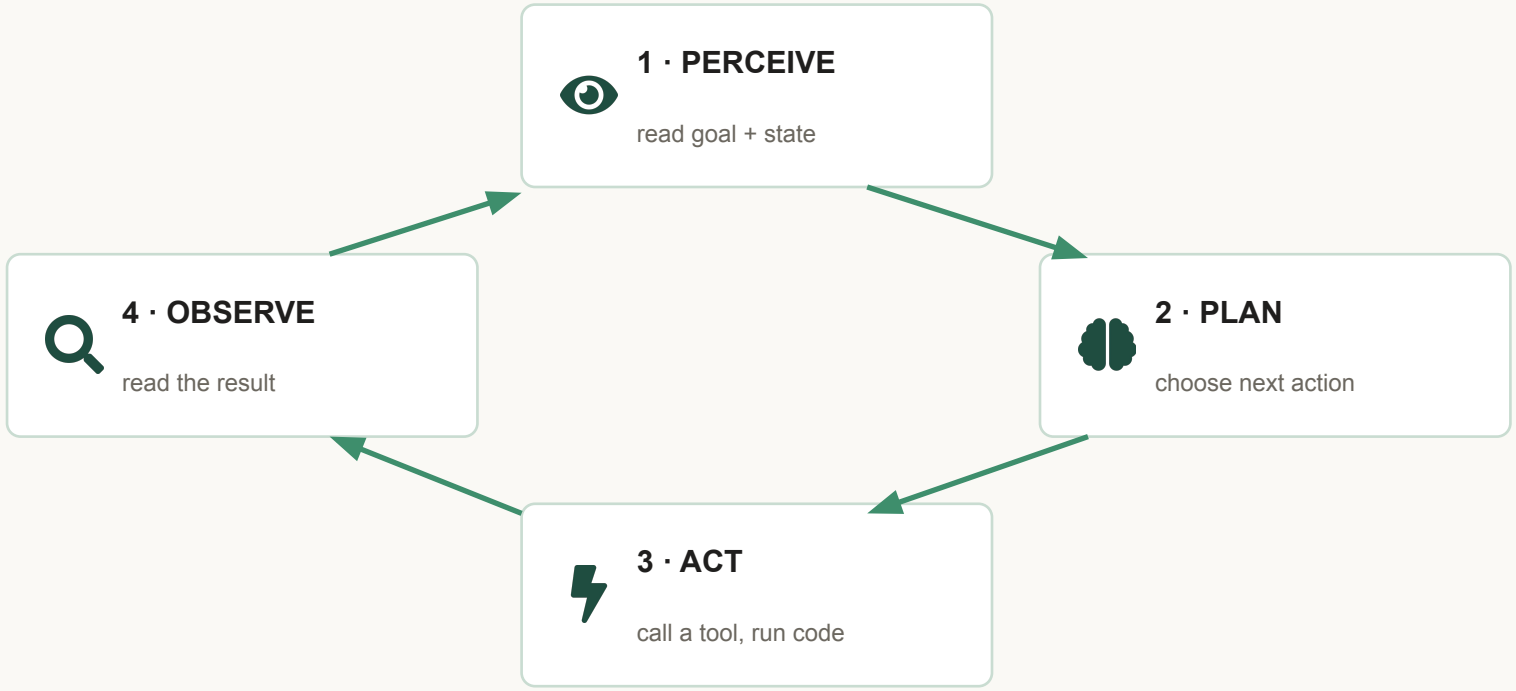


Interview answer, free of charge: "Do you know the difference between a workflow and an agent?" is 2026's FizzBuzz. Now you do — in Anthropic's own words.

Source: anthropic.com/engineering/building-effective-agents

The loop, drawn properly

Four steps in a cycle. Two ways out: the goal test passes — or the budget is spent.



EXIT · DONE

The goal test passes. The agent stops and reports — with evidence.

EXIT · BUDGET SPENT

Steps, tokens, time, money. A production agent always has a leash.

In 20 minutes you will watch this exact loop stream live — with every step traced.

CHECKPOINT 01 · YOU ARE THE ENGINEER

Vote before we move on

A fintech needs monthly compliance reports. The process is fixed and known: pull transactions › categorize › check against rules › draft report › human review › file. Same six steps, every month. What do you build?

A An autonomous agent that decides its own steps each month

B A workflow — the six steps hard-coded, an LLM filling in each one

C A multi-agent system with a supervisor and six specialists

D A fine-tuned model trained on last year’s reports

Answer: B — and why the rest are traps

B · A workflow with predefined code paths
The path is known in advance and never changes — exactly Anthropic's definition of a workflow. You get predictability, debuggability, and low cost, and the LLM still does the hard cognitive work inside each step.

A · Autonomous agent — Autonomy buys flexibility you don't need here — and pays for it in cost, latency, and unpredictable failure modes. Agents are for problems where you CAN'T write the path down.

C · Multi-agent supervisor — "Add complexity only when it demonstrably improves outcomes" — Anthropic's own rule. Six fixed steps demonstrably don't need six agents.

D · Fine-tuned model — Fine-tuning changes how a model behaves, not how a process is orchestrated. Wrong layer of the stack — and stale the day regulations change.

Pattern to remember: known path → workflow · unknown path → agent · proven bottleneck → only then add agents.

II

The Patterns

Five shapes cover almost every production system — this is the Anthropic playbook.

“Find the simplest solution possible, and only increase complexity when needed.”

— Anthropic, Building Effective Agents

The corollary they wrote next: “consider adding complexity only when it demonstrably improves outcomes.” Demonstrably = measured. Which is why Act IV of tonight is evals — you cannot follow this advice without them.

Patterns 1–3: fixed paths, composed well

Prompt chaining

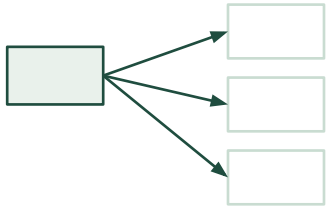
· the assembly line

Decompose into fixed steps; each LLM call consumes the last one’s output. Use when the task splits cleanly.

e.g. *draft → critique → rewrite → format*

Routing

· the triage desk

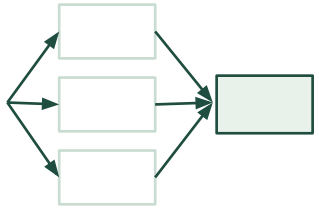


Classify the input first, then dispatch to a specialized prompt, model, or path. Cheap queries stay cheap.

e.g. *support: billing / technical / refund*

Parallelization

· many hands, one merge



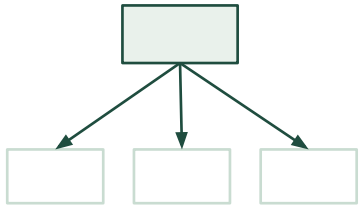
Independent subtasks fan out simultaneously; results merge. Speed — or diverse votes on one question.

e.g. *review 8 contracts at once, aggregate*

Patterns 4–5, and the agent itself

Orchestrator–workers

· the project manager

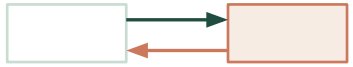


A lead LLM decomposes the task at runtime and delegates to workers. Use when you can't predict the subtasks.

e.g. *research: plan → parallel searches → synthesize*

Evaluator–optimizer

· writer + critic



One LLM produces, another grades against criteria, loop until pass. Trades latency and cost for quality.

e.g. *translation refined until the critic accepts*

The agent

· the loop, unleashed

Plans, acts, observes environmental feedback, repeats — path chosen at runtime. Maximum power, maximum need for guardrails and evals.

e.g. *"resolve this support ticket" — end to end*

Every framework you'll meet — LangGraph, CrewAI, AutoGen, the Claude Agent SDK — is a way of expressing these six shapes.

Four questions, asked in order

1	**Can you write the steps down in advance?**	Yes → workflow. No → keep going.
2	**Is one capable model + tools enough?**	Yes → single agent. Prove it isn't before adding more.
3	**Is the bottleneck breadth (parallel work)?**	Yes → orchestrator–workers or parallelization.
4	**Is the bottleneck quality (needs a critic)?**	Yes → evaluator–optimizer. Measure the gain.

Default answer: a single LLM call with retrieval. Most "we need agents" conversations end at question 1.

ACT II · THE HONEST TRADE-OFF

Multi-agent works — and it is expensive

Anthropic published the numbers from their own production research system. Both of them.

+90.2%

better than single-agent Claude Opus on their internal research eval — orchestrator + parallel subagents

Anthropic — "How we built our multi-agent research system"

~15×

more tokens than a normal chat interaction — the same system, the same report

Same source. They published the cost too — that's what honest engineering looks like.

The lesson is not "use multi-agent." It is: they could ONLY make this call because they measured both numbers. Value justified the tokens — for research. It might not for your use case.

You have already met these patterns

Klarna’s assistant

routing + guardrails + instant handoff

Incoming chats are triaged and routed to narrow paths; hard guardrails bound what the assistant may do; anything outside scope hands to a human instantly. Narrow scope, measured weekly — the $60M column from earlier.

Klarna, company reports

Morgan Stanley’s advisor assistant

retrieval workflow · human on the loop

A retrieval pipeline grounded in a vetted research corpus, shaped around the advisor’s real workflow — not a free-roaming agent. 98% of advisors use it daily; 30-minute research now takes seconds.

Morgan Stanley / OpenAI

Neither is exotic. Both are Act-II patterns chosen deliberately — and both sit at rung 2–3 of the trust ladder coming in Act IV.

DEMO 01 · SHOWN TONIGHT — YOURS THIS WEEK

An agent, live — with every step traced

```
# pip install -U langchain langchain-anthropic ddgs
# export ANTHROPIC_API_KEY=...   LANGSMITH_TRACING=true
# export LANGSMITH_API_KEY=...   (free tier)
from langchain.agents import create_agent
from ddgs import DDGS

def web_search(query: str) -> str:
    """Search the web for current information."""
    return str(DDGS().text(query, max_results=5))

agent = create_agent("anthropic:claude-sonnet-4-5",
                     tools=[web_search])
agent.invoke({"messages": [{"role": "user", "content":
  "Which sectors got VC funding in Pakistan this year? Cite sources."}]})
```

Ships tonight as Resource Pack 01 — Wednesday's help session is the install clinic.

WHAT TO WATCH

The loop, visible — model call → tool call → observation → model call, streaming

The trace — every step lands in LangSmith as a run tree, automatically — two env vars, zero code

The evidence — the final answer cites what the tool returned — grounding, live

The cost — tokens and latency per step, right in the trace. Budgets are real.

Tonight: we walk a pre-captured run — you run it yourself this week.

III

Knowledge & Context

How systems know your business — and why context is a finite resource.

ACT III · GROUNDING

Models arrive brilliant — and blind to your business

Retrieval-Augmented Generation: fetch the right evidence, let the model write from it.

Without grounding

“What’s our refund policy for enterprise clients?” → a fluent, confident, invented answer. The model has never seen your policy.

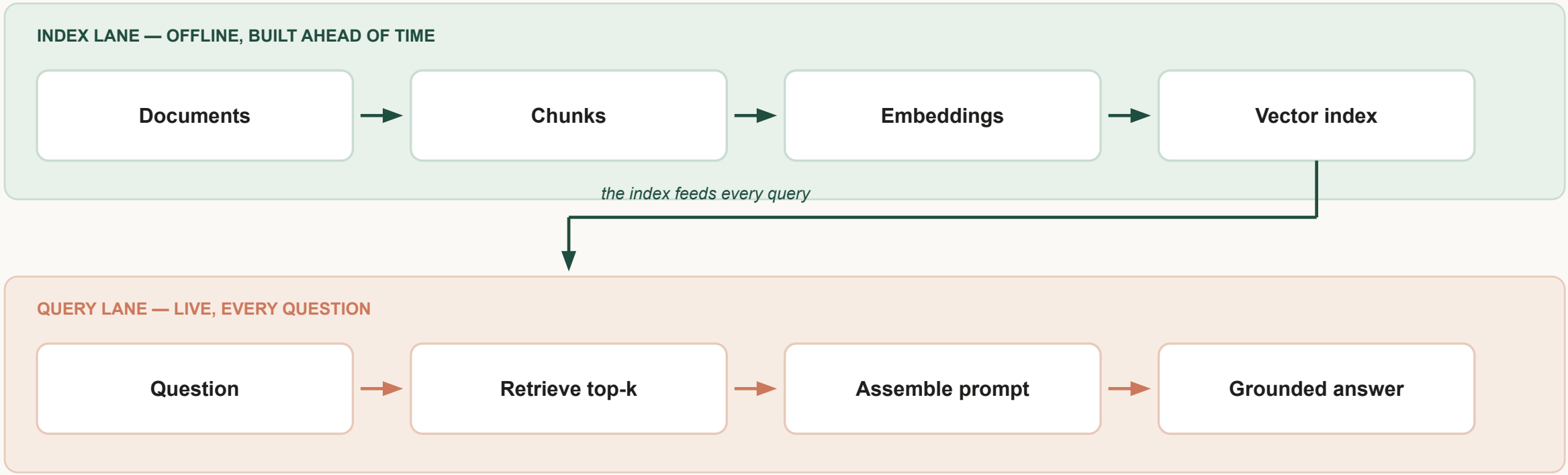
With RAG

The system retrieves the actual policy section, places it in context, and the model answers FROM it — with a citation you can click and check.

Why industry loves RAG: update the documents and the system knows new things — no retraining, no fine-tune, no waiting.

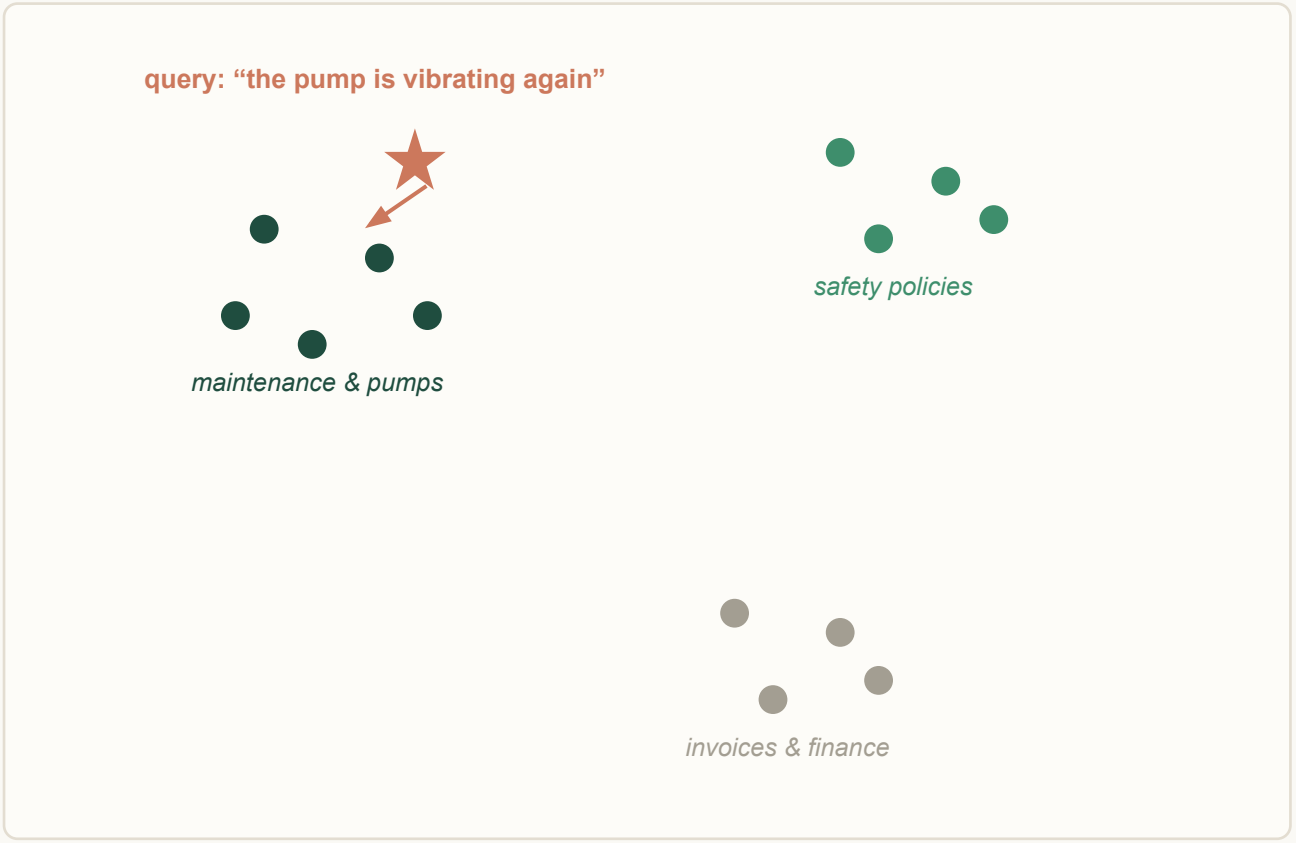
RAG as a system — two lanes, one prompt

The index is built ahead of time. Every question rides the query lane — through the index.



THE LINE TO REMEMBER The LLM is not the database. The retrieved context is the evidence.

Retrieval finds meaning, not keywords



Text → coordinates

An embedding model maps every chunk to a vector — hundreds of numbers that encode meaning.

Similar meaning → nearby points

“vibration alert” lands next to “abnormal oscillation” with zero shared words. Ctrl-F never finds that.

Search = nearest neighbours

FAISS, Qdrant, pgvector — the Week 3 toolbox — return the closest points in milliseconds.

Context engineering — the job title behind the buzzword

“Context must be treated as a finite resource with diminishing marginal returns.” — Anthropic. Models have an attention budget; your job is “the smallest possible set of high-signal tokens.”

Compaction

Summarize the transcript, restart the window with the summary. The desk stays small.

Structured notes

The agent writes durable notes OUTSIDE the window — memory that survives the session.

Sub-agents

Focused tasks get clean windows; only distilled results come back to the lead.

Just-in-time retrieval

Don’t preload everything — fetch data with tools at the moment it’s needed.

Notice: prompt engineering asks “what do I say once?” Context engineering asks “what does the model see at EVERY step of a long-running task?” Agents made this the harder — and better-paid — question.

Source: anthropic.com/engineering/effective-context-engineering-for-ai-agents

When RAG lies — four failures, four fixes

The document was never there

fix · Measure retrieval hit-rate on a test set — before blaming the model.

Chunking split the evidence

fix · Chunk by structure — sections and headings — not a fixed character count.

Nothing relevant — answered anyway

fix · Score threshold + a hard rule: below it, say “I don’t know.”

Model ignored the context

fix · Grounded prompt: answer ONLY from context, cite the chunk.

Spot the theme: every fix begins with a measurement. Act IV is where measurement lives.

CHECKPOINT 02 · YOU ARE THE ENGINEER

Vote before we move on

Your support bot answers a refund question using the 2022 policy. The correct 2024 policy is in the knowledge base. The answer sounded perfect. Where do you look FIRST?

A Swap in a bigger, smarter model

B The retrieval logs — which chunks were actually fetched, with what scores

C Rewrite the system prompt to say “always use the latest policy”

D Fine-tune the model on all the policy documents

CHECKPOINT 02 · THE DISSECTION

Answer: B — and why the rest are traps

B · The retrieval logs

Generation can only be as good as the evidence handed to it. First question in any RAG debugging session: did the right chunk even reach the context? Separate retrieval quality from answer quality — always.

A · Bigger model — No model, at any size, can cite a document that was never retrieved. You'd pay more to get the same wrong answer, more fluently.

C · Prompt fix — The prompt governs how context is used — it cannot repair context that is missing or wrong. (It's the SECOND thing you check.)

D · Fine-tuning — Freezing policies into weights is exactly what RAG exists to avoid — stale again at the next policy update, and you lose the citation.

Debugging order: retrieval → context assembly → prompt → model. Cheapest first, and usually guilty first.

IV

Verification

“Verification is the biggest bottleneck in AI.”

— said at our webinar. This act is that sentence, taken seriously.

ACT IV · WHY EVALS

Monday it worked. Wednesday it lied.

Nothing changed in your code. Everything changed in the inputs. That is normal — LLM systems are probabilistic.

“It worked in the demo”

…on the five inputs you tried, phrased the way you phrase things, on a good day. Users bring the other 10,000 phrasings.

“We’ll catch issues in prod”

You’ll catch the loud failures. The quiet ones — plausible wrong answers — are the ones that cost trust and money.

The engineering answer

A dataset of real cases, run on every change, graded automatically. Regression testing for behavior — that is what “evals” means.

Webinar echo: *“anyone can wire an agent — the AI engineer is the one who can write the evaluation pipeline.”*

ACT IV · OBSERVABILITY

Anatomy of a trace

A trace is the flight recorder of one request — every step, timed, costed, and inspectable. This is LangSmith's home screen.

```
▾ agent_run   ·  4.8s   ·  $0.011

    ▾ model_call №1 — plan   ·  1.1s

        "I should search for current data first"

    ▸ tool: web_search("…VC funding 2026")   ·  0.9s

        5 results returned → observation

    ▾ model_call №2 — synthesize   ·  2.6s

        final answer + citations   ·  1,842 tokens
```

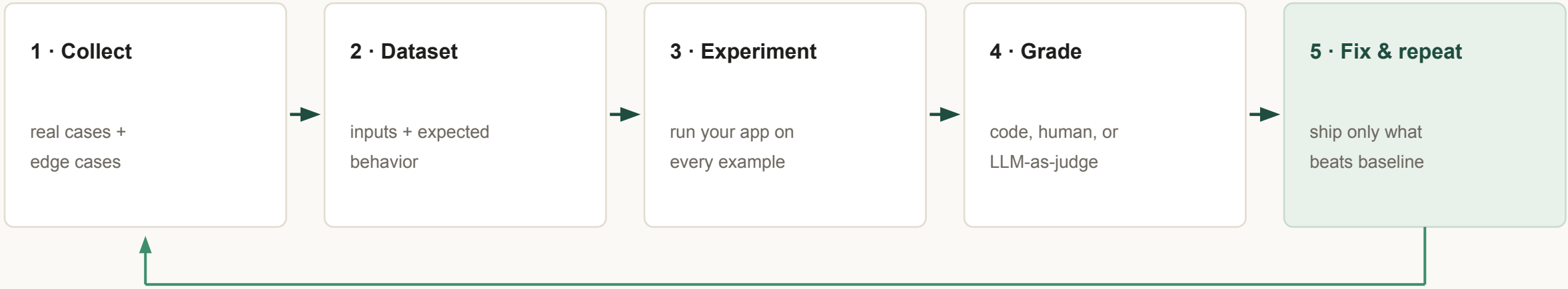
Read it like a story — plan → act → observe → synthesize. The loop from Act I, recorded.

Debug with your eyes — wrong answer? Click the exact step where it went wrong. No print statements.

Cost & latency per step — the 15× problem from Act II becomes visible — and fixable.

62% of production teams — have full tracing (LangChain, 2026). This screen is their day job.

The eval loop — the dataset IS the spec



every prompt change, model swap, or new tool goes back through the loop

"The dataset IS the spec." Twenty good examples with expected outcomes beat twenty pages of requirements — because these requirements execute. Start collecting from your very first user conversation.

Three kinds of grader — use them in this order

1

Code checks

Exact match, regex, JSON schema valid, contains-citation, latency < X. Deterministic, free, fast.

START HERE — cover everything code CAN judge

2

Human review

Domain experts label a sample. Slow and costly — but it is the ground truth that calibrates everything else.

THE CALIBRATION SET

3

LLM-as-judge

A model grades outputs against a rubric you write. Scales judgment — but the judge has biases, so you validate it against the human labels.

SCALE — AFTER 1 & 2 EXIST

The judge can cheat — LLM judges drift, prefer long answers, and grade their own style kindly. That is why the human calibration set exists. (Also from our webinar: “it can cheat.”)

ACT IV · AGENT-SPECIFIC

For agents, grade the journey — not just the destination

Final answer quality

Correct, grounded, complete? The classic eval — still necessary, no longer sufficient.

Trajectory

Did it take a sane path? Right tools, right order, no loops, no wasted steps?

Tool-call correctness

Right tool, right arguments, results actually used — not ignored.

Policy adherence

Stayed inside permissions, escalated when required, respected budgets.

In production teams this four-way grading IS the eval suite — multi-turn reasoning, tool-call correctness, policy adherence, trajectory. Week 5, you build one. This slide is a job description.

DEMO 02 · THE LANGSMITH TOUR

LangSmith, end to end — trace → dataset → experiment

1 · Open the trace	The Demo-1 run, as a tree: model calls, the tool call, tokens, latency, cost per step.
2 · Save to a dataset	One click: the input + ideal output become a test case. Five more make a starter spec.
3 · Run an experiment	Same dataset, two prompt versions, side by side — vibes become a scoreboard.
4 · Attach an LLM-as-judge	A rubric evaluator scores groundedness — and we read its reasoning out loud.

WHY THIS TOOL

89% of teams run observability; LangSmith is the reference implementation. Free developer tier — you will all have accounts by next week.

ALTERNATIVES EXIST

Langfuse (open-source), Braintrust, Arize Phoenix. The concepts transfer 1:1 — learn one deeply, speak them all.

Tonight we tour a pre-captured project — your own traces go live this week (Resource Pack 01).

The trust ladder — autonomy is earned with evidence

1

Suggest

Agent drafts, human does. Zero risk, instant value.

2

Approve each action

Agent acts, human clicks OK per step.

3

Act, escalate exceptions

Agent runs; humans see edge cases only.

4

Autonomous + audit

Full autonomy — logs, budgets, rollback.

What moves you up a rung? Eval scores, trace history, incident record. Klarna and Morgan Stanley live at 2–3. Nobody serious starts at 4.

CHECKPOINT 03 · YOU ARE THE ENGINEER

Vote before we move on

You rewrote your agent's system prompt and the demo "feels better." Your CTO asks: is it ACTUALLY better — and can you prove it by Friday? What is the minimum credible answer?

A Ship it and watch user feedback for a month

B A dataset of representative cases + an experiment comparing old vs new prompt on the same cases

C Three hours of careful manual testing on the new prompt

D Upgrade to a bigger model so the question matters less

Answer: B — and why the rest are traps

B · Dataset + comparative experiment

Same inputs, both versions, graded the same way — that is the only apples-to-apples comparison. It is reproducible, fast, and it becomes your regression suite forever after. This is precisely LangSmith's dataset → experiment flow from Demo 2.

A · Ship and watch — A month of users as your test harness — slow, confounded by traffic mix, and the failures land on customers first. Feedback complements evals; it can't replace them.

C · Manual testing — Not reproducible, not comparable (you'll subconsciously test the new prompt kindly), and it evaporates — next change, you start from zero.

D · Bigger model — Changes the system under test instead of answering the question — and now you have TWO unmeasured changes. Also: 33% of teams say quality is the barrier; few say model size.

The sentence to keep: if it isn't in a dataset, it isn't a requirement — it's a hope.

V

The Road & Your Part

Vision, gateways, the 24-week map — and the assignment that starts your portfolio.

ACT V · MONTH 4 PREVIEW

Vision is not dead — it is where LLMs still lose

> *"LLMs are not performing there — and now they want experts."*

— our webinar, on computer vision hiring. Detection, inspection, tracking, OCR on messy scans, edge deployment: still specialist work, still hiring.

Industrial vision — inspection, tracking, OCR on messy scans, edge deployment — still hires specialists precisely because generic LLMs underperform there.

Wk 13 · Detection & transfer learning

MMDetection, Detectron2, fine-tuned detectors

Wk 14 · Data curation

FiftyOne, CVAT — where vision projects are actually won

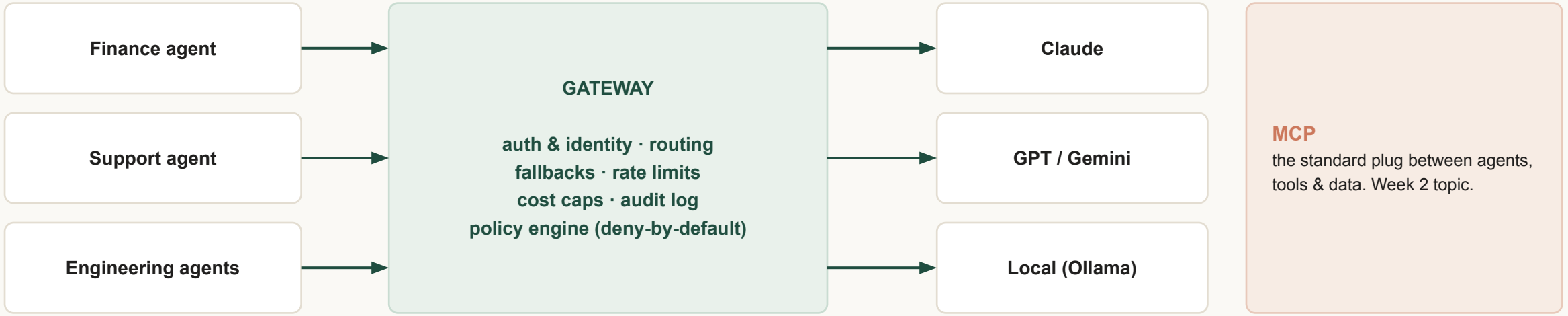
Wk 15 · Diffusion & ControlNet

generation as a controllable production tool

Wk 16 · VLMs — agents that SEE

CLIP, VQA, multimodal agents: vision meets everything from tonight

The AI gateway — where agents meet the enterprise



Why enterprises insist on it: one governed door to every model — identity on every call, spend visible per team, one place to switch providers when pricing or quality changes. Enterprises build these on MCP with identity propagation — and in Week 7 you build a small one with LiteLLM.

ACT V · THE FULL MAP

All 24 weeks — screenshot this

M1 · Agent Foundations

1 · Landscape, loop, patterns (tonight)
2 · Tool use, function calling, MCP, context eng.
3 · RAG: embeddings, vector DBs, chunking
4 · Build week: LangGraph agent + LangSmith

M2 · Production Systems

5 · Evals: datasets, graders, LLM-as-judge
6 · Memory, multi-agent, deep agents
7 · AI gateways, routing, serving, cost
8 · Fine-tuning: SFT, LoRA, QLoRA

M3 · Under the Hood

9 · PyTorch ecosystem, transfer learning
10 · Deep networks, training dynamics
11 · Transformer internals, attention
12 · Compression, quantization, edge

M4 · Vision & Multimodal

13 · Detection: MMDetection, Detectron2
14 · Data curation: FiftyOne, CVAT
15 · Diffusion, ControlNet
16 · VLMs & multimodal agents

M5 · Ship & Operate

17 · Deployment: FastAPI, Docker, serving
18 · Monitoring, drift, online evals
19 · Safety, governance, red-teaming
20 · Architecture: caching, queues, scale

M6 · Capstone

21 · Design reviews
22 · Build sprint
23 · Evaluate & harden
24 · Demo day + technical defense

Every topic from the published outline is present. Order changed; the promise did not.

ACT V · YOUR TURN

Nominate one automation candidate

From work or study you actually know. This seeds your capstone — the best ones get built over six months.

It happens at least weekly
frequency is where ROI lives

Success has a clear definition
if you can't grade it, you can't eval it

The documents or data already exist
no data, no grounding, no project

A mistake would be recoverable
start at trust-ladder rung 1, not 4

Say it out loud now — I want five nominations before we close. Keep yours in hand — the next three slides turn it into your first product.

CHALLENGE 01 · YOUR FIRST PRODUCT

Build the thing businesses are begging for

Pick a real business — an insurance company, a hospital chain, a textile exporter, a school network. Yours to choose. Its knowledge lives in four messy places: policy documents, a SQL database of transactions, working spreadsheets, and scanned PDFs.

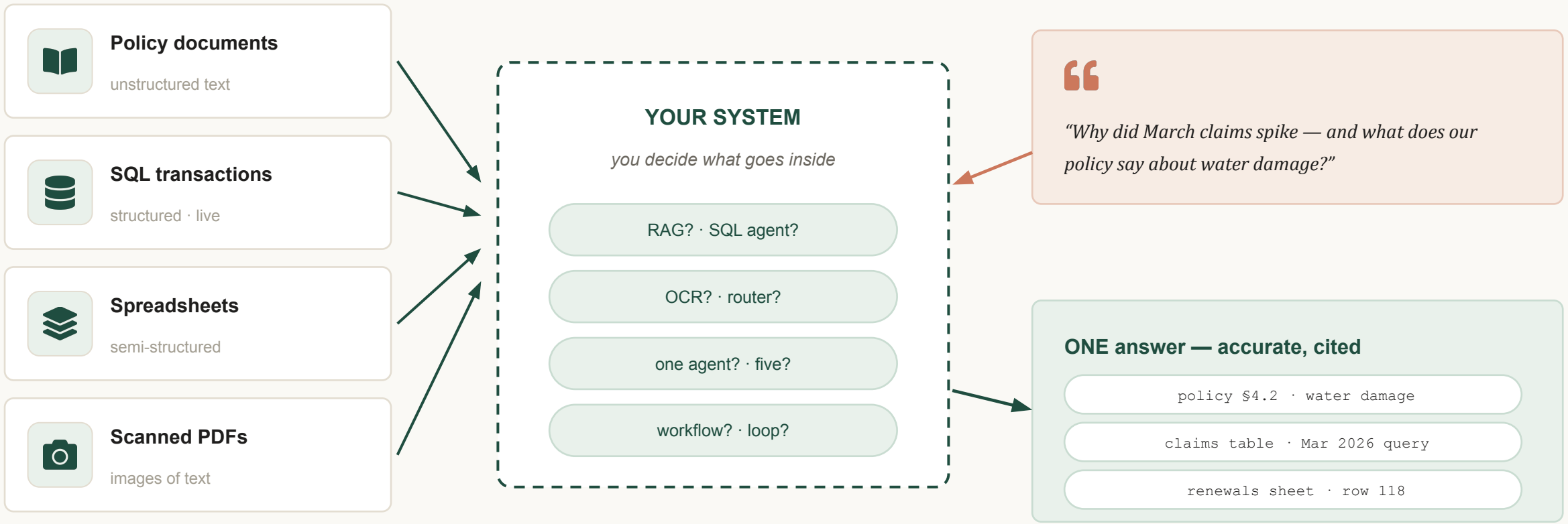
The owner walks in and asks anything — “Why did March claims spike?” … “What does our policy say about water damage?” — and gets ONE accurate answer, with citations to the exact source.

This is not an exercise. Cross-source question answering is the #1 request in enterprise AI right now — and from tonight, it is your product.

Everything you need to know was covered tonight. Grounding, tools, tracing, evals, gateway thinking — it is all in this deck. The HOW — the architecture, the steps, the escalations — is deliberately not given. That part is yours.

Full brief drops in Discord tonight as CHALLENGE_01.

Four messy sources. One trustworthy answer.



NON-NEGOTIABLES cited answers · traced end-to-end in LangSmith · evals you designed and ran · gateway thinking: model choice, cost, fallback

CHALLENGE 01 · HOW THIS WILL WORK

Development is cheap now. Decisions are not.

The how is yours

No recipe, no starter repo. The approach, the architecture, the escalation paths — the human decisions are what we want to see. That is the actual skill this industry pays for.

Realism wins

Real policy wording, believable transactions, actually-scanned pages. The more real your data, the more real your product — toy data can only ever prove a toy.

You are not alone

Stuck is normal; stuck is not failure. Bring what you TRIED to the Wednesday clinic and Discord — we help from there. “How do I start?” after zero attempts is the only wrong question.

START THIS WEEK

- Run Resource Pack 01 — it is your tracing + eval scaffold
- Pick your business · start drafting the four data sources
- Bring your plan — not your questions — to Wednesday’s clinic

ADDITIONAL RESOURCE · NOT HOMEWORK

LangSmith Essentials — building reliable agents

LangChain Academy · ~2 hours · free · academy.langchain.com

One evening, disproportionate payoff — ideally before Week 2.

THE CANON

Your reading list is the industry's reading list

Everything cited tonight, in one place. These are the documents hiring managers quote.

Anthropic — Building Effective Agents

workflows vs agents · the five patterns · simplicity

`anthropic.com/engineering/building-effective-agents`

Anthropic — Effective Context Engineering for AI Agents

attention budget · compaction · notes · sub-agents

`anthropic.com/engineering/effective-context-engineering-for-ai-agents`

Anthropic — How We Built Our Multi-Agent Research System

+90.2% and the 15× token bill — honest trade-offs

`anthropic.com/engineering/multi-agent-research-system`

LangChain — State of Agent Engineering 2026

1,340 respondents · 57% in production · quality #1

`langchain.com/state-of-agent-engineering`

LangSmith — Evaluation docs

datasets · experiments · evaluators · LLM-as-judge

`docs.langchain.com (LangSmith → Evaluation)`

Anthropic Academy + LangChain Academy

free structured courses — and the CCA-F path for the ambitious

`anthropic.com/learn · academy.langchain.com`

Links posted in Discord tonight. Read one per week — by Month 2 you speak the industry's language natively.

UNTIL NEXT WEEK

The industry standard is now your standard.

Tonight's recording lifetime access — the demos reward a slower second watch

Wednesday help session: install clinic + Challenge 01 planning (Discord)

Next week Week 2 — Tool use, function calling & MCP: how agents touch the world

Resource Pack 01 + the CHALLENGE_01 brief — both drop in whatsapp group tonight.