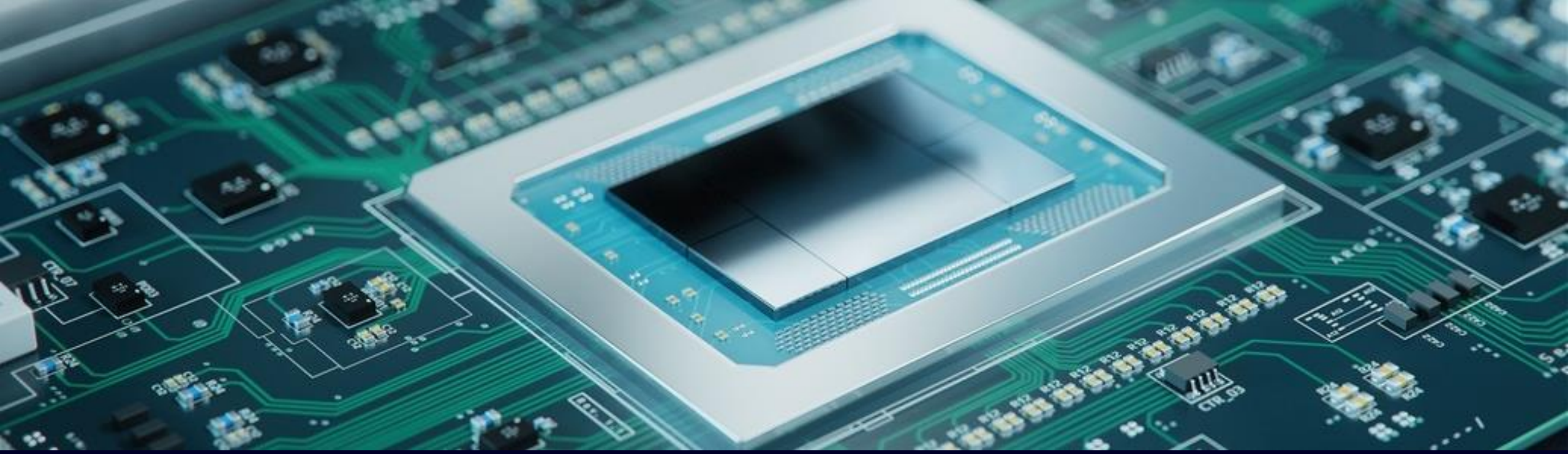


Agentic AI For RTL Signoff

Gen AI for Chip Design Flows using Questa Toolkit

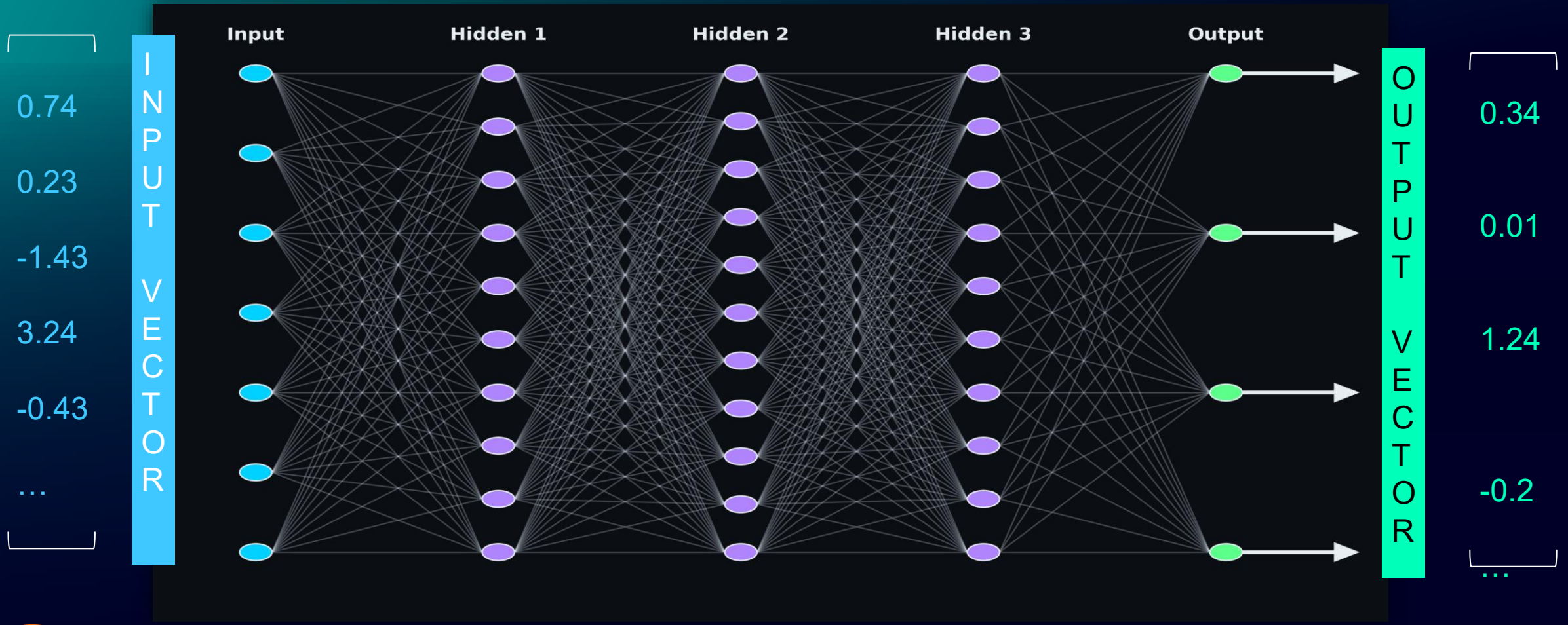
Ronen Shoham
Siemens EDA,Digital Verification Technology (DVT)
DVCon 2026

2026 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES
SANTA CLARA, CA, USA
March 2 - 5, 2026

SIEMENS

Large Language Models

At the core of an LLMs is a Deep Neural Network (DNN) – Processing number vectors



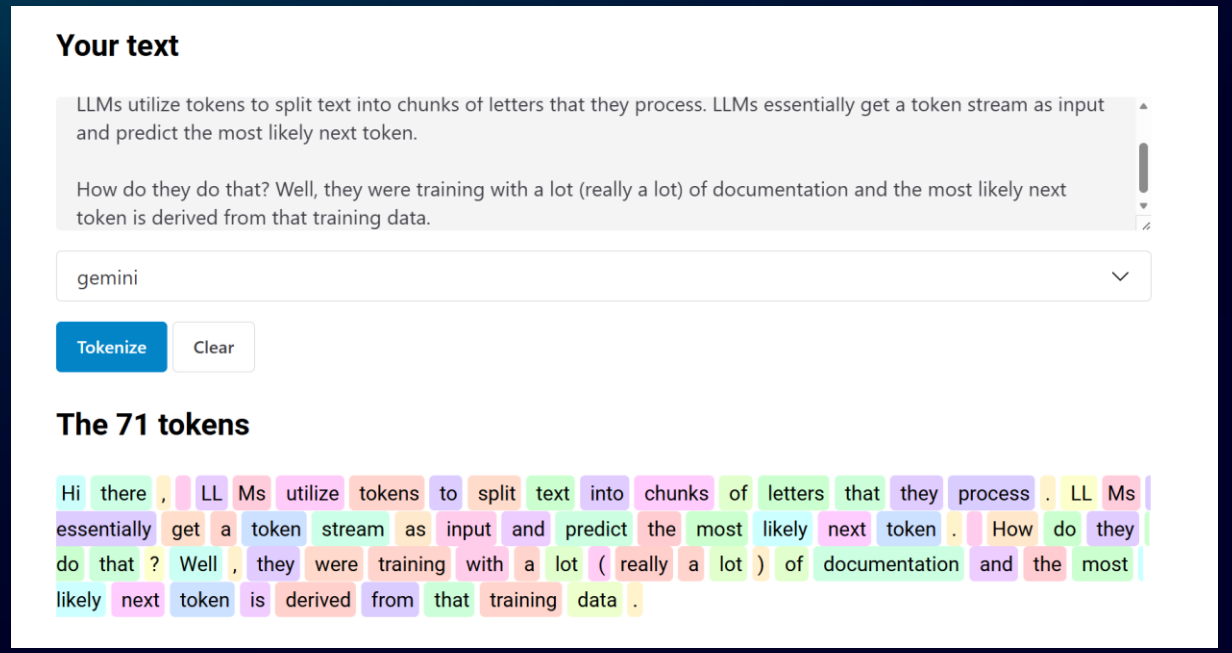
DNNs rely on large-scale parallel math, which GPUs are optimized to perform far faster than CPUs

SIEMENS

From language to numbers: begin by tokenizing the input

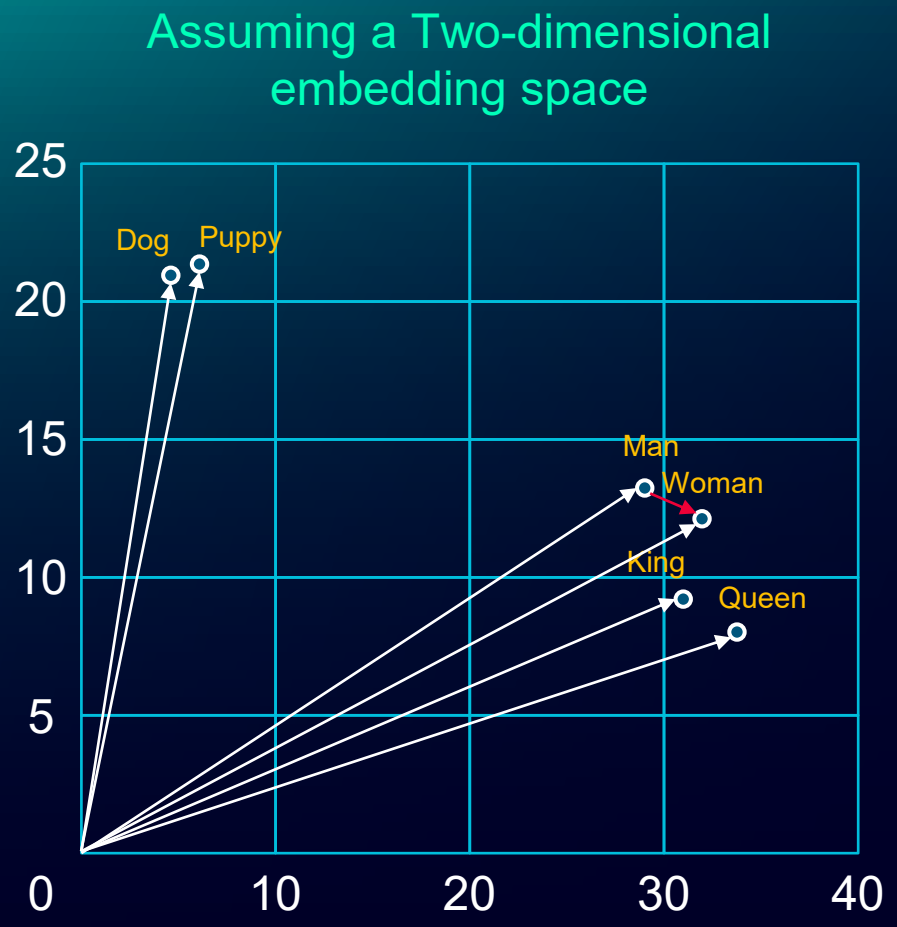
We use tokens because whole words are too many and too unpredictable.

Tokenizing breaks text into **smaller** pieces, which keeps the vocabulary manageable and lets the model handle **new words**, **typos**, and **slang**.



Token visualizer from: (https://tokens-lpj6s2duga-ew.a.run.app/)

From language to numbers: derive semantic meaning using embeddings



LLMs transform tokens into a N-dimensional vector, called embedding

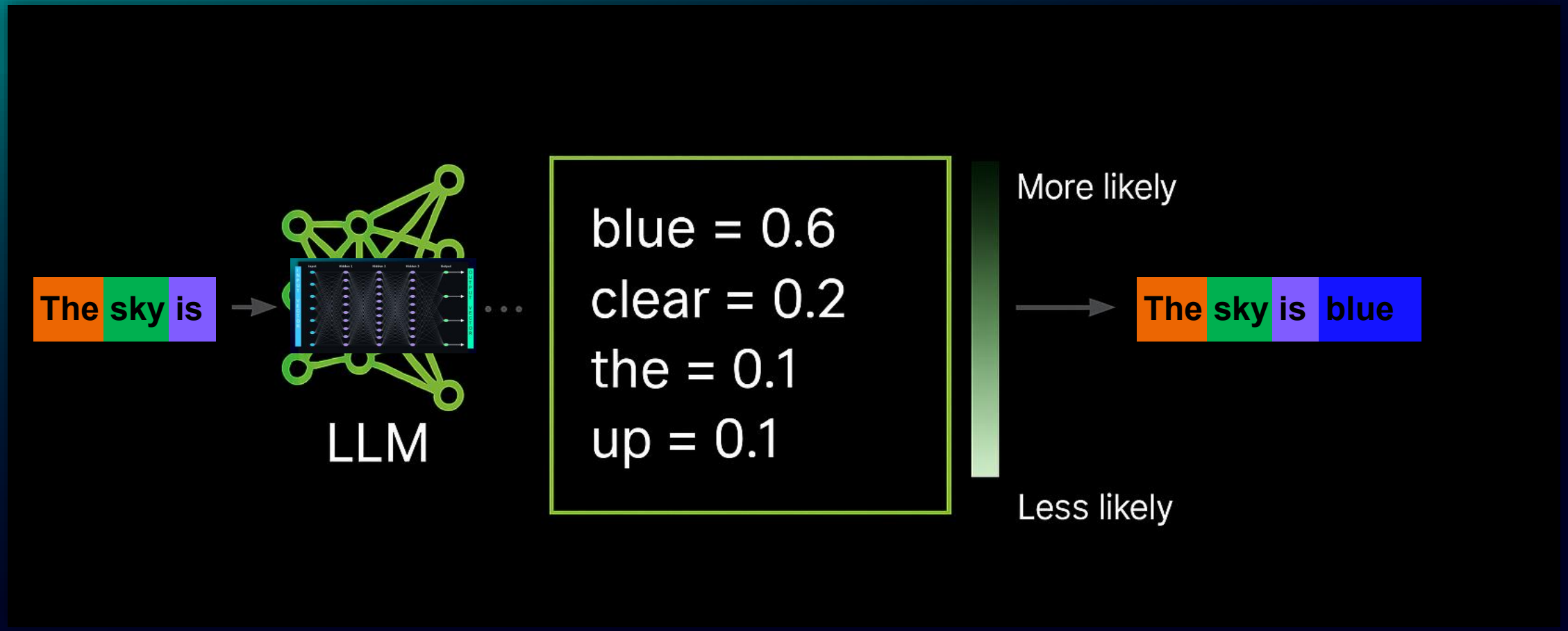
(e.g. GPT-4 has 3,072 dimensional embedding)

The embedding represents the ***semantic*** of the word and is used to understand the similarity and relationship between two words.

{King} – {Man} + {Woman} ~ {Queen}

Text embeddings are a further specialization that represent text-level meaning as vectors, at granularities from phrase to document

With tokenizer and embeddings, LLM can predicting the next token



LLMs charge by tokens: you pay for both input and output tokens, so longer prompts and longer answers cost more

LLM alone cannot have a useful conversation

What is my name?

I don't know your name

My name is Joe

Hi Joe

What is my name?

I don't know your name

My name is Joe
What is my name?

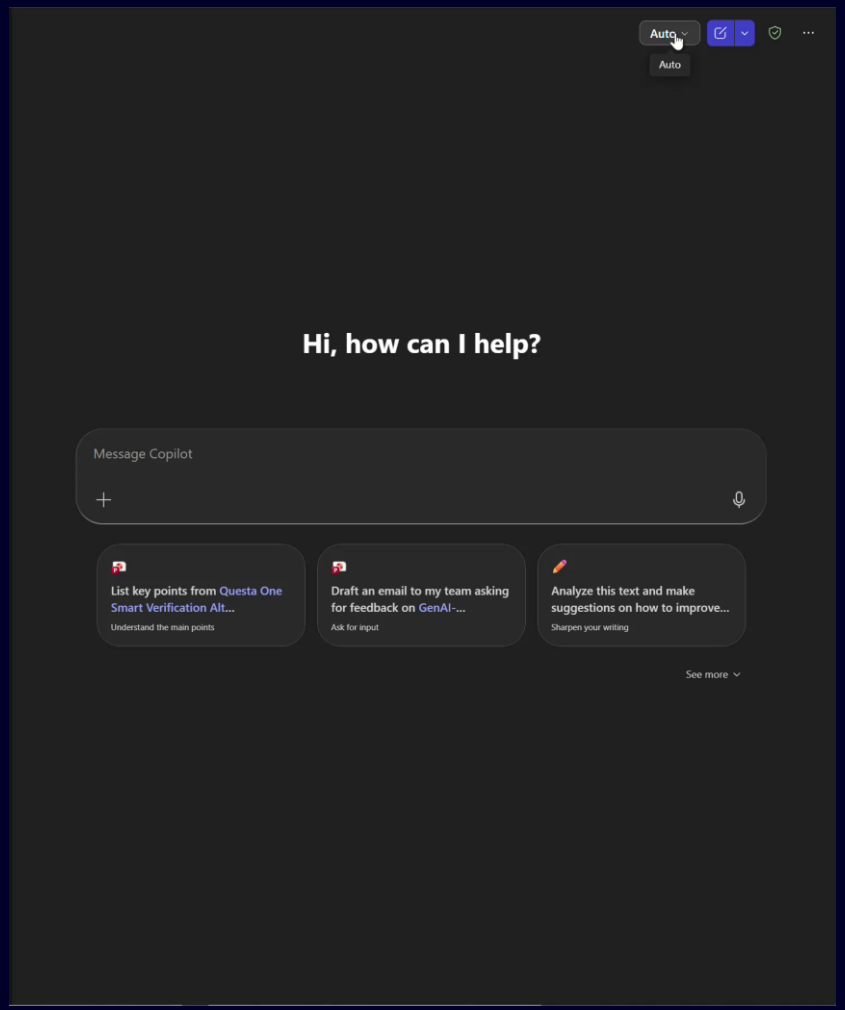
Your name is Joe

LLMs have no memory or state, everything is sent using the prompt!

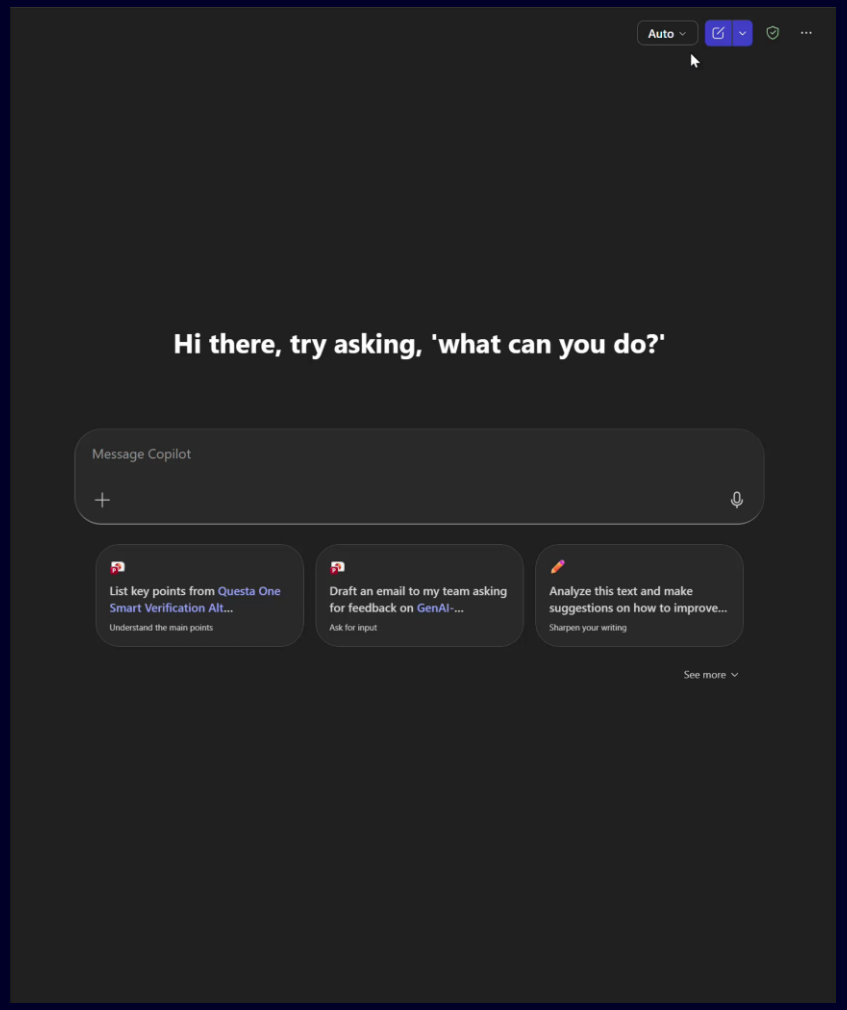
Beyond just LLMs

Chats, Workflows, and Agents

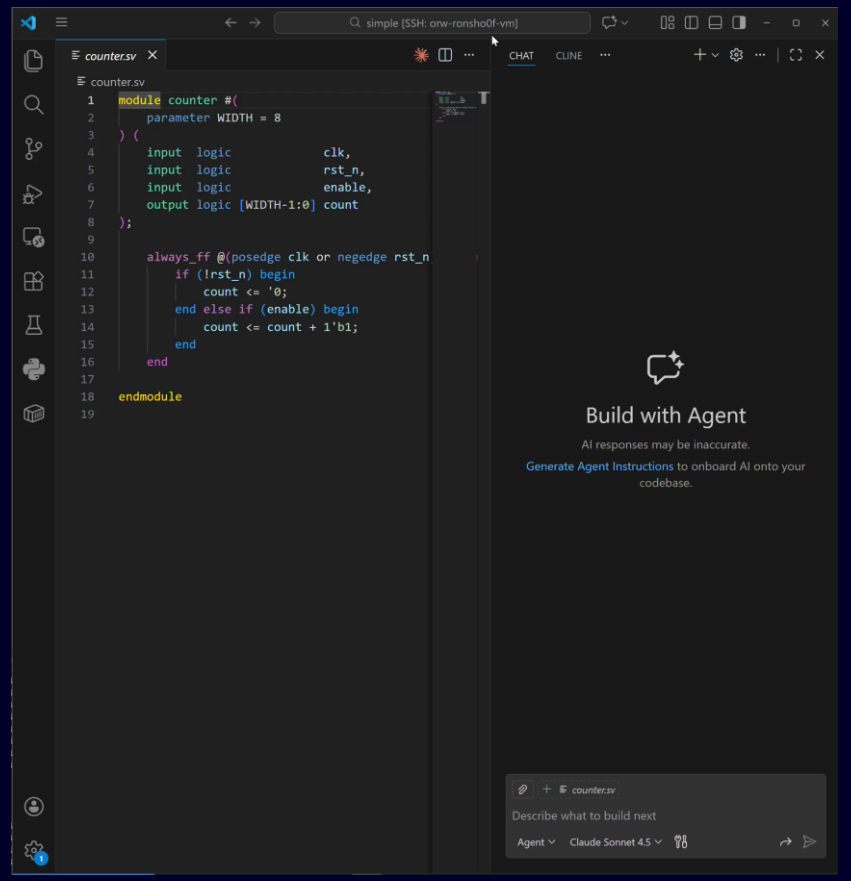
Chat



Copilot Deep Think

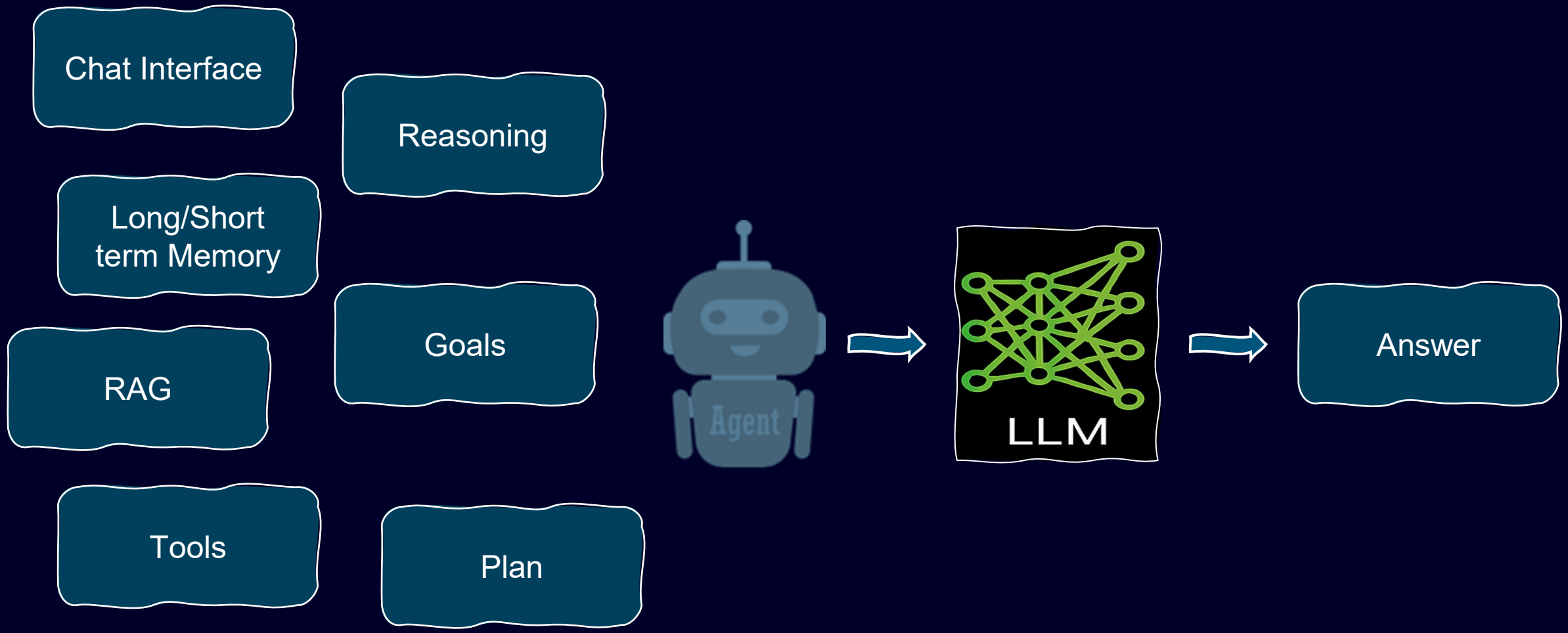


Agentic Platform



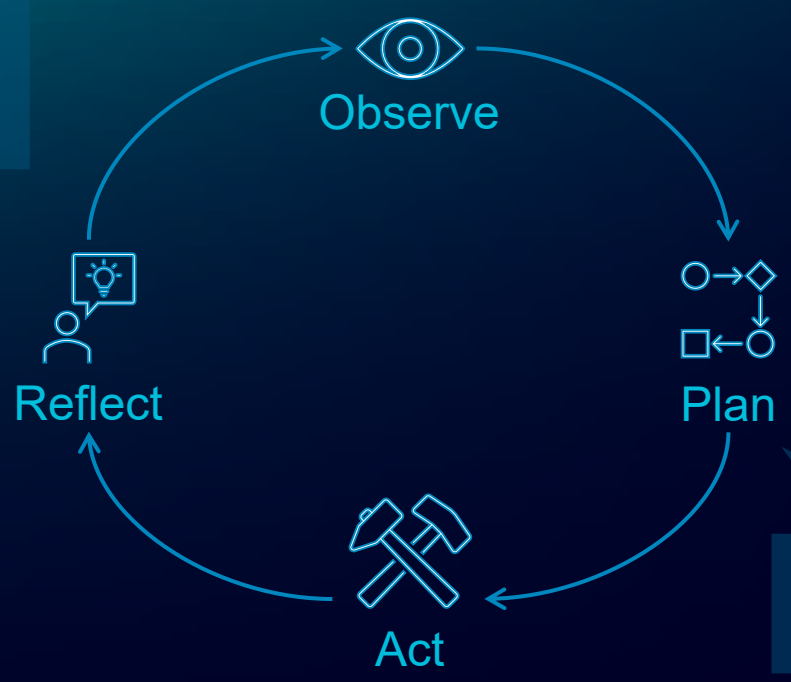
SIEMENS

What is an Agent? Software that interacts with LLM (and Tools?)



Gen AI agent is loosely defined and interpreted differently across contexts

The Agent Loop



Uses available data to form observations that guide the next step

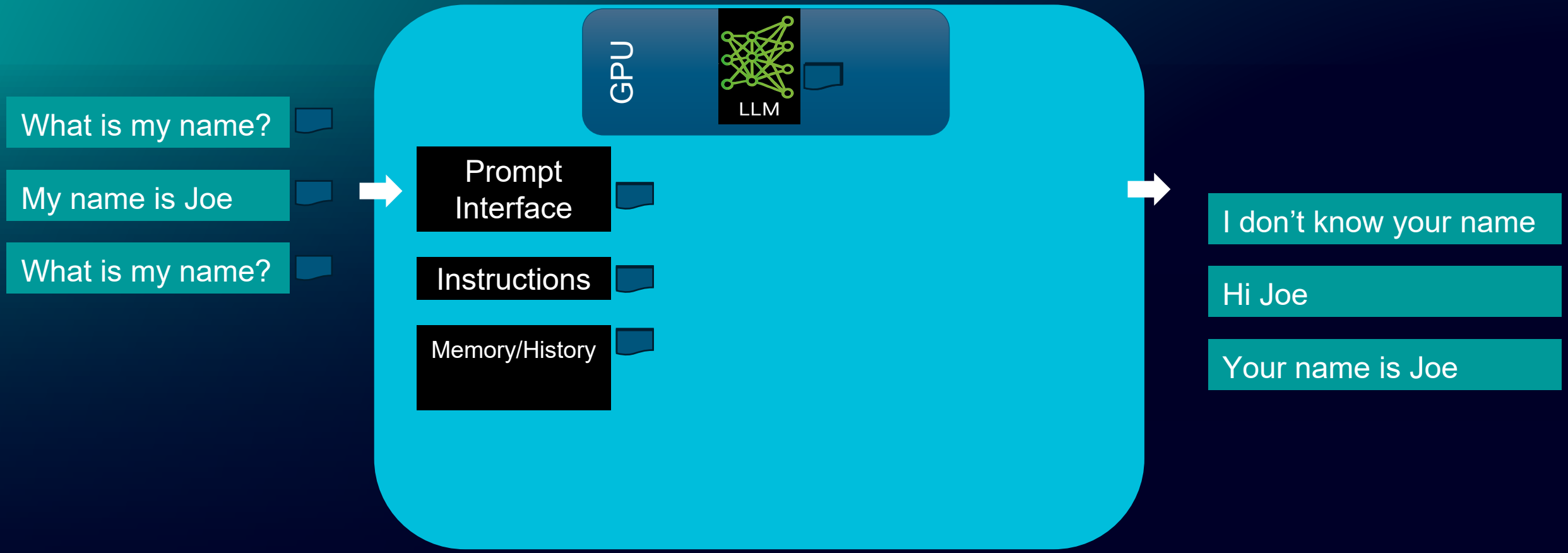
Planning can be done by reasoning (using LLM) or hard coded

External systems are engaged via tools, and their results are captured

Feedback is incorporated, and a decision is made to revise, store a lesson, stop, or request user input/approval before proceeding

	Chat	Deep Think	Agentic Platform
Observe	✓	✓	✓
Plan		✓	✓
Act			✓
Reflect		✓	✓

Adding memory: now we can have a conversation



Memory management in an LLM-based flow provides continuity across interactions

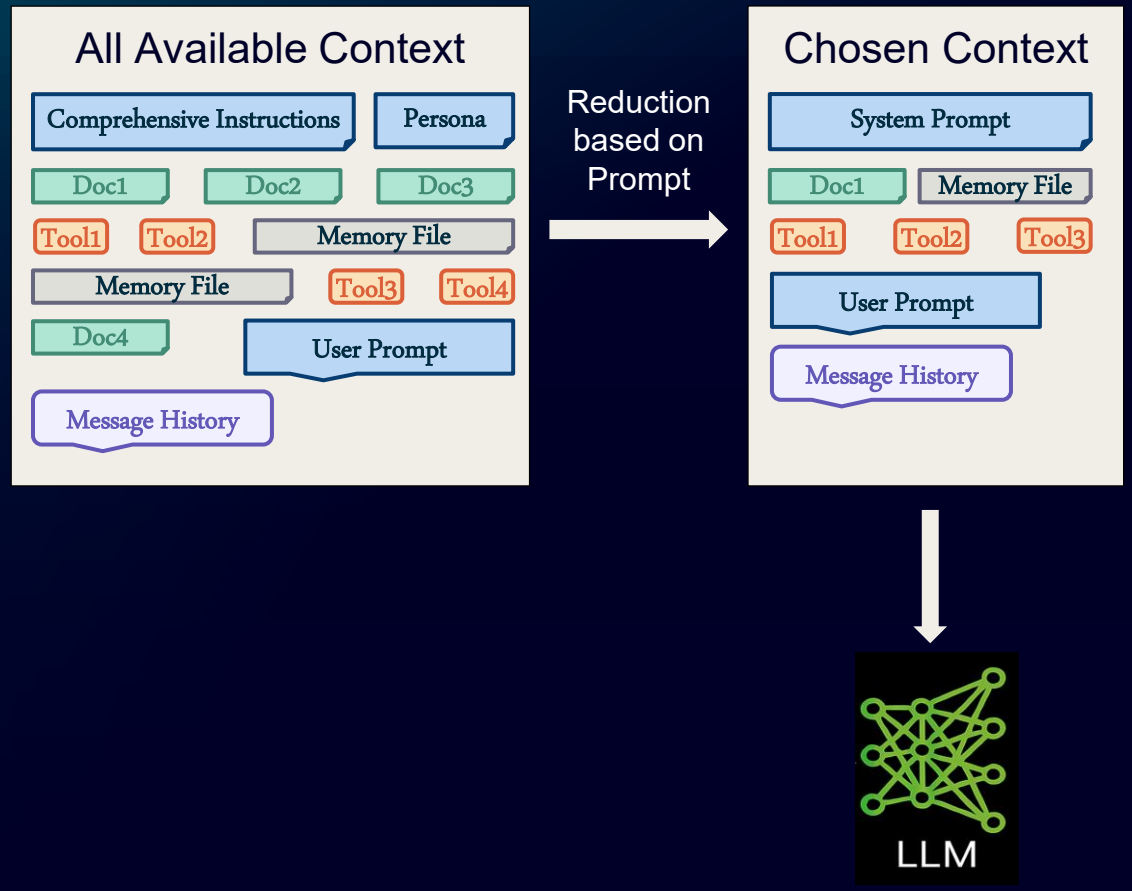
Context Windows Are Limited, So Management Is Required

Context windows are bounded
Limits differ by model, and overflow triggers truncation or refusal

Accuracy degrades with excess
Long or noisy context dilutes attention and hurts grounding

Costs scale with tokens
More input/output tokens directly raise spend; wasted tokens waste budget

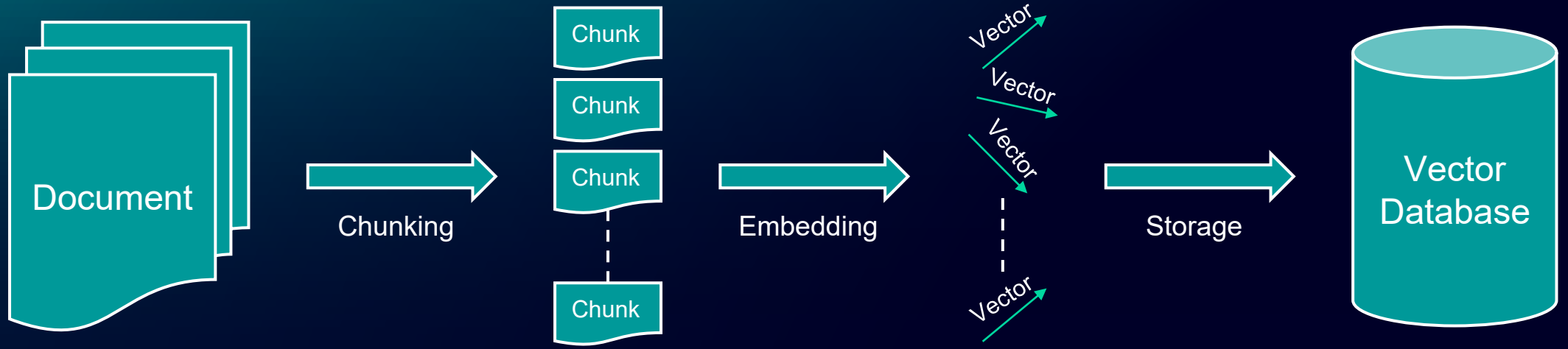
Latency rises with length
Longer prompts/outputs and retrieval/tool hops add noticeable delay



Supplying LLMs with precise, efficient context is a core challenge to achieve good results

Reading large documents: Retrieval Augmented Generation

How are RAG Database built?



Knowledge

Meaningful Sections

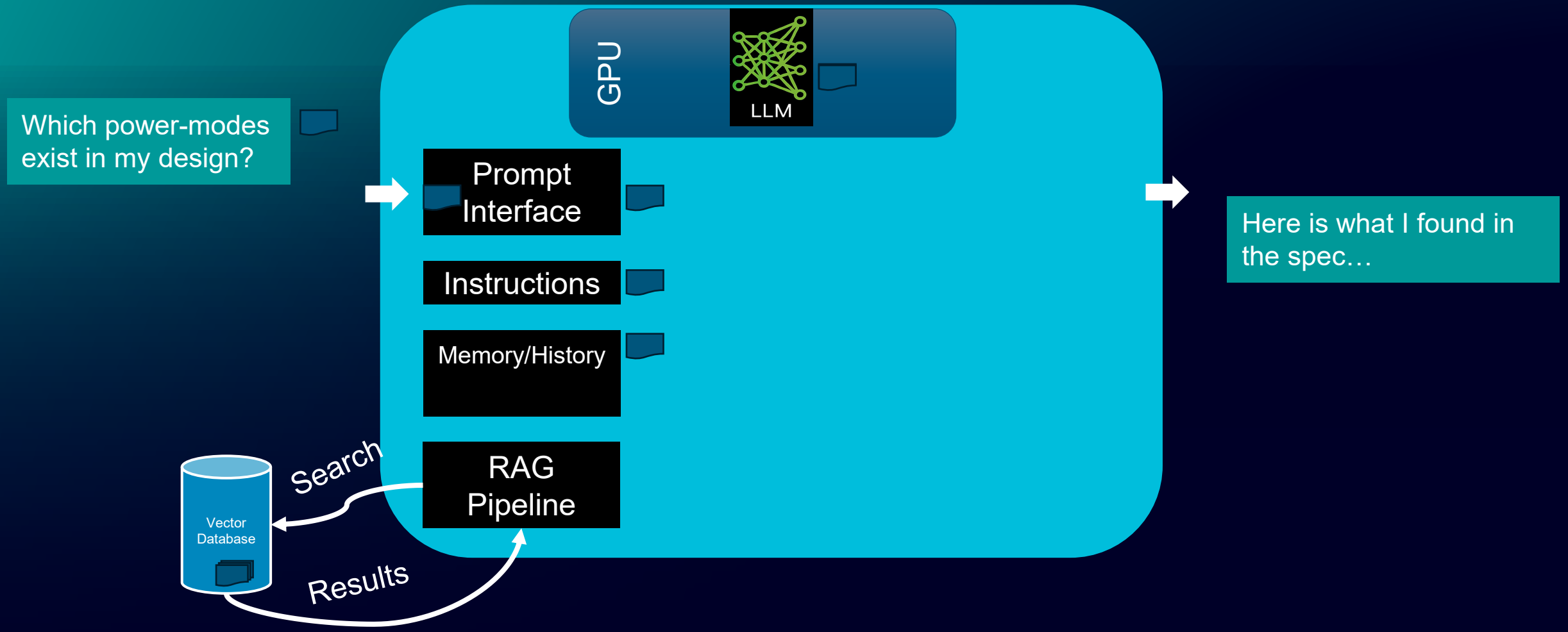
Semantic Meaning

Semantic Search

RAG Ingestion Pipeline

SIEMENS

Adding RAG: Read up to date large documents



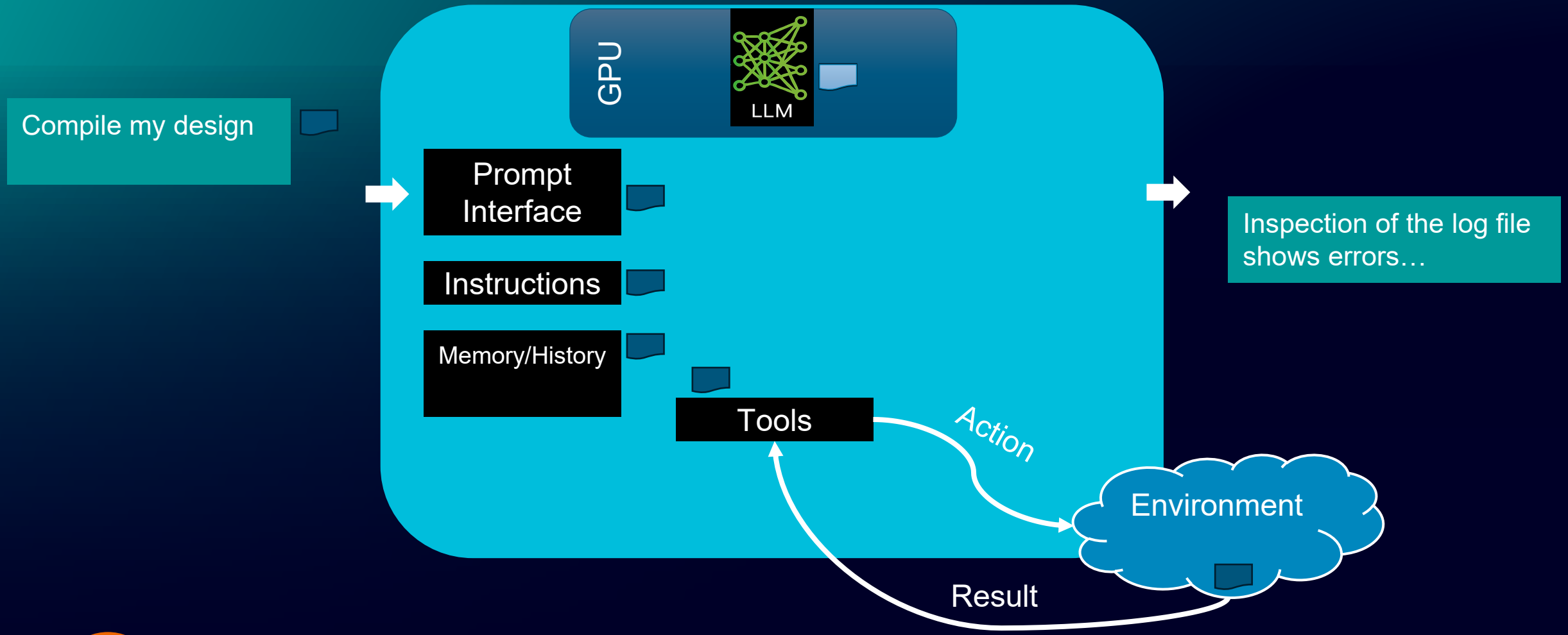
RAG search lets the agent pull in fresh and sensitive information not in training, while keeping the context smaller by retrieving only what's needed

Interacting with the environment

How can an Agent use tools?

*LLMs **can not** call tools,*
*but LLMs can **instruct** (with words/token)*
*their calling **agent** to call a tool for them*
*and **use the result** in the next prompt*

Calling tools: Agents can be actors in workflows



With added tools, the agent can control and run EDA tools and flows

How can we use tools with different agentic framework?

MCP

- Model Context Protocol
- An **open standard** that lets AI apps/agents connect to **external tools, data sources, and workflows** via a simple client–server protocol
- Think “USB-C for AI”

Who created it & stewardship

- **Introduced by Anthropic** on **Nov 25, 2024**
- Now an **open project under the Linux Foundation** with community governance

Why it exists

- To replace one-off, vendor-specific connectors
- Enable using tools from different vendors
- Giving models secure, scalable access to real data and actions

How it’s used

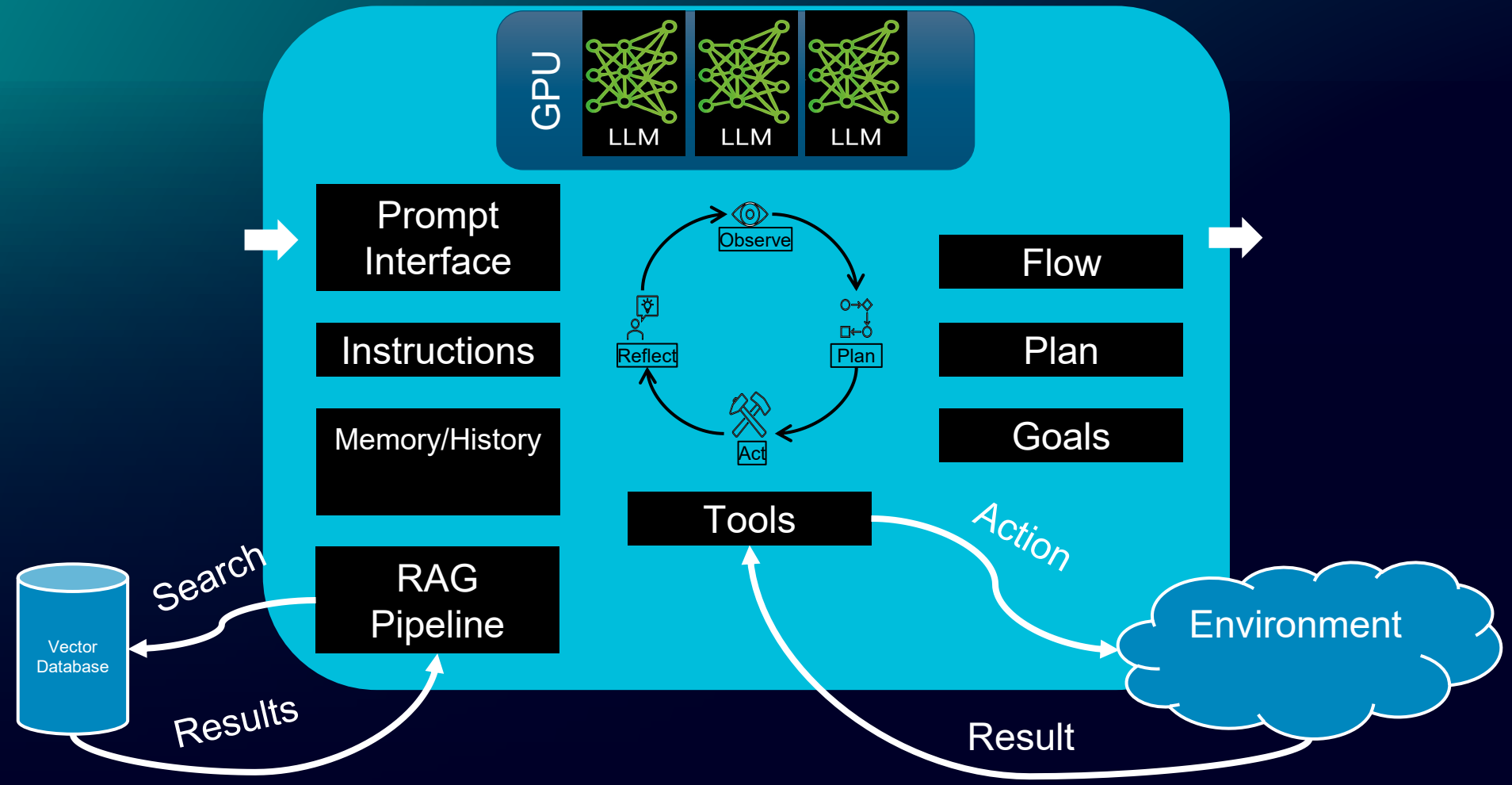
- **MCP servers** expose capabilities
- **MCP clients/hosts** (AI apps) discover the servers and call them
- Can run **locally or hosted**

What capabilities are exposed

- **Tools** - actions the model may request (e.g., “Compile,” “query DB”)
- **Resources - read-only** data the model can see (files, DB rows, APIs).
- **Prompts** — reusable **prompt templates/workflows** discoverable by clients

Too many tool descriptions increase cognitive load and can confuse the model!

Putting it all together



How to Describe an LLM?

Vendor
The company that builds the model

Family/Generation
The named release line that shares a core architecture and training baseline

Tier/Variant
The sub-choice within a family that trades off speed/cost vs. depth

Representative models across vendors (families/tiers)

Vendor	Family/Generation	Tiers/Variants
OpenAI	GPT4.o	Mini/(standard)
	GPT5.2	Instant/Thinking/Pro
	GPT5.x	5.1-codex, 5.3-codex
Anthropic	Claude3.5	Haiku, Sonnet, Opus
	Claude4.5	Haiku, Sonnet, Opus
Google	Gemini1.5	Flash-Lite/Flash/Pro
	Gemini2.5	Flash-Lite/Flash/Pro

How to choose which LLM to use?

- **Fit to task**
  Writing, coding, reasoning, multimodal, tool use
- **Context window**
  How much text it can read at once (relevant to long docs/codebases)
- **Tooling**
  Can it call tools/browse/invoke code/handle RAG
- **Speed & cost**
  Rough latency and price tier
- **Deployment & safety**
  Cloud vs. self-host, data isolation, guardrails

Example for Comparing Claude models

Feature	Claude Sonnet 4.5	Claude Haiku 4.5	Claude Opus 4.5
Description	Our smart model for complex agents and coding	Our fastest model with near-frontier intelligence	Premium model combining maximum intelligence with practical performance
Claude API ID	`claude-sonnet-4-5-20250929`	`claude-haiku-4-5-20251001`	`claude-opus-4-5-20251101`
Claude API alias[1]	`claude-sonnet-4-5`	`claude-haiku-4-5`	`claude-opus-4-5`
AWS Bedrock ID	`anthropic.claude-sonnet-4-5-20250929-v1:0`	`anthropic.claude-haiku-4-5-20251001-v1:0`	`anthropic.claude-opus-4-5-20251101-v1:0`
GCP Vertex AI ID	`claude-sonnet-4-5@20250929`	`claude-haiku-4-5@20251001`	`claude-opus-4-5@20251101`
Pricing[2]	$3 / input MTok $15 / output MTok	$1 / input MTok $5 / output MTok	$5 / input MTok $25 / output MTok
Extended thinking	Yes	Yes	Yes
Priority Tier	Yes	Yes	Yes
Comparative latency	Fast	Fastest	Moderate
Context window	200K tokens / 1M tokens (beta)[3]	200K tokens	200K tokens
Max output	64K tokens	64K tokens	64K tokens
Reliable knowledge cutoff	Jan 2025[4]	Feb 2025	May 2025[4]
Training data cutoff	Jul 2025	Jul 2025	Aug 2025

https://platform.claude.com/docs/en/about-claude/models/overview

Choosing an LLM is hard! it depends on your task, data, constraints, and personal preferences, so research before you decide

The missing piece

Good prompting …

How we ask the question matters

Context One:

- *Plan me a trip to Paris*

Context Two:

- *Plan me a trip to Paris*
- Here is some information about me:
  - I don‘t not like flying
  - I like seeing countryside
  - I usually stay in a city for only 2 or less days
  - I have only one week to travel
  - I will travel 1 week before christmas
  - My starting point is from Stockholm
  - I like information in table format

Which answer will be better?

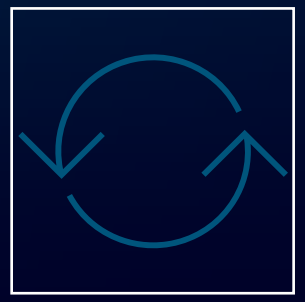
Prompting Guidelines

Think about the Content

- Set a Persona & Tone
- State what to do
- State what not to do
- Use Chain of Thought (CoT)
- Specify output format
- One-shot or Many-shot examples

Ensure Clarity and Style

- Be Clear and Specific
- Use Active Voice
- Structure with Formatting
- Keep Sentences Concise
- Use Consistent Terminology



Improve Through Iteration

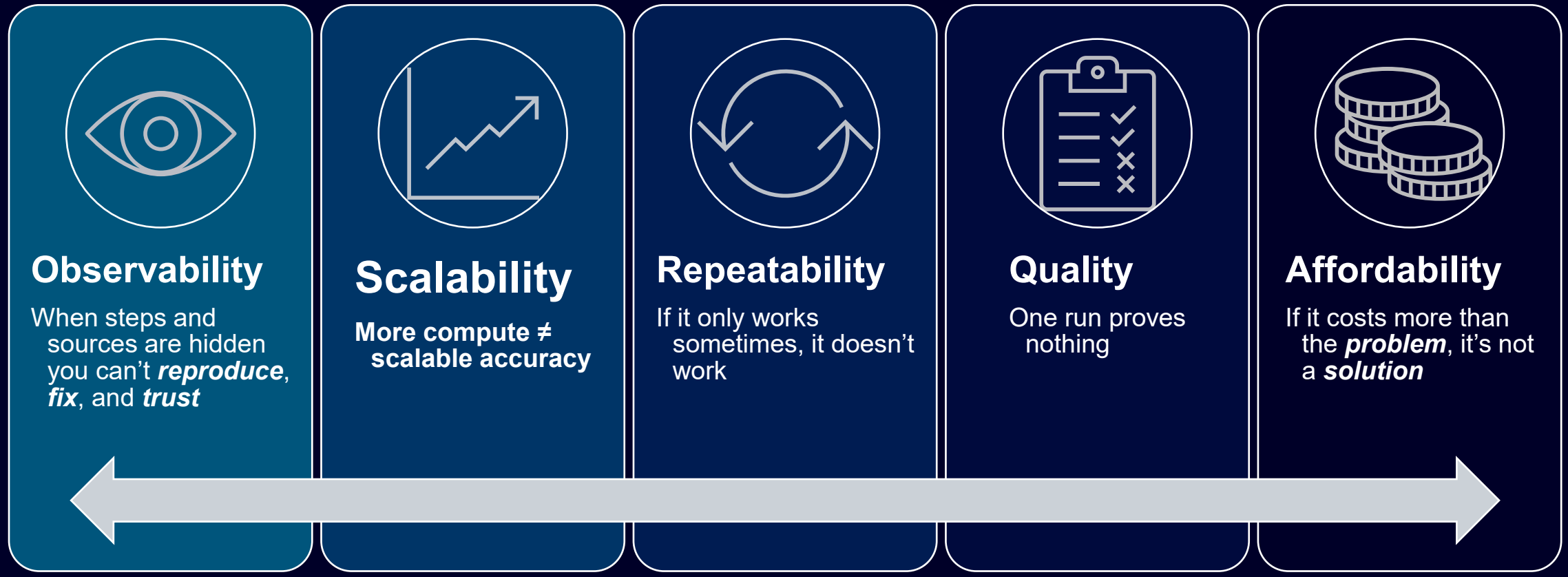
- Test different prompt versions
- Compare results to see which performs best
- Refine the prompt for better clarity and accuracy

LLMs can help you refine, optimize, and clarify your own prompts

```
# Persona and Tone
You are a hardware design engineer. Use a spartan, direct tone. Prioritize clarity and correctness

# Task
Create a parameterized SystemVerilog counter module.
The module must include:
- A width parameter
- An optional enable parameter
- Clock, active-low reset, and counter output ports

# Restrictions
- Do not generate a testbench
- Do not add comments, explanations, narrative text, or multiple files
- Do not introduce external packages or dependencies.
- Output SystemVerilog code only

# Chain-of-Thought Steps
Follow this sequence and reflect it in the final code structure:
1. Define the micro-architecture: parameters, ports, and state.
2. Create the module skeleton: header, parameter list, port list, declarations.
3. Implement the sequential logic: reset handling and increment behavior.
4. Add SVA assertions for reset correctness and increasing count when enabled.

# Output Format
Produce one SystemVerilog file containing the complete module and its assertions.

# Micro Architecture Example (Structure Only) …
```

Questa One Agentic Toolkit

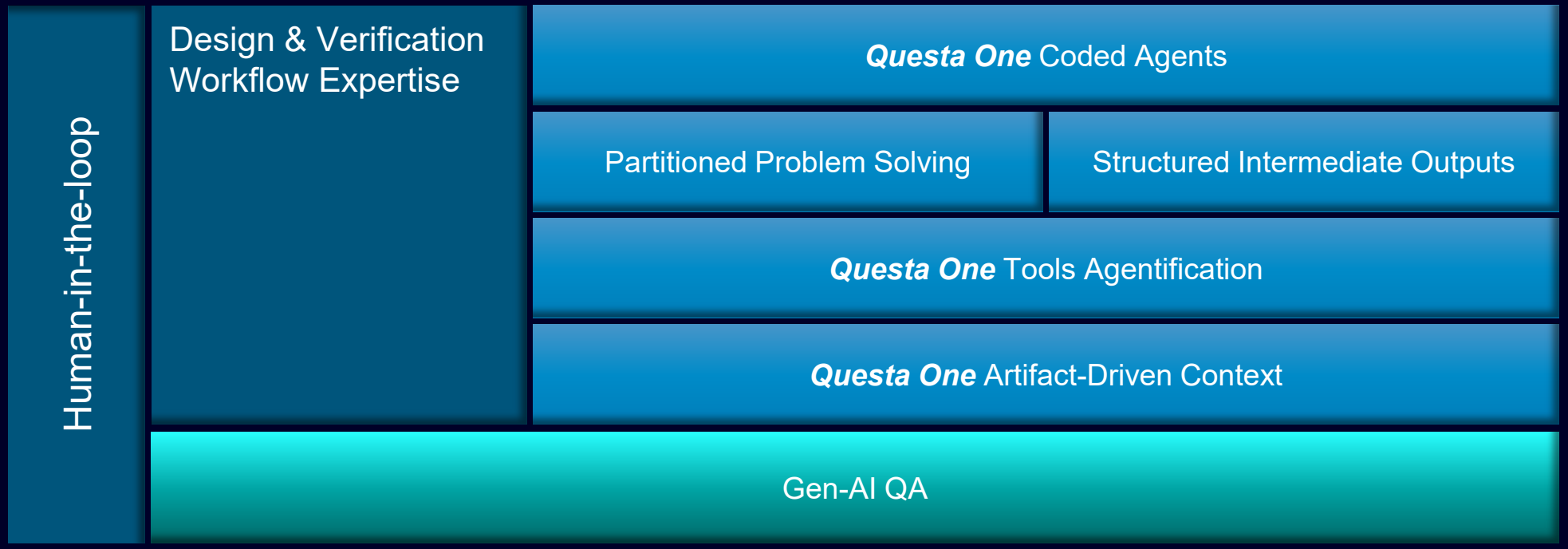
Agentic Challenges
From Demo to Production: Bridging the Gap for Agentic AI



A slick demo without a plan for how to solves these is a dead end!

Our Approach

Delivering Trustworthy AI: Building Reliability and Control using ***Questa One***



Chip Design Gen-AI flows require unique ***knowledge*** and ***tools***

SIEMENS

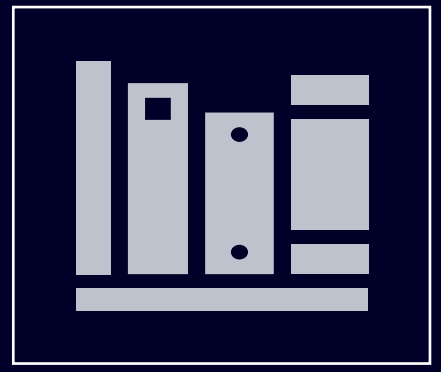
Which AI Coding Agent/Framework should you use?

*** These are illustrative platform examples—not recommendations or advice

Choose to what fit the use case and your needs, what's available in your environment, and team preference - often the right answer is to use more than one

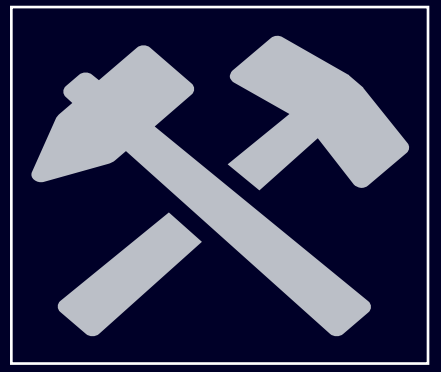
Platform-Agnostic Toolkit: Plug into any stack without lock-in

Prompt Library



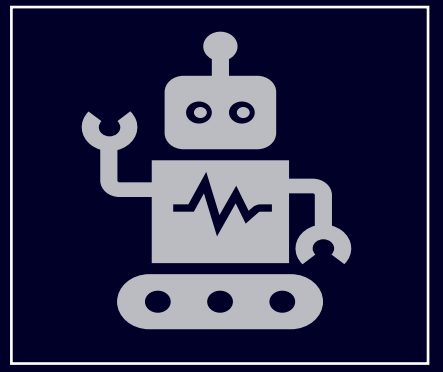
- Bootstrap operations, accelerate adoption
- Guidance for effective prompting

Tools



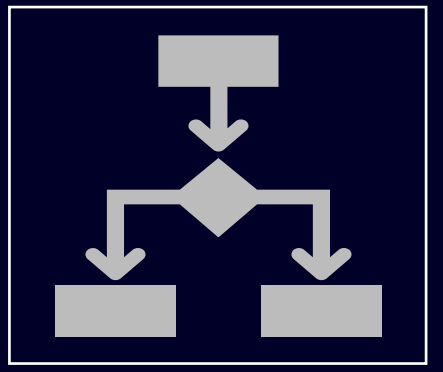
- **MCP Servers:** Agentify ***Questa One*** tools for reliable execution
- **Context Management**: Domain-specific knowledge & task context for informed actions

Agents



- Dedicated agents for specific problems (e.g., Verification, Debug)
- Structured, step-by-step processes for complex tasks

Examples



- Getting started package for rapid adoption
- Agentic workflow Python code example

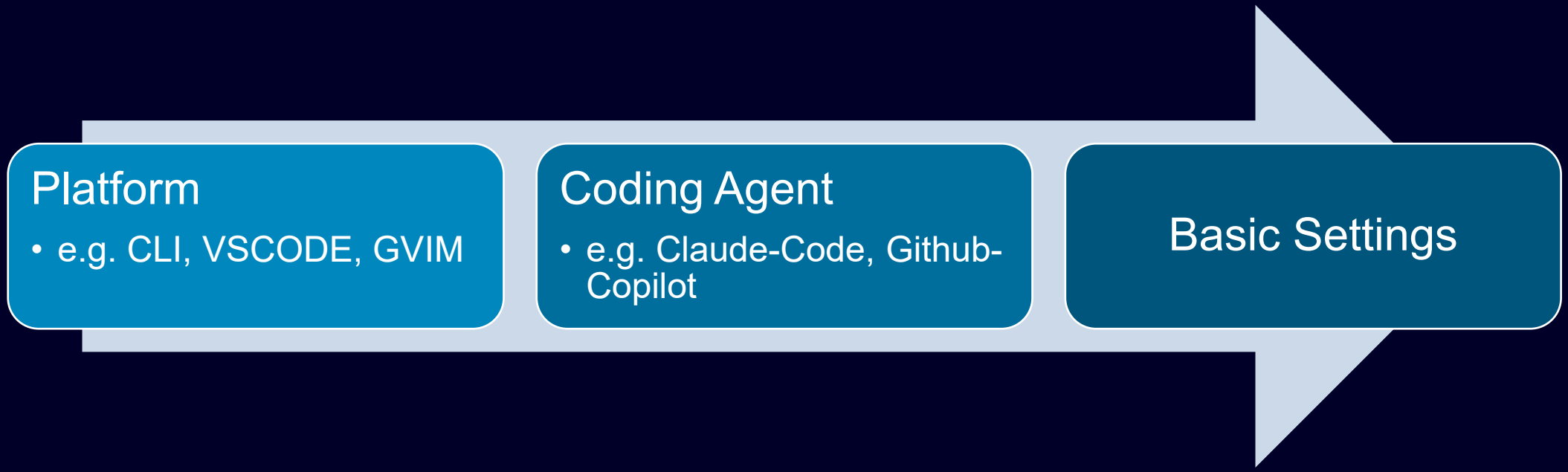
2026 DESIGN AND VERIFICATION DVCON UNITED STATES SANTA CLARA, CA, USA March 2-3, 2026

Questa One Agentic Flows

Gen AI for Chip Design in action

SIEMENS

Getting Started

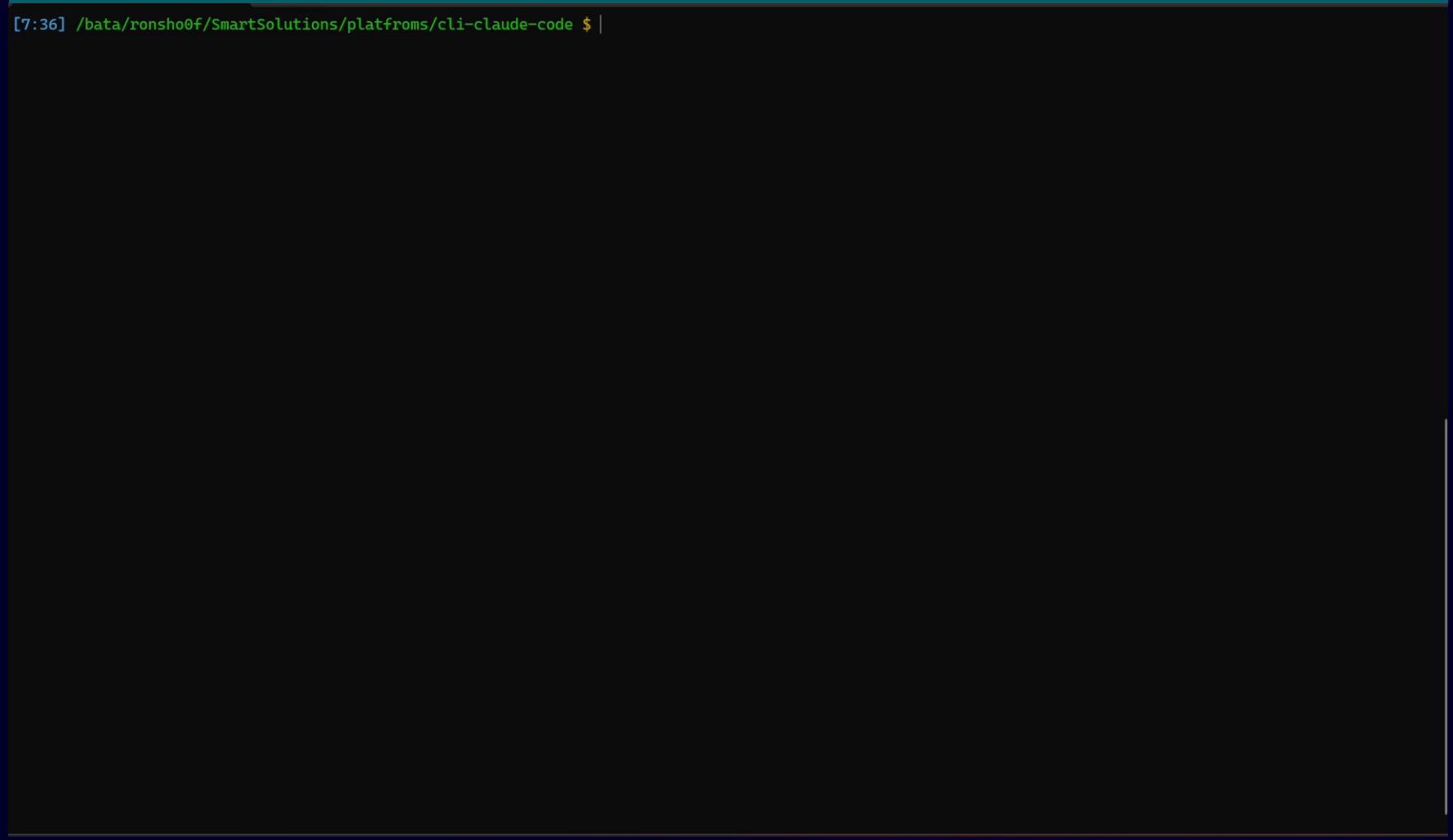


Installation and setup of the subscription are not covered, as these steps vary based on each company's internal processes, legal considerations, and requirements

SIEMENS

Getting Started

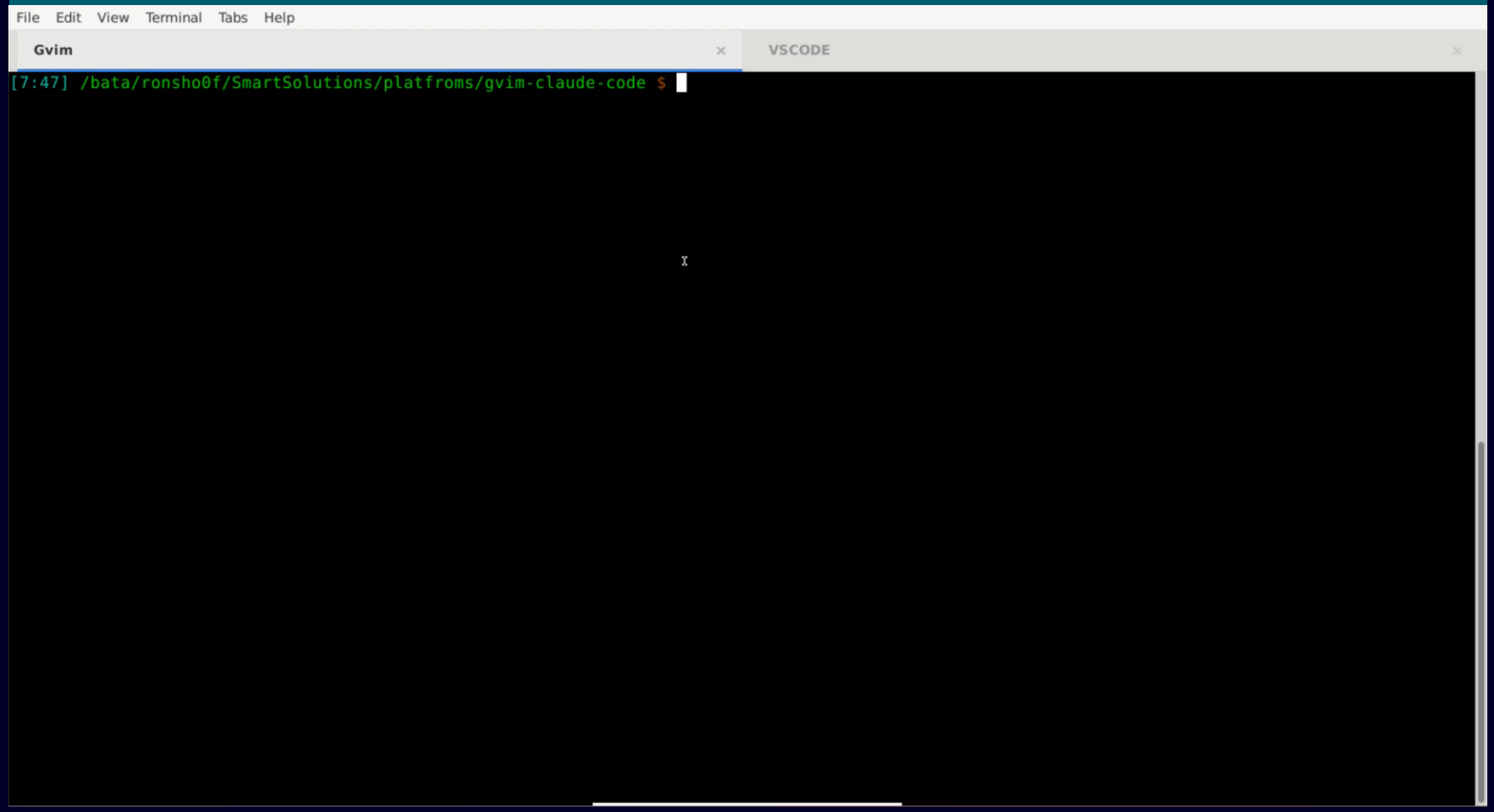
CLI / Claude-Code



SIEMENS

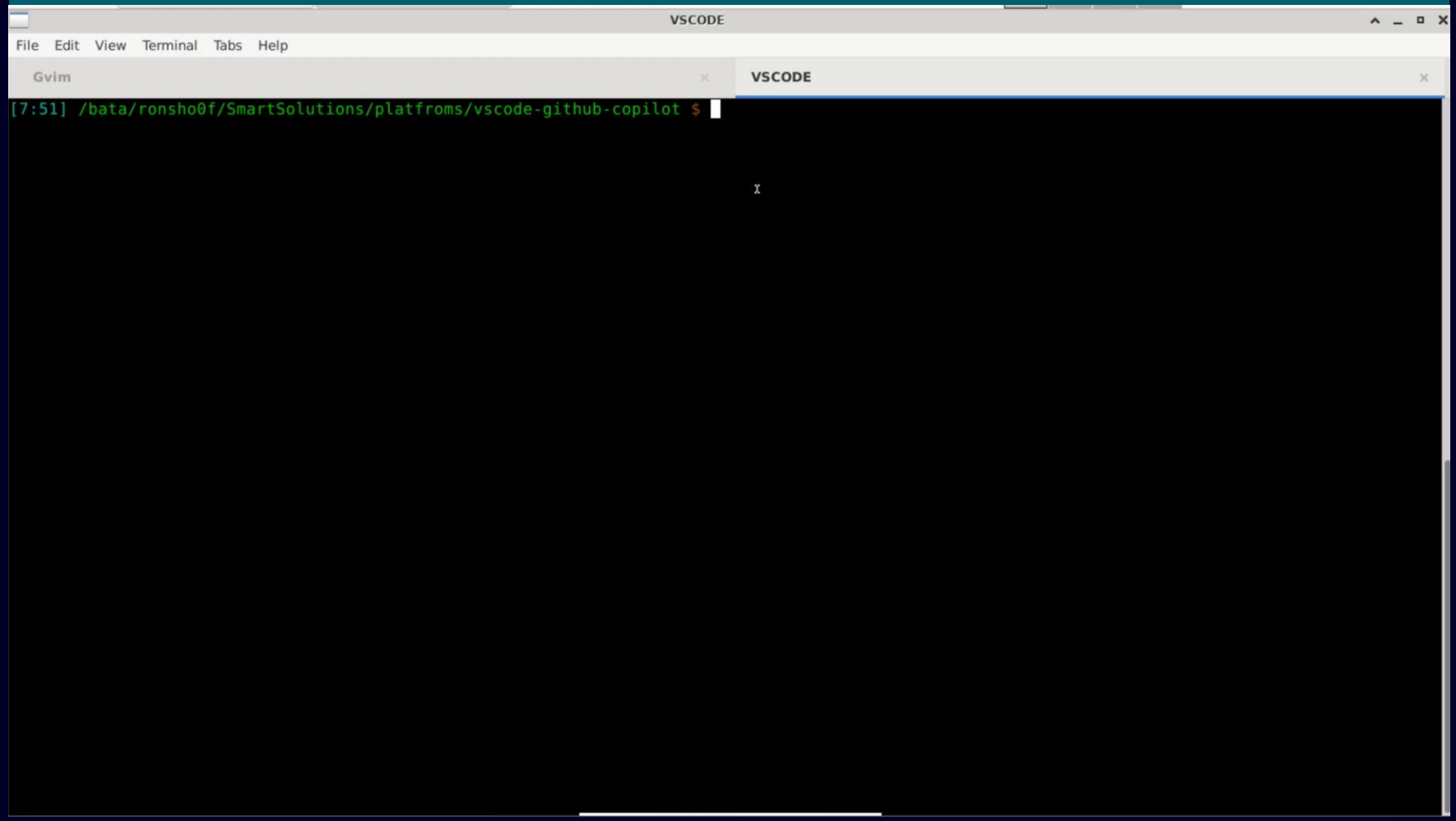
Getting Started

Gvim / Claude-Code



Getting Started

VS-Code / Github-Copilot



SIEMENS

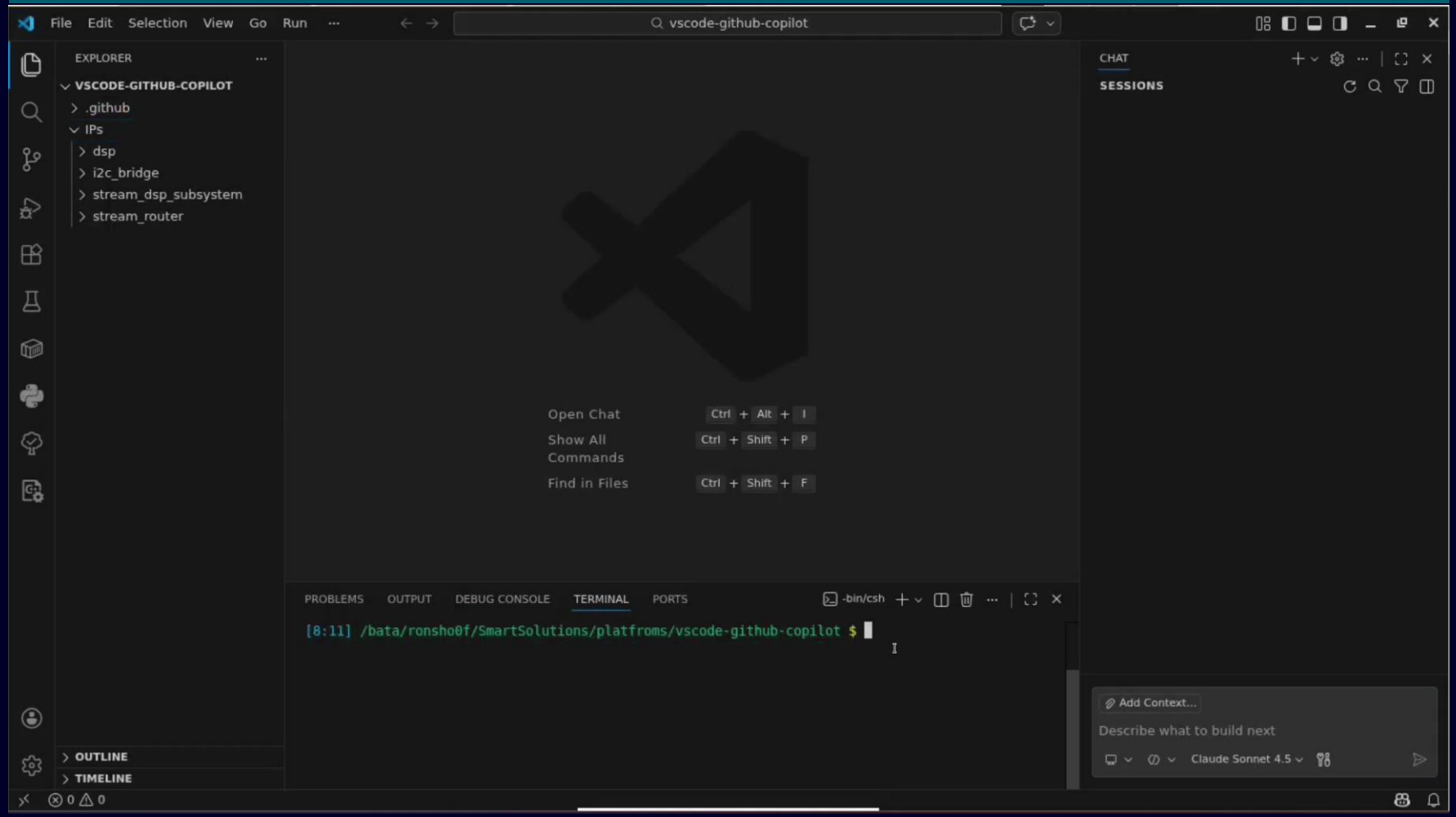
Using Toolkit

CLI / Claude-Code

```
[8:05] /bata/ronsho0f/SmartSolutions/platfroms/cli-claude-code $
[8:05] /bata/ronsho0f/SmartSolutions/platfroms/cli-claude-code $
[8:05] /bata/ronsho0f/SmartSolutions/platfroms/cli-claude-code $
[8:05] /bata/ronsho0f/SmartSolutions/platfroms/cli-claude-code $
[8:05] /bata/ronsho0f/SmartSolutions/platfroms/cli-claude-code $
```

Using Toolkit

VS-Code / Github-Copilot



SIEMENS

Lint Agent

Lint check and fixing agent

Assist user to run Lint and suggest fixes for auto-fixable issues

SIEMENS

Verification Planning Agent

Smart Flows

Faster Engineers with Verification Planning Assist

Generative AI

Debug Agent

QuestaOne Agentic Toolkit

SIEMENS

SIEMENS

Q & A

Disclaimer

Subject to changes and errors. The information given in this document only contains general descriptions and/or performance features which may not always specifically reflect those described, or which may undergo modification in the course of further development of the products. The requested performance features are binding only when they are expressly agreed upon in the concluded contract.