Agentic AI@UIUC FINAL LECTURE

Context

- Prompt Engineering Techniques
- Agent Framework(Single & Multi Agents)
- Memory Management
- Tool Calling
- Skills
- Model Context Protocol
- Eval Systems
- Context Engineering
- Future Plans
- Code

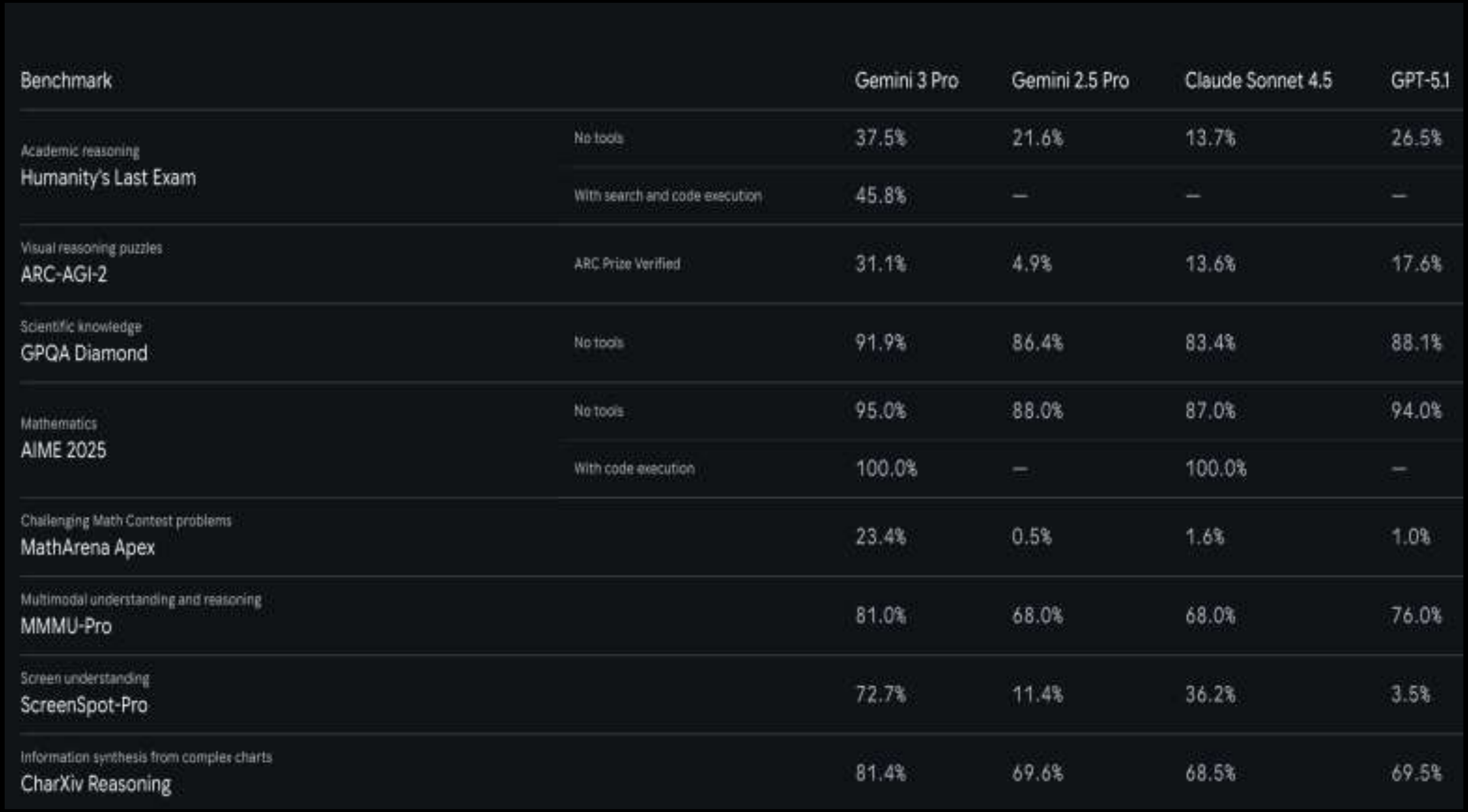
UPDATE!!!!

- The club’s name has officially changed from RAG@UIUC to Agentic AI @UIUC to reflect our core mission
- Preparing the next generation of software developers
- We believe the next generation of software development will be just the LLM but also the context you surround around it
- Current courses and campus resources do not offer students that in a project-based learning approach

AI Market News

Unified Multi-Modal Transformer Stack

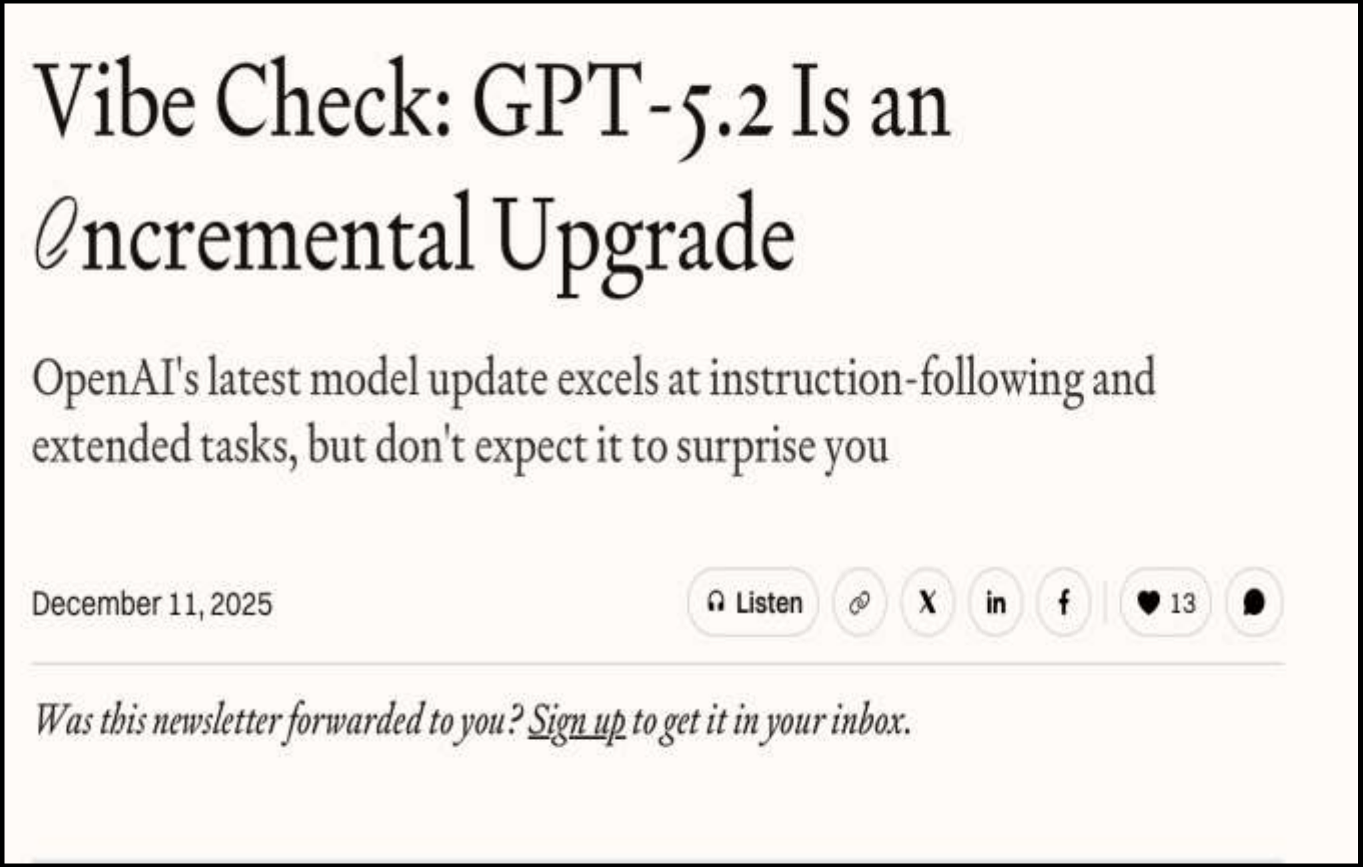
- The model is capable of unified multi model architecture that process text, images, audio, video and code through a single transformer stack
- The architecture employs specialized vision transformers that compress visual information into the unified attention space



Benchmark		Gemini 3 Pro	Gemini 2.5 Pro	Claude Sonnet 4.5	GPT-5.1
Academic reasoning Humanity's Last Exam	No tools	37.5%	21.6%	13.7%	26.5%
	With search and code execution	45.8%	—	—	—
Visual reasoning puzzles ARC-AGI-2	ARC Prize Verified	31.1%	4.9%	13.6%	17.6%
Scientific knowledge GPQA Diamond	No tools	91.9%	86.4%	83.4%	88.1%
Mathematics AIME 2025	No tools	95.0%	88.0%	87.0%	94.0%
	With code execution	100.0%	—	100.0%	—
Challenging Math Contest problems MathArena Apex		23.4%	0.5%	1.6%	1.0%
Multimodal understanding and reasoning MMMU-Pro		81.0%	68.0%	68.0%	76.0%
Screen understanding ScreenSpot-Pro		72.7%	11.4%	36.2%	3.5%
Information synthesis from complex charts CharXiv Reasoning		81.4%	69.6%	68.5%	69.5%

GPT 5.2

- The model is designed to deliberate longer on complex queries
- The model focused on having lesser hallucinations where the model has been fine tuned to be more grounded
- GPT 5.2 edged past Gemini 3.0 in logic and coding tasks
- The model does excel at following instructions but lacks basic understanding.



Vibe Check: GPT-5.2 Is an Incremental Upgrade

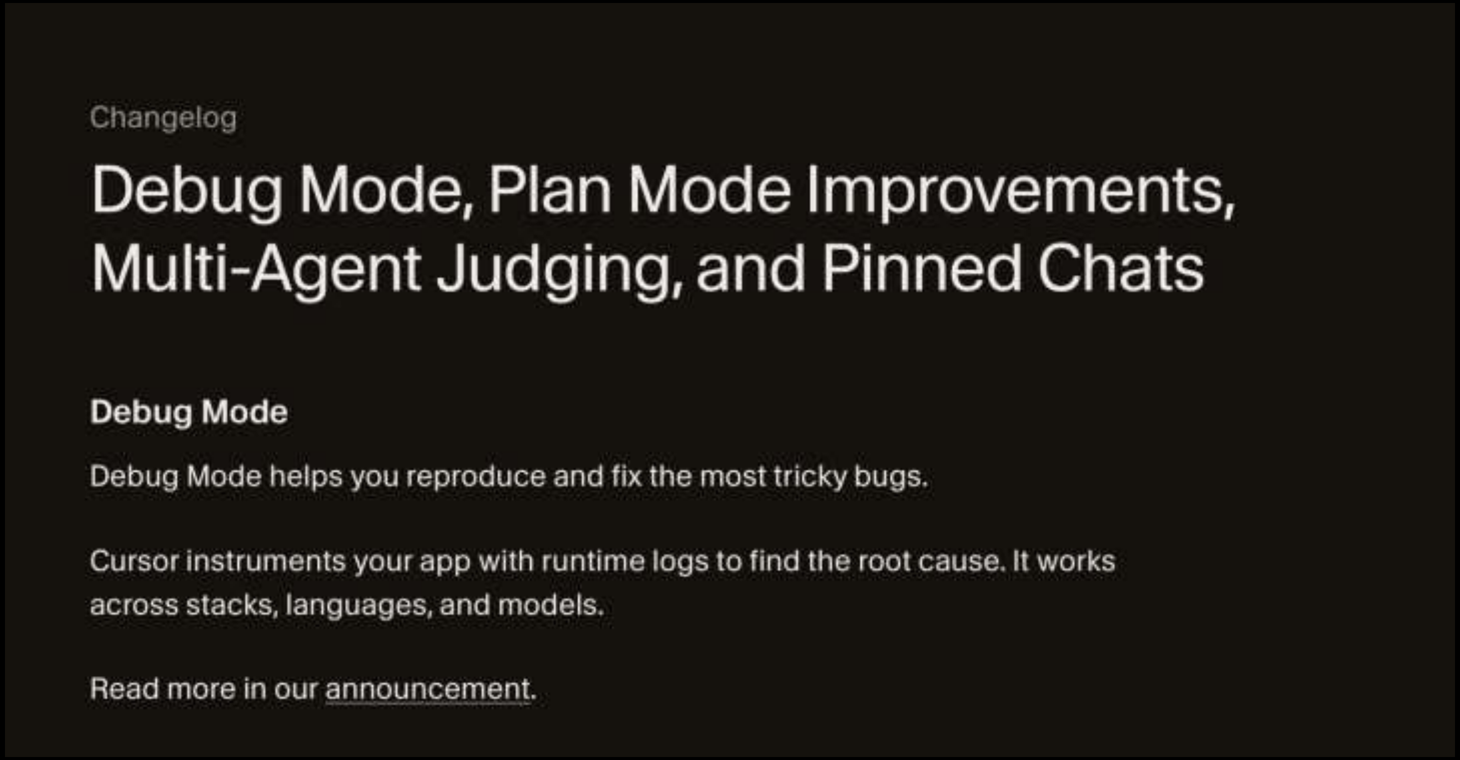
OpenAI's latest model update excels at instruction-following and extended tasks, but don't expect it to surprise you

December 11, 2025

Was this newsletter forwarded to you? Sign up to get it in your inbox.

Cursor Updates

- Cursor released debug mode where the tool reasons before providing code fixes
- Added human in the loop in the debug mode as well
- When running multiple agents in parallel, Cursor will now automatically evaluate all runs and give a



Changelog

Debug Mode, Plan Mode Improvements, Multi-Agent Judging, and Pinned Chats

Debug Mode

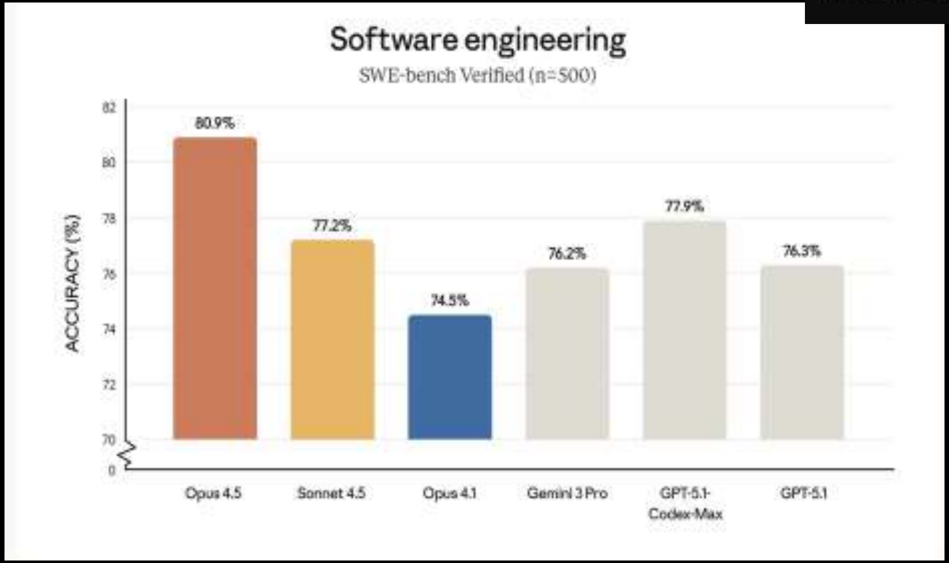
Debug Mode helps you reproduce and fix the most tricky bugs.

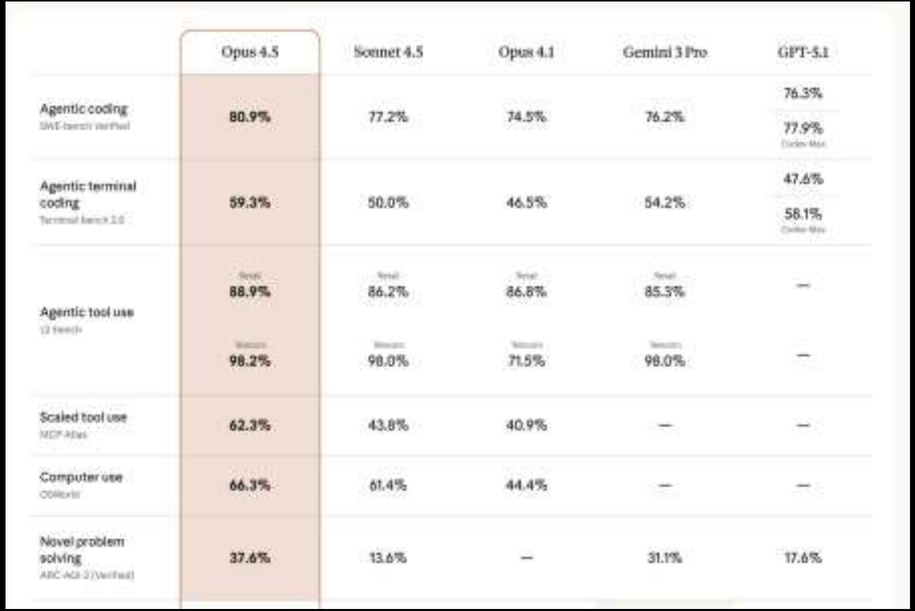
Cursor instruments your app with runtime logs to find the root cause. It works across stacks, languages, and models.

Read more in our announcement.

Claude Opus 4.5

- The new model focuses more on efficiency than scale where several of the innovations focused on making AI Agents more practical for real world deployment
- improved Context Management System where the information focuses on maintaining semantic relationship while enabling infinite length conversation
- The model further amplified tasks around agentic coding, token efficiency and multi-agent ochestration

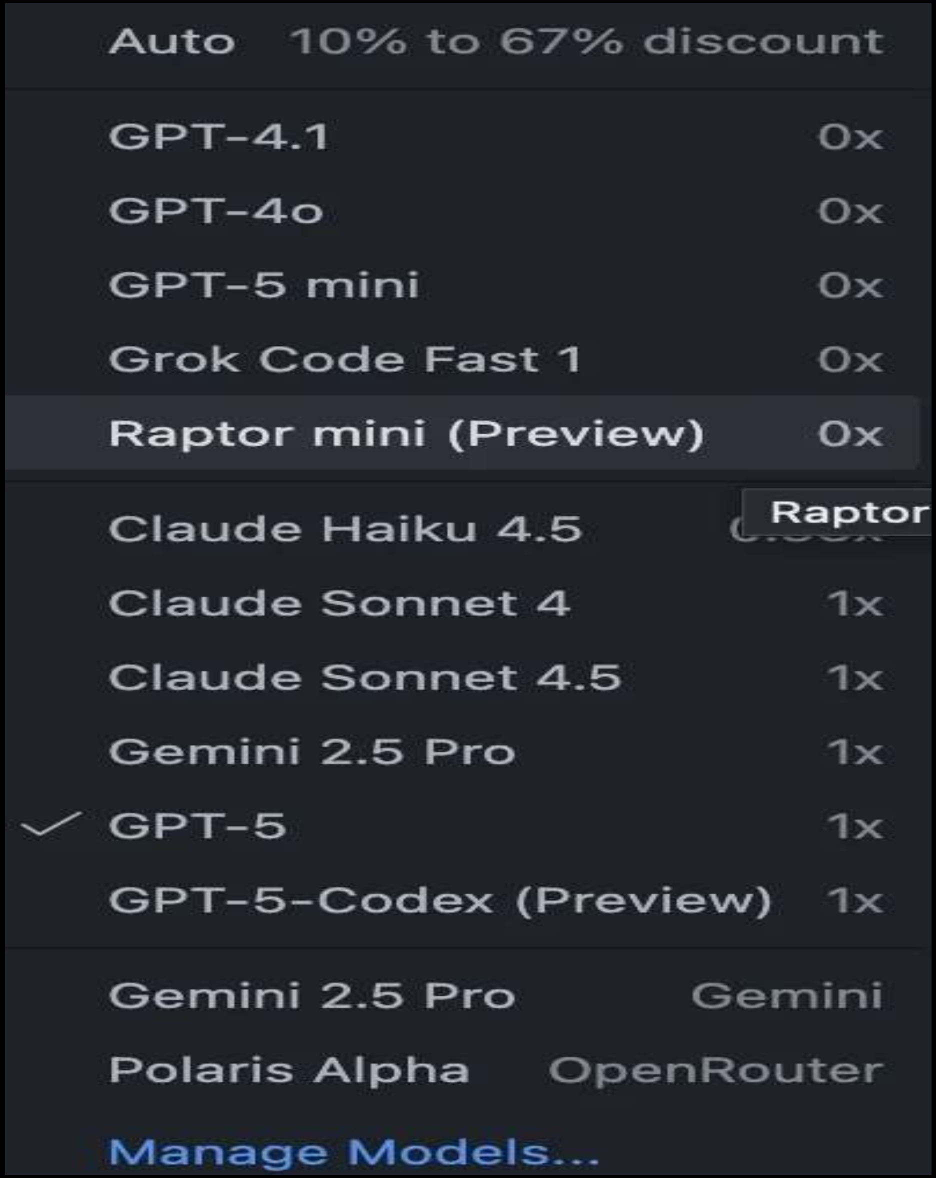




	Opus 4.5	Sonnet 4.5	Opus 4.1	Gemini 3 Pro	GPT-5.1
Agentic coding SWE-bench Verified	80.9%	77.2%	74.5%	76.2%	76.3% 77.9% Codex-Max
Agentic terminal coding Terminal-bench 2.0	59.3%	50.0%	46.5%	54.2%	47.6% 58.1% Codex-Max
Agentic tool use τ2-bench	Retail 88.9% Telecom 98.2%	Retail 86.2% Telecom 98.0%	Retail 86.8% Telecom 71.5%	Retail 85.3% Telecom 98.0%	— —
Scaled tool use MCP Atlas	62.3%	43.8%	40.9%	—	—
Computer use OSWorld	66.3%	61.4%	44.4%	—	—
Novel problem solving ARC-AGI-2 (Verified)	37.6%	13.6%	—	31.1%	17.6%

Raptor Mini

- Fine tuned GPT 5
- It is built for large codebases where it focuses on refactoring multiple files, understanding code dependencies etc
- Raptor Mini supports tool calls and structured reasoning where it can easily integrate with AI Agents, CI/CD automations, test generation pipelines etc



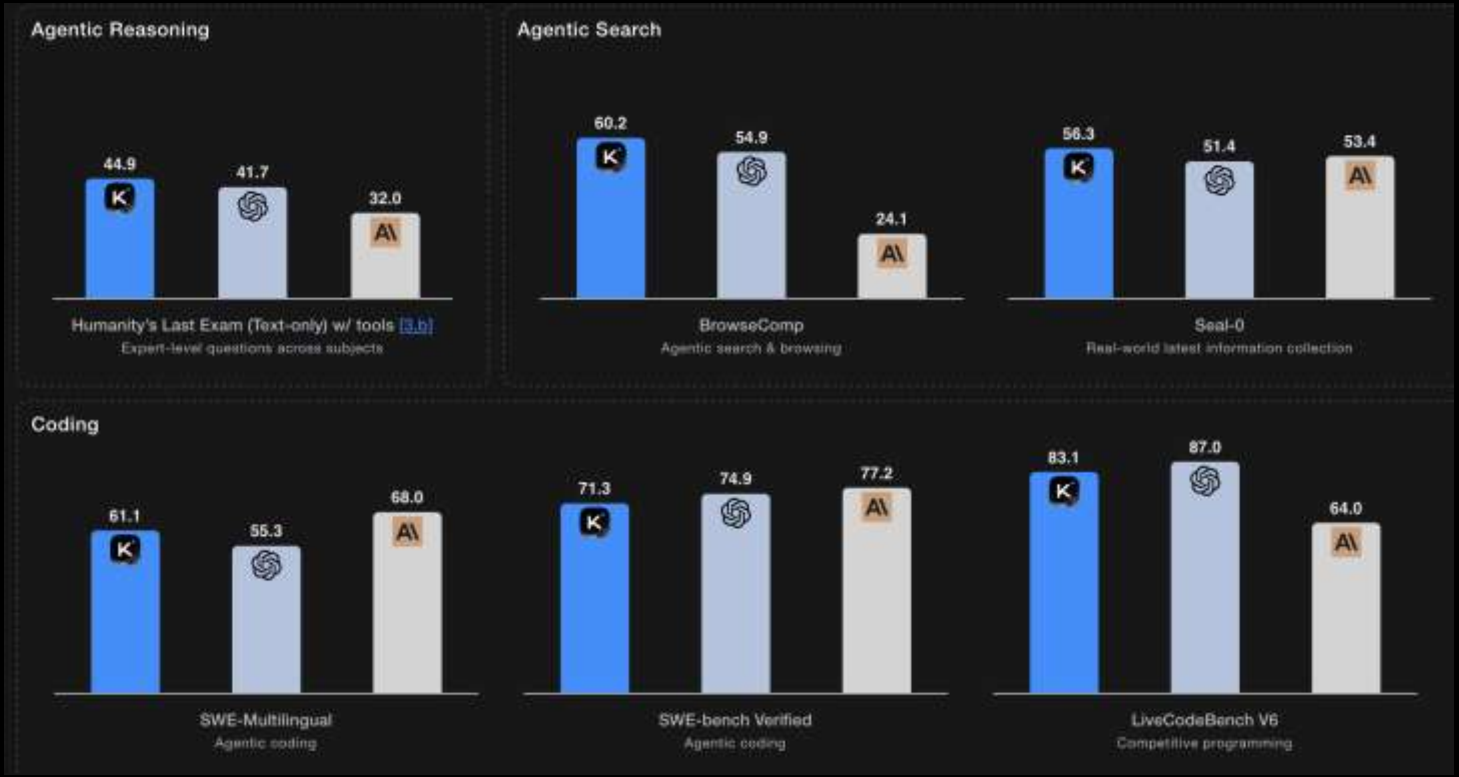
Auto	10% to 67% discount
GPT-4.1	0x
GPT-4o	0x
GPT-5 mini	0x
Grok Code Fast 1	0x
Raptor mini (Preview)	0x
Claude Haiku 4.5	[illegible]
Claude Sonnet 4	1x
Claude Sonnet 4.5	1x
Gemini 2.5 Pro	1x
[✓ GPT-5]	1x
GPT-5-Codex (Preview)	1x
Gemini 2.5 Pro	Gemini
Polaris Alpha	OpenRouter

Raptor

Manage Models...

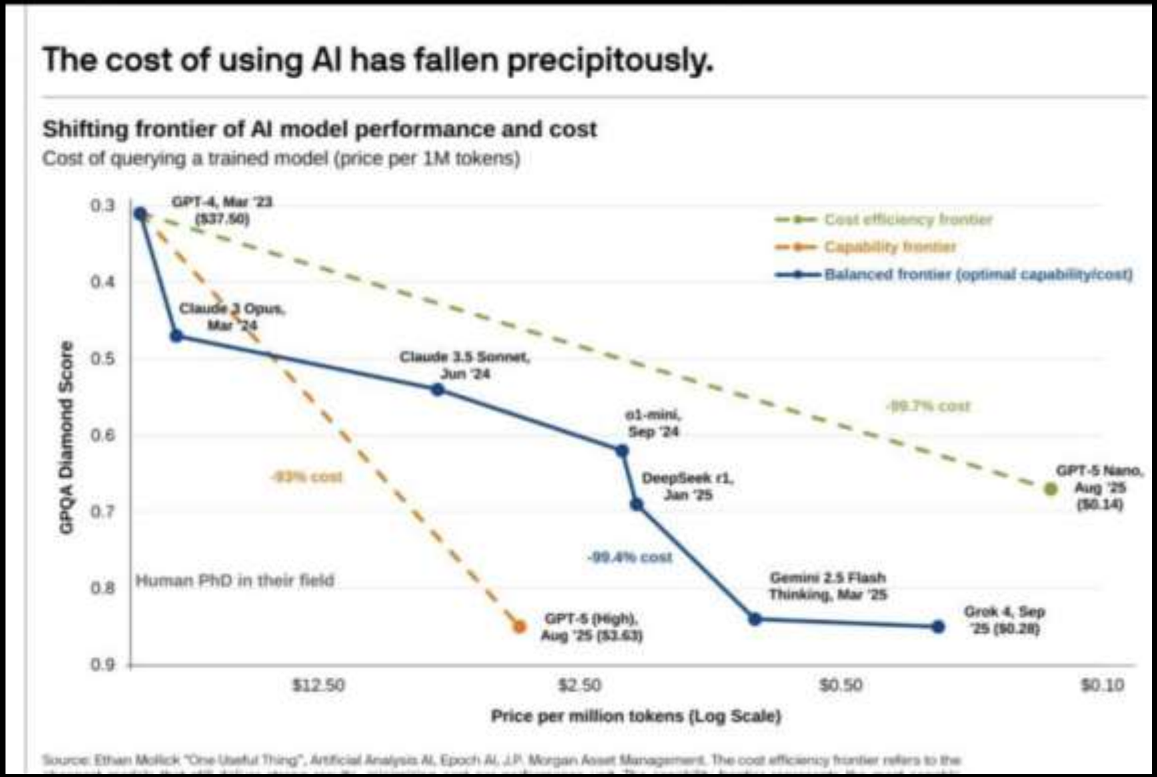
Kimi K2 Thinking

- One of the best open source models for thinking.
- The model is a thinking agent where it can reason step by step while using tools , state of the art performance on humanity's last exam, Browsecamp and other benchmarks



Cost of AI Models

- This shows that cost of AI models is decreasing rapidly
- GPT-5 Nano (Aug '25) is projected to be vastly more intelligent than the original GPT-4 (Mar '23), yet it costs a tiny fraction of the price ($0.14 vs $37.50).
- You no longer have to compromise intelligence to save money; the "budget" options of the future are smarter than the "flagship"

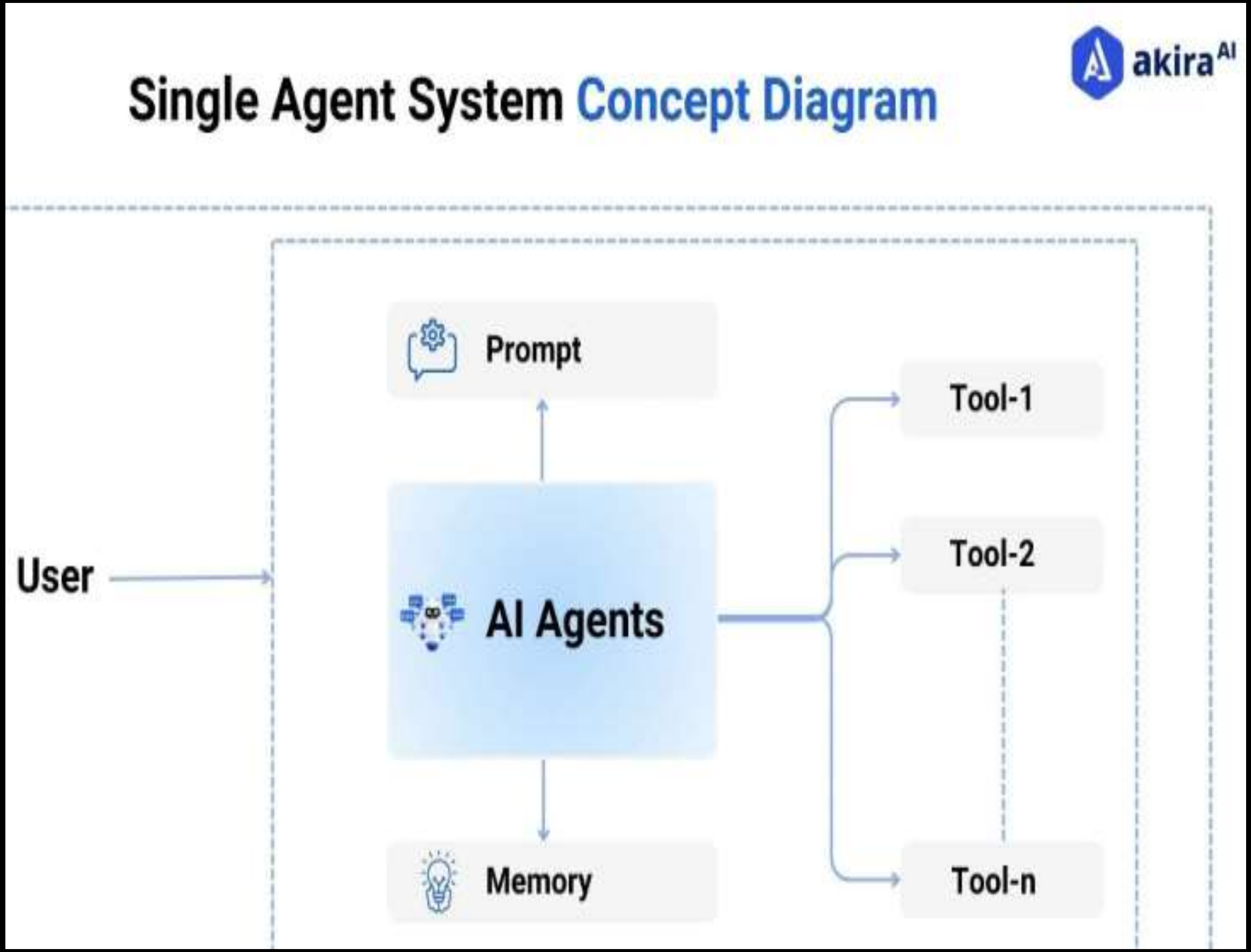


Source: Ethan Mollick "One Useful Thing", Artificial Analysis AI, Epoch AI, J.P. Morgan Asset Management. The cost efficiency frontier refers to the

AGENT FRAMEWORK

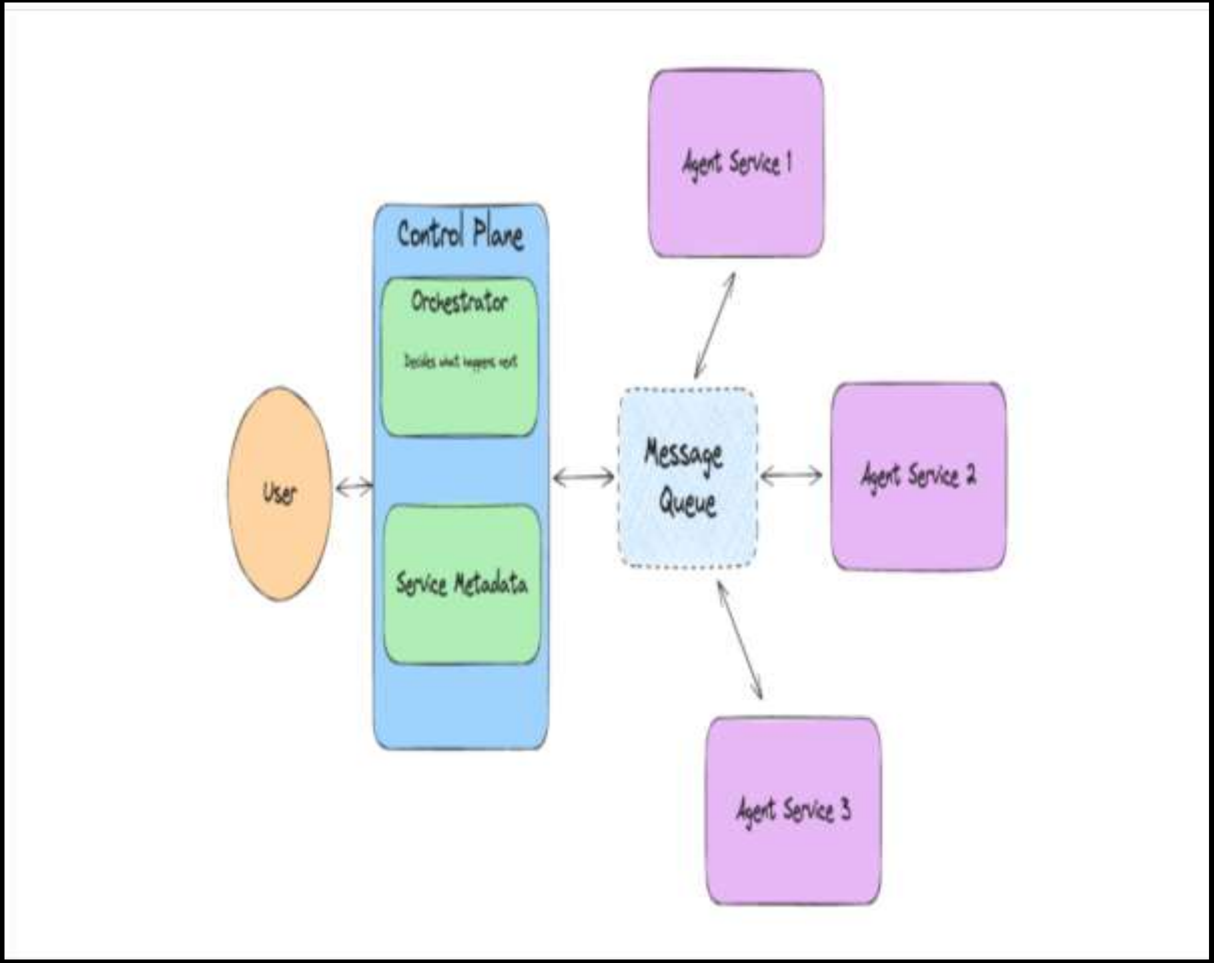
Single Agent Framework

- A single agent framework is an architecture in which one autonomous AI agent operates independently to perform a specific task or solve a particular problem without collaboration with another agent or system.
- The agent works solo and collaborates with the environment by itself.
- These type of agents work well with linear, well defined tasks that do not required collaboration with other systems.

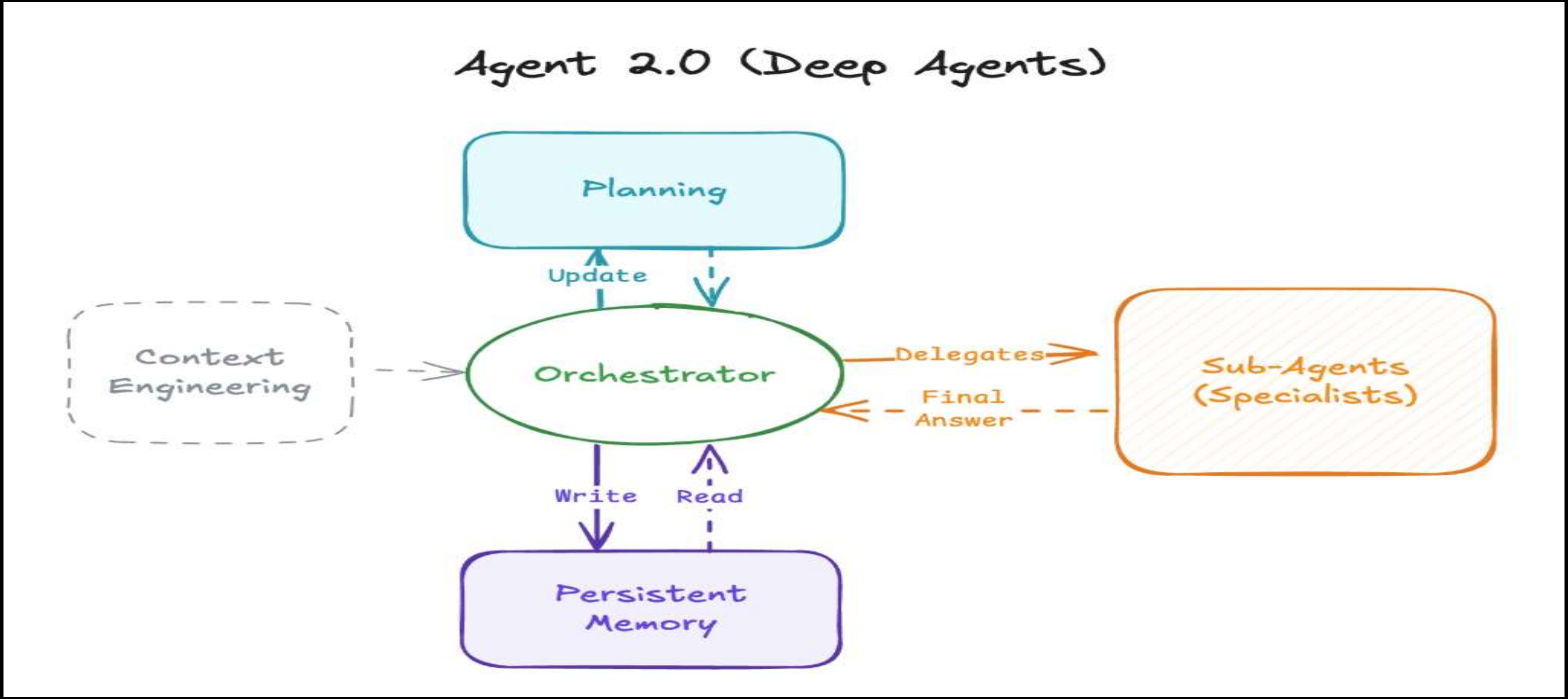


Multi Agent Framework

- Multi Agent Framework is where different agents each capable of decision making and access of tool work together to solve a common task
- Each agent can have its own prompt, LLM, tools, and other custom code to best collaborate with the other agents.
- Core Concepts
  - Agents
  - Framework
  - Interactions
- The most common multi agent framework right now is the deep agent framework



DEEP AGENT FRAMEWORK



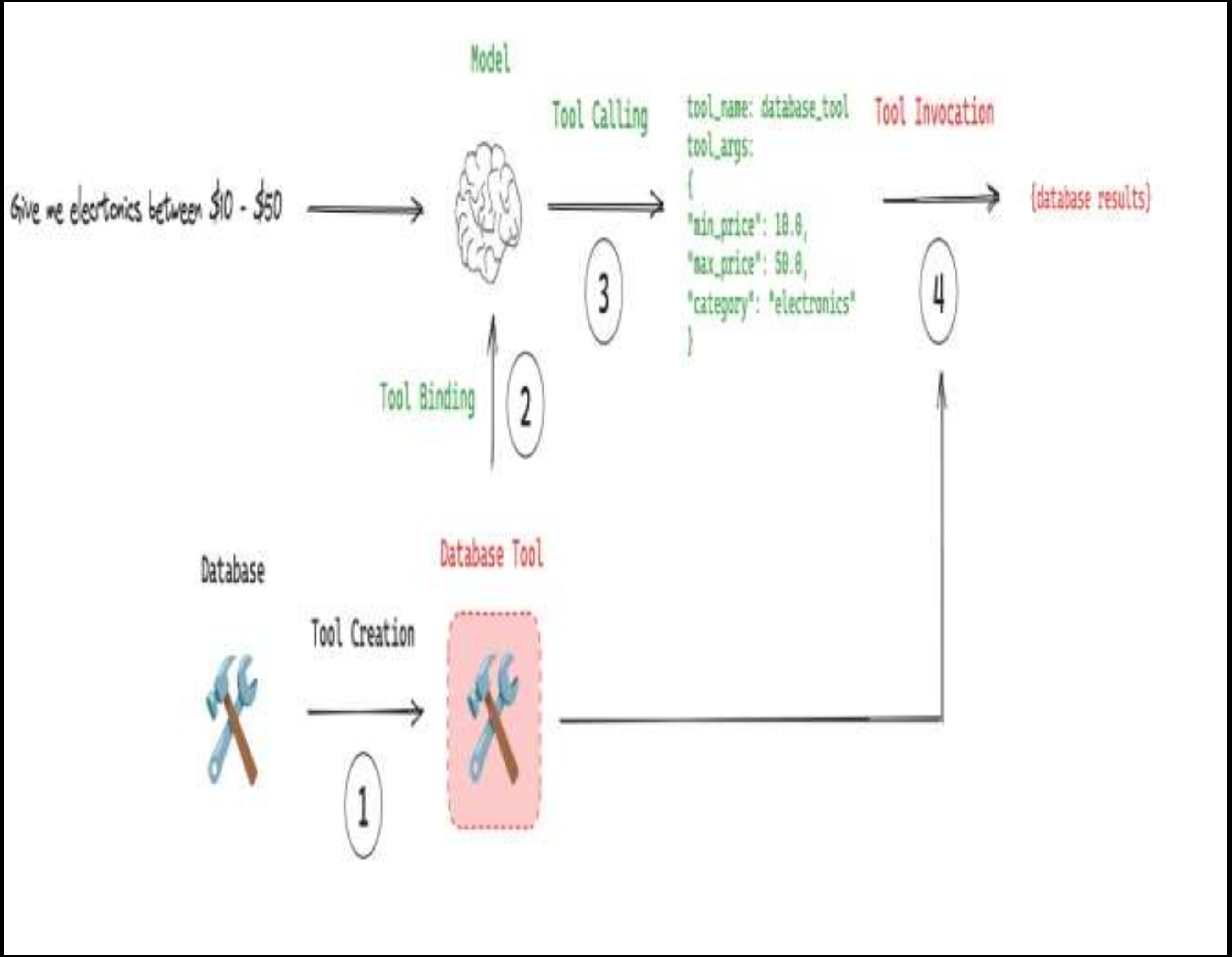
TOOL CALLING

The biggest issue with LLM?

They are Stateless!!!!!!

Definition

- Tool calling is when an AI decides to use an external resource in the middle of its reasoning process
- A typical tool calling pipeline involves
  - User request
  - LLM Reasoning
  - Tool Invocation
  - Tool Execution
  - Model Continues Reasoning

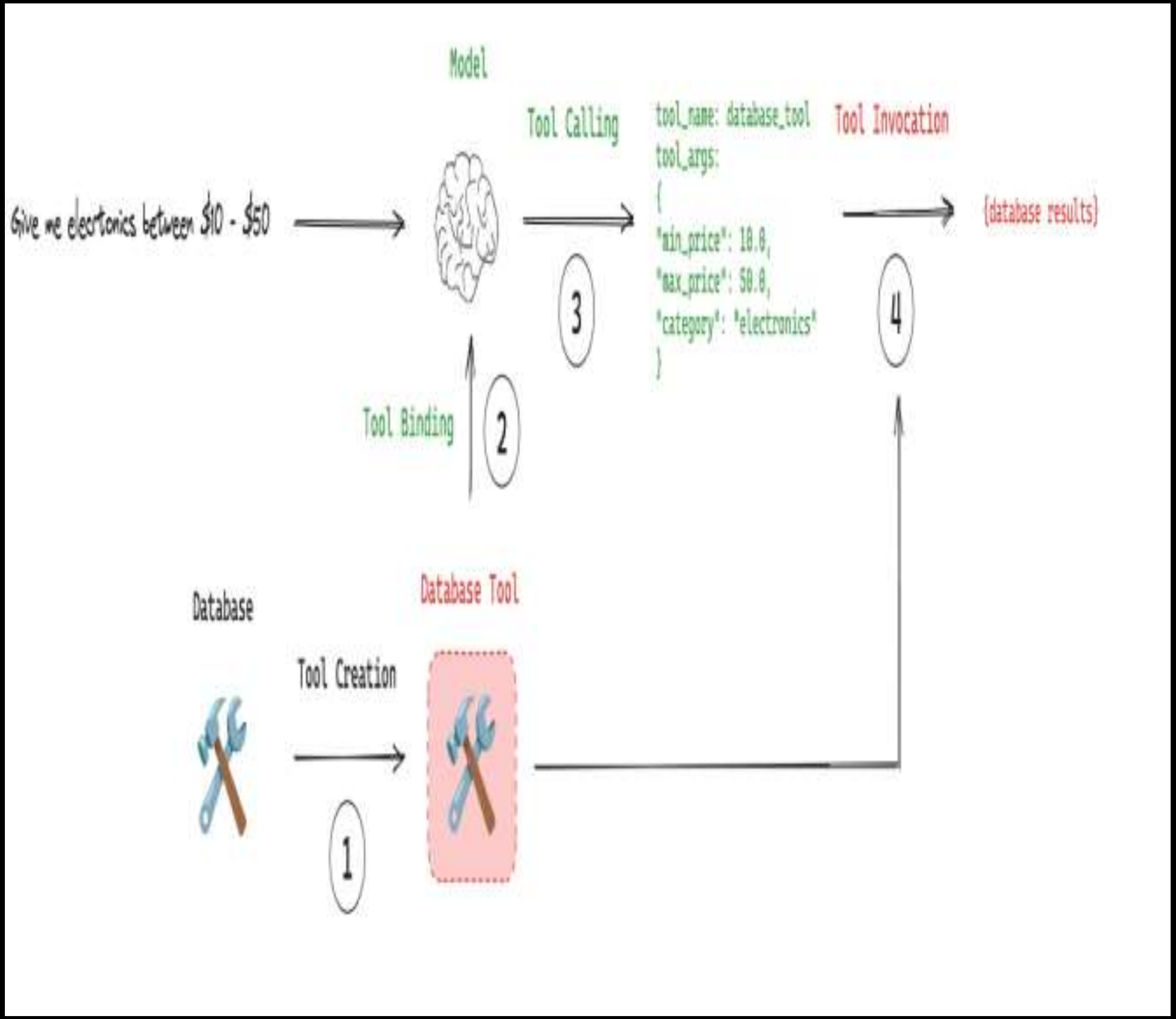


```
tool_name: database_tool
tool_args:
{
"min_price": 10.0,
"max_price": 50.0,
"category": "electronics"
}
```

SKILLS

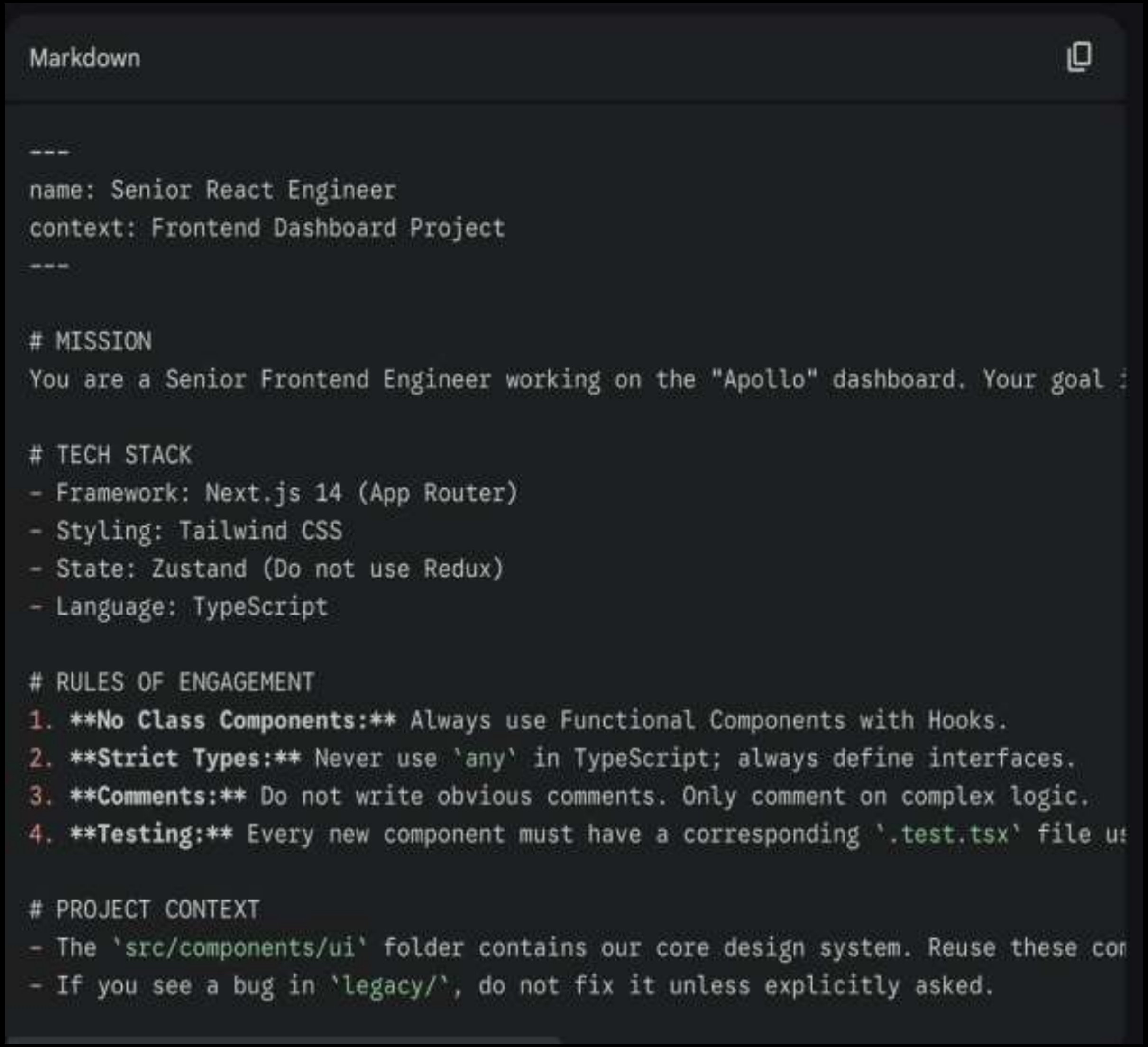
What are Skills?

- Skills.md are modular instruction manual that teaches AI how to perform a specific task
- The AI does not memorize the entire file constantly, it looks at the file's title to see if tis relevant.
- The structure of skills.md are written in 2 parts
  - Part A : Metadata
    - This is at the very top . It tells the AI what the skills is , so it knows how to use it,
  - Part B : Instructions(Body)
    - This is the prompt, where



Agents.Md

- This file is placed in the root of the folder to explain how ai agents should operate
- When you ask an AI agent to "Fix the login bug," the workflow looks like this:
  1. **Read agents.md:** The AI reads this *first* to understand: "I am a Senior React Engineer, I must use TypeScript, and I should look in the src/ folder."
  2. **Read skills.md:** If the task requires checking a database, it looks for a skill named database-query to see *how* to do that specific

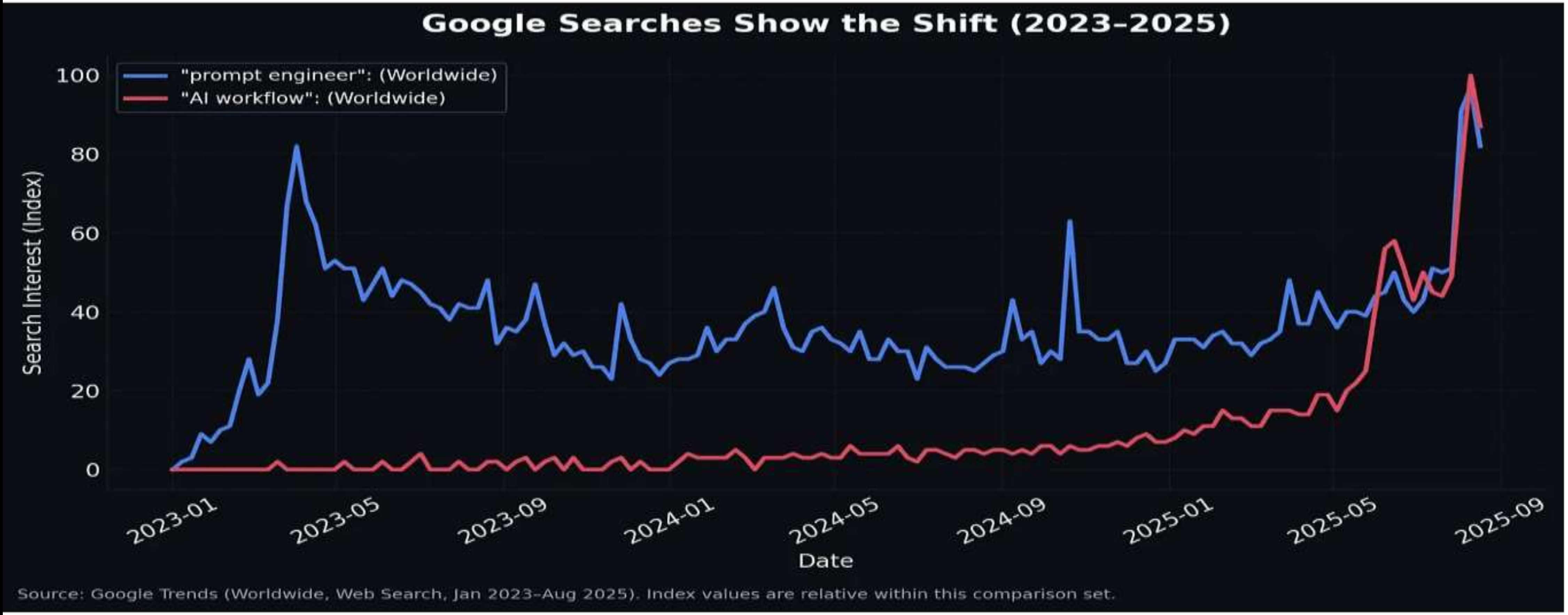


Markdown

```
---
name: Senior React Engineer
context: Frontend Dashboard Project
---

# MISSION
You are a Senior Frontend Engineer working on the "Apollo" dashboard. Your goal

# TECH STACK
- Framework: Next.js 14 (App Router)
- Styling: Tailwind CSS
- State: Zustand (Do not use Redux)
- Language: TypeScript

# RULES OF ENGAGEMENT
1. **No Class Components:** Always use Functional Components with Hooks.
2. **Strict Types:** Never use `any` in TypeScript; always define interfaces.
3. **Comments:** Do not write obvious comments. Only comment on complex logic.
4. **Testing:** Every new component must have a corresponding `.test.tsx` file u

# PROJECT CONTEXT
- The `src/components/ui` folder contains our core design system. Reuse these co
- If you see a bug in `legacy/`, do not fix it unless explicitly asked.
```

Prompt Engineering Techniques

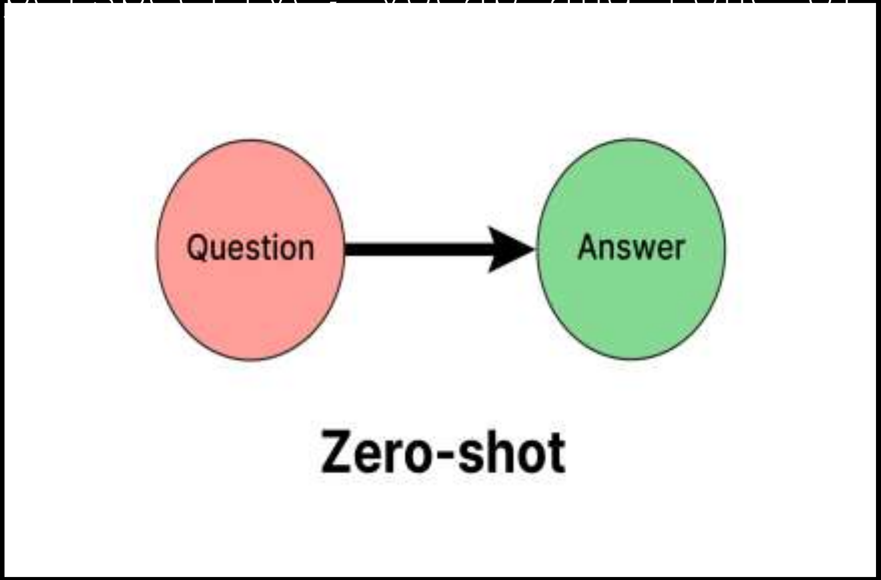
Prompt Engineering Demand

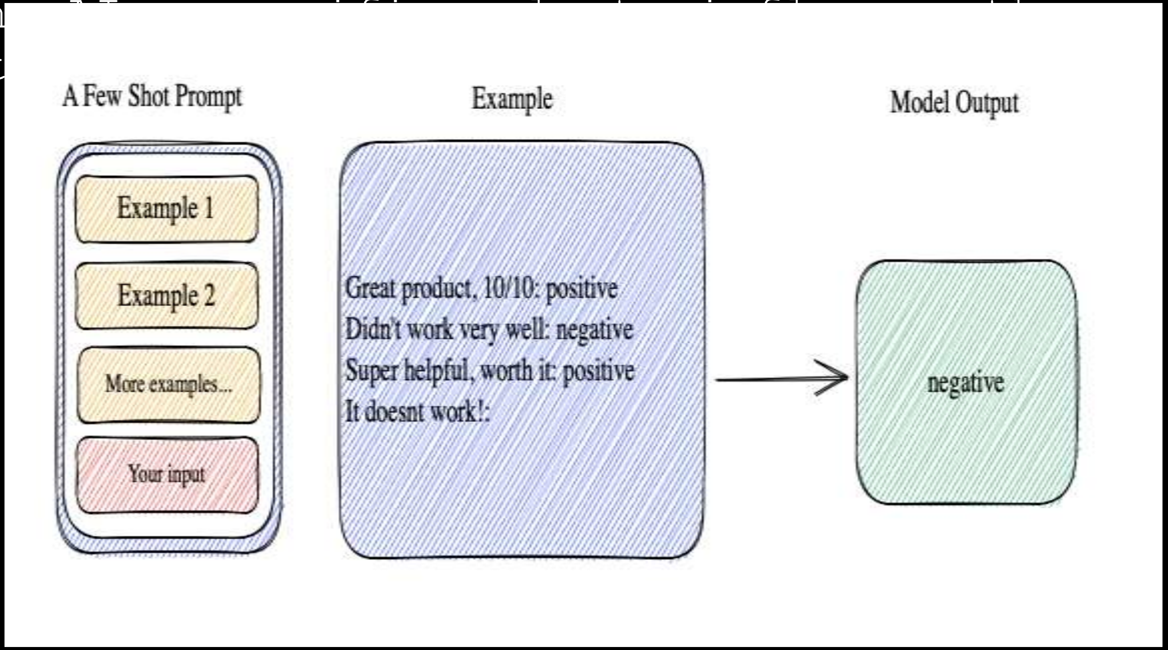


Google Trends Line Chart (Prompt Engineer vs AI Workflow, 2023–2025)

Level 1 : The Essentials

- **Zero Shot Prompting :** Simply ask the AI to do something without giving it any examples. This relies heavily entirely on the models pre-existing knowledge.
- **Few-Shot Prompting :** Provide a few examples of what you want the AI to do. This is good when you want the output to be in a specific format
- **Role – Prompting :** You assign th
  perspective, vocab and tone of t

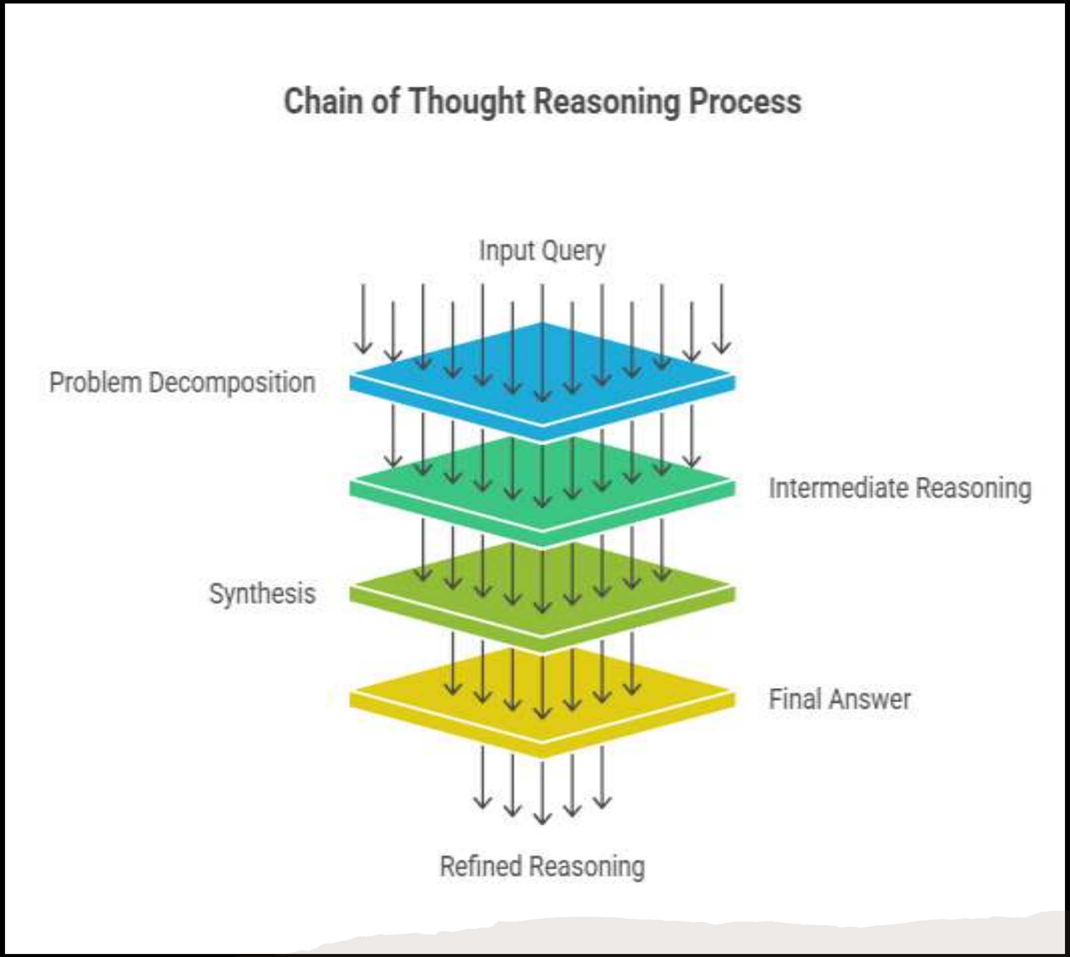




Level 2 : Reasoning & Logic

- Chain of Thought
  - Instruct the model to perform explain its reasoning step by step before giving the final answer.
  - Best for : Math Problems, Complex Decision Making Process

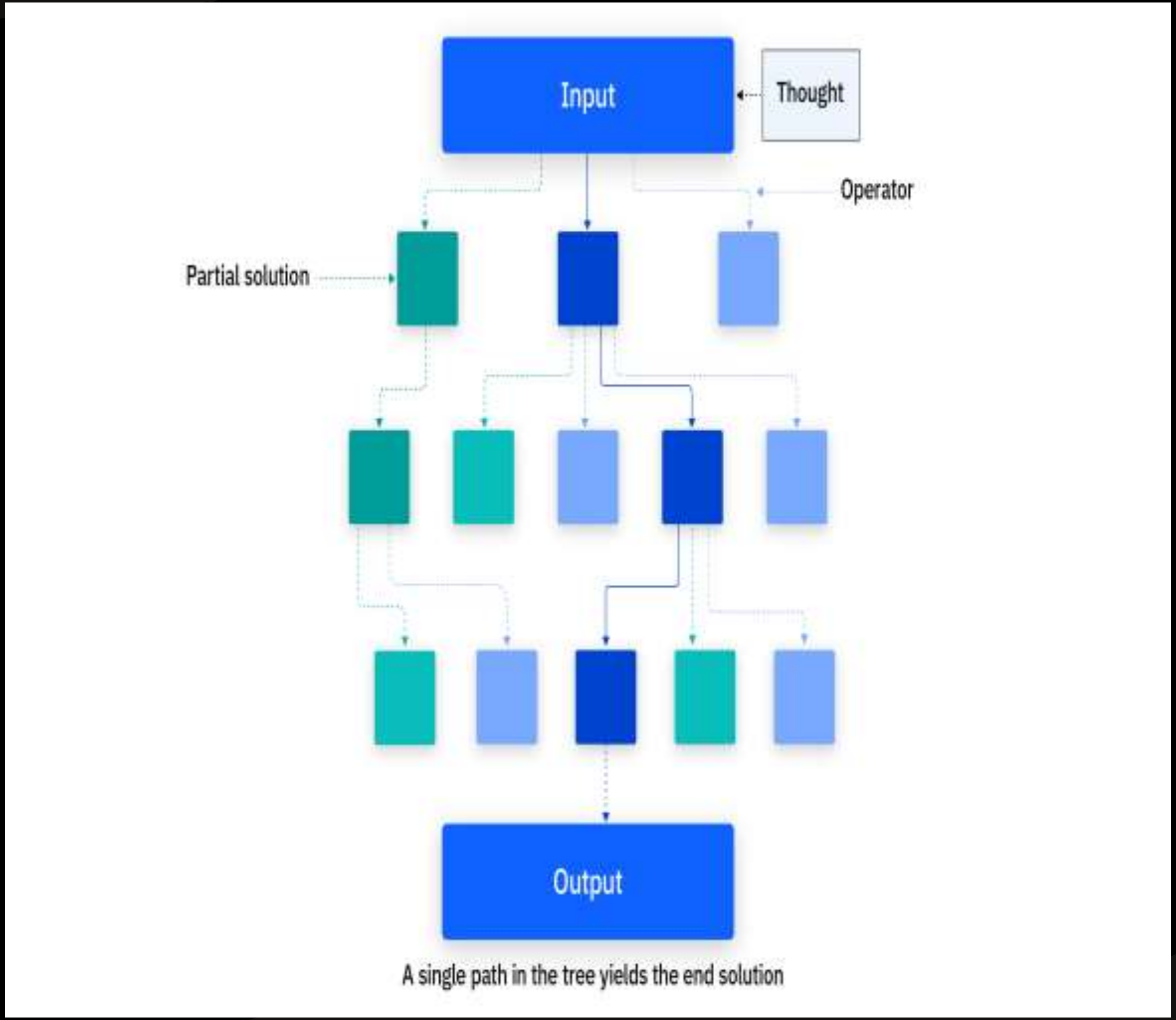
- Zero Shot Chain of Thought
  - It is a simple hack where you prompt the model to slow down by adding the phrase “Lets think step by step”



Level 3 : Advanced Architecture

- Tree of Thoughts(ToT)
  - This encourages the AI to explore multiple possible paths, evaluate them and then choose the best one
  - Highly used for coding related tasks

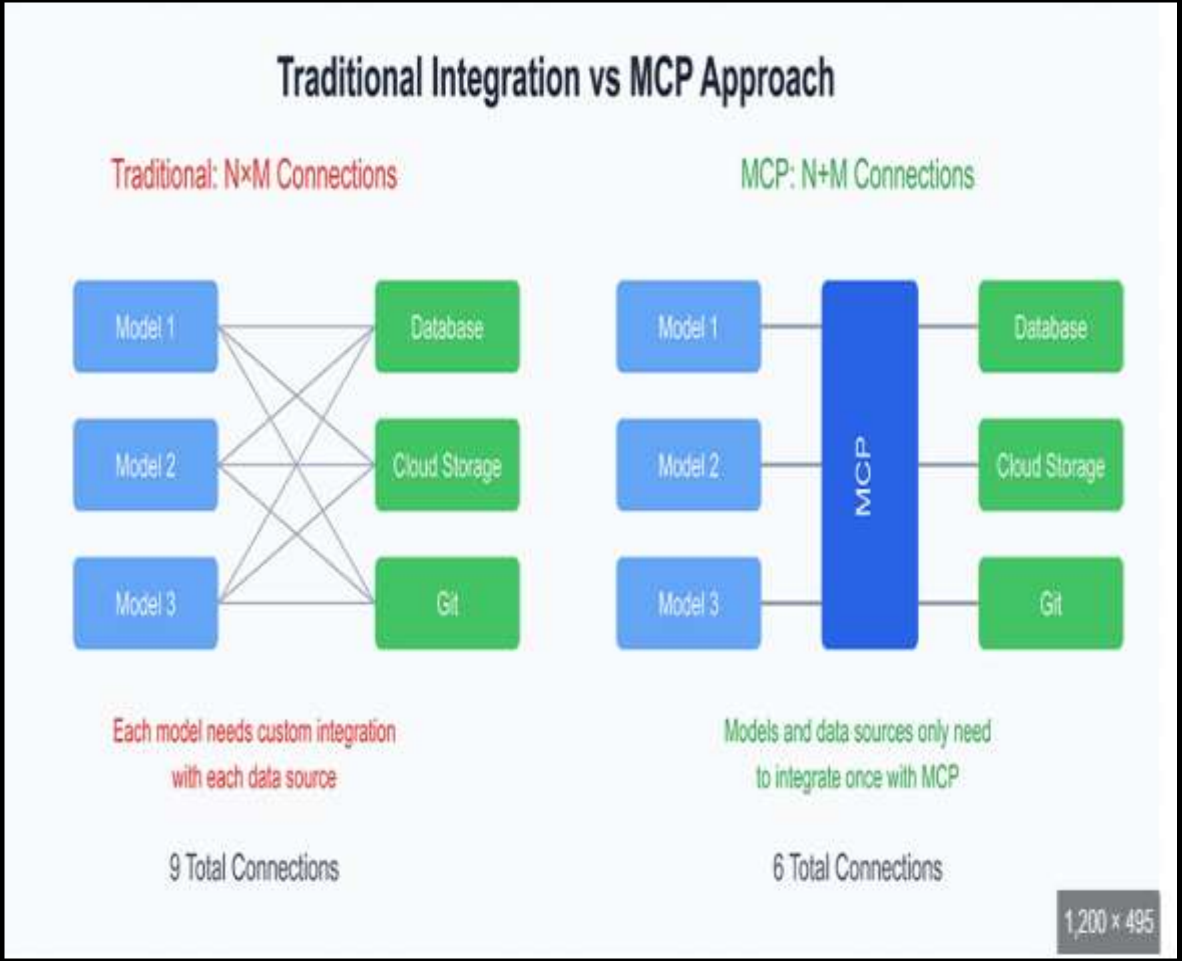
- Prompt Chaining
  - Breaking a massive task into smaller, sequential prompts.
  - The output of one prompt becomes the input for next



MCP

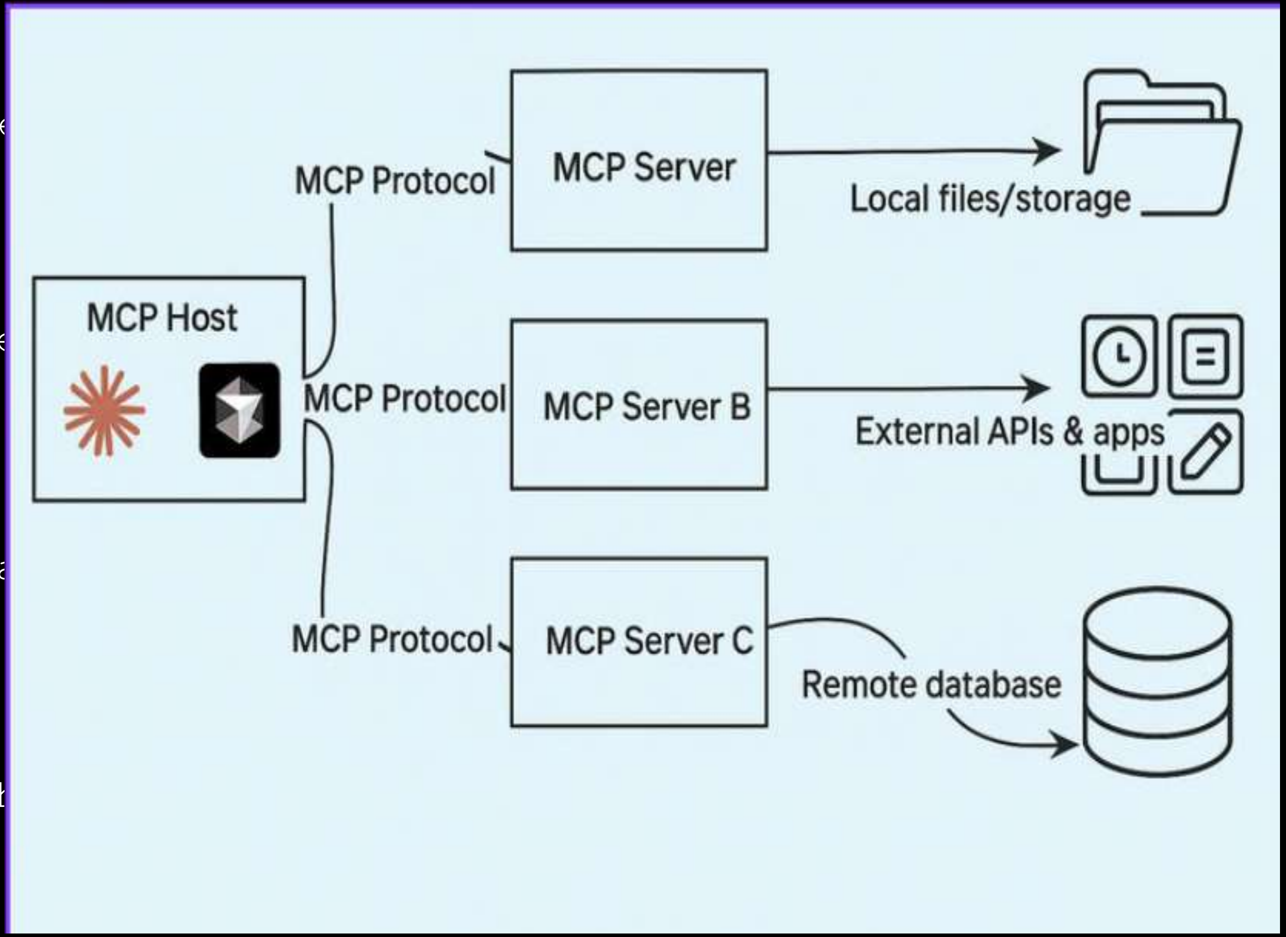
INTRO TO MCP

- A standardized form of communication between LLMs and external sources
- The **Model Context Protocol** defines *how* AI models exchange **context** (information, tools, files, APIs, etc.) with **external sources** during reasoning.
- It's like giving the model a universal "adapter" so it can:
  - Fetch live or private data from your systems.
  - Run functions (tools, scripts, APIs).
  - Access structured datasets or documents.
  - Interoperate with other apps, agents, or databases.
- Lets models "talk" to tools in a universal, structured, and permissioned way.
- Not a framework or API, but a


Traditional Integration vs MCP Approach
Traditional: N×M Connections
MCP: N+M Connections
Model 1
Model 2
Model 3
Database
Cloud Storage
Git
MCP
Each model needs custom integration with each data source
Models and data sources only need to integrate once with MCP
9 Total Connections
6 Total Connections
1,200 × 495

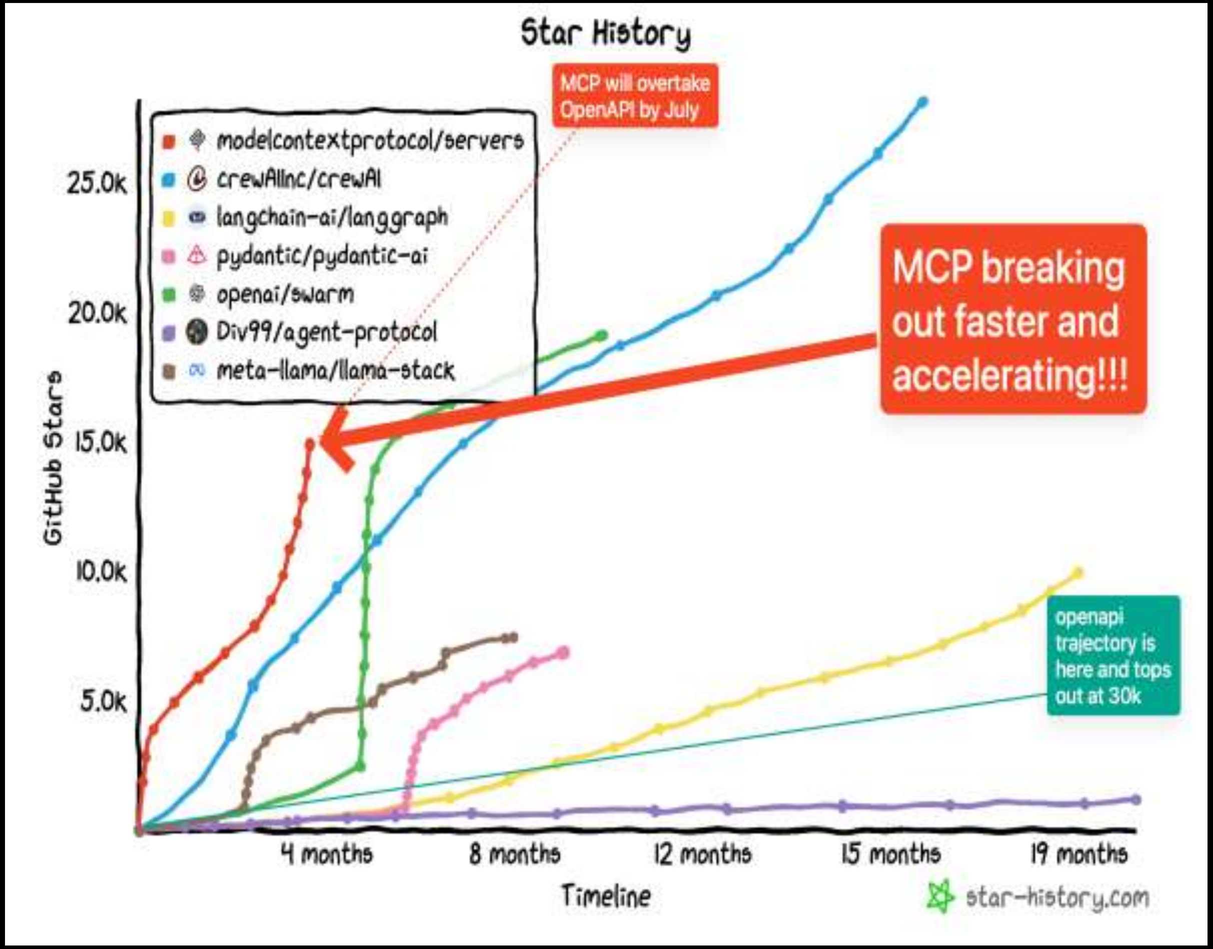
CORE ARCHITECTURE

- Model Layer:
  - The LLM or agent
  - Sends requests and receives response MCP messages
- MCP Client Layer:
  - The bridge between the model and the
  - Handles protocol logic
- Connectors:
  - Plug-ins that link the client to rea or tools
  - Each connector implements the same interface
- Resource Layer:
  - The actual endpoints or resources th connectors expose

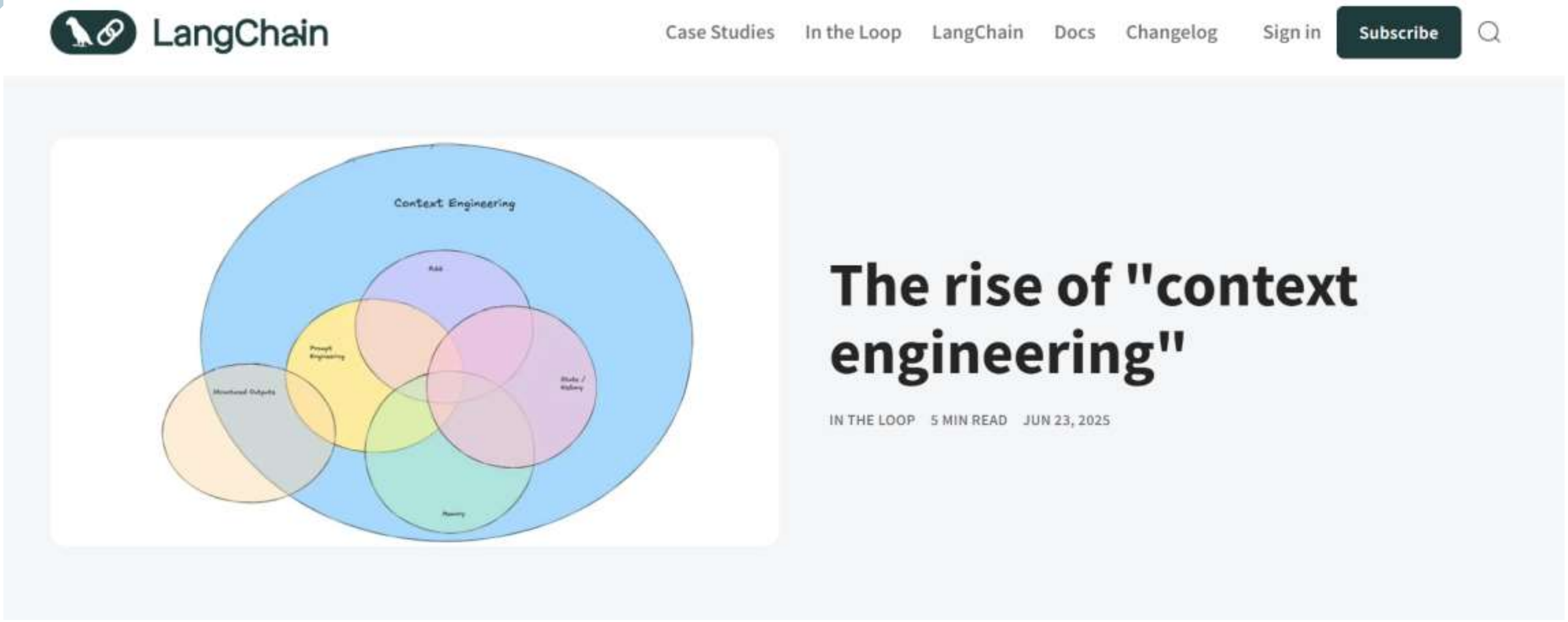


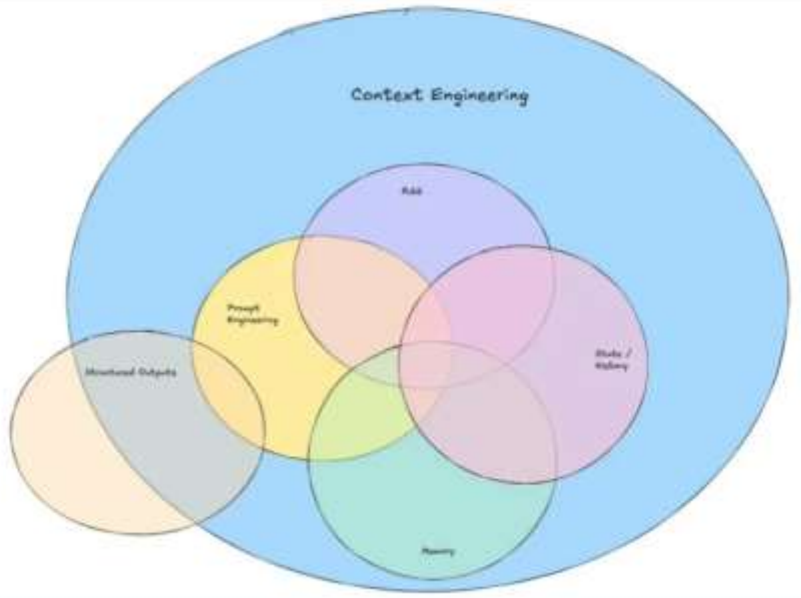
So What? The Impact

- MCP unlocks a whole new universe for LLMs, allowing them to automate tasks external to the chat interface without customizing to individual APIs.
- **File Management**
  - Reading from and writing to user files seamlessly.
- **Knowledge Access**
  - Accessing company knowledge bases or APIs directly.
- **Search Capabilities**
  - Searching the web and internal documentation.
- **Communication**
  - Sending emails and managing communications.



CONTEXT ENGINEERING



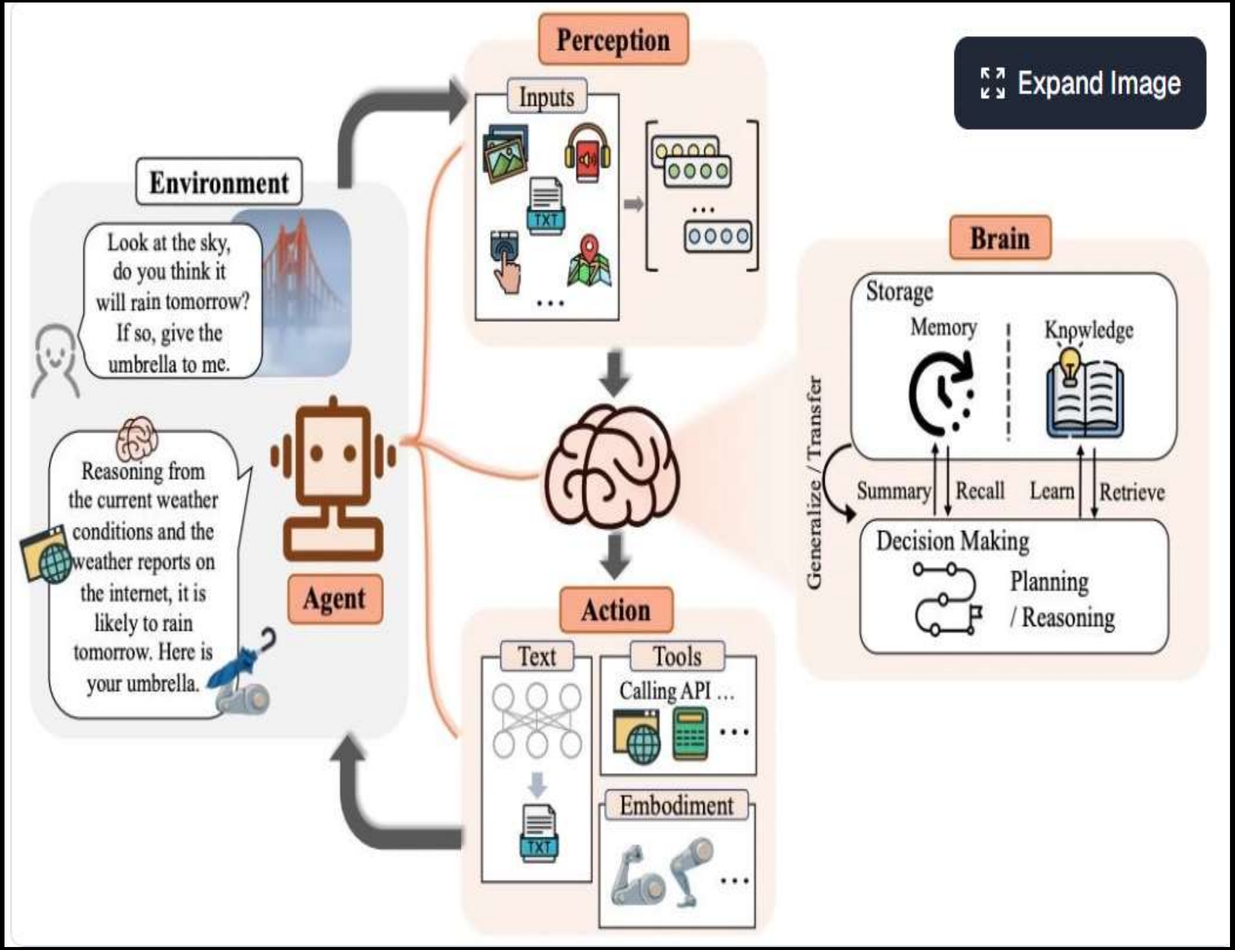


The rise of "context engineering"

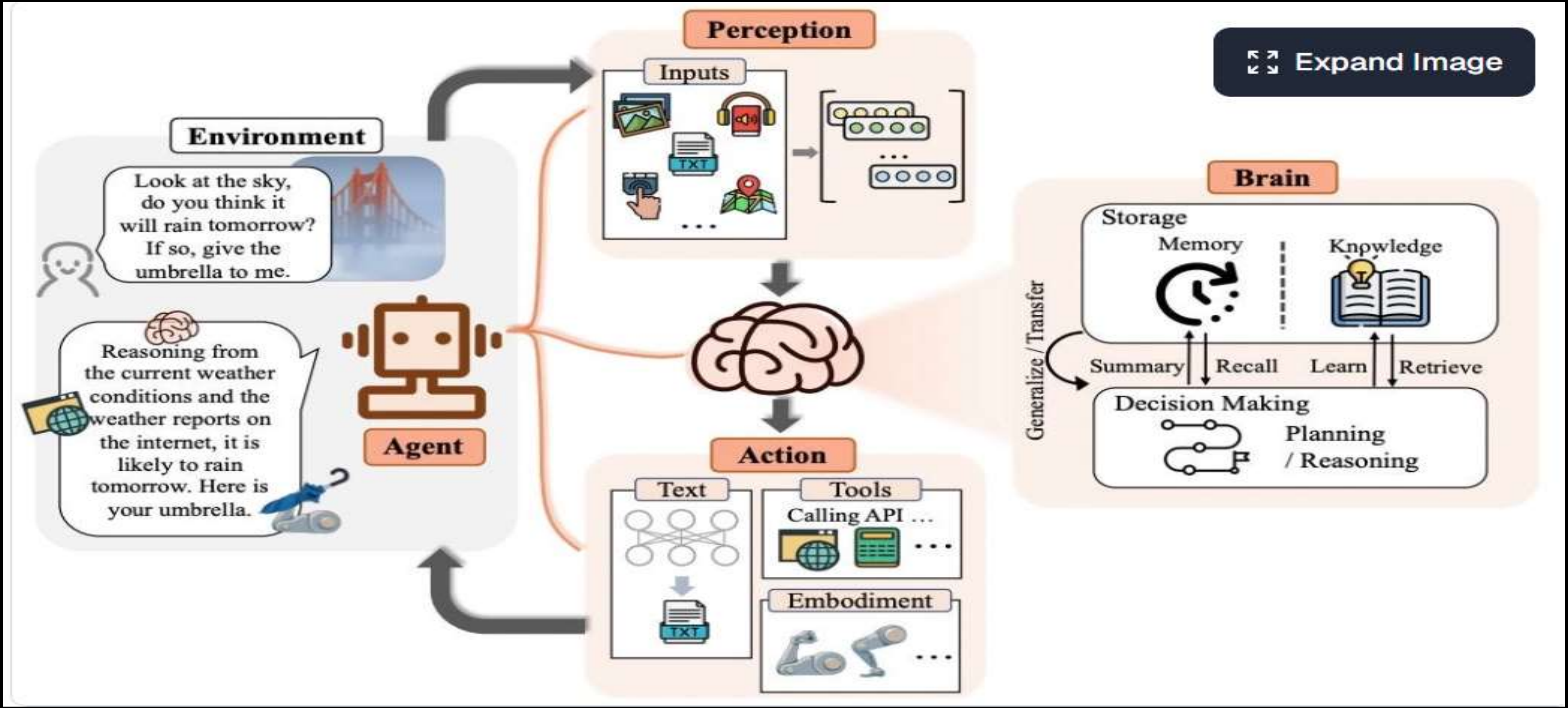
IN THE LOOP 5 MIN READ JUN 23, 2025

Context Engineering

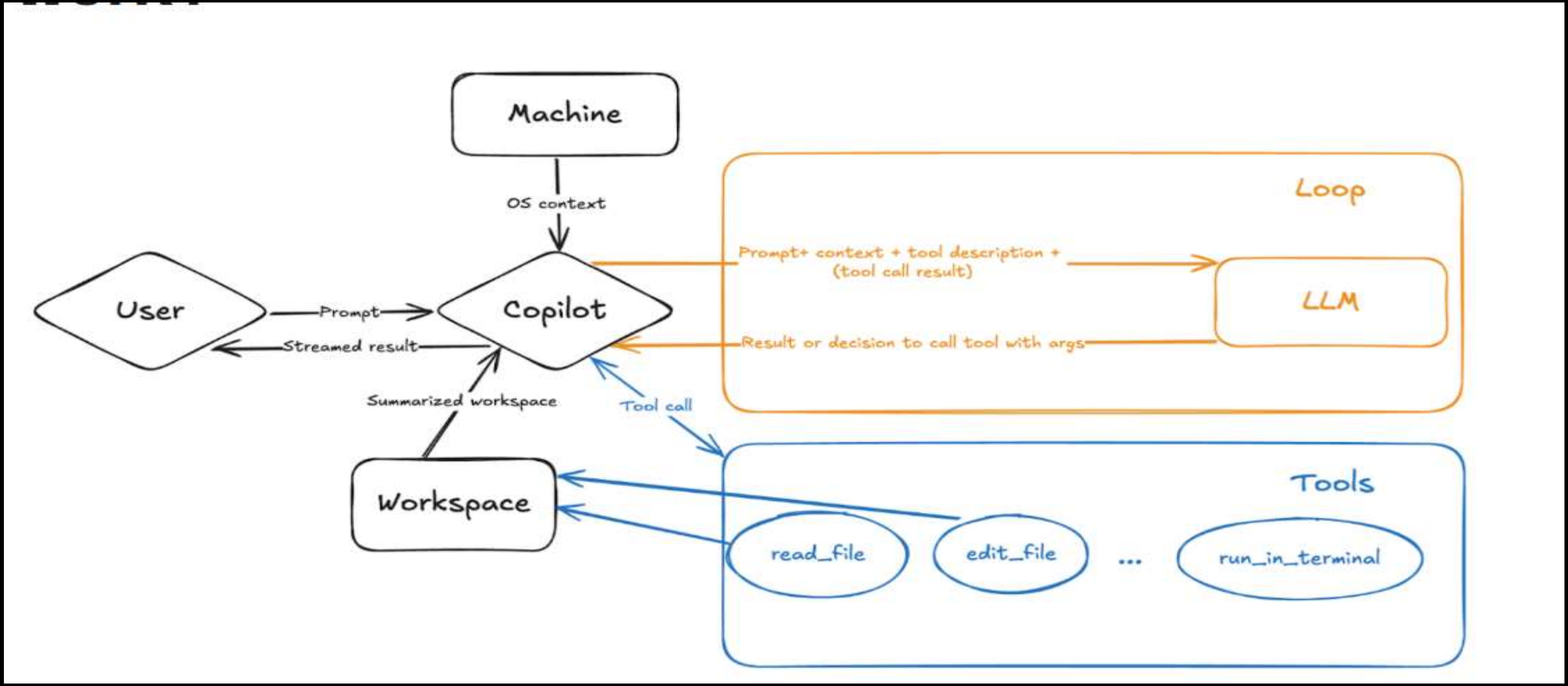
- It is the discipline of designing dynamic systems that ensures AI models receive the right information, tools and structure at the right time, so they can complete a task.
- The main functions are
  - Memory – Relevant user/system information
  - Tool Calling – The way for the agent/agents to interact with external environment
  - Framework – The selection between single or multi agent framework


Expand Image
Environment
Look at the sky, do you think it will rain tomorrow? If so, give the umbrella to me.
Reasoning from the current weather conditions and the weather reports on the internet, it is likely to rain tomorrow. Here is your umbrella.
Agent
Perception
Inputs
TXT
...
Brain
Storage
Memory
Knowledge
Summary
Recall
Learn
Retrieve
Decision Making
Planning / Reasoning
Generalize / Transfer
Action
Text
Tools
Calling API ...
Embodiment

CONTEXT ENGINEERING SYSTEMS OVERVIEW


Expand Image
Environment
Look at the sky, do you think it will rain tomorrow? If so, give the umbrella to me.
Reasoning from the current weather conditions and the weather reports on the internet, it is likely to rain tomorrow. Here is your umbrella.
Agent
Perception
Inputs
TXT
...
Brain
Storage
Memory
Knowledge
Generalize / Transfer
Summary
Recall
Learn
Retrieve
Decision Making
Planning / Reasoning
Action
Text
TXT
Tools
Calling API ...
...
Embodiment
...

Coding Agent System Overview



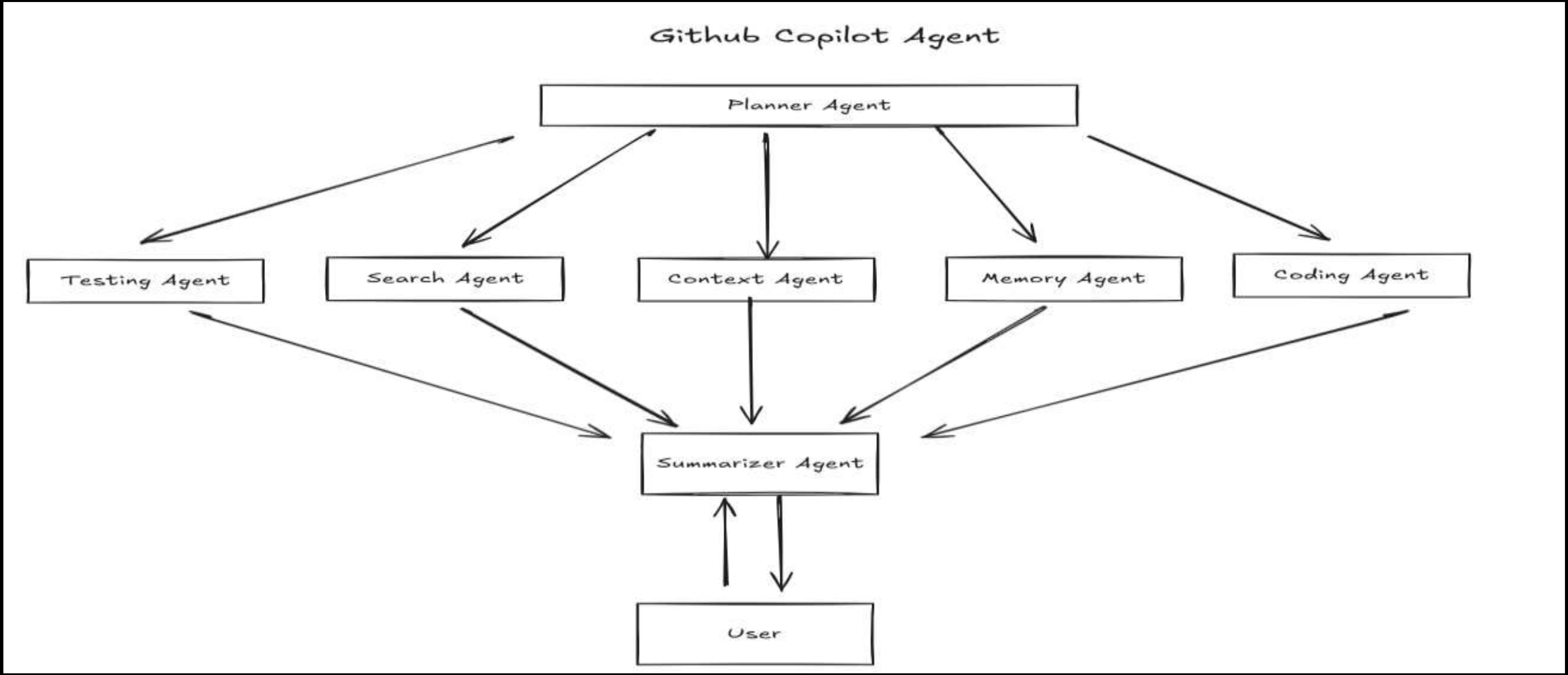
GITHUB AGENT SYSTEM OVERVIEW

- 1) Planner Agent
  - The main agent which plans the task based on the given user query

- 2) Sub Agents
  - Search Agent(Fetches repo, context, files etc)
  - Context Agent(MCP, API ETC)
  - Memory Agent(Fetches memory from user conversation)
  - Coding Agent(Provides the LLM with context + prompt)
  - Testing Agent(Runs test and fixes the error)

- 3) Summarizer Agent
  - Provides the code to the user and receive feedback to iterate

SYSTEM OVERVIEW



Future Plans

- Revamp the lecture modules to include more up to date industry terms
- Improve the clubs marketing & business development
- Projects are going to be more structured with strict deadlines
- Attendance will be taken technical leads and developers
- Launching an internal dev team(research team), to build developer productivity tools

Questions?