

Justin Milligan · Healthlead.io · February 2026

Building Effective Agents in 1Q26

From Prompts to Agents

Prepared for Consultare

Why Listen to Me?

Custom CRM

Built TheraTrack from scratch using AI

SaaS Clone

Replicated multi-million dollar application

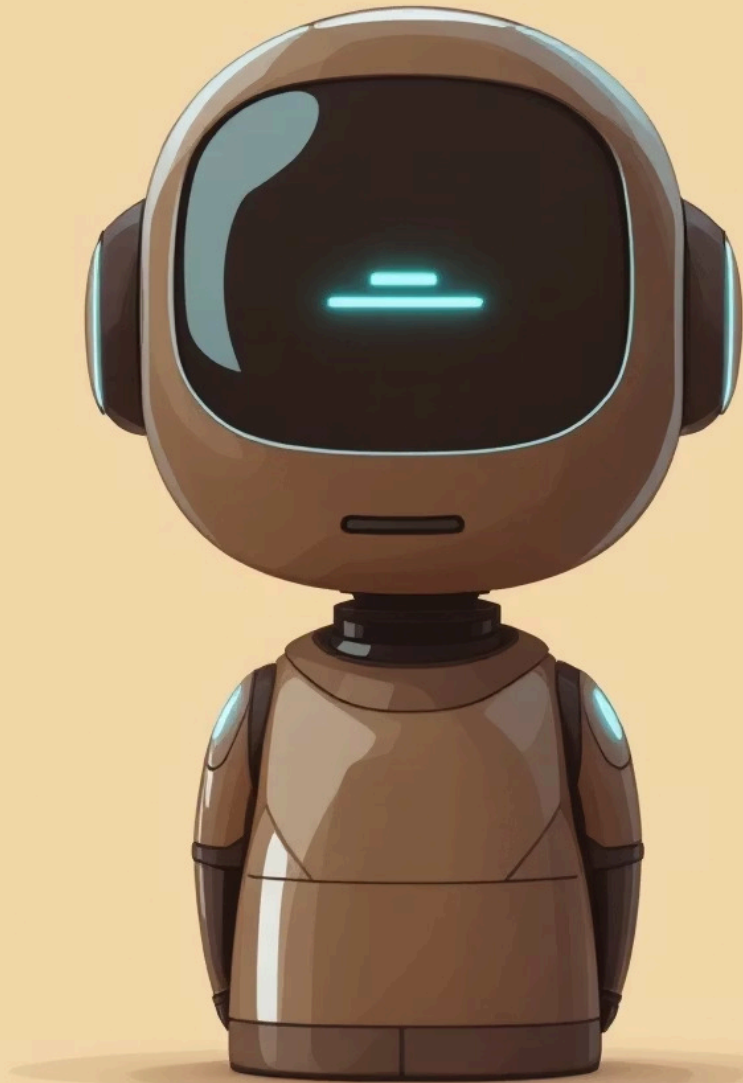
Patent Filed

Patent-pending product developed

100+ Page Website

No designer, no developer, no agency

Built in the last 7 months. Not an AI researcher - a practitioner. Everything here comes from doing as a sole employee.



Two ways to use AI. Most people only use one.

Oracle Mode

You ask. AI answers from general knowledge.

Tool Mode

You provide context. AI works within YOUR reality.

Same AI. Dramatically different results.

Oracle Mode vs. Tool Mode — When to Use Each

Oracle Mode	Tool Mode
Exploring a new topic	Working in YOUR systems
Brainstorming ideas	Producing deliverables
Quick research	Anything client-facing
Low stakes	High stakes
No context to provide	You HAVE examples, docs, data

Same Problem. Two Approaches.

SCENARIO

Client says the invoice doesn't match the proposal

1

Oracle Mode

"How do I fix invoice discrepancies?"

Result: Generic list. Reconcile line items, check for scope changes, review payment terms.

2

Tool Mode

"Here's the proposal, the SOW, and the invoice (paste).
What doesn't match?"

Result: The proposal quoted 60 hours for Phase 2 but was billed 84. The extra hours came from two change requests that were approved but not added to SOW.



Make Oracle Mode Better



Persona

Think like a senior SAP B1 consultant...



Scoping

Only consider X context / exclude Y.



Best Practice Discovery

What would an elite firm do?

Force AI to reason from a specific vantage point instead of averaging everything it knows.

Tool Mode / Context Engineering

1

Context Loading

Here's our schema + a working query. Now write this one.

2

Few-Shot Examples

Here are 3 scope docs we wrote. Write one like these.

3

Reference + Task

Here's last month's report + this month's data. Go.

4

Constraints

Only use these tables. If unsure, ask.

Use ChatGPT/Claude projects or CLI Terminal with folder system on your computer.

Projects Are Powerful

The screenshot shows the ChatGPT interface with a sidebar on the left containing navigation options like 'New chat', 'Search chats', 'Images', 'Apps', 'Deep research', 'Codex', 'GPTs', and 'Projects'. The 'Projects' section is expanded to show a project named 'Main Street Grant - Amex'. The main content area displays the project name at the top, followed by a 'New chat in Main Street Grant - Amex' input field. Below this, there are two tabs: 'Chats' and 'Sources', with an orange arrow pointing to the 'Sources' tab. Under the 'Sources' tab, there is an 'Add sources' button and a list of three source files:

- mainstreet-amex-shop-small-grant-program-webpage.md**
File · Jan 19, 2026
- MSA-Amex-Shop-Small-Grants-Program-FAQs-Rev.-1.13.pdf**
PDF · Jan 19, 2026
- MSA-Amex-Shop-Small-Grants-Program-Terms-Rev.-1.13.pdf**
PDF · Jan 19, 2026

The screenshot shows the ChatGPT interface with the same sidebar as the first image. The 'Projects' section is expanded to show the 'Main Street Grant - Amex' project. The main content area displays the project name at the top, followed by a 'New chat in Main Street Grant - Amex' input field. Below this, there are two tabs: 'Chats' and 'Sources', with an orange arrow pointing to the 'Chats' tab. Under the 'Chats' tab, there is a list of two chat messages:

- Grant Application Review**
yes provide a diff checklist Jan 23
- Submission Deadline Info**
when is the submission deadline? Jan 21

Three Techniques That Upgrade Everything



Chain of Thought

Think through this step by step before answering.



Meta-Prompting

What context would you need to give me the best output?



Iterative Refinement

Prompt 1: Outline → Prompt 2: Expand → Prompt 3: Critique

Use these strategies to improve "runtime" prompting. These will help you craft better prompts.



Now The Good Stuff - What We're Covering Today

1

How to make AI output trustworthy

2

How to build agents that work

3

How to get better every week



One principle runs through all three: give AI what good looks like.

💡 BIG IDEA 1

How to Trust AI Output

The #1 reason people underuse AI is trust.

The fix isn't 'double-check everything.' The fix is a system.



Getting A Trustable Output

1

Specific Context

 WEAK

"Write me a project update."

2

Clear Constraints

 STRONG

"Here's last week's update, this week's Jira tickets, and the client's priorities. Write this week's update in the same format. Only reference work that's in the tickets."

3

Defined Output Format

4

Cross-Check

Smaller Steps = Higher Trust



One Big Prompt

⚠️ Process this invoice and update the billing tracker.

WHERE DID IT GO WRONG? NO IDEA.



Decomposed into Steps

Extract — PO number, line items, amounts, tax, total from this invoice.

Compare — Do the line items match the original proposal? Flag mismatches.

Format — Add to the billing tracker using this template row.

Each step is easy to check. Fix errors before they compound.

13 Ways to Show AI What Good Looks Like

Method	What you give it
Golden examples	"Here's one we're proud of. Match this."
Working code	"Here's a query that works. Write the next one like it."
Templates	"Use this format — these sections, this order, this length."
Previous versions	"Here's last month's report. Do this month's."
Manual run + corrections	Do it in conversation. Every correction teaches it.
Output schema	"Return: field 1, field 2, field 3. Nothing else."
Checklists	"Every output must include X, Y, and Z."
Rubrics	"Score on these 4 dimensions, 1-3 scale."
Style guides	Paste your brand voice doc, naming conventions, writing rules.
SOPs and process docs	Paste the internal wiki — how YOUR company does it.
Annotated bad examples	"Here's a bad one. Here's exactly what's wrong with it."
Constraints	"Only use these tables. Never assume. Ask if unsure."
Meta-prompt	What would you need from me to do this well?"

When 'Good' is Hard to Define

Easy

SQL queries, reports, templates (you have examples). Show the AI.

Hard

Website design, strategy docs, unfamiliar bugs.

No clear 'right answer.'

A rubric translates subjective quality into measurable dimensions.

Good website

accessibility, load time, visual hierarchy, mobile, brand

Good strategy doc

specificity, feasibility, risk coverage, goal alignment

Good debug analysis

root cause, evidence, proposed fix, side effects

How to Build a Rubric

01

Ask

What does "good" look like for THIS task?

02

Can you point to examples?

→ Use Tool mode

03

Can't?

→ Break quality into 3-5 measurable dimensions

04

Define levels

Define poor / acceptable / excellent for each

05

Use the rubric

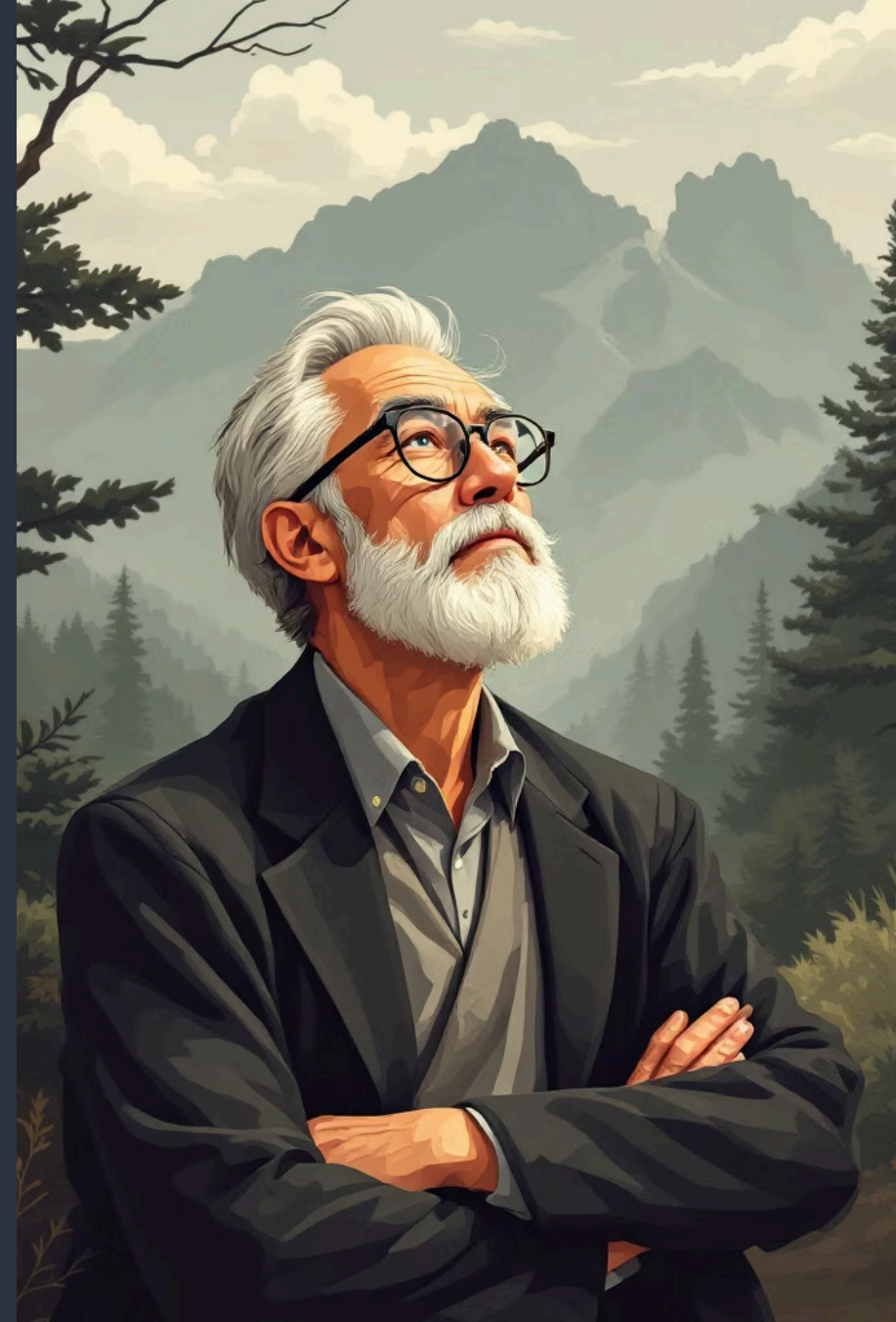
To evaluate AND to instruct the AI

Ask AI to Build the Rubric FOR YOU

This alone puts you ahead of 95% of AI users.

Example Prompt

I'd like to discuss how i should improve the slide content before building the PPT. What would you suggest for a rubrick on how to judge the quality of the presentation? How would a professional TEDx Presenter judge their presentation?



Consultare



AI Training & Coaching Program Development for Consultare.net — A consulting engagement to deliver AI education (talks, seminars, 1-on-1 coaching, small group training).

Reply...



Opus 4.6 Extended ▾

Evaluating presentation quality for diverse audiences

Last message 21 minutes ago



Claude business account pricing and features

Last message 29 days ago

AI training program proposal for Consultare

Last message 29 days ago

Initial project outreach and scoping email

Last message 29 days ago

Memory

Only you



Purpose & context Justin is developing and delivering AI training and coaching services for consulting companies, with his primary active...

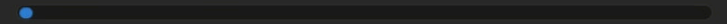
Last updated 12 minutes ago

Instructions



Engagement Overview **Client:** Consultare.net / Consultare.io
Target Audience: 5 employees across different roles ## Known...

Files



1% of project capacity used

healthlead/consultare
main

GITHUB

submissions.json
518 lines

JSON

2026-01-20_meeting_with_gustavo_(presi...
97 lines

TEXT



Rubric Applied: "AI That Works" Presentation (V1)

Dimensions chosen before scoring — first draft evaluation

Dimension	1 (Poor)	2 (Acceptable)	3 (Excellent)
Audience Calibration 2	One skill level dominates; others lost or bored	Lands for the middle; extremes partially addressed	Every skill level has moments designed for them
Actionability 3	Inspirational but no clear "do this tomorrow"	Some takeaways but vague on application	Every section has a framework or template they use immediately
Interactive Element 1	Pure lecture — audience only listens	One moment where audience tries something	Multiple exercises producing real output they keep
Clear Next Steps 2	Ends with "any questions?" and nothing else	Generic challenge or homework assigned	Each person knows what happens next for them specifically

Overall: **8 / 12** — strong content, needs evidence and one interactive moment

Two Tools. Zero Blind Spots.



Generate

with Tool A. Different training data.



Validate

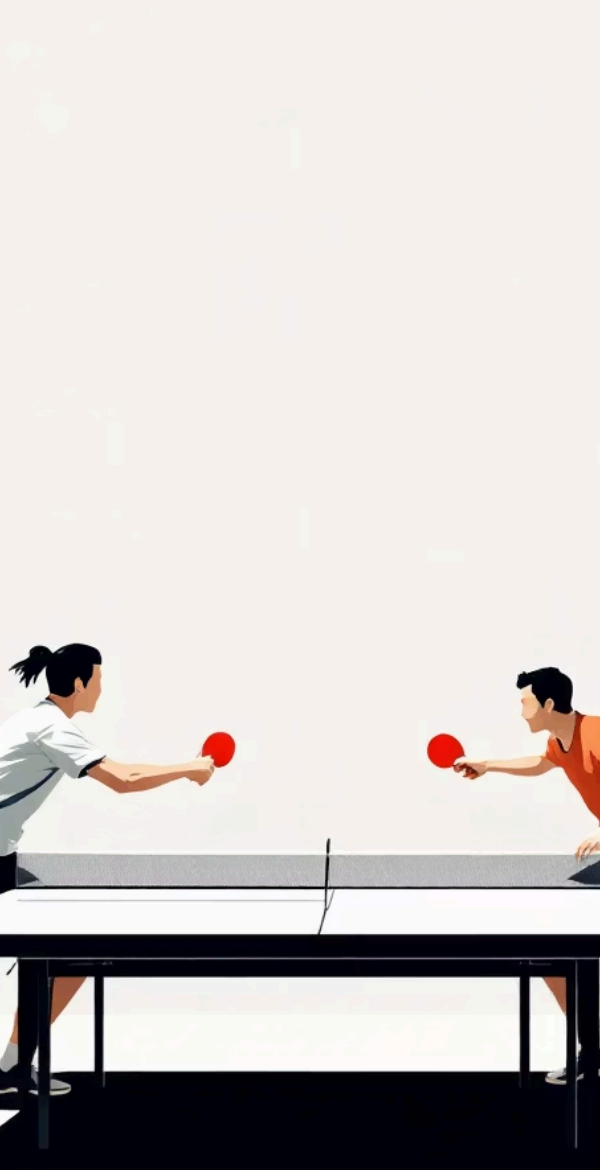
with Tool B. Different blind spots.

Double-check regularly

Don't trust the first output. Spot-check facts, numbers, and logic every time — especially early on. Trust is earned through repetition, not assumption.

Use other LLMs for different perspectives

Each model sees the problem differently. A second model isn't a spellchecker — it has different blind spots. Where they agree, confidence goes up. Where they diverge, you know exactly where to look.



Ping-Pong: 2-LLMs, 1-Answer



LLM A (Claude) generates

First draft of the work using context engineering tools.



LLM B (ChatGPT) critiques

Is this accurate? What needs to change?



LLM A (Claude) responds

Here's what ChatGPT said. **Don't be a syncophant. It doesn't have all of your context so it may not be right. Explain where & why it's wrong.**



LLM B (ChatGPT) validates

Here is Claude's rebuttal. What is right? What is wrong.?

Each round sharpens the output. Stop when they converge.

Three Validation Techniques

Cross-Check

Generate in A, score in B with your rubric.

Use for: code, queries, data

Adversarial Review

Generate in A, attack in B: 'Find everything wrong.'

Use for: debugging, risk

Parallel + Diff

Same prompt to both, compare where they diverge.

Use for: docs, planning

In Practice: Scope Document Validation



Generate

ChatGPT writes the scope doc from requirements



Score or Validate

Check against rubric: Requirements clear? Boundaries defined? Effort broken down? Risks covered?



Cross-check

Paste into Copilot: "What's missing? What's ambiguous? What would a client push back on?"



Result

Copilot flags no out-of-scope section and the effort estimate has no assumptions listed

Copy These. Use Them Tomorrow.



Rubric Check

Score this against this rubric: [rubric]. Be specific about what's missing.



Second Opinion

I asked another AI to do this. Here's what it produced. What would you do differently?



Stress Test

Poke holes in this. What assumptions is it making? What edge cases does it miss?

The image shows a VS Code editor with two main panels. The top panel displays a JSON file named `submissions.json` with the following content:

```
{
  "formId": "pbD9kV",
  "fetchedAt": "2026-02-23T20:45:41.957Z",
  "totalSubmissions": 12,
  "submissions": [
    {
      "id": "aQ7DxEv",
      "submittedAt": "2026-02-23T18:11:46.000Z",
      "respondentId": "q4PRpDG",
      "answers": {
        "Name": "Juan Vasquez",
        "Job Title / Role": " Business Intelligence Analyst ",
        "Location": [
          "Miami / Doral"
        ],
        "Which Jira activities are part of your regular work?": [
          "I don't use Jira regularly"
        ]
      }
    }
  ]
}
```

The bottom panel shows a terminal window with the following output:

```
consultare main • ? ) claude-skip
Claude Code v2.1.56
Opus 4.6 · Claude Max
~/repos/consultare

) here is my presentation thus far. for my big idea #1, am i missing anything that would help our avatars? what would a professional presenter do differently given the wide audience range?'/home/jrm/Downloads/Justin-Milligan-Healthleadio-February-2026 (1).pdf'

● Read 1 file (ctrl+o to expand)

● I've reviewed the full PDF (20 slides) against the slide brief and the audience profiles. Here's my assessment of Big Idea 1.

What's working well

- The trust framing is universal – everyone in the room has been burned by a hallucination
- Rubric progression is the strongest arc in the section (concept → how to build → let AI build it → score THIS presentation → validation techniques)
- The self-scoring slide (9/15 on your own deck) is brilliant dogfooding – it makes the concept impossible to dismiss
- Ping-pong pattern is genuinely advanced and will hook Avanesch
- The invoice mismatch example in the opening section (slide 3) is the best admin anchor in the whole deck

What's missing from the brief that hurts

Three slides from your brief got cut, and they're the three most useful for your basic-level avatars:

1. "Smaller Steps = Higher Trust" (brief slide 12) – Decomposition. This is the single most practical trust technique for beginners. "Break one big prompt into four small ones, verify each step." The admins and PMs need this more than anything. Without it, Big Idea 1 jumps from "define good output" straight to "build rubrics" – that's a big cognitive leap for someone who's barely prompting.
2. "Define the Output Before You Ask" (brief slide 10) – The three role-spanning examples (bug diagnosis format for Avanesch, client email format for PMs, data table format for admins). Slide 9's "Defined Output Format" pillar names the concept but doesn't show anyone how. The examples did that.
3. "Copy These. Use Them Tomorrow" (brief slide 16) – The six prompt templates. This was the most actionable takeaway for beginners. The 13-method table is a reference: the templates are executable. Beginners need copy-paste starting points, not a menu of 13 concepts.
```



This is Claude's feedback on my Big Idea #1 for am i missing anything that would help our avatars? what would a professional presenter do differently given the wide audience. do you agree/disagree and why?...



Analysis

I **completely agree** with this feedback. Claude has identified the primary risk in your current "Big Idea 1" draft: **the "Trust Gap" between high-level concepts and low-level execution.**

While your content is sophisticated, it currently risks alienating the two admins in the room (40% of your audience) while asking the PMs to make a massive cognitive leap from "general prompting" to "rubric engineering." [🔗 +2](#)

Here is why I agree with the specific points and how they align with your presentation's goals:

1. The Necessity of Decomposition (Slide 12)

I agree that jumping straight to rubrics is a "big cognitive leap." [🔗 +1](#)

- **Why it's vital:** For your admin avatars who handle invoicing and Jira data, a rubric might feel like "extra work." [🔗 +2](#)

Rubric: V1 → V2 Comparison

Same dimensions, scored before and after revision

Dimension	V1	V2	What Changed	3 (Excellent) Looks Like
Audience Calibration	2	3	Decomposition slide bridges beginners. Ping-pong + Judge + Multi-LLM give advanced users new material.	Every skill level has moments designed for them
Actionability	3	3	Already strong. Added 13-method table and 5-step agent build — more reference material on top of solid base.	Every section has a framework or template they use immediately
Interactive Element	1	2	Real screenshots of the work: Claude Code, ping-pong, rubric on THIS deck. Evidence, not hypotheticals. Still watching, not doing.	Multiple exercises producing real output they keep
Clear Next Steps	2	2	Unchanged. Same weekly challenge, same closing. Missing personalized "here's what we work on 1-on-1" moment.	Each person knows what happens next for them specifically

Overall: **8 / 12** → **10 / 12** — one live exercise away from max score

💡 BIG IDEA 2

How to Build Agents That Work

An agent is just a workflow that an AI runs... for you.

It's just a set of instructions, nothing else. No mystery.

What An AI Agent Is

① Software that uses an LLM to make decisions and take actions, not just answer questions.

Reads context

Ingests documents, data, logs, messages

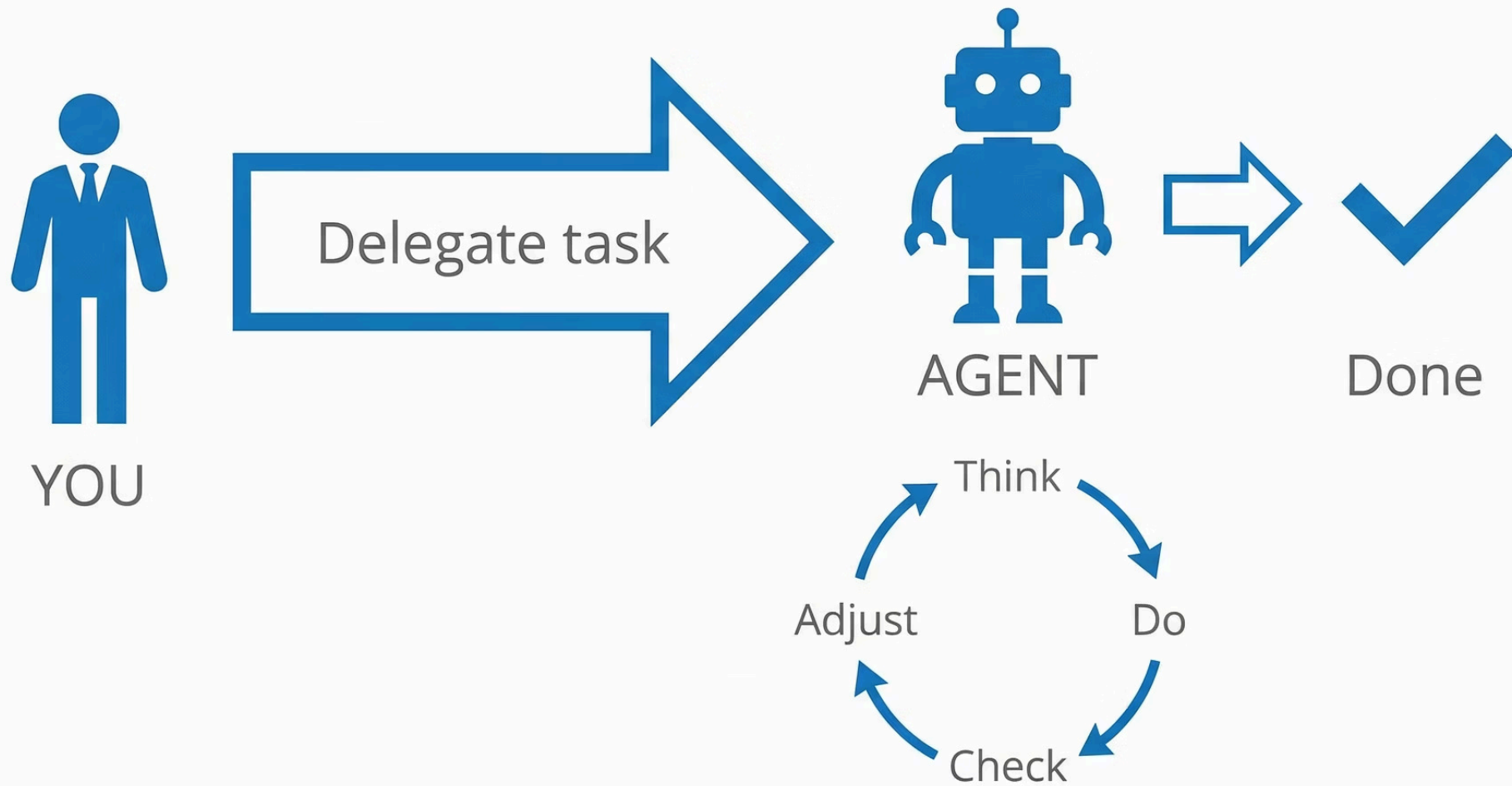
Makes judgments

Classifies, prioritizes, extracts, summarizes

Takes action

Writes output, routes work, triggers next steps

Simple Agent Loop



Simple Examples

Without Agent

You: "Research this prospect for me"

AI: Gives you info

With Agent

You: "Research 50 prospects, score them, prioritize, generate call briefs"

Agent: Does all of it, delivers a report

More Examples

- "Analyze 6 months of customer complaints and surface the top 3 issues"
- "Monitor competitors' pricing pages and alert me to changes"

Agent Examples

📄 Agents turn messy, human-generated information into something you can act on.



Error Log → Diagnosis

Messy error log → Classified diagnosis + root cause + suggested fix



Transcript → Scope Doc

Rambling transcript → Structured scope document



Support Ticket → Triage

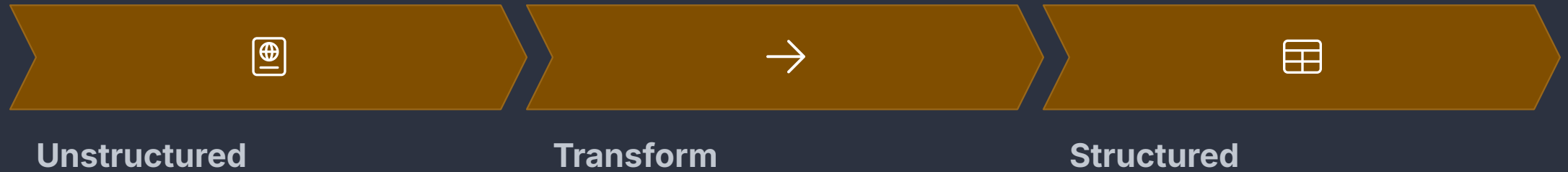
Ambiguous support ticket → Category, priority, assignee



Invoices → Line Items

Pile of invoices → Extracted line items, totals, GL codes

What An Agent Isn't



If you need the same output every time, that's code — not an agent.



Before You Automate, Solve It Once



Load real context

Paste the actual messy inputs



Work through the prompts

What questions, what order?



Check the output

Correct? Complete? Edge cases?



Iterate

Every correction becomes a lesson

You are the Orchestrator and Judge during the manual process. When you're done, you have: the context, the prompts, the corrections to show AI what 'good' looks like. That's the blueprint for the Orchestrator and Judges.

The Judge



Did it follow instructions?

Did it skip steps or take shortcuts?



Did it drift?

Is it still solving the original problem?



Is the output right?

Does it match the rubric or expected format?



Is it safe?

Does the output cause security holes?

The Orchestrator tells the Agents what to do. The Judge keeps the agent honest.



Three Roles: Orchestrator, Agents, & Judges

Input — "A prompt, document, data, or trigger"

Agent — "Does the work. One task, focused."

Output — "The finished result"

Orchestrator "Routes the work. Decides what happens next."

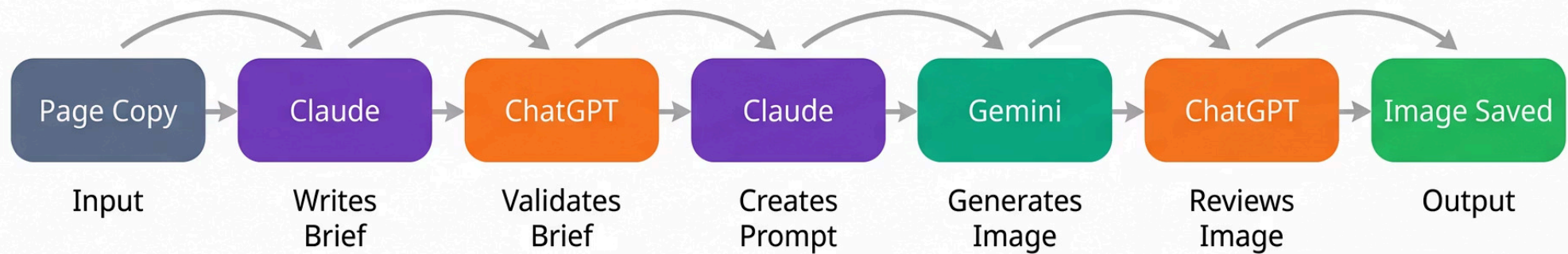
Judge — "Checks the output. Pass, retry, or escalate."

Loop — "Orchestrator gives output to next agent in the chain."

Multi-LLM Agentic Workflow

AI Models Working Together Autonomously

Three different AI models collaborate - each doing what it does best



■ Claude (Anthropic)

■ Gemini (Google)

■ ChatGPT (OpenAI)



Now Let AI Design the Workflow



Post-Mortem

What was the goal? Where did you struggle? What corrections did I make?



Agent Proposal

How would you turn this into a repeatable workflow?



Quality Control

Where could this produce bad output? Where do we need guardrails?

```
1 # Foot & Ankle City-Group Page -- Build Log
2
3 **Date:** 2026-02-18
4 **Region:** Foot & Ankle
5 **Slug:** `foot-ankle`
6 **Conditions (4):** Achilles Tendonitis, Heel Spurs, Plantar Fasciitis, Shin Splints
7 **Template:** `src/pages/[cityState]/pain-relief/back-spine/index.astro` (gold standard)
8 **Status:** Staging-gated (`prerender = false`)
9
10 ---
11
12 ## 0. Post-Mortem: How to Build the Next Group Page Faster
13
14 This was the second city-group page (after back-spine). The build took longer than necessary due to
15 plan gaps, image iteration, and component bugs that should now be resolved. The next group page
16 (shoulder-arm, hip-leg, or head-jaw) should be significantly faster.
17
18 ### What the next developer needs to create (3 files only):
19
20 - Copy data file: `src/data/city-group/{region}-copy.ts` -- 3 cities, each with `deckText`,
    `paragraphs` (3), `conditionCards` (one per condition). Paragraph 2 MUST link all conditions with
    `class="text-xanthous hover:text-xanthous/80 underline"`. Triple-check every city has every condition
    linked.
21
22 - Quiz config: Add a `{region}Config: QuizConfig` to `src/data/city-group/quiz-config.ts` and
    register it in `quizConfigByRegion['{region}']`. Test 1 persona path per condition to verify correct
    "Most Likely" scoring.
23
24 - Page file: `src/pages/[cityState]/pain-relief/{region}/index.astro` -- copy back-spine or
    foot-ankle as template. Change: imports, regionSlug, copy import, hero image, diagram image, image
    maps, conditionExtras, FAQs, schema description, availableService name.
```

Agent Failure Is Usually From:

1

Contradictions

Instructions conflict with each other

2

Ambiguity

"Summarize the ticket" means something different every run

3

Inference instead of code

Letting AI guess what should be calculated

4

Jumping the harness

Agent skips steps or takes shortcuts

5

No checkpoints

8 steps in a row, bad data at step 3 ruins everything

 Tip: Ask AI to review your workflow for these specific common failure points.

Let Agents Auto Run After Refining



Webhook

System sends a signal (e.g., new Jira ticket created)



Schedule

Timer fires (e.g., every Friday at 4 PM)



Polling

Agent checks periodically (e.g., every 15 min, check for new invoices)



Message

Someone emails/messages a bot (e.g., invoice to invoices@company.com)

A session agent only needs to be right when you're watching. A long-running agent needs to be right at 3 AM on a Tuesday.

Build An Agent in 5 Steps

1

Define the goal

What specific problem are we trying to solve? (Oracle mode for exploration)

2

Define the context

What data, tools, or APIs needed for an agent? (Tool mode for execution)

3

Manually go through the process

Do the entire process with your LLM. Step by step until you get a good result.

4

Post Mortem (Refine Process)

Ask: Now that we are here, how could we get here faster and more efficiently?

5

Build The Agent

Develop this workflow as an Agentic Loop with a final post mortem step to evaluate what can be done better.

Hand hold first few iterations until you're confident in outcome. I recommend Claude Web with Opus 4.6 for designing agents.

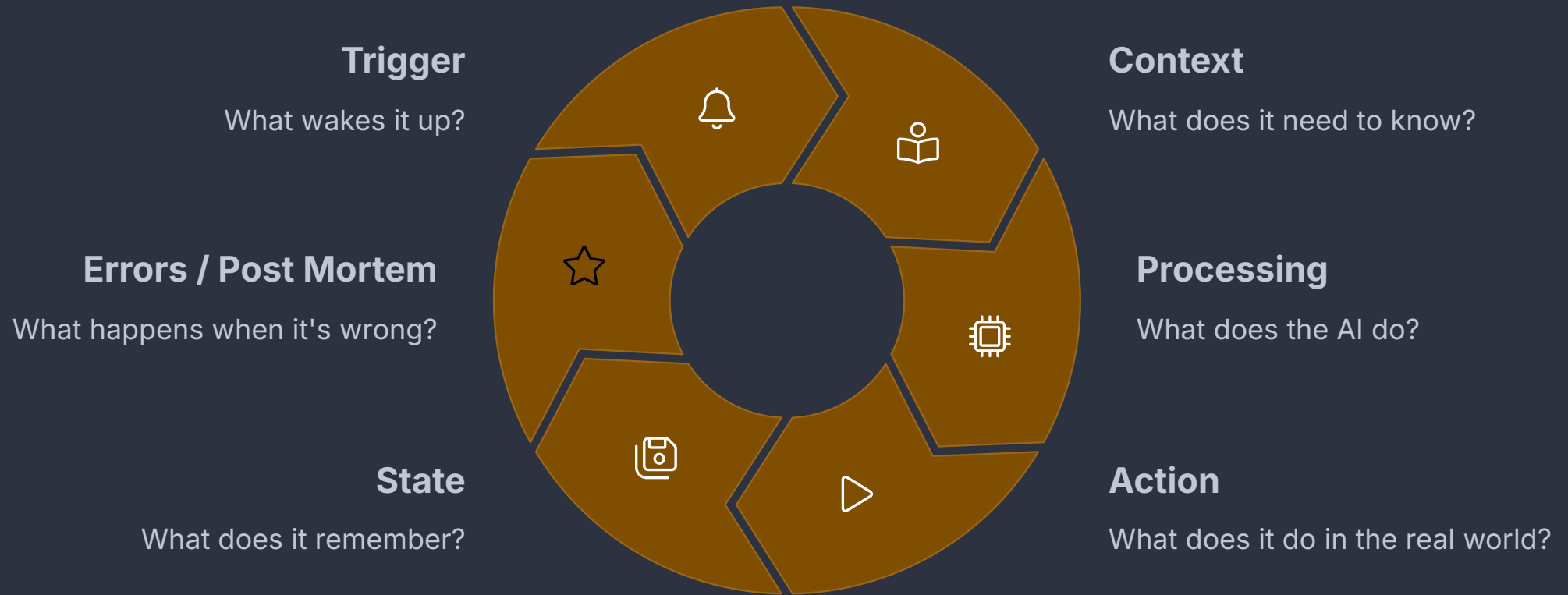
💡 BIG IDEA 3

Get Better Over Time

Solve Memory. Conquer The World.

Converting AI from stateless into an autonomous agent

6 Parts of Long-Running Agents



How Agents Learn



State Logging

- What inputs it received
- What it did
- What it produced
- What was accepted or corrected



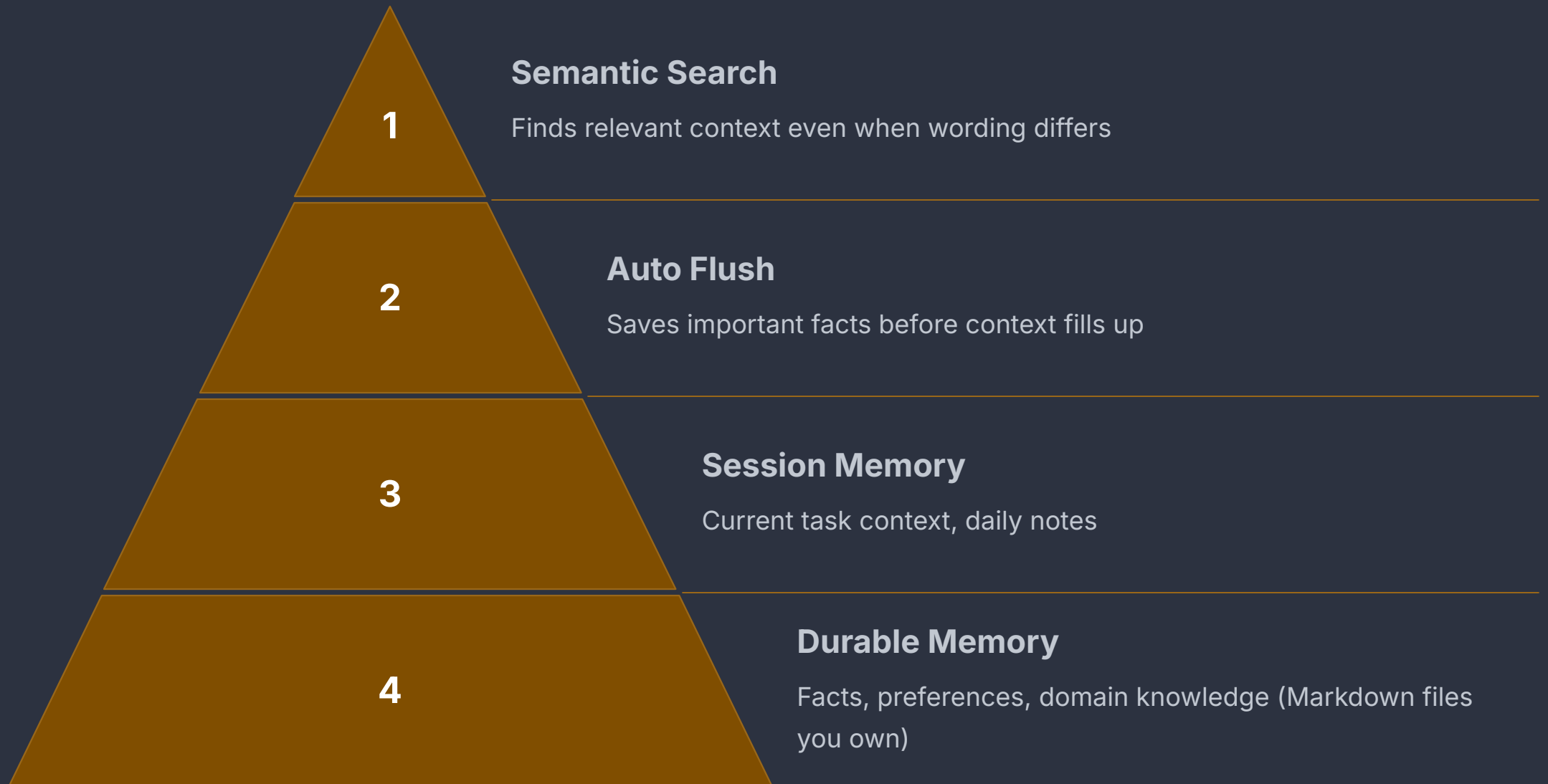
Automatic Post-Mortem

- What went wrong?
- What could be better?
- How do I update my instructions?

The corrections become the training. The agent gets better without you rebuilding it.

Memory Is Everything — OpenClaw and 2026

Bigger models. Faster inference. Better context. Deeper context windows.



5 Levels of Agency

1

Level 1: Manual

Chat session, start to finish

2

Level 2: Reusable

Saved prompts or Custom GPT

3

Level 3: Triggered

Runs when an event happens

4

Level 4: Scheduled

Runs on a clock

5

Level 5: Adaptive

Learns from corrections

Three Ideas. One Principle.

Trustable AI output

Build rubrics. Cross-check. Never eyeball it.

Build agents through iteration

Unstructured → structured. Manual first. Let AI design the agent.

Get better over time

Post-mortem loop. Memory is everything.

Give AI what good looks like — context over questions, always.

Your Challenge This Week

- 1 Pick ONE repetitive task
- 2 Do it with AI — give it real context (Tool mode)
- 3 Build a rubric and/or cross-check process with a second tool
- 4 Ask AI for a post-mortem: what worked, what didn't, what to save

Questions?

What from today surprised you?

What are you most skeptical about?