



# **PRACTICAL AI INTEGRATION WITH JAVA**

## A Hands-On Workshop

Craig Walls & Dan Vega ~ Broadcom



# *INTRODUCTION*



# Workshop Agenda

- 9:00 AM - 9:30 AM :: Introduction & What is AI
- 9:30 AM - 10:30 AM :: Spring AI Introduction
- 10:30 AM - 10:45 AM :: Break
- 10:45 AM - 12:00 PM :: Spring AI Features
- 12:00 PM - 1:00 PM :: Lunch
- 1:00 PM - 2:00 PM :: Large Language Model Limitations (Part 1)
- 2:00 PM - 2:15 PM :: Break
- 2:15 PM - 3:15 PM :: Large Language Model Limitations (Part 2)
- 3:15 PM - 3:30 PM :: Break
- 3:45 PM - 4:30 PM :: Observability
- 4:30 PM - 5:00 PM :: Open Source Models

# **Craig Walls**

# ABOUT ME

Learn more at [danvega.dev](https://danvega.dev)

👤 Husband & Father

🏡 Cleveland

☕ Java Champion

💻 Software Development 23 Years

🌿 Spring Developer Advocate

📖 Author



# Fundamentals of Software Engineering

From Coder to Engineer

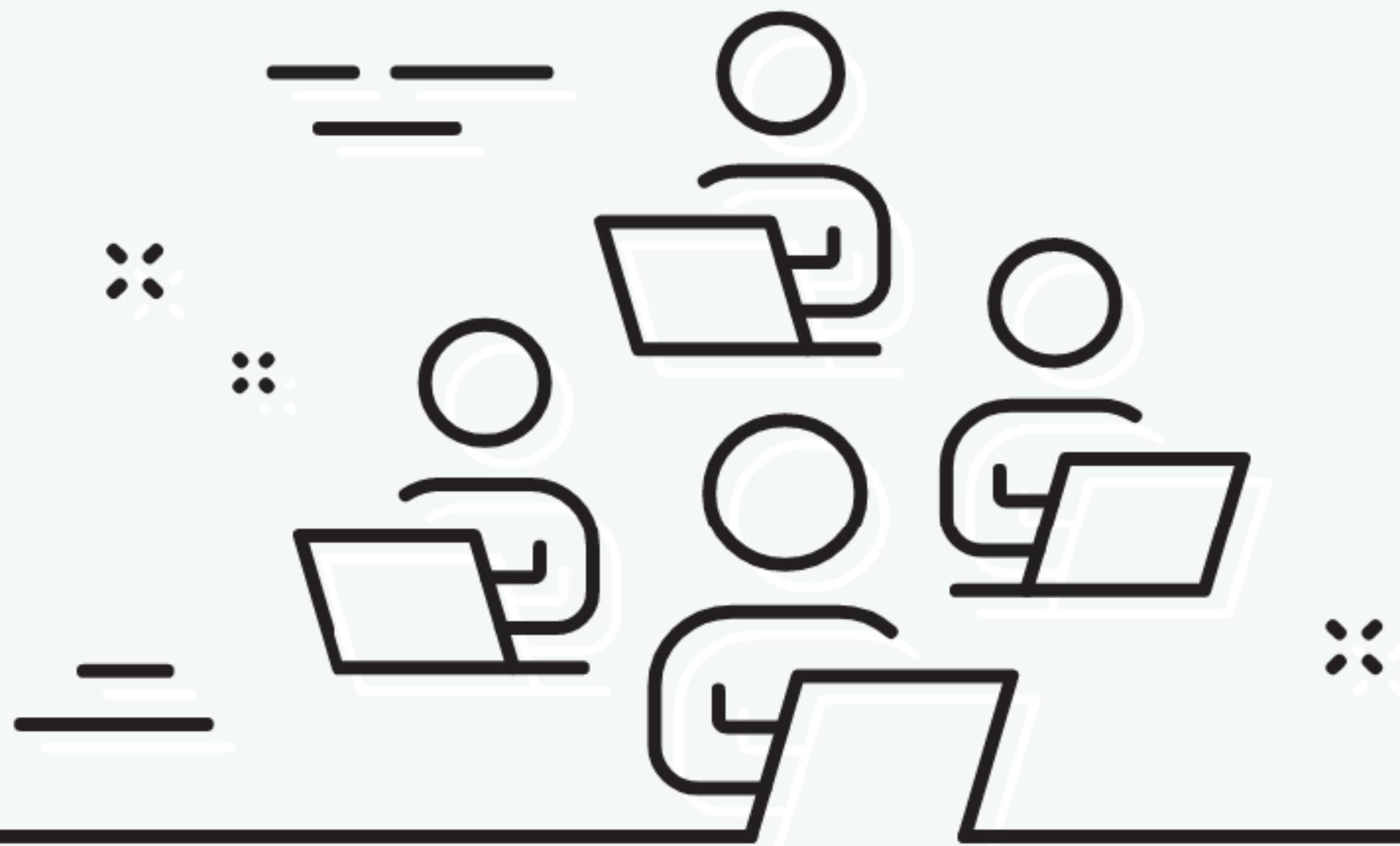
Early  
Release  
RAW &  
UNEDITED



Nathaniel Schutta  
& Dan Vega



# OFFICE HOURS



<https://www.springofficehours.io>

# **PREREQUISITES AND RESOURCES**

## **Prerequisites**

- Familiar with the Java programming Language
- Familiar with Spring (but not Spring AI)
- IDE (IntelliJ)

## **Resources**

- <https://docs.spring.io/spring-ai/reference>
- <https://github.com/danvega/spring-ai-workshop>

This is a workshop. It is meant  
to be interactive

<https://www.habuma.com/openai-key.html>

# WHAT IS AI



# WHY ARE WE TALKING ABOUT AI ON THE JVM?



# PYTHON LIBRARIES FOR AI/ML



## Core Deep Learning Frameworks

- **TensorFlow** – originally dominant, still heavily used in production/enterprise.
- **PyTorch** – the current favorite for research and a lot of industry adoption.

## Classical ML / Data Science Foundations

- **Scikit-learn** – standard for traditional ML (regression, classification, clustering, pipelines).
- **NumPy** – numerical computing foundation (arrays, linear algebra).
- **Pandas** – data manipulation and analysis (DataFrames).

## Transformers / NLP

- **Transformers (Hugging Face)** – state-of-the-art NLP (and increasingly multimodal).

# THE JAVA AI OPPORTUNITY



- **Enterprise Reality Check** Most large-scale enterprise applications run on the JVM. When these organizations want to integrate AI capabilities, they face a choice: build isolated Python services that require new infrastructure and teams, or leverage their existing JVM expertise and infrastructure.
- **Performance and Scalability** The JVM's mature runtime optimizations, garbage collection, and Just-In-Time compilation provide excellent performance for AI workloads. While Python dominates model training, the JVM excels at serving models in production where latency, throughput, and reliability matter most.
- **Ecosystem Maturity** The JVM offers battle-tested libraries for distributed computing (Spark, Akka), data processing (Kafka, Hadoop), web services, and enterprise integration. This existing ecosystem can accelerate AI application development without reinventing the wheel.
- **Language Diversity Advantage** Java provides enterprise stability and extensive tooling. Groovy offers Python-like experience. This gives teams flexibility to choose the right tool for each AI task while staying within one runtime.
- **Developer Talent Pool** There are millions of JVM developers worldwide. Training existing teams on JVM-based AI tools is often faster and more cost-effective than hiring specialized Python AI teams or forcing Java developers to context-switch between languages.

# James Ward

Principal Developer Advocate at AWS

 **James Ward**   
@JamesWard

I know most people here won't believe it, but I can guarantee you that in 2 years, the majority of AI Agent workloads will run on the JVM.

12:36 AM · Sep 9, 2025 · 14.8K Views

 22  33  131   24 

 Post your reply 

 **James Ward**   
@JamesWard · 4h

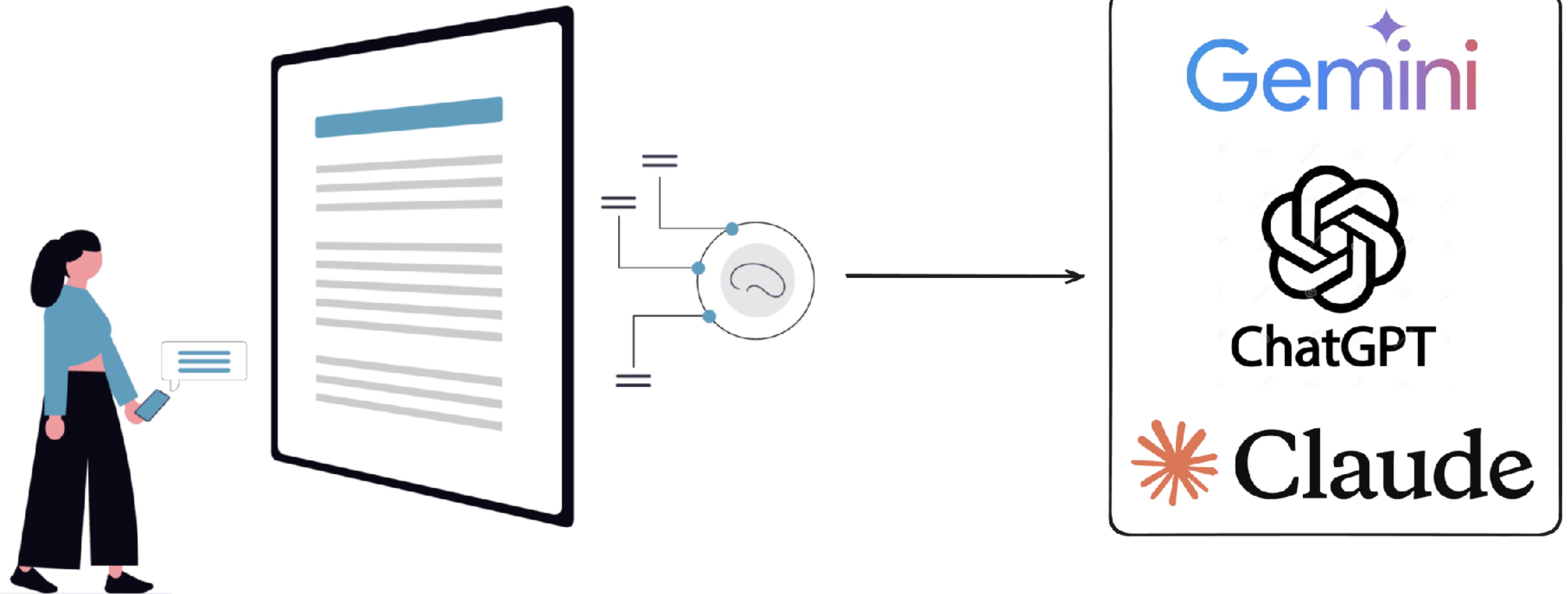
Here's why: AI Agents are just integration systems that need to be secure, observable, scalable, etc. The JVM is exactly the place where the majority of these types of workloads run today. And with great JVM agent frameworks already here, there is no reason to build these on different tech.

 1   4   704 

 **James Ward**   
@JamesWard · 3h

And most importantly, AI Agents need to be reliable. And the JVM is where the majority of workloads that need reliability go. JVM Agent frameworks like Embabel and Koog are proving that domain-oriented agent orchestration increases reliability.

 1   4   710 



```
#!/bin/bash
echo "Calling Open AI..."
MY_OPENAI_KEY="YOUR_API_KEY_HERE"
PROMPT="Tell me an interesting fact about Java"

curl https://api.openai.com/v1/chat/completions \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $MY_OPENAI_KEY" \
-d '{"model": "gpt-5", "messages": [{"role": "user", "content": "'"${PROMPT}"'}]}'
```

```
{  
  "id": "chatcmpl-ABNbjZ5oRbo720evnCX2arPufJCYK",  
  "object": "chat.completion",  
  "created": 1727275719,  
  "model": "gpt-4o-2024-05-13",  
  "choices": [  
    {  
      "index": 0,  
      "message": {  
        "role": "assistant",  
        "content": "Sure! Did you know that Java was initially designed with interactive television in mind? James Gosling and his team at Sun Microsystems started the project in 1991 under the name \"Oak.\" The name was later changed to \"Java\" after discovering there was already a programming language called Oak. Java's versatility has made it one of the most popular programming languages for a wide range of applications, far beyond its initial intended use for TV set-top boxes!",  
        "refusal": null  
      },  
      "logprobs": null,  
      "finish_reason": "stop"  
    }  
  ],  
  "usage": {  
    "prompt_tokens": 14,  
    "completion_tokens": 90,  
    "total_tokens": 104,  
    "completion_tokens_details": {  
      "reasoning_tokens": 0  
    }  
  },  
  "system_fingerprint": "fp_e375328146"  
}
```

```
public static void main(String[] args) throws IOException, InterruptedException {
    var apiKey = "YOUR_API_KEY_HERE";
    var body = """
        {
            "model": "gpt-4o",
            "messages": [
                {
                    "role": "user",
                    "content": "Tell me an interesting fact about Java"
                }
            ]
        }""";
}

HttpRequest request = HttpRequest.newBuilder()
    .uri(URI.create("https://api.openai.com/v1/chat/completions"))
    .header("Content-Type", "application/json")
    .header("Authorization", "Bearer " + apiKey)
    .POST(HttpRequest.BodyPublishers.ofString(body))
    .build();

var client = HttpClient.newHttpClient();
var response = client.send(request, HttpResponse.BodyHandlers.ofString());
System.out.println(response.body());
}
```

# **WHAT ARE THE CHALLENGES**

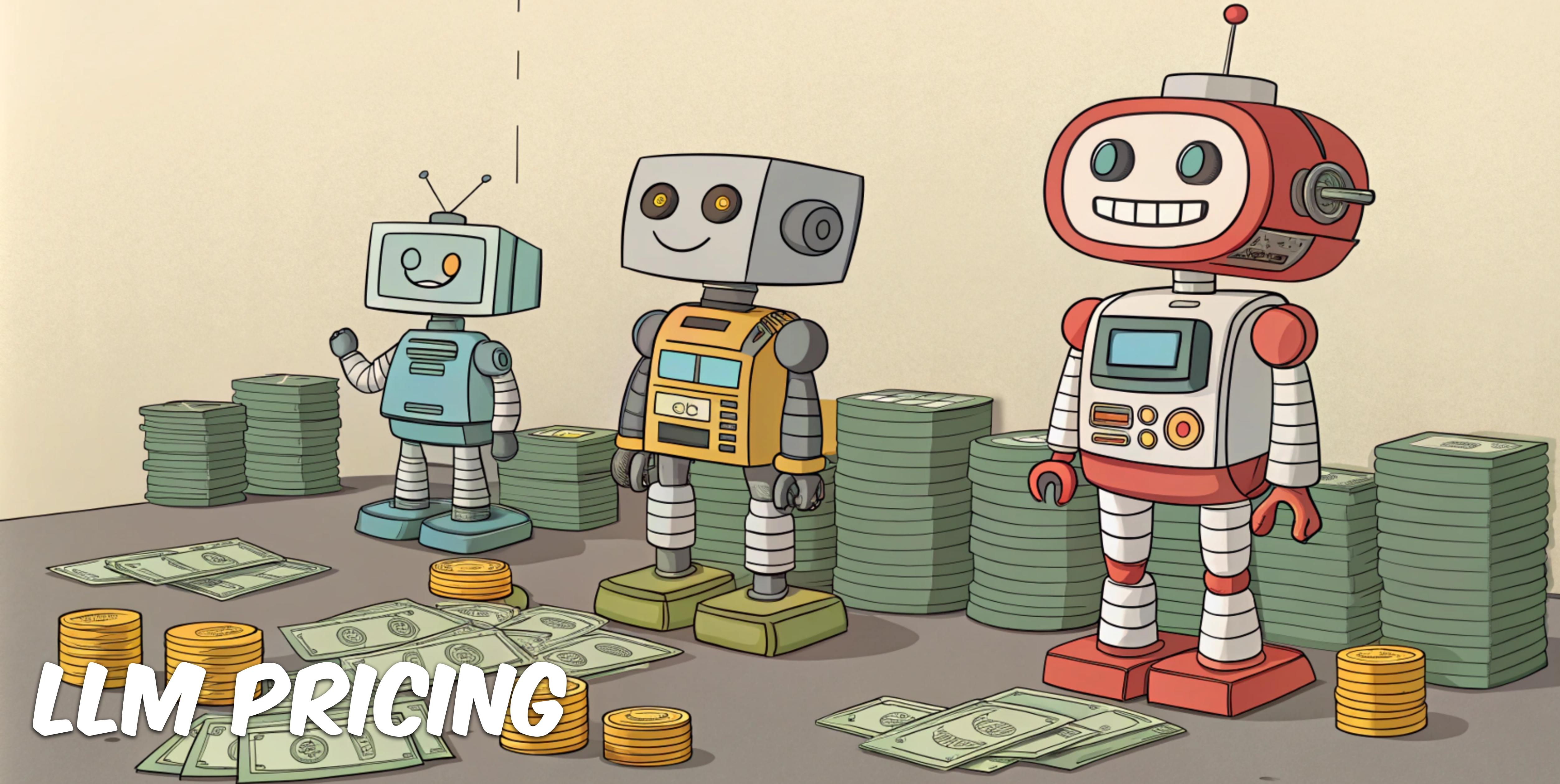
**What does a framework provide you?**

- Model abstraction and portability
- Unified API across providers
- Structured output parsing
- Prompt template management
- Token counting and cost management
- Retry logic and error handling
- Response streaming and async processing
- Memory and conversation management
- Embedding and vector operations
- Function calling integration
- Observability and monitoring
- Security and compliance

**SPRING AI PROVIDES US SO MUCH**  
**MORE THAN A FACILITY FOR**  
**MAKING REST API CALLS**



# LLM PRICING



## Free

Explore how AI can help with everyday tasks

- ✓ Access to GPT-4.1 mini
- ✓ Real-time data from the web with search
- ✓ Limited access to GPT-4o, OpenAI o4-mini, and deep research
- ✓ Limited access to file uploads, data analysis, image generation, and voice mode
- ✓ Code edits with the ChatGPT desktop app for macOS
- ✓ Use custom GPTs

Have an existing plan? See [billing help](#)

\$0 / month

[Get Free ↗](#)

## Plus

Level up productivity and creativity with expanded access

- ✓ Everything in Free
- ✓ Extended limits on messaging, file uploads, data analysis, and image generation
- ✓ Standard and advanced voice mode with video and screensharing
- ✓ Access to deep research and multiple reasoning models (OpenAI o3, OpenAI o4-mini, and OpenAI o4-mini-high)
- ✓ Access to a research preview of GPT-4.5, our largest model yet, and GPT-4.1, a model optimized for coding tasks
- ✓ Create and use projects, tasks, and custom GPTs
- ✓ Opportunities to test new features

\$20 / month

[Get Plus ↗](#)

Limits apply >

## Pro

Get the best of OpenAI with the highest level of access

- ✓ Everything in Plus
  - ✓ Unlimited access to all reasoning models and GPT-4o
  - ✓ Unlimited access to advanced voice, with higher limits for video and screensharing
  - ✓ Access to OpenAI o1 pro mode, which uses more compute for the best answers to the hardest questions
  - ✓ Extended access to deep research
  - ✓ Extended access to Sora video generation
  - ✓ Access to a research preview of Operator
  - ✓ Access to research preview of Codex agent
- Unlimited subject to abuse guardrails. [Learn more](#)*

\$200 / month

[Get Pro ↗](#)

## GPT-5

The best model for coding and agentic tasks across industries

### Price

Input:

\$1.250 / 1M tokens

Cached input:

\$0.125 / 1M tokens

Output:

\$10.000 / 1M tokens

## GPT-5 mini

A faster, cheaper version of GPT-5 for well-defined tasks

### Price

Input:

\$0.250 / 1M tokens

Cached input:

\$0.025 / 1M tokens

Output:

\$2.000 / 1M tokens

## GPT-5 nano

The fastest, cheapest version of GPT-5 —great for summarization and classification tasks

### Price

Input:

\$0.050 / 1M tokens

Cached input:

\$0.005 / 1M tokens

Output:

\$0.400 / 1M tokens

GPT-4o & GPT-4o mini

GPT-3.5 & GPT-4

GPT-3 (Legacy)

Tell me an interesting fact about The Groovy Programming Language

Clear

Show example

Tokens

11

Characters

65

Tell me an interesting fact about The Groovy Programming Language

Text Token IDs

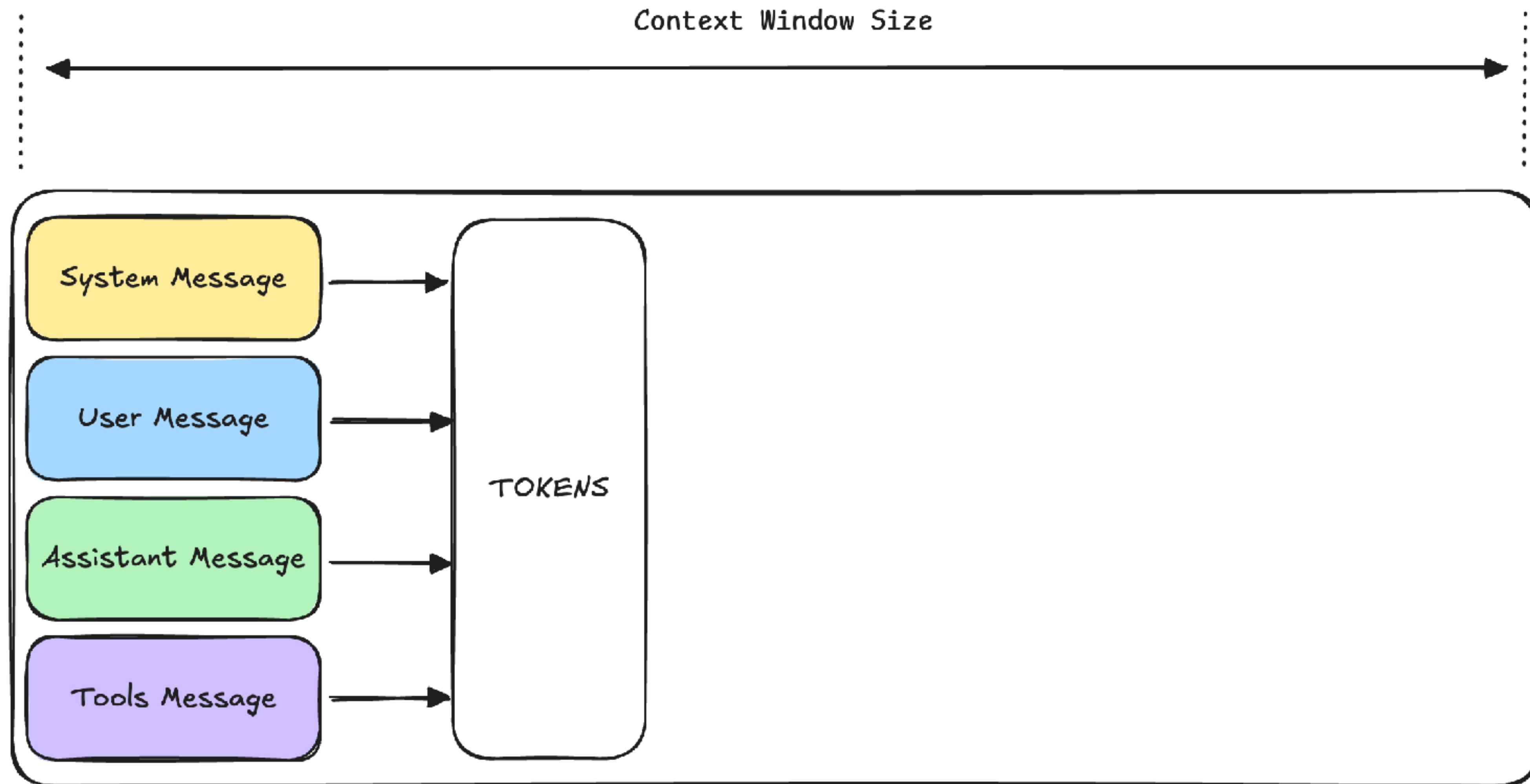
A helpful rule of thumb is that one token generally corresponds to ~4 characters of text for common English text. This translates to roughly  $\frac{3}{4}$  of a word (so 100 tokens  $\approx$  75 words).

If you need a programmatic interface for tokenizing text, check out our [tiktoken](#) package for Python. For JavaScript, the community-supported [@dbdq/tiktoken](#) package works with most GPT models.

[60751, 668, 448, 9559, 2840, 1078, 623, 16453, 62360, 65103, 20333]

Text Token IDs

# Context Window



Model	Context window	Pricing (per 1M input / output tokens)
GPT-5 (OpenAI)	~400K tokens	\$1.25 / \$10.00 (cached input: \$0.125)
GPT-5 Mini (OpenAI)	~400K tokens	\$0.25 / \$2.00 (cached input: \$0.025)
GPT-5 Nano (OpenAI)	~400K tokens	\$0.05 / \$0.40 (cached input: \$0.005)
Claude Sonnet 4 (Anthropic)	200K tokens	\$3.00 / \$15.00
Claude Opus 4.1 (Anthropic)	200K tokens (32K output)	\$15.00 / \$75.00
Gemini 2.5 Flash-Lite (Google)	1M tokens	\$0.10 / \$0.40
Gemini 2.5 Flash (Google)	1M tokens	\$0.30 / \$1.25
Gemini 2.5 Pro (Google)	1M tokens (2M roadmap)	≤200K: \$1.25 / \$10.00 • >200K: \$2.50 / \$15.00
Grok 3 (xAI)	131K tokens	\$3.00 / \$15.00 (cached input: \$0.75)

# SPRING AI INTRODUCTION



# GETTING STARTED

## A Tour of Spring AI Features

- Spring AI Reference Documentation
  - Currently Version: 1.0.2
  - 1.10-M2 (Lots of goodies here)
- Sign up for an API Key - **OpenAI**
- [start.spring.io](https://start.spring.io)
- Chat Client & Chat Model
  - Blocking vs Non Blocking (Streaming Responses)
  - Response Types (Content / ChatResponse)

# SPRING AI FEATURES

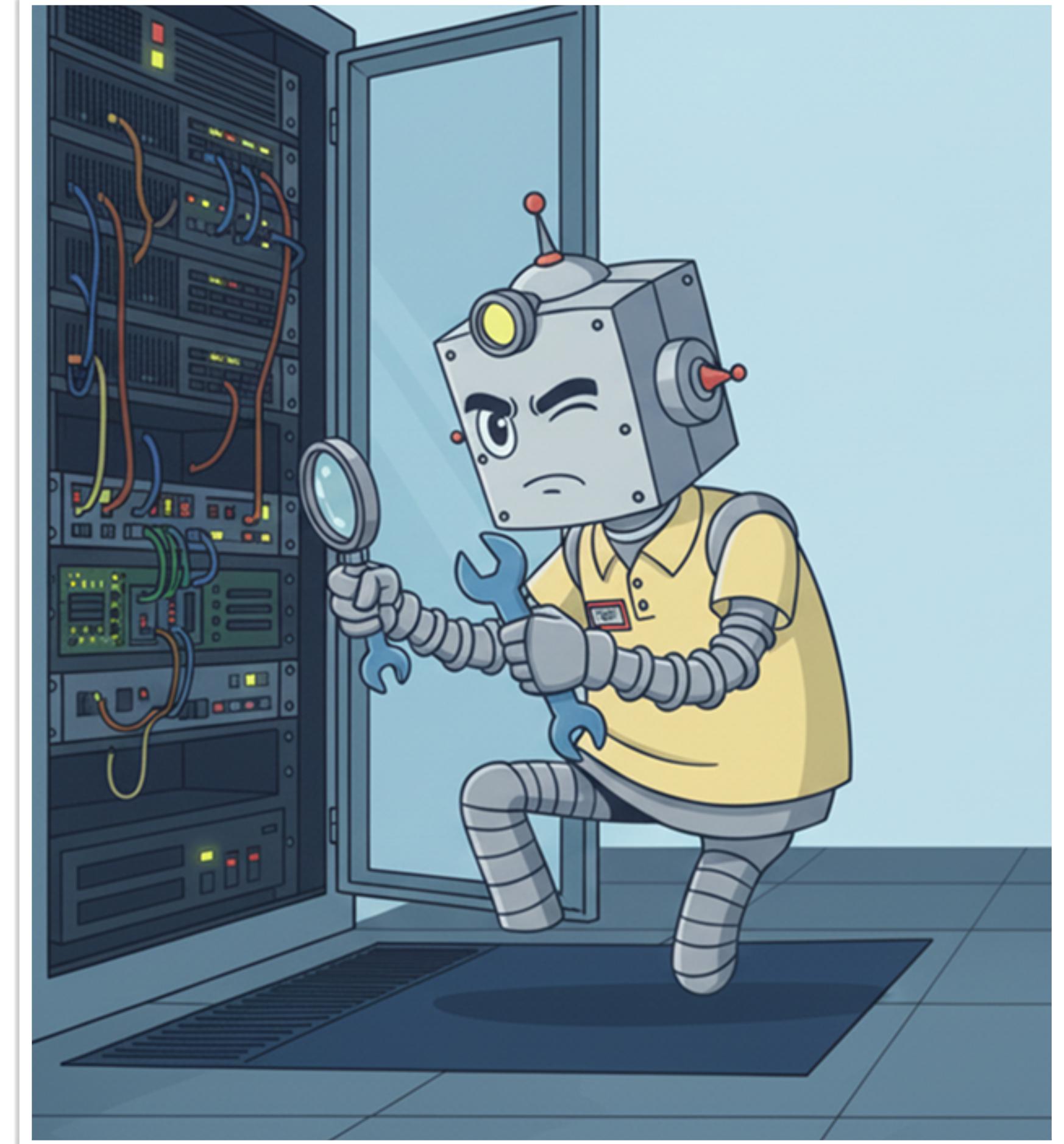


# LLM LIMITATIONS



# **PROBLEM STATEMENT**

How do we augment the limitations of Large Language Models?





Hallucinations



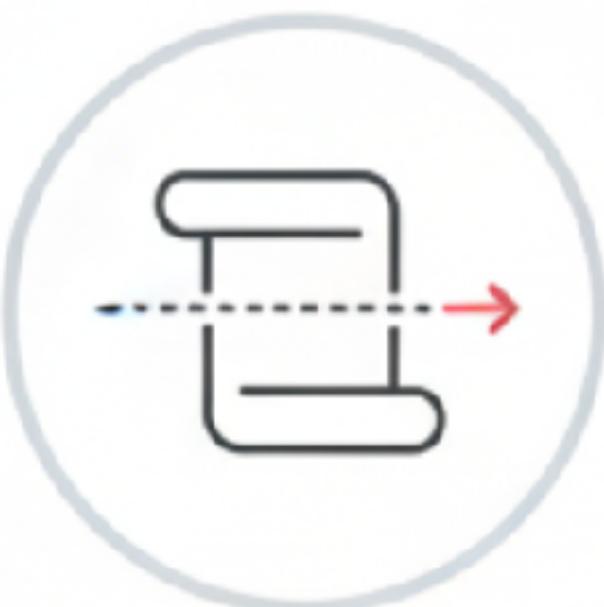
Stale Data



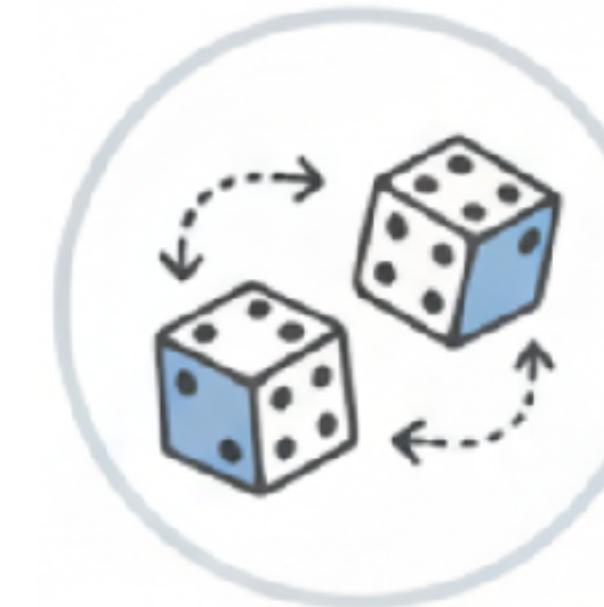
Bias & Safety



Domain Gaps



Context Window



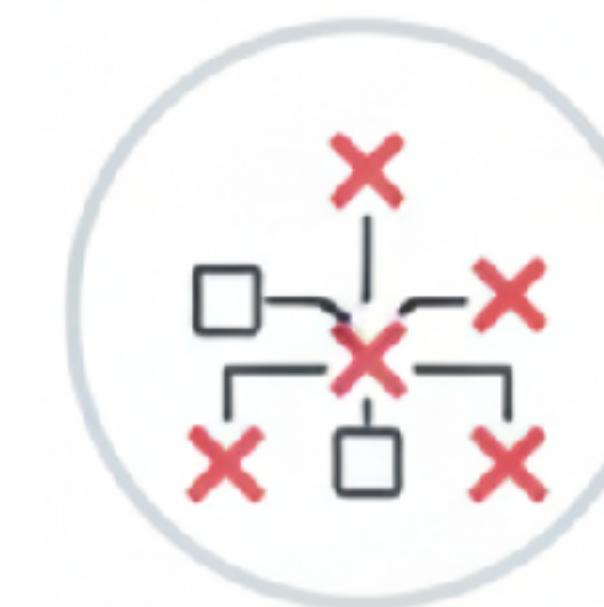
Non-Deterministic



Privacy & Security

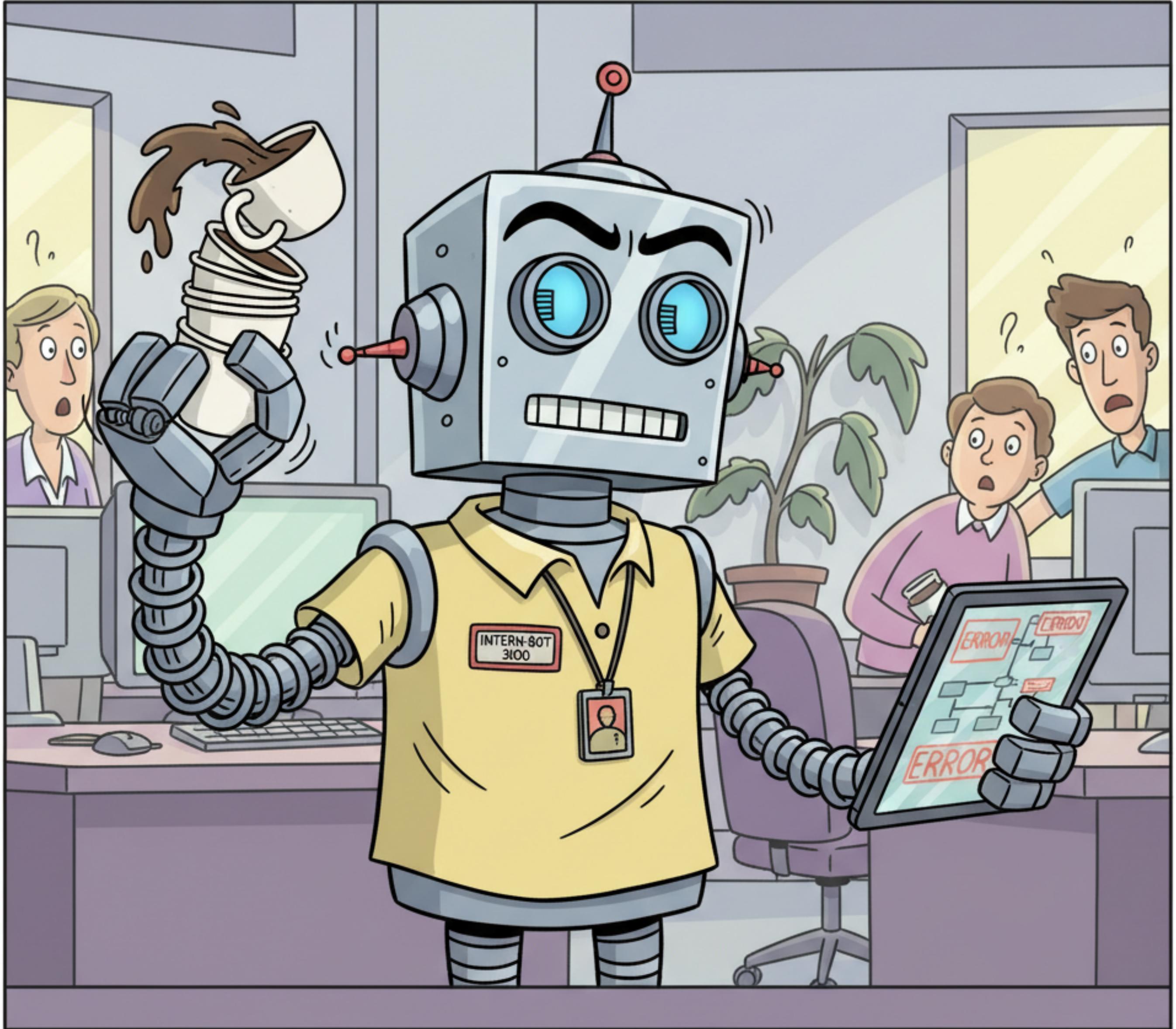


Cost & Latency

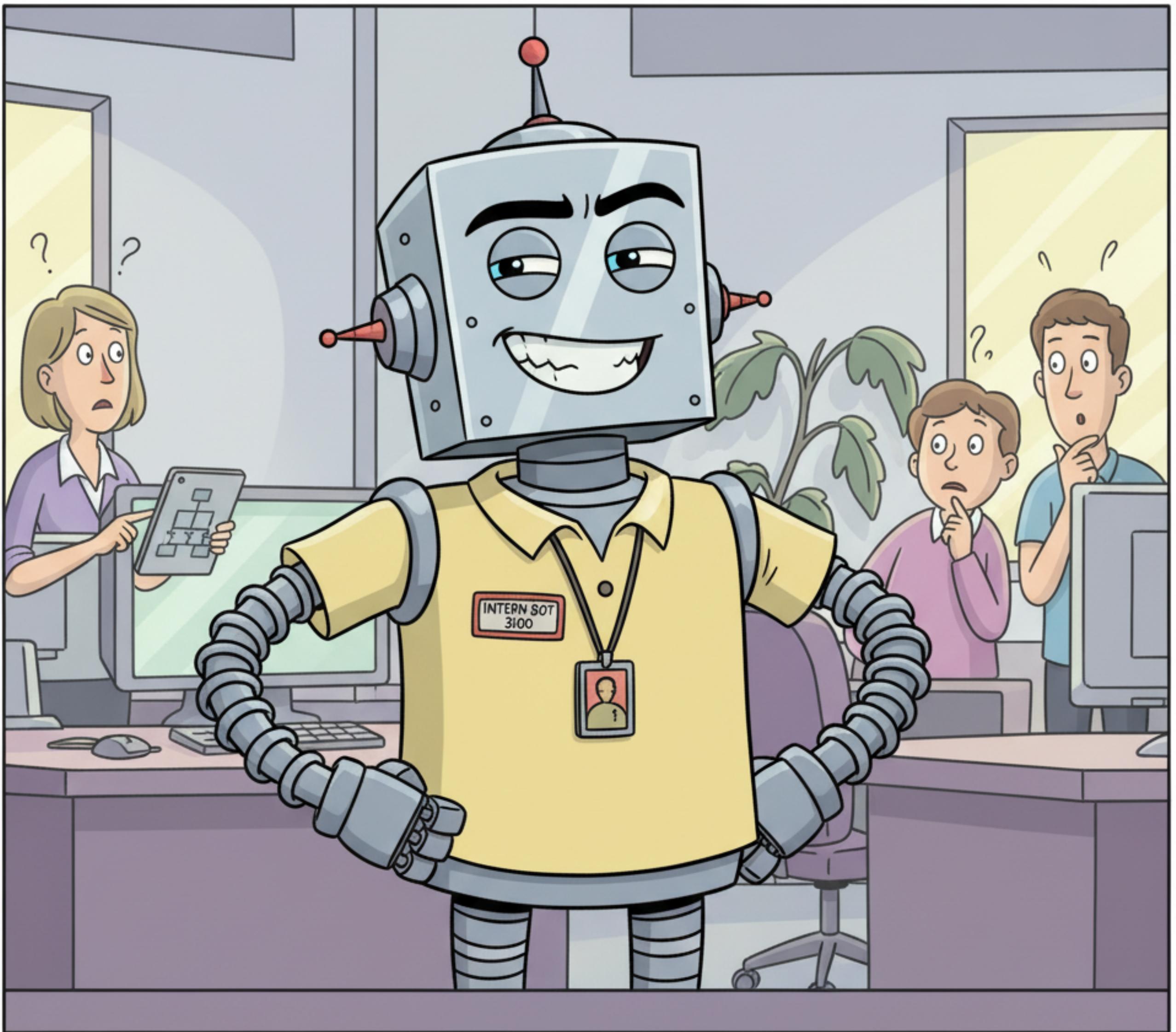


Weak Reasoning

**LLMS ARE LIKE  
SUPER-SMART INTERNS**



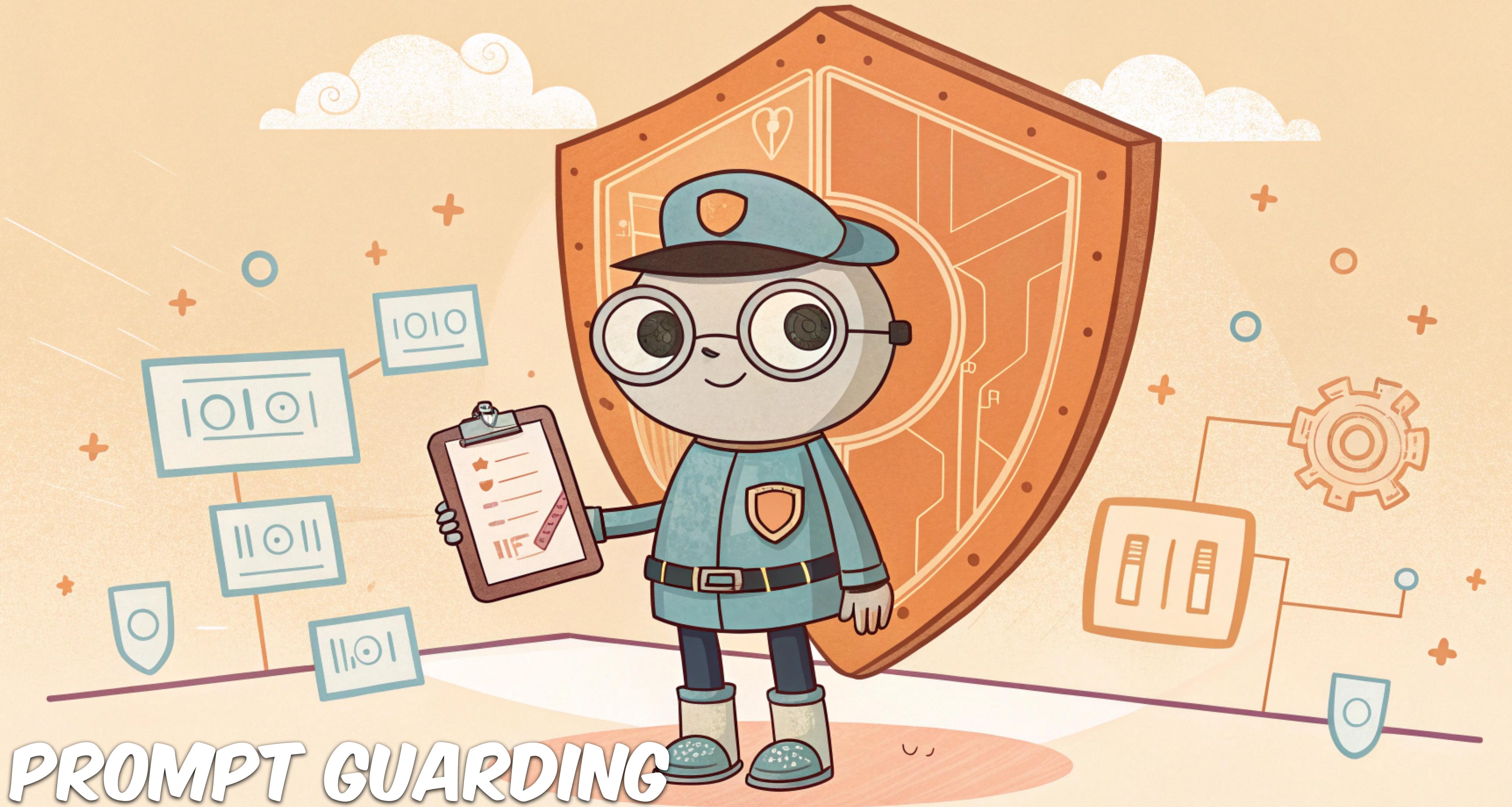
**LLMS ARE CONFIDENT**



# LLM LIMITATIONS

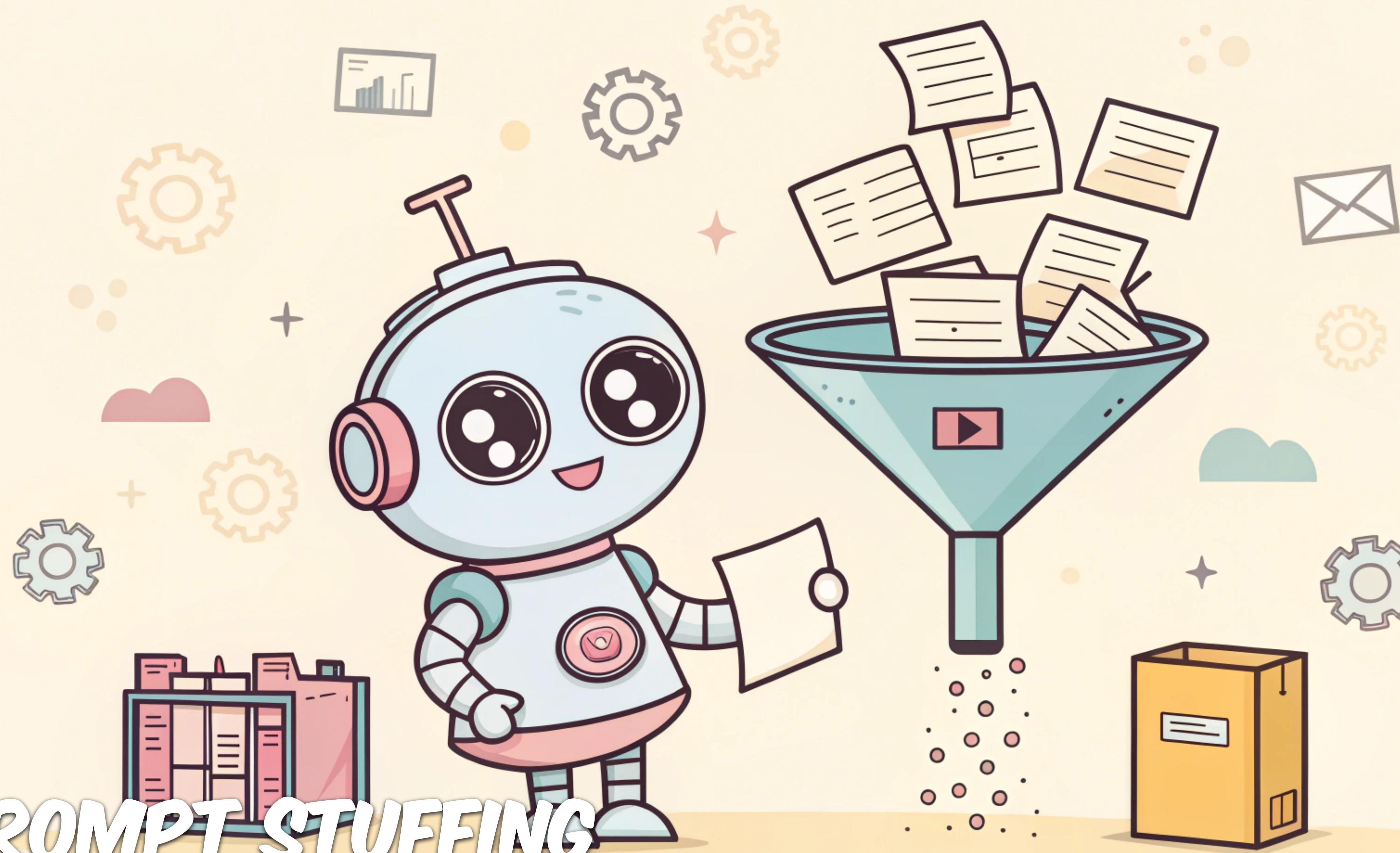
Here's our Swiss-army lineup for taming any limitation

#	Lever	Purpose
1	 <b>Prompt Guarding</b>	Encode rules that constrain the model's behavior (tone, honesty, refusal policy).
2	 <b>Prompt Stuffing / RAG</b>	Inject fresh, task-specific context so the model quotes facts instead of guessing.
3	 <b>Tools / Function Calling</b>	Let the model invoke code or APIs for real-time data, calculations, or business logic.
4	 <b>MCP (Resources + Tools)</b>	Package those tools as reusable, versioned endpoints every client can share.

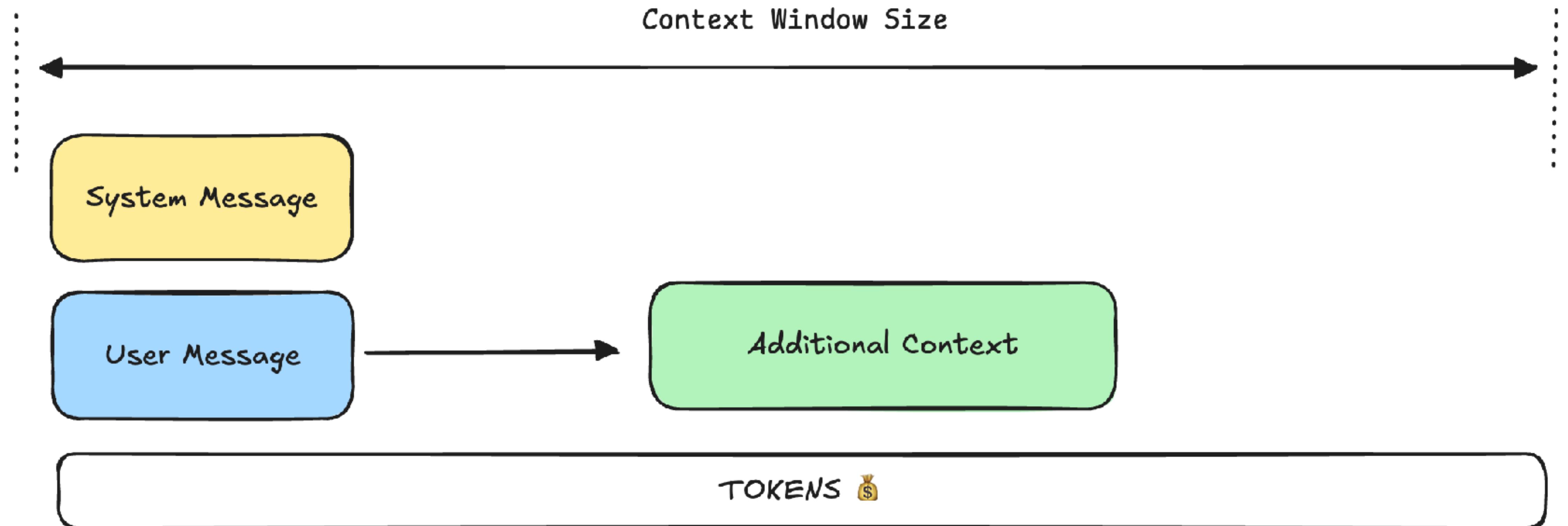


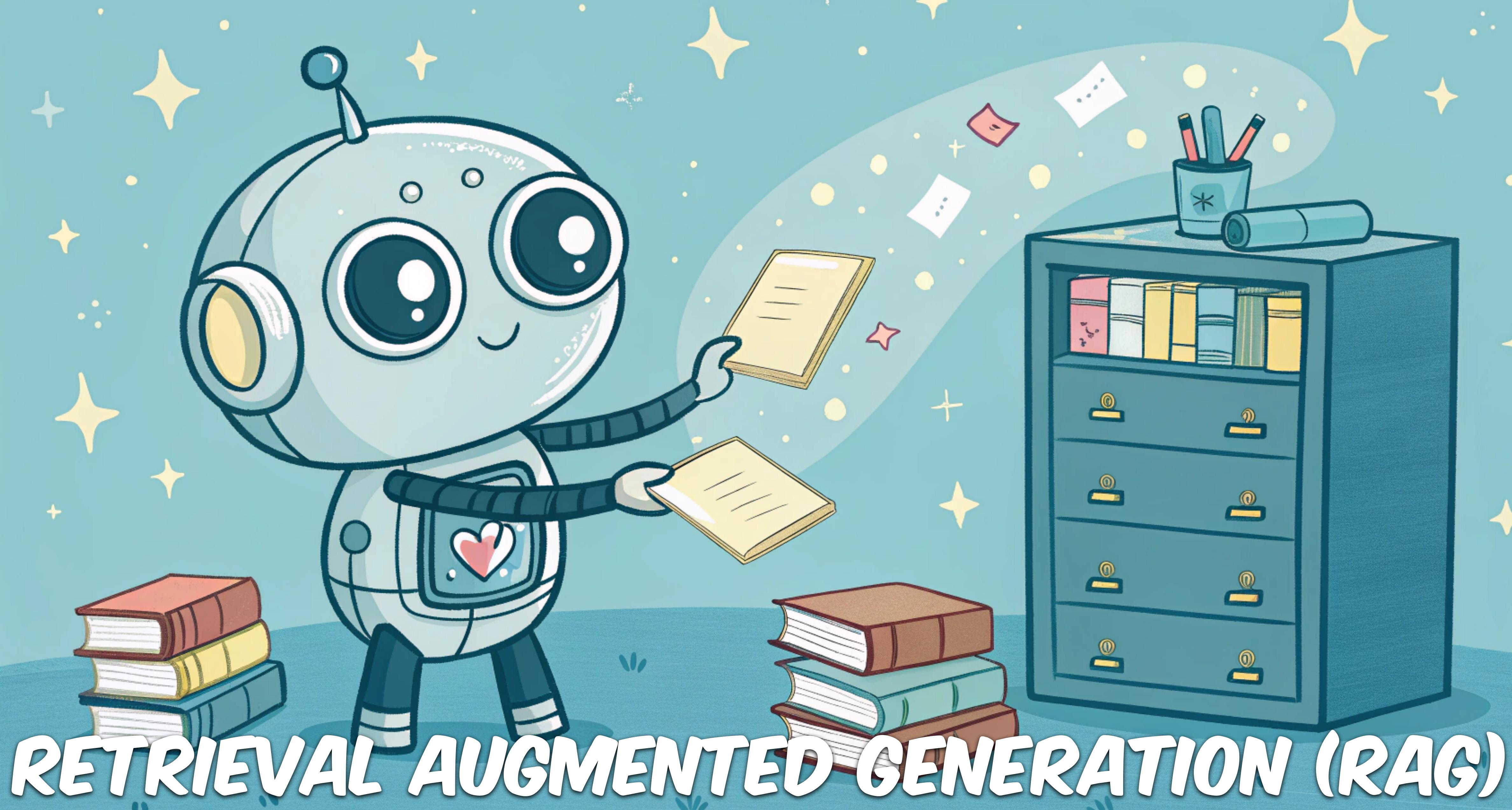
# PROMPT GUARDING

# PROMPT STUFFING



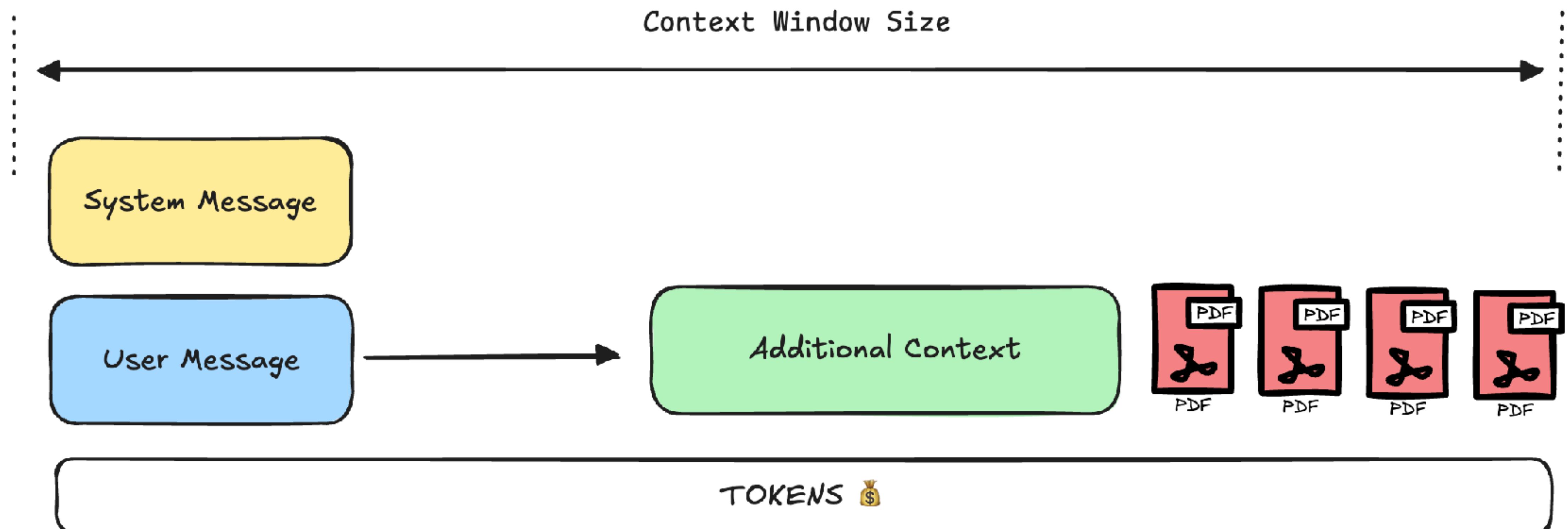
# Context Window

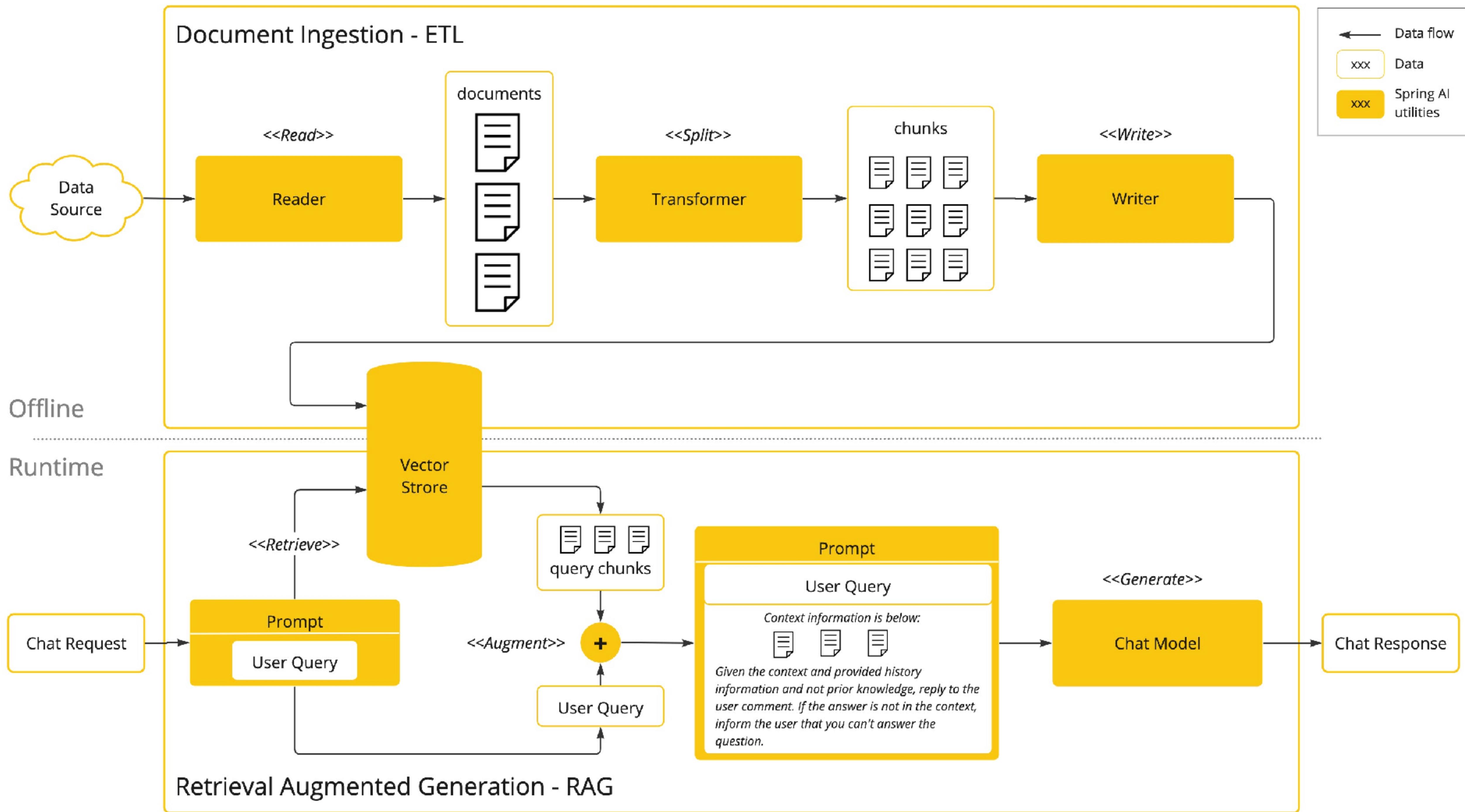


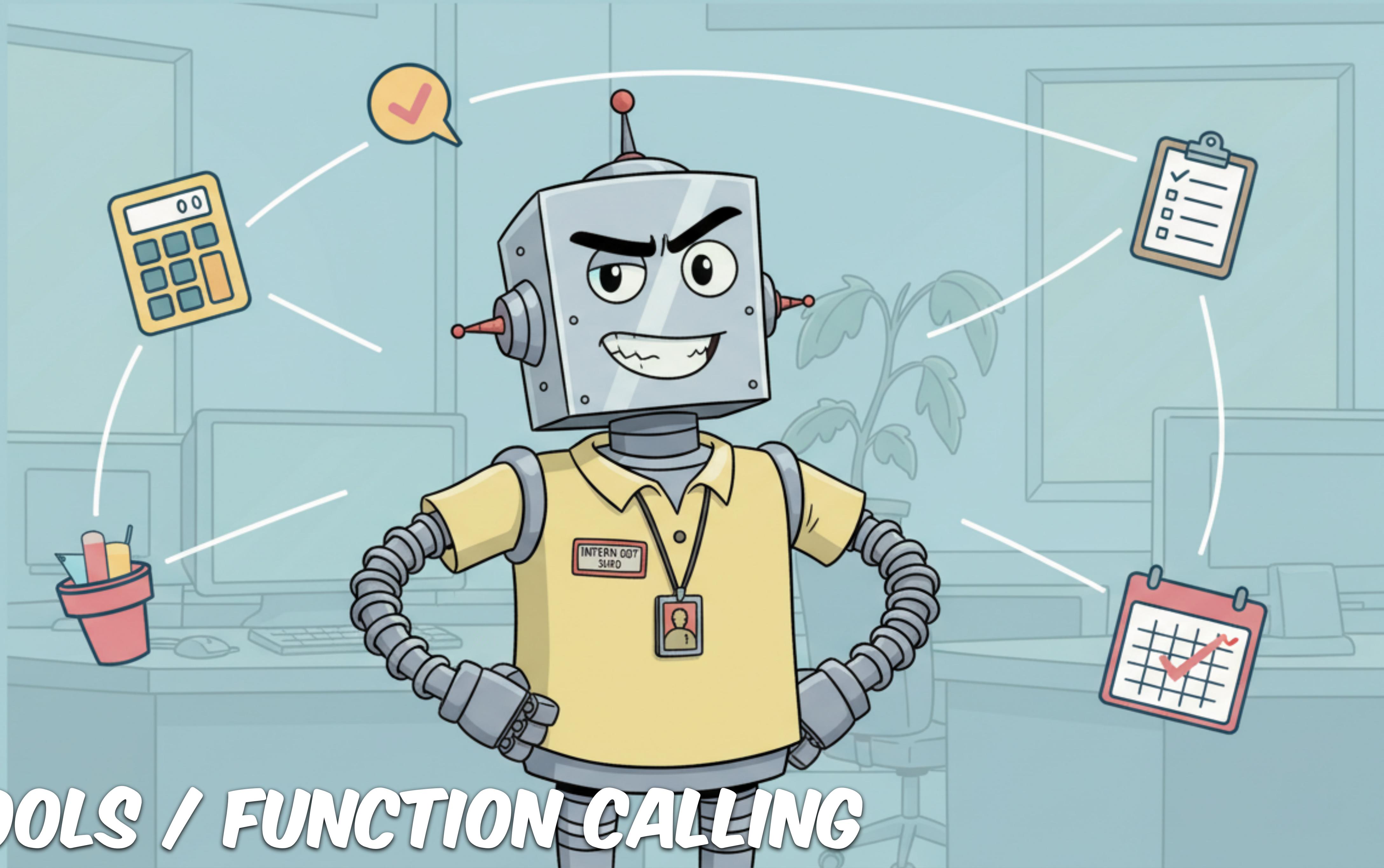


# RETRIEVAL AUGMENTED GENERATION (RAG)

# Context Window

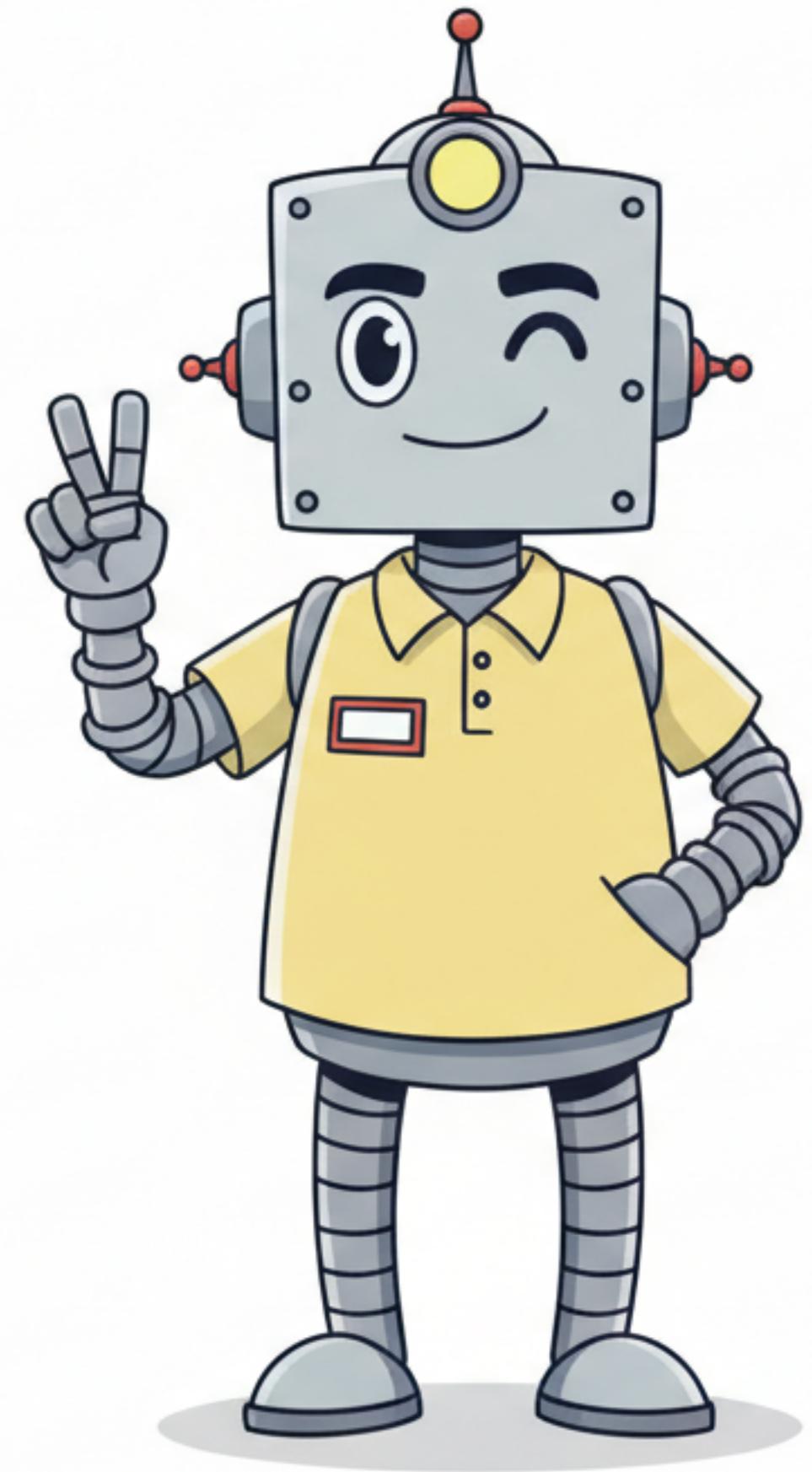






# TOOLS / FUNCTION CALLING

Information Retrieval

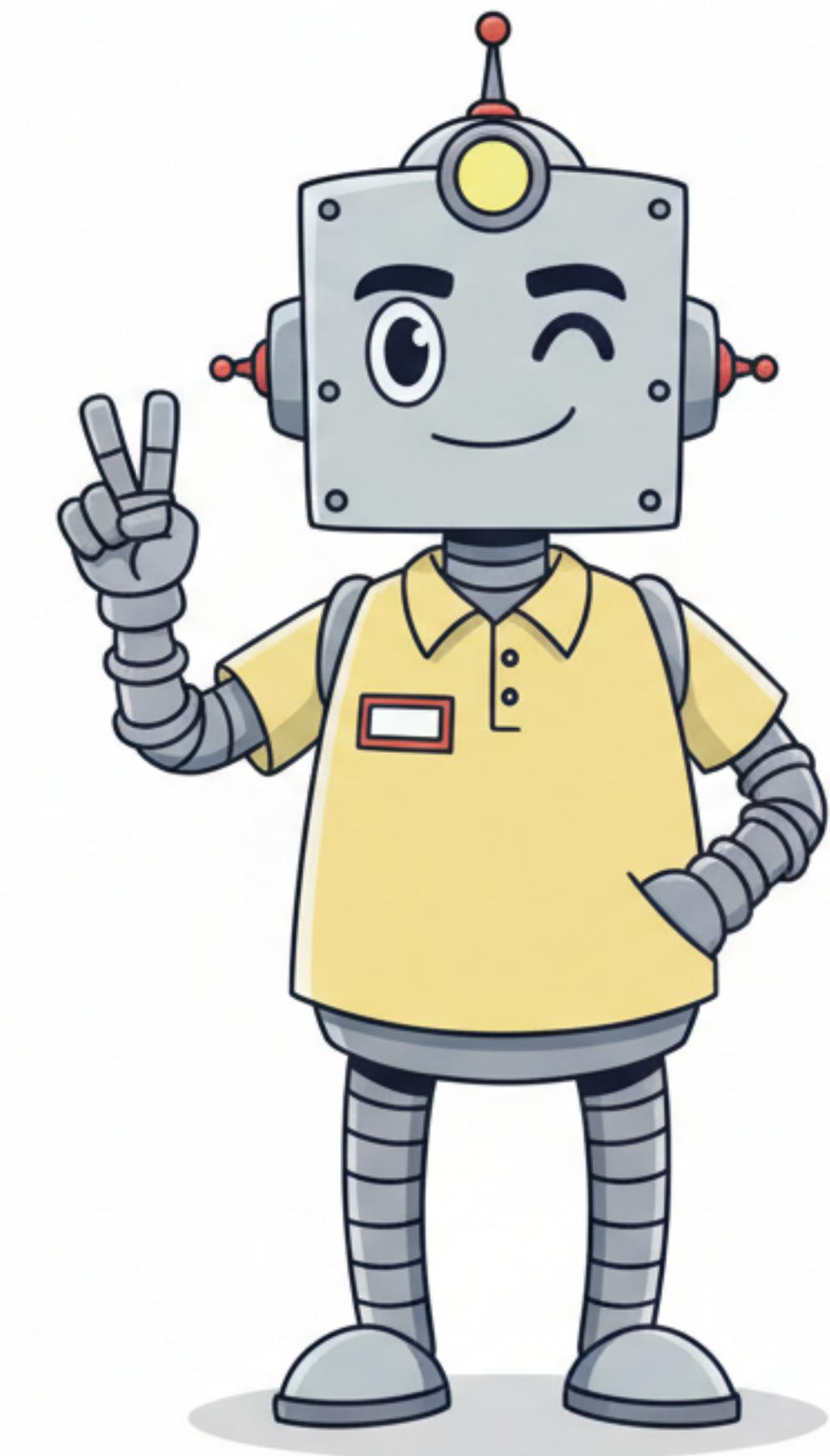


Taking Action

# TOOL CALLING / FUNCTION CALLING

## Information Retrieval

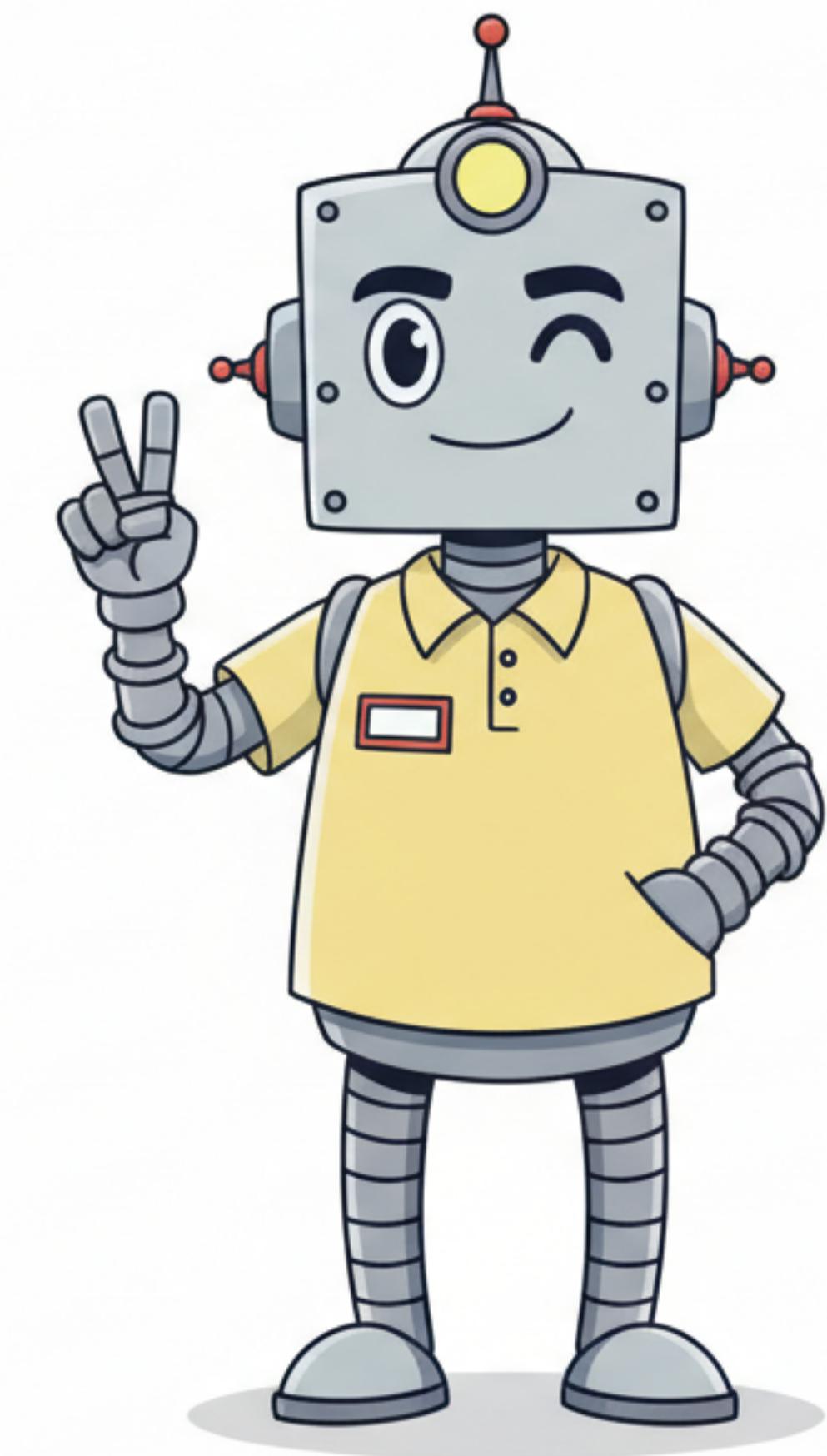
Tools in this category can be used to retrieve information from external sources, such as a database, a web service, a file system, or a web search engine. The goal is to augment the knowledge of the model, allowing it to answer questions that it would not be able to answer otherwise. As such, they can be used in Retrieval Augmented Generation (RAG) scenarios. For example, a tool can be used to retrieve the current weather for a given location, to retrieve the latest news articles, or to query a database for a specific record.



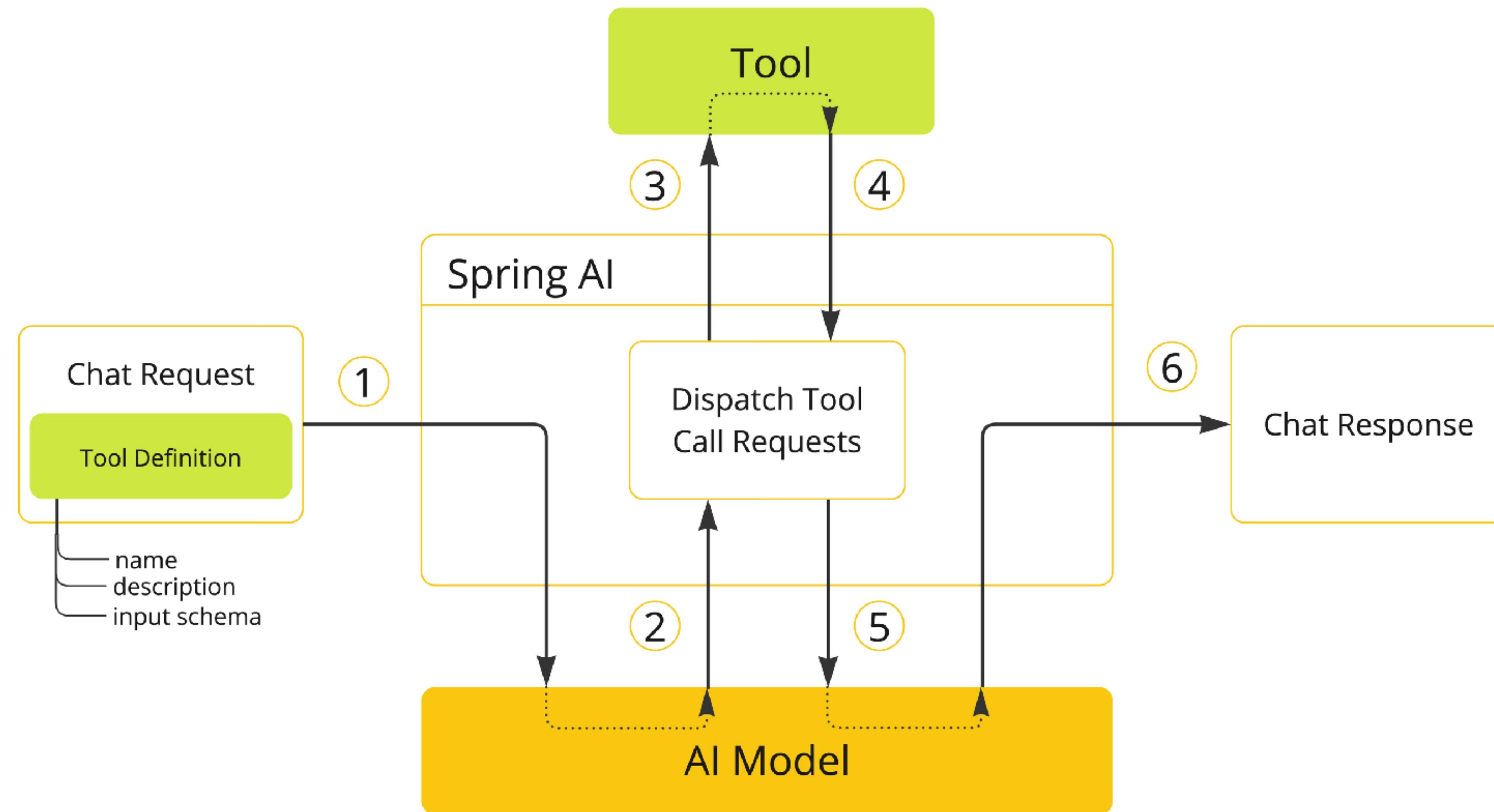
# TOOL CALLING / FUNCTION CALLING

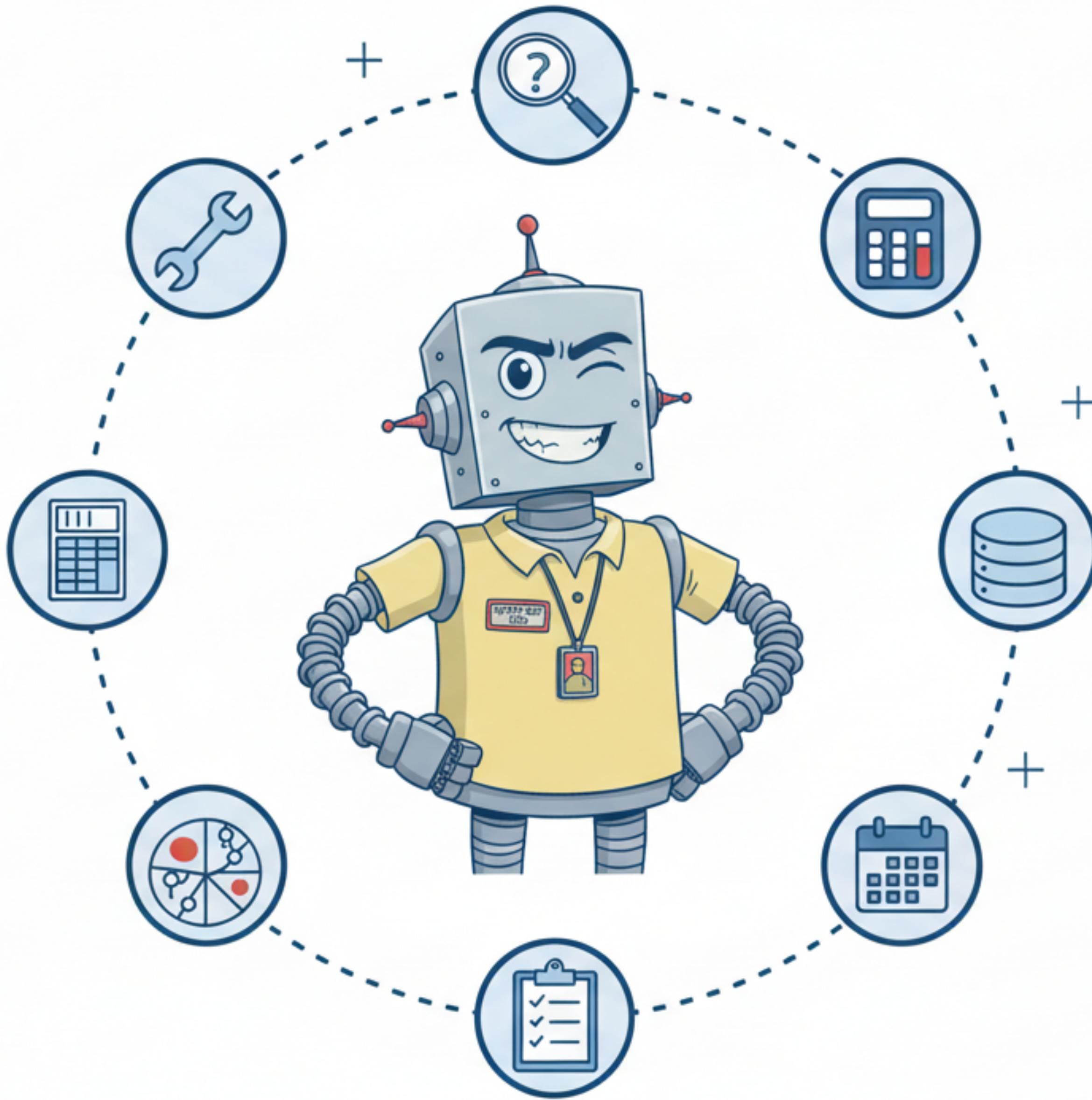
## Taking Action

Tools in this category can be used to take action in a software system, such as sending an email, creating a new record in a database, submitting a form, or triggering a workflow. The goal is to automate tasks that would otherwise require human intervention or explicit programming. For example, a tool can be used to book a flight for a customer interacting with a chatbot, to fill out a form on a web page, or to implement a Java class based on an automated test (TDD) in a code generation scenario.



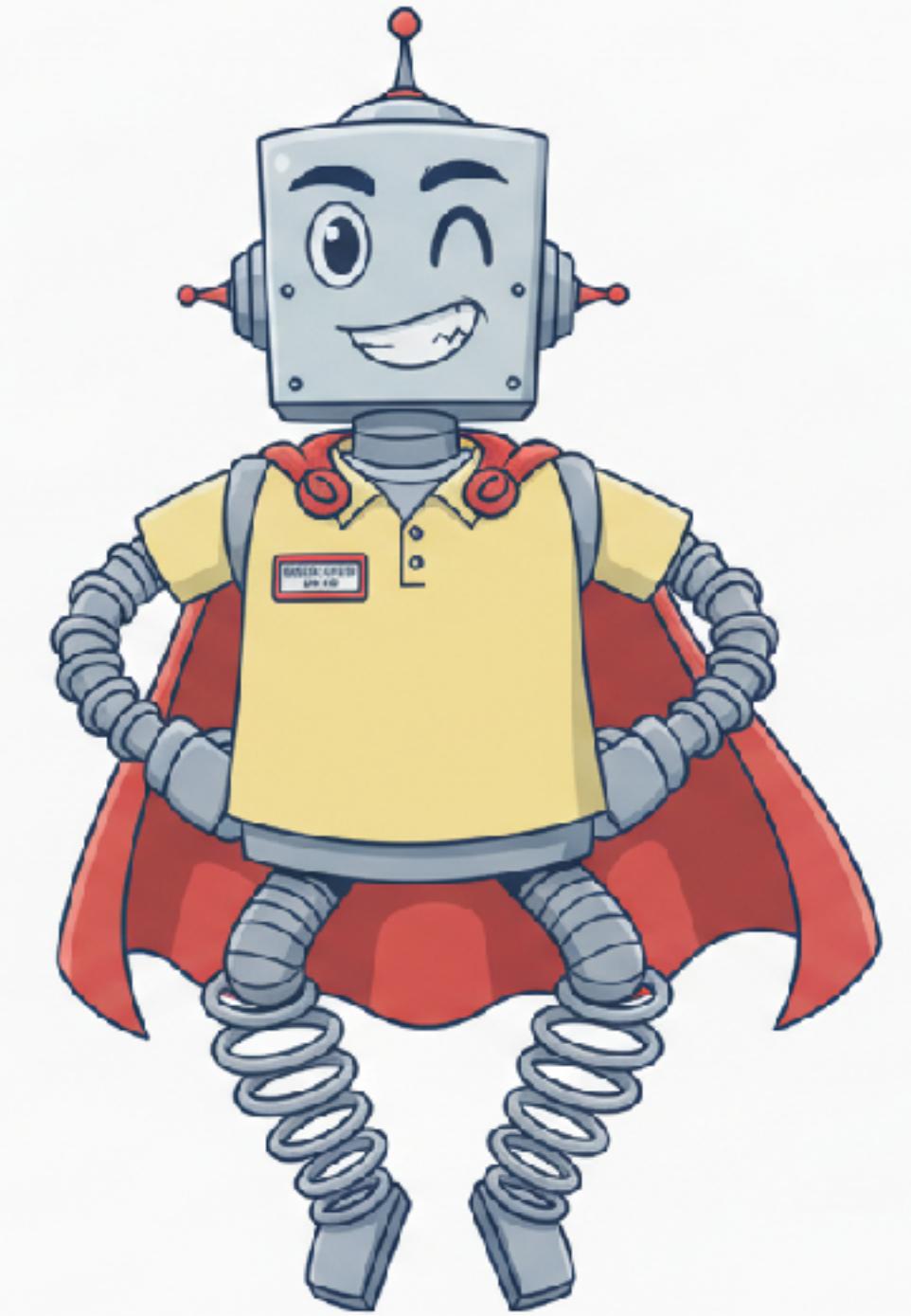
```
public class DateTimeTools {  
  
    @Tool(description = "Get the current date and time in the user's timezone")  
    String getCurrentDateTime() {  
        return LocalDateTime.now().atZone(LocaleContextHolder.getTimeZone().toZoneId()).toString();  
    }  
  
}
```





**MCP**

# **MODEL CONTEXT PROTOCOL**



# **WHAT IS MODEL CONTEXT PROTOCOL (MCP)?**

MCP is a protocol that defines how to talk to AI models in a consistent, structured way.

# **MOTIVATION**

Models are only as good as the context  
you provide them

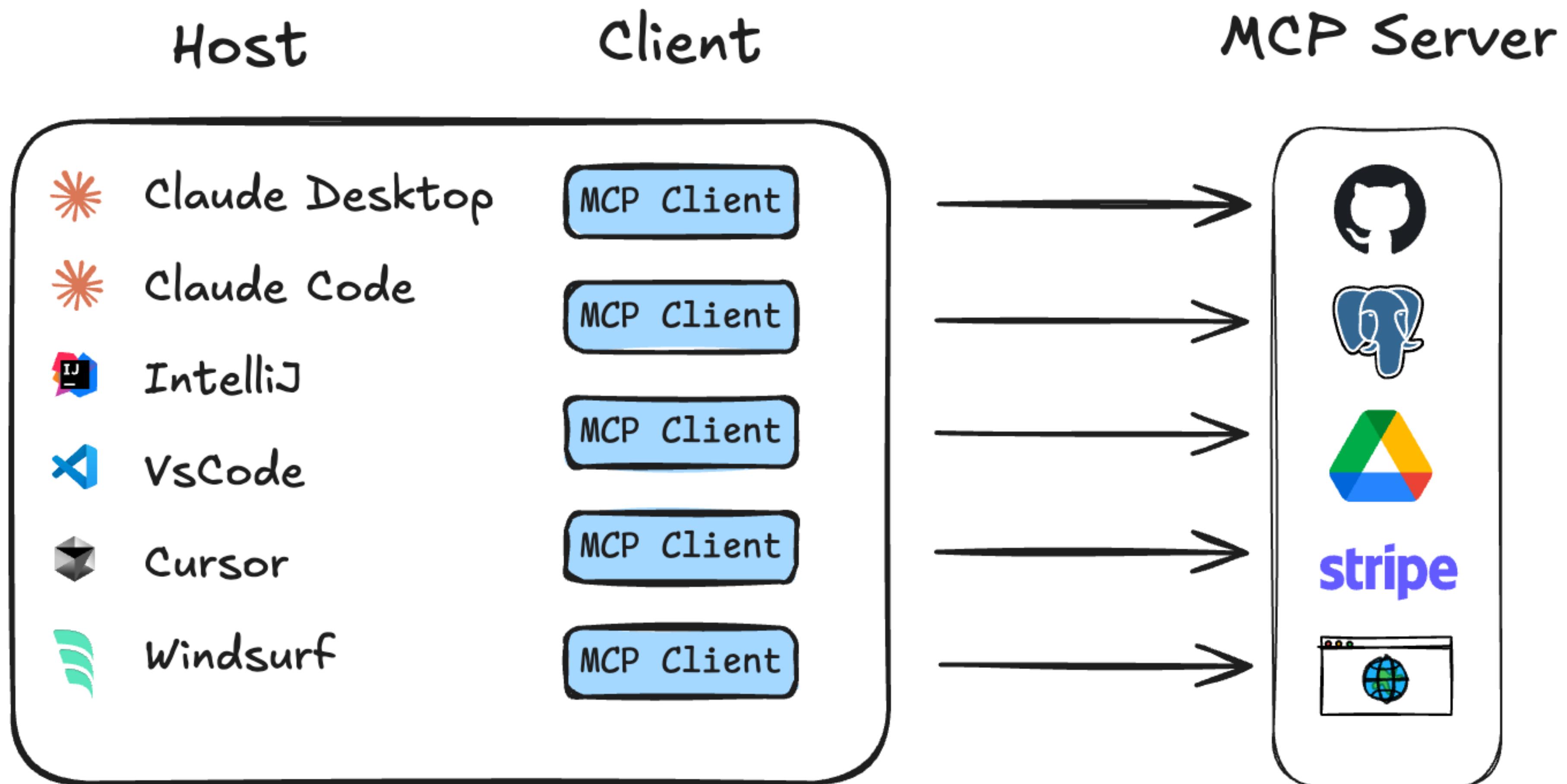
# BENEFITS OF MCP

- **Modularity:** 🧠 This lets the AI app stay light while still being deeply aware of your environment.
- **Reusability:**💡 Once you write an MCP server (say, one that surfaces project files), it can serve any app that supports the MCP protocol.
- **Language Agnostic:**💻 Works across programming languages and environments without language-specific dependencies.
- **Fine-Grained Control:**🔒 It gives you programmable control over what your AI sees and can do
- **Privacy and Security:**🛡️ You stay in control of what your AI knows.



# COMPONENTS OF MCP

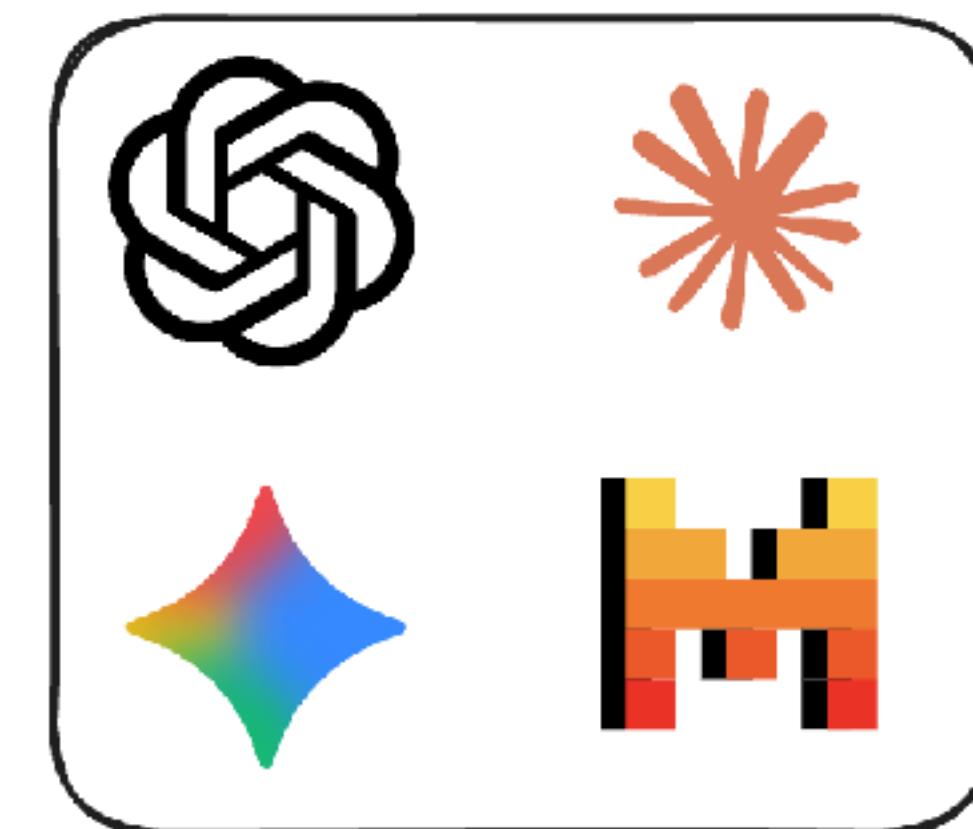
## Participants



# HOW IS AN MCP SERVER DIFFERENT THAN AN API?



Provider Agnostic



# HOW IS THIS DIFFERENT FROM TOOL USE?



# PRIMITIVES

# MCP PRIMITIVES

MCP primitives are the most important concept within MCP. They define what clients and servers can offer each other. These primitives specify the types of contextual information that can be shared with AI applications and the range of actions that can be performed.



# **MCP PRIMITIVES: TOOLS**

**What:** Executable functions that AI applications can invoke to perform actions (e.g., file operations, API calls, database queries)

**When:** AI needs to take action beyond just generating text - when it needs to DO something in the real world

**Examples:**

- `read_file()` - Read contents of a specific file
- `write_file()` - Create or modify files
- `execute_sql()` - Run database queries
- `send_email()` - Send messages via email API
- `git_commit()` - Commit changes to repository
- `slack_post()` - Send messages to Slack channels
- `web_search()` - Search the internet
- `calculate()` - Perform mathematical operations

# MCP PRIMITIVES: RESOURCES

**What:** Data sources that provide contextual information to AI applications (e.g., file contents, database records, API responses)

**When:** AI needs to understand or reference existing information before responding or taking action

## Examples:

- file://project/README.md - Documentation and project files
- database://users/profile/123 - User records and data
- git://repo/commit/history - Version control information
- slack://channel/messages - Chat history and conversations
- calendar://events/today - Schedule and meeting data
- email://inbox/recent - Email content and metadata

# MCP PRIMITIVES: PROMPTS

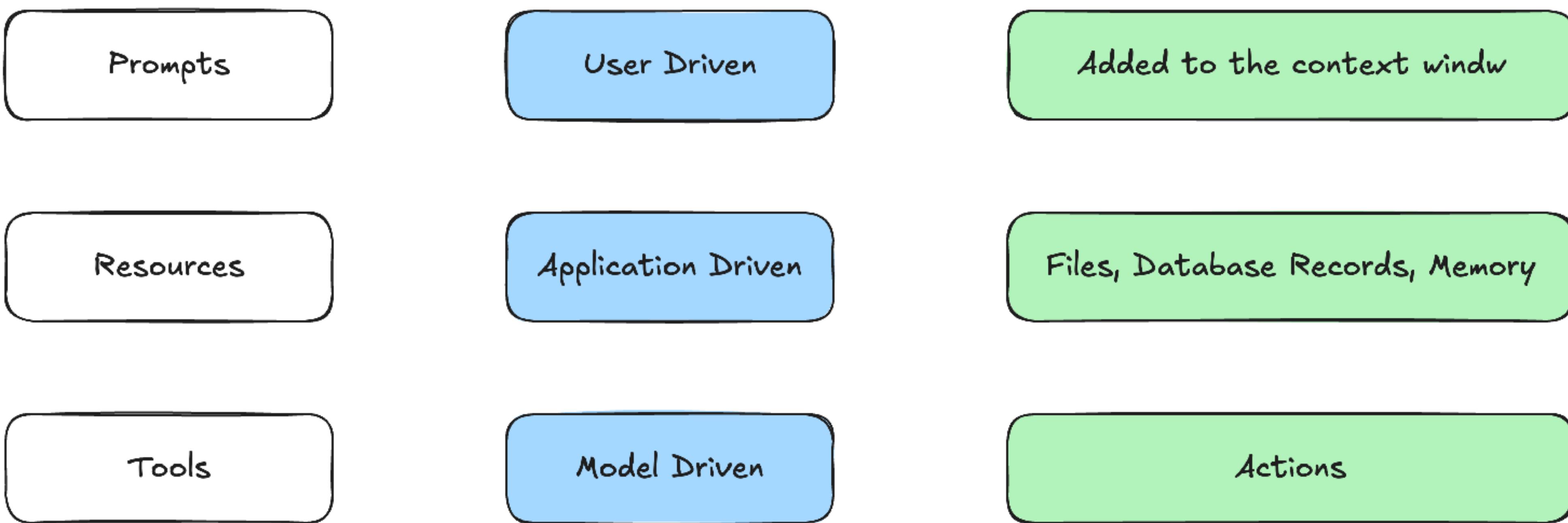
**What:** Reusable templates that help structure interactions with language models (e.g., system prompts, few-shot examples)

**When:** You need consistent, well-crafted prompts across different conversations or want to standardize AI behavior patterns

**Examples:**

- code\_reviewer - Template for reviewing pull requests with specific criteria
- meeting\_summarizer - Structured format for extracting action items from transcripts
- technical\_writer - Guidelines for creating documentation in company style
- bug\_triager - Template for categorizing and prioritizing issues
- customer\_support - Consistent tone and approach for user interactions
- data\_analyst - Framework for interpreting charts and metrics
- project\_planner - Structure for breaking down tasks and timelines
- content\_moderator - Guidelines for evaluating user-generated content

# MCP PRIMITIVES: INTERACTION MODEL



# **TRANSPORTS**

**Transports in the Model Context Protocol (MCP) provide the foundation for communication between clients and servers. A transport handles the underlying mechanics of how messages are sent and received.**

# TRANSPORTS

## Standard Input/Output (stdio)

- Building command-line tools
- Implementing local integrations
- Needing simple process communication
- Working with shell scripts

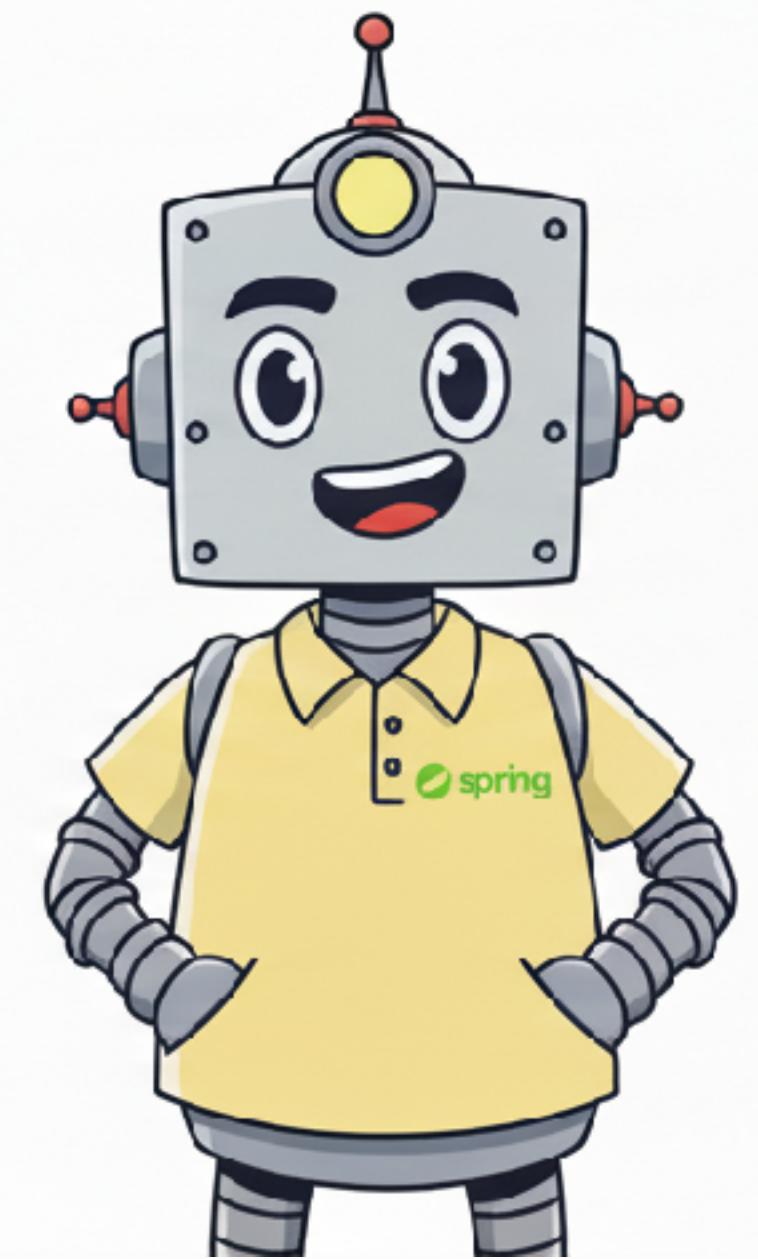
## Server-Sent Events (SSE)

- Only server-to-client streaming is needed
- Working with restricted networks
- Implementing simple updates

## Streamable HTTP

- Building web-based integrations
- Implementing bidirectional streaming
- Request / Response Streaming
- Working with modern HTTP infrastructure

# **BUILDING MCP SERVERS IN JAVA / SPRING**



[Introduction](#)[SDKs](#)[Concepts](#)[Architecture Overview](#)[Server Concepts](#)[Client Concepts](#)[Versioning](#)[Tutorials](#)[Using MCP >](#)[Server Development >](#)[Client Development >](#)[FAQs](#)

## SDKs

[!\[\]\(afbc970bc3457ad2e60b755a2604bcfb\_img.jpg\) Copy page](#)

Official SDKs for building with the Model Context Protocol

Build MCP servers and clients using our official SDKs. Choose the SDK that matches your technology stack - all SDKs provide the same core functionality and full protocol support.

## Available SDKs

[TypeScript](#)[Python](#)[Go](#)[Kotlin](#)[Swift](#)[Java](#)[C#](#)[Ruby](#)[Rust](#)

**Project**

- Gradle - Groovy     Gradle - Kotlin  
 Maven

**Language**

- Java     Kotlin     Groovy

**Spring Boot**

- 4.0.0 (SNAPSHOT)     4.0.0 (M1)     3.5.5 (SNAPSHOT)     3.5.4  
 3.4.9 (SNAPSHOT)     3.4.8

**Project Metadata**

Group dev.danvega

Artifact kcdc-mcp

Name kcdc-mcp

Description Demo project for Spring Boot

Package name dev.danvega.kcdc

Packaging  Jar     War

Java  24     21     17

**Dependencies**

**ADD DEPENDENCIES... ⌘ + B**

**Spring Web** WEB

Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

**Model Context Protocol Server** AI

Spring AI support for Model Context Protocol (MCP) servers.



**GENERATE** ⌘ + ↵

**EXPLORE** CTRL + SPACE

...

# **BUILDING AN MCP SERVER IN JAVA / SPRING**

## **Spring Programming Model**

**With some additional MCP APIs**

- Tools
- Prompts
- Resources

```
@Component
public class SessionTools {

    private static final Logger log = LoggerFactory.getLogger(SessionTools.class);
    private Conference conference;
    private final ObjectMapper objectMapper;

    public SessionTools(ObjectMapper objectMapper) {
        this.objectMapper = objectMapper;
    }

    @McpTool(name = "cyc-get-conference-data", description = "Get all conference data including sessions, tracks, rooms and conference details")
    public Conference getConferenceData() {
        return conference;
    }

    @PostConstruct
    public void init() {
        log.info("Loading Sessions from JSON file 'sessions.json'");
        try (InputStream inputStream = TypeReference.class.getResourceAsStream("/data/sessions.json")) {
            var jsonNode = objectMapper.readTree(inputStream);
            var conferenceNode = jsonNode.get("conference");
            this.conference = objectMapper.treeToValue(conferenceNode, Conference.class);
            log.info("Sessions loaded successfully: {} sessions",
                    conference.sessions() != null ? conference.sessions().size() : 0);
        } catch (IOException e) {
            log.error("Failed to read JSON data", e);
            throw new RuntimeException("Failed to read JSON data", e);
        }
    }
}
```

```
@SpringBootTest
class SessionToolsTest {

    @Autowired
    private SessionTools sessionTools;

    @Test
    void getConferenceDataReturnsValidData() {
        Conference conference = sessionTools.getConferenceData();

        assertThat(conference).isNotNull();
        assertThat(conference.name()).isEqualTo("Commit Your Code Conference (CYC25)");
        assertThat(conference.year()).isEqualTo(2025);
        assertThat(conference.dates()).containsExactly("2025-09-25", "2025-09-26");
        assertThat(conference.location()).isEqualTo("Yum! Brands International, Plano, TX");

        assertThat(conference.tracks()).isNotEmpty();
        assertThat(conference.tracks()).contains("JavaScript", "Java", "Cloud", ".NET", "Leadership", "AI");

        assertThat(conference.rooms()).isNotEmpty();
        assertThat(conference.rooms()).contains("Red Room", "Yellow Room");

        assertThat(conference.sessions()).isNotEmpty();

        Session firstSession = conference.sessions().get(0);
        assertThat(firstSession).isNotNull();
        assertThat(firstSession.title()).isNotBlank();
        assertThat(firstSession.day()).isNotBlank();
        assertThat(firstSession.time()).isNotBlank();
        assertThat(firstSession.speakers()).isNotNull();

        boolean hasSessionWithSpeakers = conference.sessions().stream()
            .anyMatch(session → session.speakers() ≠ null & session.speakers().length > 0);
        assertThat(hasSessionWithSpeakers).isTrue();
    }
}
```

# **TESTING YOUR MCP SERVERS**

- Create an executable JAR
- Test using an MCP Client
  - Spring MCP Client
  - Claude Desktop
  - Cursor / Windsurf / Junie
  - Any MCP Client of your choice

## Transport Type

STDIO

## Command

/Users/vega/.sdkman/candidates/

## Arguments

-jar /Users/vega/Downloads/spri

&gt; Environment Variables

[Server Entry](#)[Servers File](#)

&gt; Configuration

[Restart](#)[Disconnect](#)

Connected

## Logging Level

debug

System



## Tools

[List Tools](#)[Clear](#)

## spring-io-sessions

Returns all sessions for Spring I/O 2025 Conference

## spring-io-sessions

Returns all sessions for Spring I/O 2025 Conference

[Run Tool](#)

## Tool Result: Success

```
[  
  0: {  
    day: "2025-05-22"  
    time: "08:00"  
    title: "Registration"  
    type: "logistics"  
    speakers: []  
    room: "General"  
  }  
  1: {  
    day: "2025-05-22"  
    time: "09:00"  
    title: "Welcome"  
    type: "opening"  
    speakers: [  
      {  
        name: "John Doe",  
        bio: "John is a logistics expert..."  
      },  
      {  
        name: "Jane Smith",  
        bio: "Jane is a welcome committee..."  
      }  
    ]  
  }]
```

## History

6. ping

5. initialize

4. tools/call

3. tools/list

2. resources/list

## Server Notifications

No notifications yet

# OBSERVABILITY





```
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-actuator</artifactId>
</dependency>
```

# **WHY OBSERVABILITY**

- Cost can spike → need token & \$ metrics
- LLMs are nondeterministic → traces & logs for debug
- Safety / legal → need evidence when things go wrong

# **THE 3+1 PILLARS OF OBSERVABILITY**

- Metrics – latency, token counts, cost
- Logs – structured prompt / response (PII-redacted)
- Traces – end-to-end view of RAG pipeline
- Evaluations – automated quality checks (FactCheckingEvaluator)

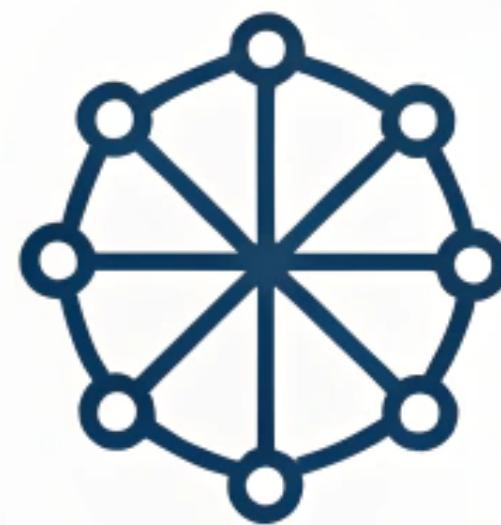
# **WHAT SPRING GIVES YOU FOR FREE**

- Micrometer metric names (gen\_ai.usage.\* , gen\_ai.client.latency)  
[docs.spring.io](https://docs.spring.io)
- Observation events → Micrometer Tracing / OpenTelemetry
- Pluggable log masking (PII filter)
- GenerationListener hook for custom metrics

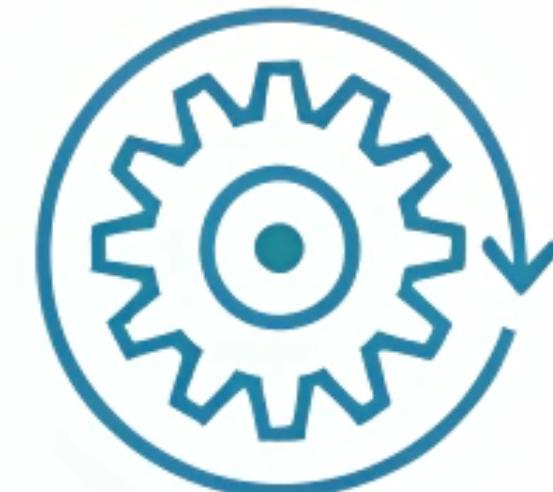
# OPEN SOURCE MODELS



# Open Source AI Model Checklist



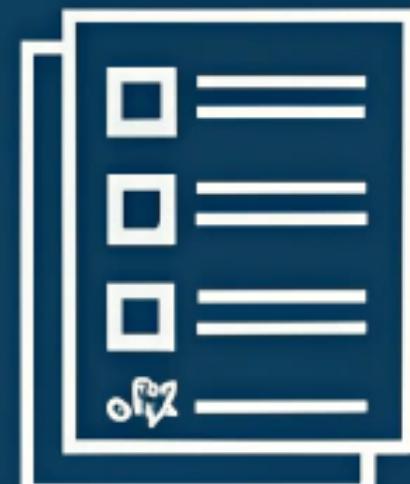
Model Weights



Source Code



Training Data



OSI

# **OPEN SOURCE MODELS - KEY CHALLENGES**

- Performance Gap vs Frontier Models
- Up-Front Hardware & Ops Overhead
- No Vendor-Grade Support or SLA
- Safety & Quality Risks (Hallucinations/Bias)
- Security & Supply-Chain Exposure
- License & IP Ambiguity
- Benchmark / Eval Burden Rests on You
- Rapid-Fire Releases - Maintenance Churn

# **WHY OPEN SOURCE MODELS**

- Transparency & Auditability
- Self-Hosting for Data Privacy & Compliance
- Lower Total Cost (No API metering)
- Low-Latency Local Development
- Full Customization & Fine-Tuning Rights
- Community-Driven Innovation & Fast Patch Cycles
- Deployment Flexibility (cloud, on-prem, edge)
- No Vendor Lock-In

# HOW TO EVALUATE OPEN SOURCE MODELS

- Licence & “Openness” – OSI-approved? any commercial-use limits?
- Capability Benchmarks – MMLU, HellaSwag, GSM-8K scores; check the Open LLM Leaderboard
- Tool / Function-Calling Support – built-in schemas for agents & Spring AI's callable() interface
- Size vs Hardware Budget – parameter count, VRAM need, quantised variants
- Context Window – 8 K, 128 K... even 1 M tokens for long-doc RAG
- Latency & Throughput – tokens/sec locally vs on-prem GPU or HF Endpoint
- Community & Release Cadence – active issues, model-card updates, patch frequency
- Security & Supply Chain – reproducible weights, SBOM, no suspicious commits



# Ollama

<https://ollama.com>

Search models

All

Embedding

Vision

Tools

Popular

## deepseek-r1

DeepSeek's first-generation of reasoning models with comparable performance to OpenAI-o1, including six dense models distilled from DeepSeek-R1 based on Llama and Qwen.

1.5b 7b 8b 14b 32b 70b 671b

14.7M Pulls 29 Tags Updated 5 days ago

## llama3.3

New state of the art 70B model. Llama 3.3 70B offers similar performance compared to the Llama 3.1 405B model.

tools 70b

1.2M Pulls 14 Tags Updated 2 months ago

## phi4

Phi-4 is a 14B parameter, state-of-the-art open model from Microsoft.

14b

454.4K Pulls 5 Tags Updated 5 weeks ago

## llama3.2

Meta's Llama 3.2 goes small with 1B and 3B models.

tools 1b 3b

8.6M Pulls 63 Tags Updated 4 months ago

## llama3.1

Llama 3.1 is a new state-of-the-art model from Meta available in 8B, 70B and 405B parameter sizes.

# Docker Model Runner

Page options ▾

**Availability:** Beta 

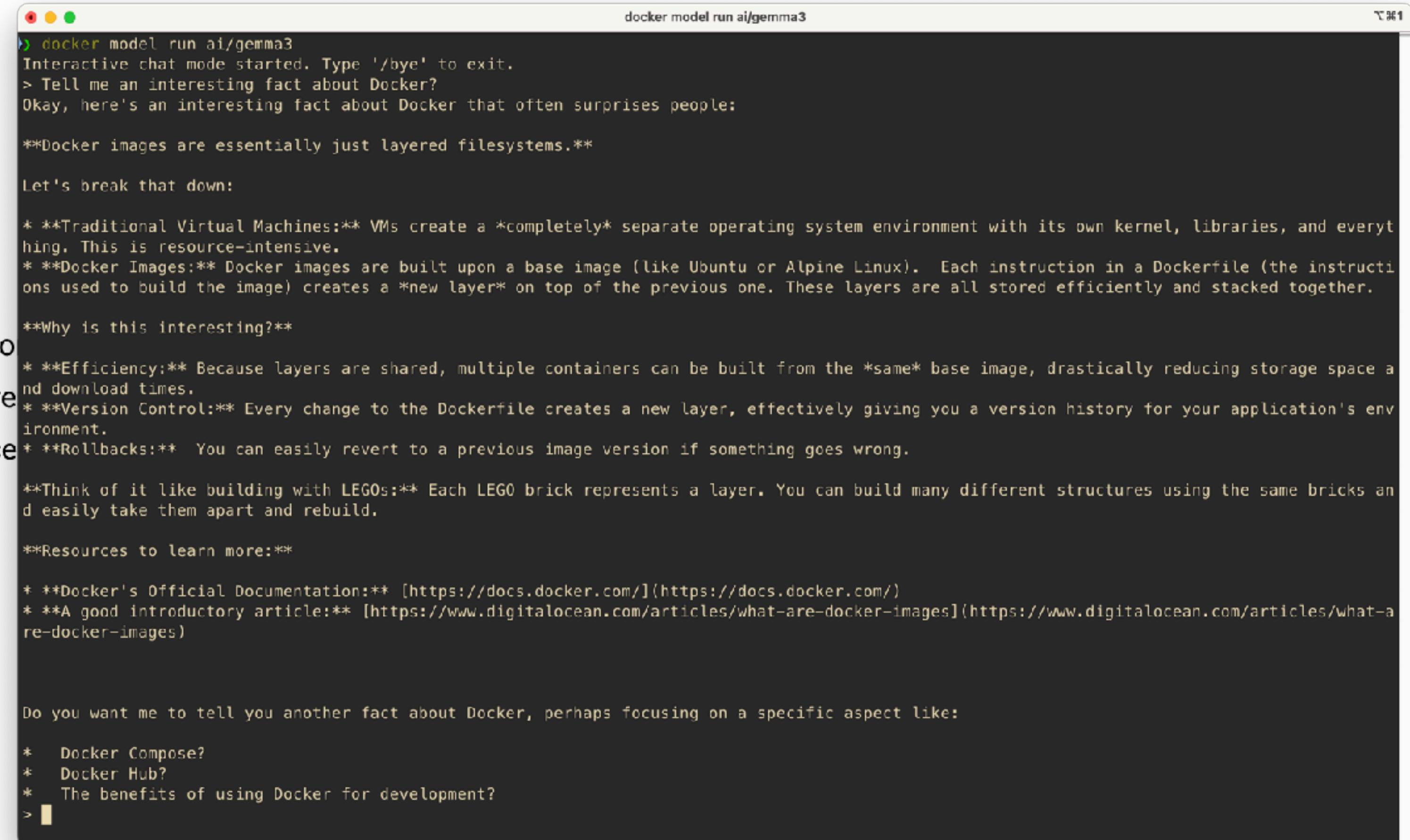
**Requires:** Docker Desktop 4.40 and later 

**For:** Docker Desktop for Mac with Apple Silicon

The Docker Model Runner plugin lets you:

- [Pull models from Docker Hub](#) 
- Run AI models directly from the command line
- Manage local models (add, list, remove)
- Interact with models using a submitted prompt or in chat mode

Models are pulled from Docker Hub the first time they're used and stored when a request is made, and unloaded when not in use to optimize resources. This takes some time — but after that, they're cached locally for faster access via [compatible APIs](#).



```
docker model run ai/gemma3
Interactive chat mode started. Type '/bye' to exit.
> Tell me an interesting fact about Docker?
Okay, here's an interesting fact about Docker that often surprises people:
**Docker images are essentially just layered filesystems.**

Let's break that down:
* **Traditional Virtual Machines:** VMs create a *completely* separate operating system environment with its own kernel, libraries, and everything. This is resource-intensive.
* **Docker Images:** Docker images are built upon a base image (like Ubuntu or Alpine Linux). Each instruction in a Dockerfile (the instructions used to build the image) creates a *new layer* on top of the previous one. These layers are all stored efficiently and stacked together.

**Why is this interesting?**
* **Efficiency:** Because layers are shared, multiple containers can be built from the *same* base image, drastically reducing storage space and download times.
* **Version Control:** Every change to the Dockerfile creates a new layer, effectively giving you a version history for your application's environment.
* **Rollbacks:** You can easily revert to a previous image version if something goes wrong.

**Think of it like building with LEGOs:** Each LEGO brick represents a layer. You can build many different structures using the same bricks and easily take them apart and rebuild.

**Resources to learn more:**
* **Docker's Official Documentation:** [https://docs.docker.com/](https://docs.docker.com/)
* **A good introductory article:** [https://www.digitalocean.com/articles/what-are-docker-images](https://www.digitalocean.com/articles/what-are-docker-images)

Do you want me to tell you another fact about Docker, perhaps focusing on a specific aspect like:
* Docker Compose?
* Docker Hub?
* The benefits of using Docker for development?
> █
```

# Open WebUI



A screenshot of a web-based AI interface titled "OpenAI / GPT 4". The interface has a dark background with a grid pattern. At the top, there's a sidebar with options like "New Chat", "Workspace", and "Search". A message from "OpenAI / GPT 4" says "SUCCESS Open WebUI - On a mission to build the best open-source AI user interface." Below this, there's a large "How can I help you today?" prompt with a "Suggested" section. Four cards are visible: "Help me study" (vocabulary for a college entrance exam), "Give me ideas" (for what to do with my kids' art), "Overcome procrastination" (give me tips), and "Tell me a fun fact" (about the Roman Empire). At the bottom, there's a "Send a Message" button and a note about LLMs making mistakes.

<https://openwebui.com>

The screenshot shows the lmstudio interface with the following components:

- Top Bar:** Chats, File, New, Settings, Model (lmstudio-community/Meta-Llama-3.1-8B-Instruct-GGUF), Eject, Advanced Configuration.
- Left Sidebar:** Secret project, C++ Filesystem... (1600 tokens), log about version of ... (56 tokens), Financial analysis (106 tokens).
- Central Area:** C++ Filesystem Implementation section with Assistant (Meta-Llama-3.1-8B-Instruct-GGUF). The title is "Filesystem Implementation in C++".
  - Text:** Before we begin, let's outline our approach:
    1. We'll create a basic directory structure with methods for creating directories, deleting directories, listing files and directories, and navigating through the directory tree.
    2. We'll implement file operations such as creating, reading, writing, and deleting files.
    3. We'll handle errors and exceptions to ensure robustness.
  - Section:** Filesystem Class
  - Code:**

```
#include <iostream>
#include <fstream>
#include <string>
#include <vector>
#include <dirent.h>

class Filesystem {
public:
    // Constructor
    Filesystem(const std::string& rootDir) : rootDir_(rootDir) {}

    // Create a new directory
    void createDirectory(const std::string& path);
```
  - Text Input:** Type a message and press Enter to send ...
- Right Sidebar:** Preset (Discard Unsaved), Coding Helper (C++) (Commit Changes, Save As New...), System Prompt (You are an incredibly good C++ engineer. Think about the problems you're about to solve step-by-step. First make a plan, and then ask the user to confirm. Only then act on it. Token count: 41), Settings, Sampling, Structured Output.



# The AI community building the future.

The platform where the machine learning community collaborates on models, datasets, and applications.

[Explore AI Apps](#)[or Browse 1M+ models](#)[Tasks](#) [Libraries](#) [Datasets](#) [Languages](#) [Licenses](#) [Other](#)[Filter Tasks by name](#)

Multimodal

[Text-to-Image](#) [Image-to-Text](#)  
[Text-to-Video](#) [Visual Question Answering](#)  
[Document Question Answering](#) [Graph Machine Learning](#)

Computer Vision

[Depth Estimation](#) [Image Classification](#)  
[Object Detection](#) [Image Segmentation](#)  
[Image-to-Image](#) [Unconditional Image Generation](#)  
[Video Classification](#) [Zero-Shot Image Classification](#)

Natural Language Processing

[Text Classification](#) [Token Classification](#)  
[Table Question Answering](#) [Question Answering](#)  
[Zero-Shot Classification](#) [Translation](#)  
[Summarization](#) [Conversational](#)  
[Text Generation](#) [Text2Text Generation](#)  
[Sentence Similarity](#)

Audio

[Text-to-Speech](#) [Automatic Speech Recognition](#)  
[Audio-to-Audio](#) [Audio Classification](#)  
[Voice Activity Detection](#)

Tabular

[Tabular Classification](#) [Tabular Regression](#)

Reinforcement Learning

[Reinforcement Learning](#) [Robotics](#)

Models 469,541

[Filter by name](#)[meta-llama/Llama-2-70b](#)

Text Generation • Updated 4 days ago • 25.2k • 64

[stabilityai/stable-diffusion-xl-base-0.9](#)

Updated 6 days ago • 2.01k • 393

[openchat/openchat](#)

Text Generation • Updated 2 days ago • 1.3k • 136

[illyasviel/ControlNet-v1-1](#)

Updated Apr 26 • 1.87k

[cerspense/zeroscope\\_v2\\_XL](#)

Updated 3 days ago • 2.66k • 334

[meta-llama/Llama-2-13b](#)

Text Generation • Updated 4 days ago • 328 • 64

[tiiuae/falcon-40b-instruct](#)

Text Generation • Updated 27 days ago • 288k • 899

[WizardLM/WizardCoder-15B-V1.0](#)

Text Generation • Updated 3 days ago • 12.5k • 332

[CompVis/stable-diffusion-v1-4](#)

Text-to-Image • Updated about 17 hours ago • 448k • 5.72k

[stabilityai/stable-diffusion-2-1](#)

Text-to-Image • Updated about 17 hours ago • 782k • 2.81k

[Salesforce/xgen-7b-8k-inst](#)

Text Generation • Updated 4 days ago • 6.18k • 57



CHECK OUT MY DEMO