

Agentic AI Design using AMD Technology



together we advance_

Jens Stapelfeldt – Global AI Lead AUP

Mario Ruiz – Senior Member Technical Staff

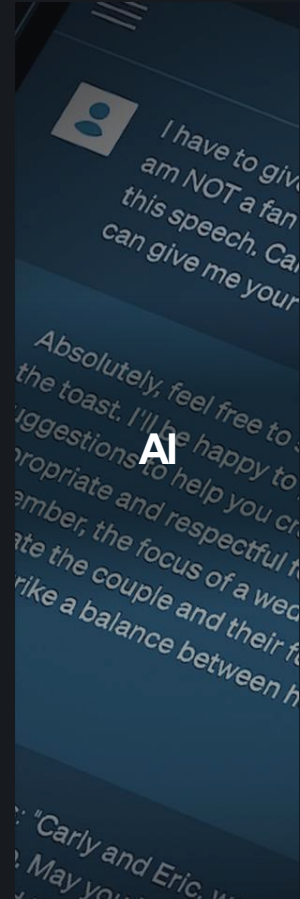
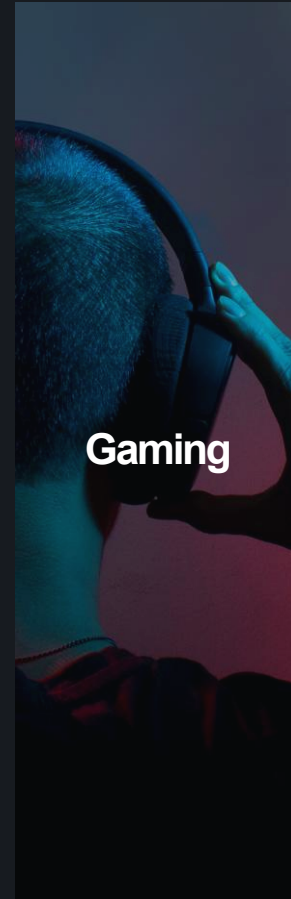
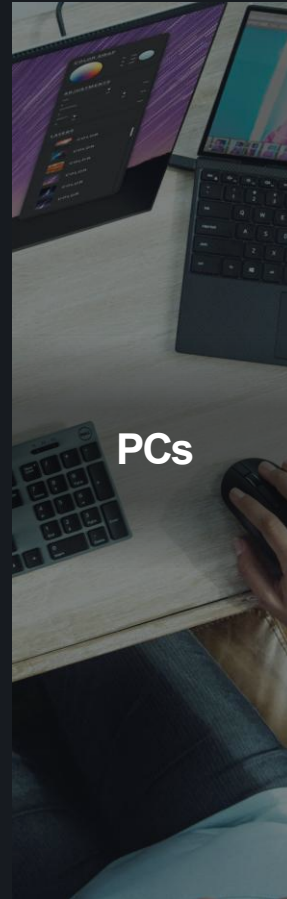
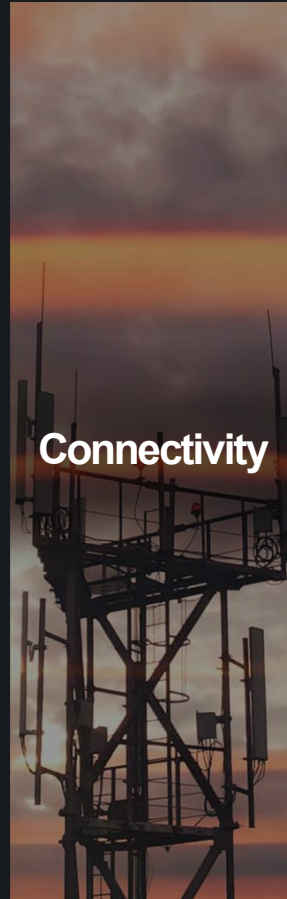
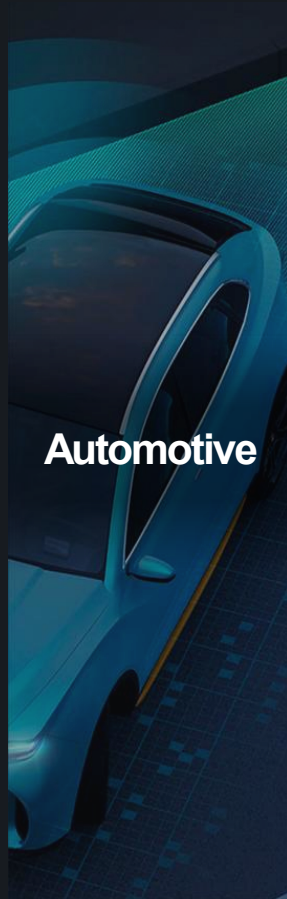
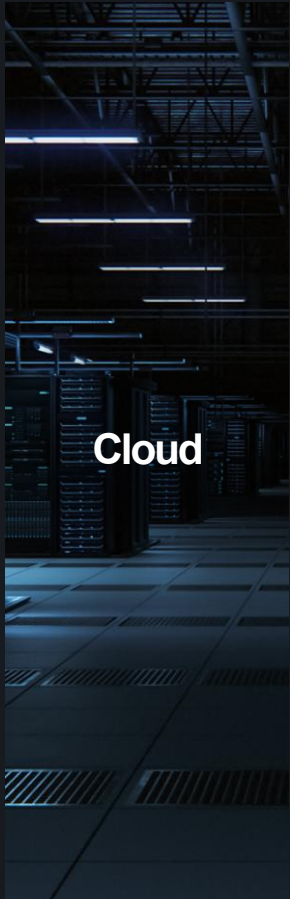


High Performance and Adaptive Computing

Creating solutions to the world's most important challenges



AMD powers the daily lives of billions



AMD High-Performance and Adaptive Computing



Data Center

Leadership performance and TCO across cloud, enterprise and AI workloads

AMD 
EPYC

AMD 
INSTINCT

AMD 
ALVEO

AMD 
VIRTEx

AMD 
KINTEX

AMD 
SOLARFLARE

AMD 
ZYNO

AMD 
VERSAL

AMD 
RYZEN



Client

Performance and energy efficiency for productivity, AI, gaming and content creation



Gaming

Stunning visuals and immersive experiences for PC and console gaming

AMD 
RADEON



Embedded

Computing leadership for diverse markets and devices

AMD 
RYZEN
Embedded

AMD 
EPYC
Embedded

AMD 
ALVEO

AMD 
KRIA

AMD 
SPARTAN

AMD 
KINTEX

AMD 
VIRTEx

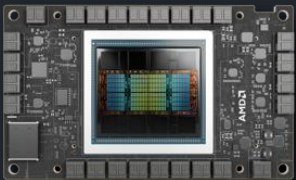
AMD 
ZYNO

AMD 
VERSAL

AMD 
ARTIX

End-to-end AI infrastructure

Leadership compute engines for AI

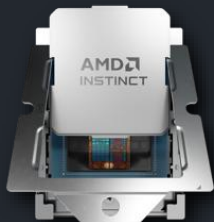


Cloud

Leadership accelerator for generative AI, LLMs and inferencing

AMD Instinct™ MI300X Accelerators

AMD Instinct™ MI325X Accelerators



HPC

World's first data center APU for HPC and AI

AMD Instinct™ MI300A Accelerators

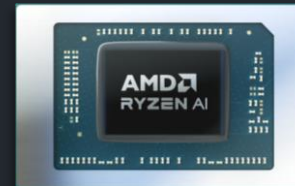


Enterprise

Broad ecosystem of OEM and solution partners

AMD Instinct™ MI300 Accelerators

5th Gen AMD EPYC™ Processors

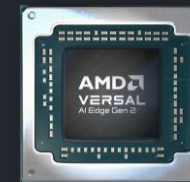


PC

The world's most powerful NPU for next-gen AI PC experiences

AMD Ryzen™ AI 300 Series Mobile Processors

AMD Ryzen™ AI PRO 300 Series Mobile Processors



Embedded

Adaptive SoC + AI for Embedded

AMD Versal™ AI Edge

AMD Zynq™ MPSoC

Cloud to end point AI software

AMD University Program (AUP): Mission

Empower academics with AMD technology to enhance teaching and learning experiences and advance state-of-the-art research.

Academic and Research Community



Students

**Affordable access
to AMD products**

Build industry relevant skills and
knowledge with AMD products



Educators

**University resources
and support**

Deliver high-quality teaching with
AMD educational resources and support



Researchers

**Leading-edge
AMD technologies**

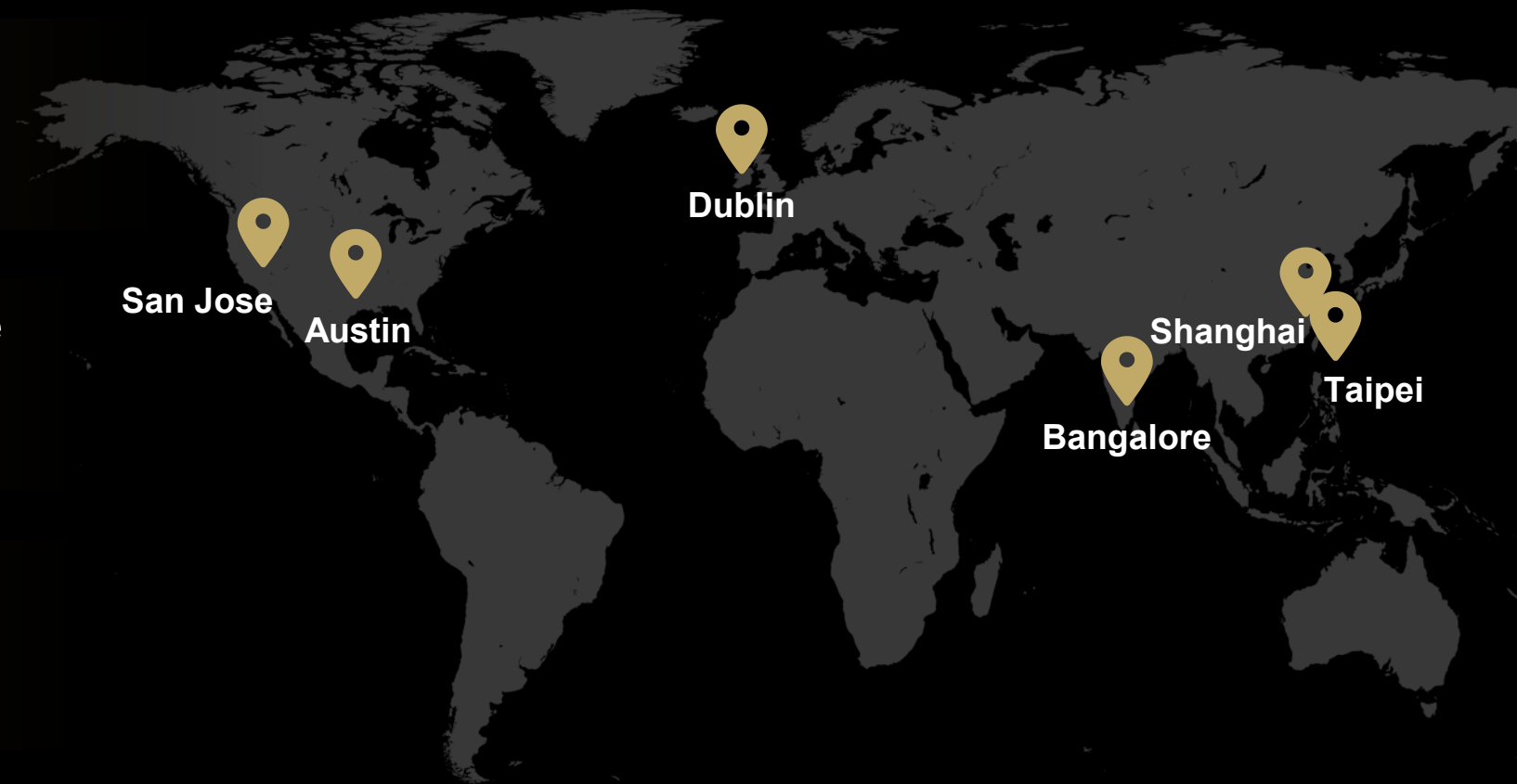
Advance state-of-the-art
research

Our Team

Dedicated world-wide
immersive technical team

Supporting High Performance
and Adaptive Compute,
focused on AI

25+ years experience
working with academia



Hardware Resources

Compute Resources

HACC	Donation Program	AI&HPC Cluster	AUP Learning Cloud	AI Developer Cloud
High-performance computing cluster for partner universities	Hardware donations to academic partners	Dedicated AI and HPC cluster access	Cloud-based learning environment for students	AMD AI Developer Cloud access for research
On-premise AMD HPC platforms	Devices, boards, software tools & research grants	On-prem and cloud-capable deployments	Low-cost dev kits and remote AI lab access	Scalable compute for AI model development

AMD Technology: AI Across the Curriculum



- Lead ECE Education
- 1st touch HW
- Low Cost
- AMD-first AI Teaching
- On-Prem Installation
- Cloud Capable
- Heterogenous AI
- Infra Research
- Pathfinding
- Top research group
- AI+ Research
- Petaflops

AMD Partners

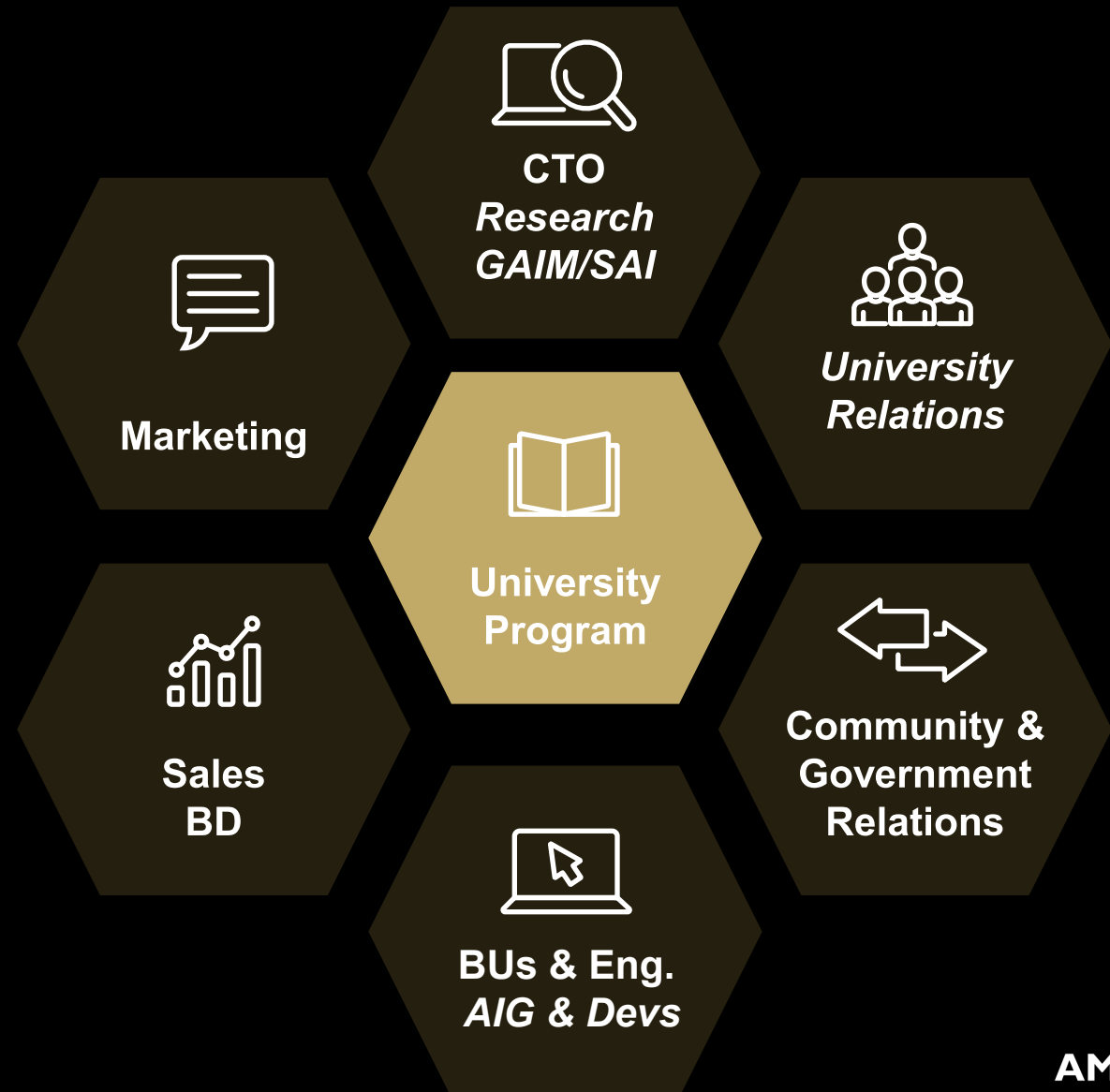
University Program

Leads technical engagement
with university community

Coordinates university
collaborations across AMD partners

Facilitates research
opportunities

aup@amd.com | www.amd.com/AUP



AMD University Program (AUP)

Empower academics with AMD technology for teaching and research



Research Programs

AI & HPC Clusters
HACC on-prem platforms
AMD Developer Cloud



Academic Donations

Devices & boards
Software tools
FAR Research grants



Teaching Resources

Teaching Solutions
AI Tutorials
AMD textbooks



Training & Events

Academic workshops
Hackathons
Contests



Solutions & Support

Low-cost dev kits
AUP Learning Cloud
Tech support

aup@amd.com | www.amd.com/AUP

Agents

Agentic AI Design using AMD Technology

Overview

1. What are AI Agents and Agentic AI
2. Introduction to prompt engineering
3. Introduction to RAG systems and structure
4. Introduction to AI agents
 - AI Agents components
 - AI Agents Types and design pattern
5. Overview of Model Context Protocol (MCP)
6. Hands-on

Agentic AI Design using AMD Technology

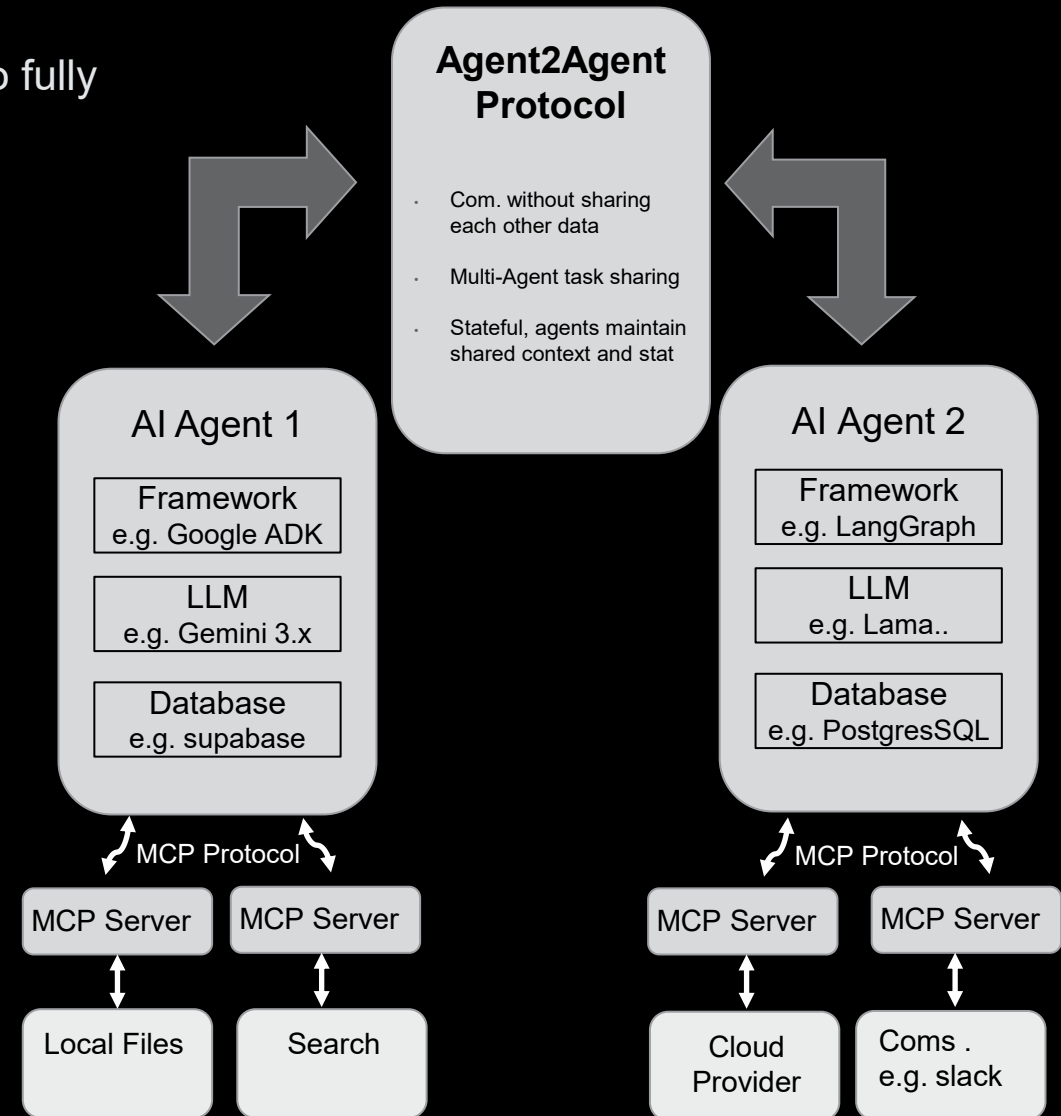
About this module and learning objective:

This module gives a comprehensive introduction to AI Agents their characteristics and architecture. We talk about basic **Architecture Components of a general Agent** and **Model Context Protocol (MCP)** as interface to connect and interact with tools. Different types of AI Agents are introduced.

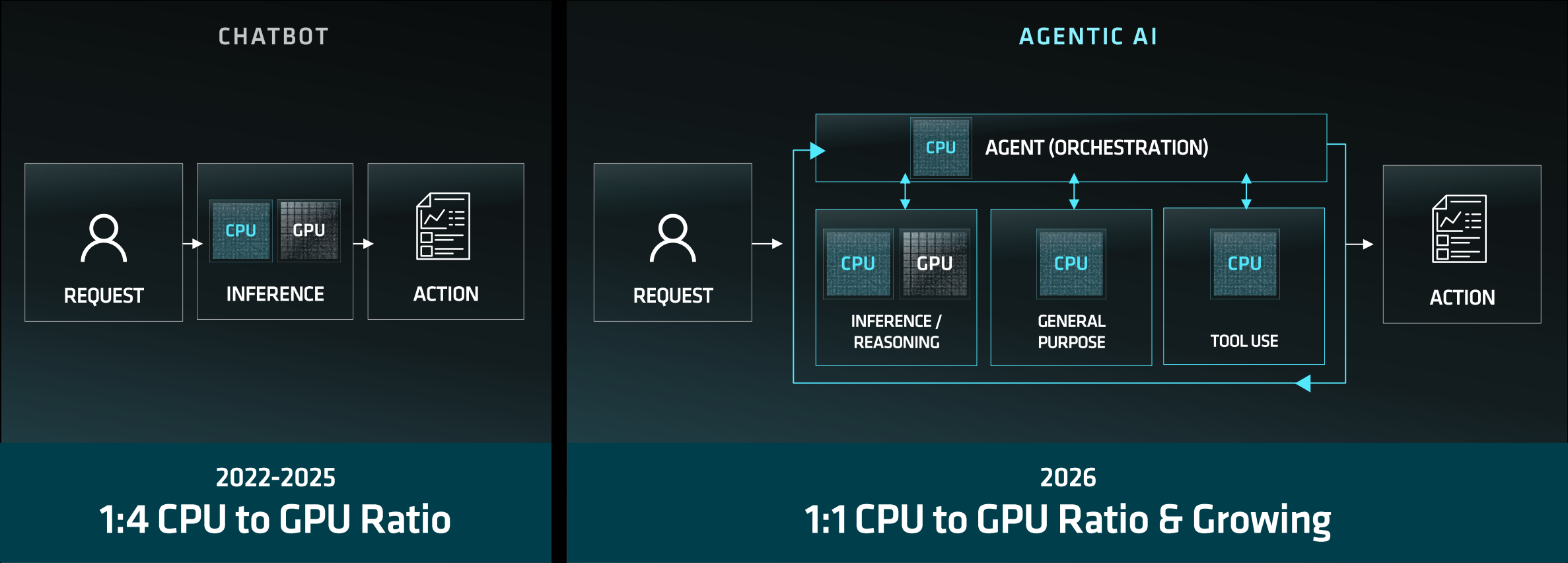
From Gen AI to AI Agents

The evolution from Gen AI, RAG (Retrieval-Augmented Generation) to fully autonomous agents represents a fundamental shift in AI capabilities.

- **Gen AI**
 - Language models learn from texts. They recognize patterns and generate responses. (the likely next word!)
- **RAG (Retrieval Augmented Generation)**
 - RAG enhances LLMs with external knowledge but lacks true autonomy.
- **Autonomous Agents**
 - Agents make independent decisions and take actions without continuous human guidance.



Chatbots to Agentic AI: A Compute Shift



Transition From Chatbots to Agents Performing Real Work

What is Prompt Engineering

What is prompt engineering?



Prompt Engineering is the practice of **designing and optimizing** the **inputs (prompts)** given to AI models like GPT-4, so that they produce the desired or **most useful outputs**. Since AI models rely on the input prompt to generate responses, how you **phrase and structure the prompt can significantly affect the quality**, relevance, and usefulness of the output.

In simpler terms, **prompt engineering** is about crafting the **right questions or instructions** to get the **most accurate and helpful responses** from AI systems.

Key elements of a prompt

Definition

Prompts are initial text inputs given to a model that guide its responses

Purpose

They serve as instructions for AI to execute specific tasks

Goal

Prompt engineering refines these inputs to enhance accuracy and relevance



Persona



Constraints



Context



Tone



Instructions



Delimiters



Examples



Data



Format



Prompt Patterns

Local Llamafire Demo

What is Retrieval-Augmented Generation (RAG)?

Retrieval-Augmented Generation (RAG) combines Large Language Models with external knowledge retrieval systems to enhance AI capabilities.

- **Enhanced Generation**
Incorporates real-time data retrieval from external sources, accessing information beyond the model's training cutoff.
- **Accuracy Boost**
Enables LLMs to query external databases for precise responses, especially for factual or domain-specific questions.
- **Hallucination Reduction**
Anchors responses in authoritative knowledge, creating factual reliability that pure generative models lack.

RAG combines LLMs' contextual understanding with retrieval systems' factual precision to deliver reliable, current, and nuanced information.

Simple RAG - The Foundation



Query Input

User provides a prompt or question



Document Retrieval

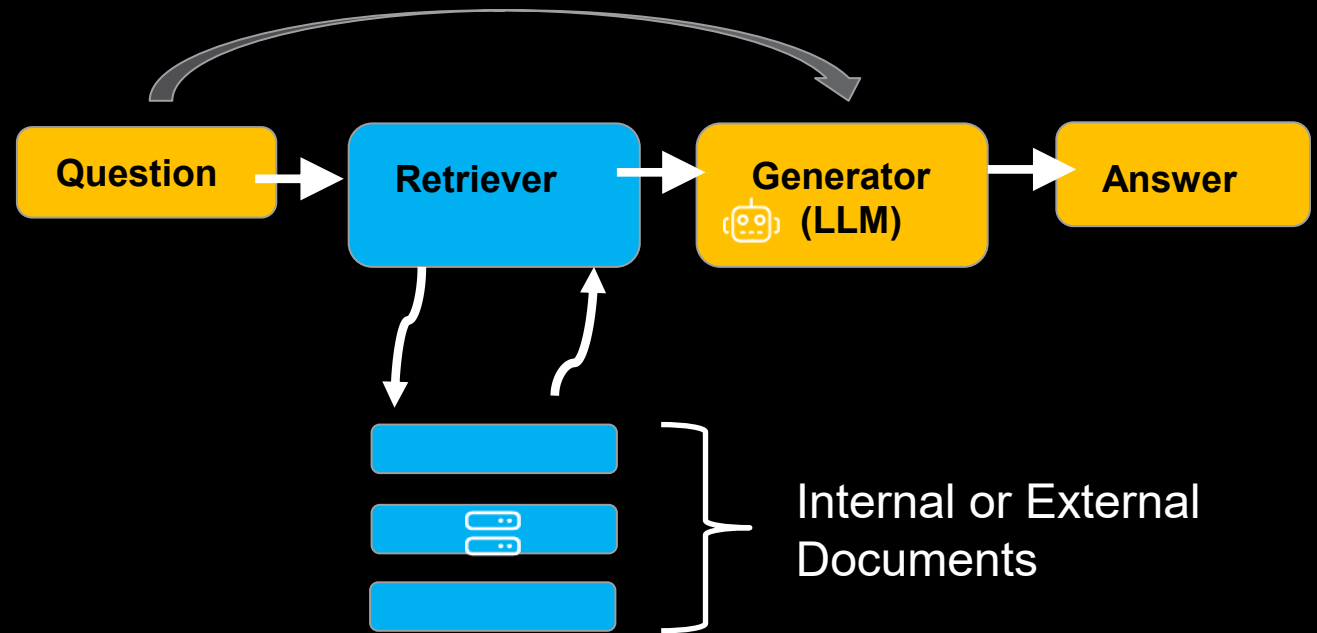
System searches fixed database for Relevant information



Generation

Model creates response grounded in retrieved data

LLM Basic Blocks
RAG Blocks



What Defines an AI Agent?

Perception

AI agents gather information through sensors or digital inputs, collecting data about their environment and user needs.

Processing

They analyze information using models and algorithms to understand context and make rational decisions.

Action

Agents execute tasks through digital or physical means to accomplish goals and complete objectives.

An AI agent perceives its environment, processes information, and takes action to achieve goals. Unlike basic programs, agents display rational behavior by selecting actions that optimize success in meeting their objectives.

Key Characteristics of AI Agents



Autonomy

Operating independently, making decisions based on internal logic and environmental inputs.



Reactive & Proactive Behavior

Responding to environmental changes while taking initiative to achieve objectives.



Adaptability

Learning from experiences to improve performance and decision-making over time.



Goal-Oriented Focus

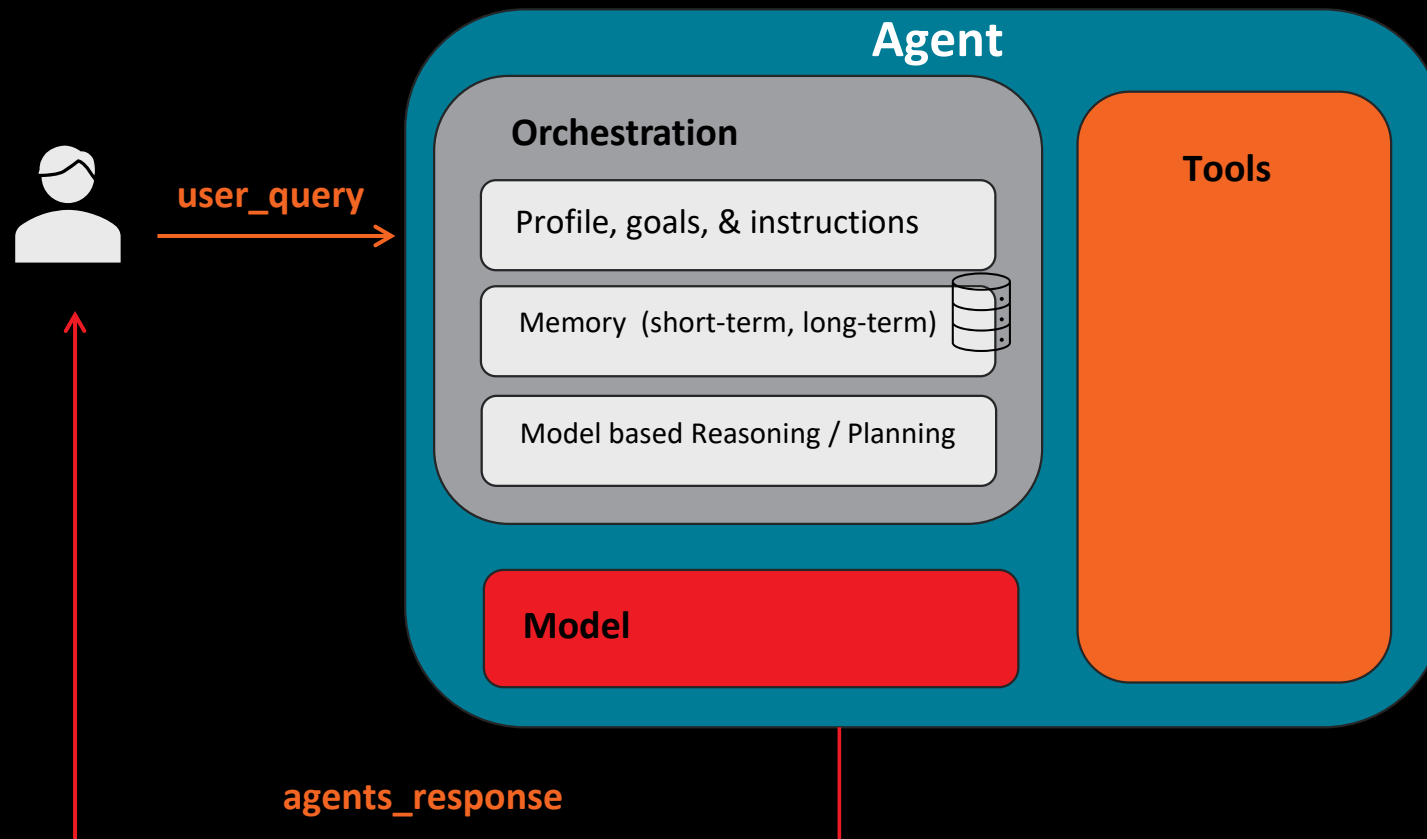
Working toward predefined objectives while optimizing outcomes within constraints.

AI agents also demonstrate interactivity (communicating with humans and other systems) and persistence (operating continuously in dynamic environments). These capabilities enable them to solve complex problems beyond traditional software approaches.

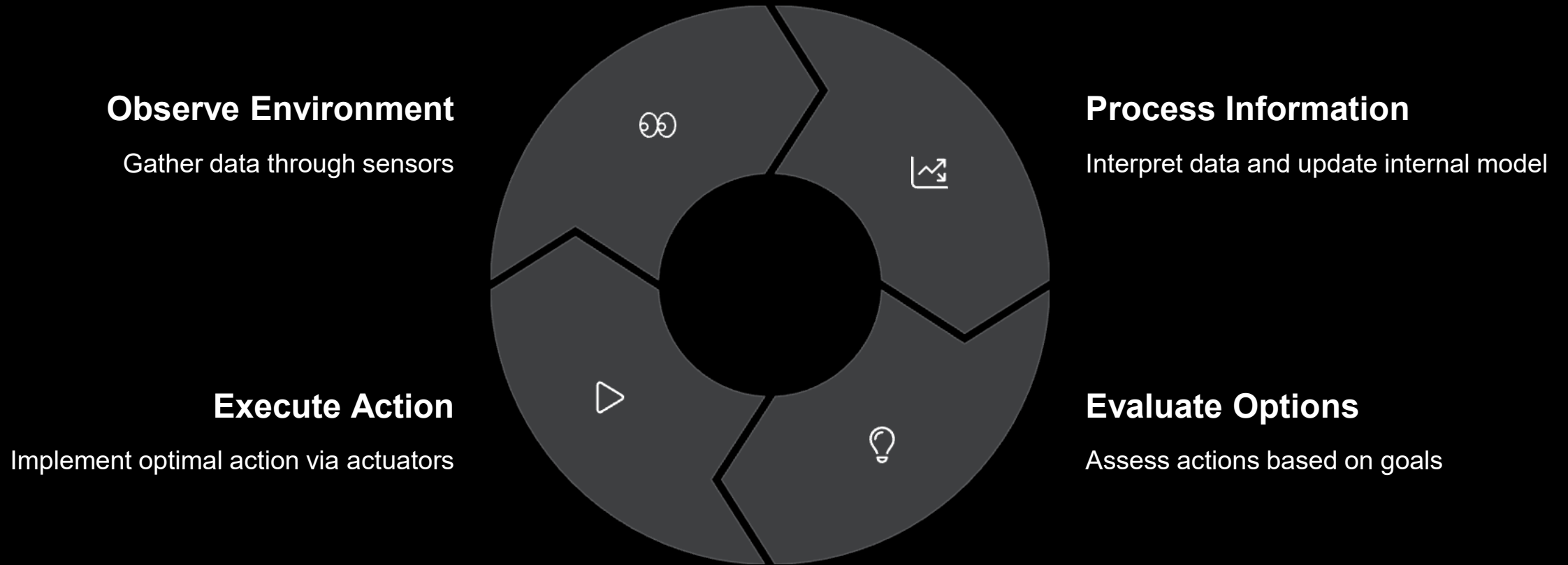
Architecture and Components of a general Agent

Components in Agents:

- **The Orchestration layer**
Organizes information, performs some internal reasoning, and uses that reasoning to inform its next action or decision making
- **The Model**
LLM depending on the use case
- **Tools**
Tools enable the Agent to interact with the outside world real time data



The Continuous Adaptation Cycle of AI Agents



This cycle enables adaptability to unexpected changes, learning through outcome comparison, and goal-directed behavior. The same process applies across AI systems - from chatbots generating text responses to robot vacuums planning cleaning routes.

Levels of Program Sophistication



1 Simple Program

Follows fixed rules without considering consequences or context. Like a basic thermostat that turns on at a set temperature threshold with no adaptation.

2 Responsive Program

Uses more complex rules with context awareness, such as a thermostat that considers time of day. Lacks true intelligence or learning capability.

3 True AI Agent

Analyzes multiple factors, predicts outcomes, learns from experience, and balances competing objectives while optimizing for long-term goals.

The progression from simple to AI programs represents a fundamental shift in capability. Simple programs follow fixed instructions, responsive ones add conditional logic, while AI agents employ advanced perception, reasoning, and learning to navigate complex environments.

AI Agent Architecture in Code - SmartThermostat

```
class AdaptiveThermostatAgent:

    def observe_environment(self):
        sensors = {
            "temperature": read_temp_sensor(),
            "current_time": system_clock(),
            "energy_cost": fetch_energy_tariff(),
            "forecast": get_weather_outlook(),
            "preferences": load_user_profile()
        }
        return sensors

    def decide(self, context):
        action_space = self.available_strategies()
        scored_actions = []

        for strategy in action_space:
            simulated_context = self.simulate(context, strategy)
            score = self.evaluate(simulated_context)
            scored_actions.append((strategy, score))

>>> next page >>>
```

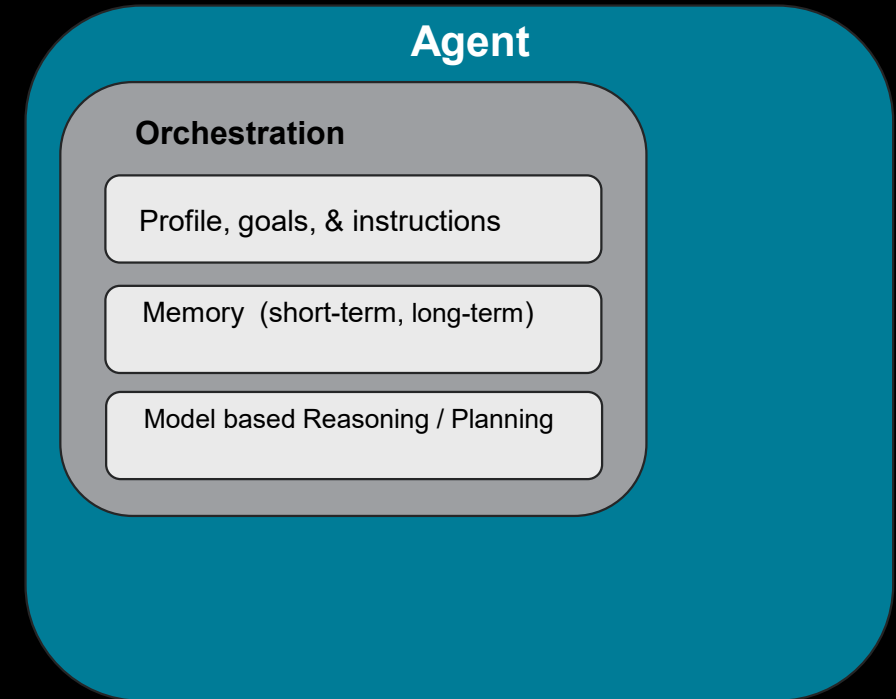
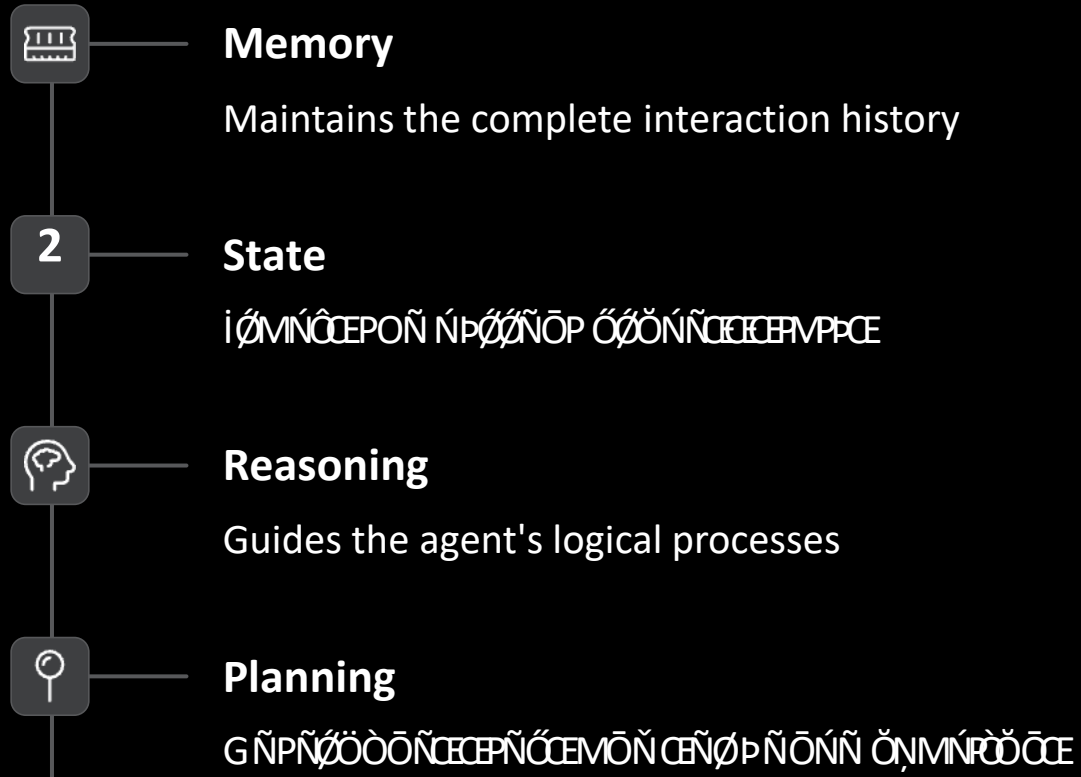
```
        return max(scored_actions, key=lambda x: x[1])[0]

    def available_strategies(self):
        return [
            MaintainTemperature(),
            StandardCooling(),
            EnergySavingCooling(),
            EarlyCooling(),
            DelayUntilCheaperEnergy()
        ]

    def execute(self, chosen_strategy):
        chosen_strategy.apply()
        self.collect_feedback()
        self.refine_policy()

    def run(self):
        world_state = self.observe_environment()
        strategy = self.decide(world_state)
        self.execute(strategy)
```

The Orchestration Layer: The Control Center



The orchestration layer serves as the AI agent's control center, processing queries in any format (text, audio, video, or image). When receiving a request like "I want to host a webinar for my students," this layer activates to coordinate the appropriate response. Through its four core components, this layer maintains conversation history, tracks process state, guides reasoning, and plans action sequences. It continuously communicates with the language model to process information and execute decisions effectively.

Understanding Memory in AI Agents

Definition

Memory in AI Agents refers to the ability of agents to store, retrieve, and update context over time to improve task performance, enable continuity, and foster autonomy.

Memory Types:

- Short-Term Memory (STM): Transient memory used for immediate reasoning within a session or short task window.
- Long-Term Memory (LTM): Persistent memory retained across sessions, interactions, and goals.

Importance

Without STM, agents lose the thread of context during complex tasks. Without LTM, they become reactive rather than adaptive.

Short-Term Memory (STM) : The Working Memory

- **Key Characteristics**

Volatile and session-scoped, stored in vector stores, memory buffers, or in-context prompts. Often managed using token windows with specific limits in LLMs.

- **STM Mechanisms**

Prompt Engineering: Selectively injecting prior messages into prompts. Memory Buffers: Maintaining a sliding window of recent interactions. Retrievers: Dynamically retrieving context-relevant documents.

- **Limitations**

Token limits cap the amount of context. Doesn't persist across sessions. Not designed for learning or updating knowledge.

Long-Term Memory (LTM) : Identity and History

- **Persistent Storage**
Cross-session memory in external databases
- **Indexed & Retrievable**
Organized for efficient access
- **Personalization**
 - Tailoring interactions over time
- **Knowledge Building**
Foundation for learning and improvement

Long-term memory allows agents to build knowledge over time, personalize interactions, and improve performance through experience. It's crucial for creating AI systems that can maintain coherence across multiple sessions and adapt to user preferences.

Key components of Memory Architectures

- **Embeddings Generator**

Converts text chunks into high-dimensional vectors for semantic search, used by both STM and LTM for retrieval and indexing.

- **Vector Store**

Stores long-term memory and enables approximate nearest-neighbor search using tools like FAISS, Weaviate, Pinecone, or pgvector.

- **Memory Controller**

Governs what to remember and when, deciding memory write policies and retrieval strategies.

- **Retriever-Augmented Generator (RAG)**

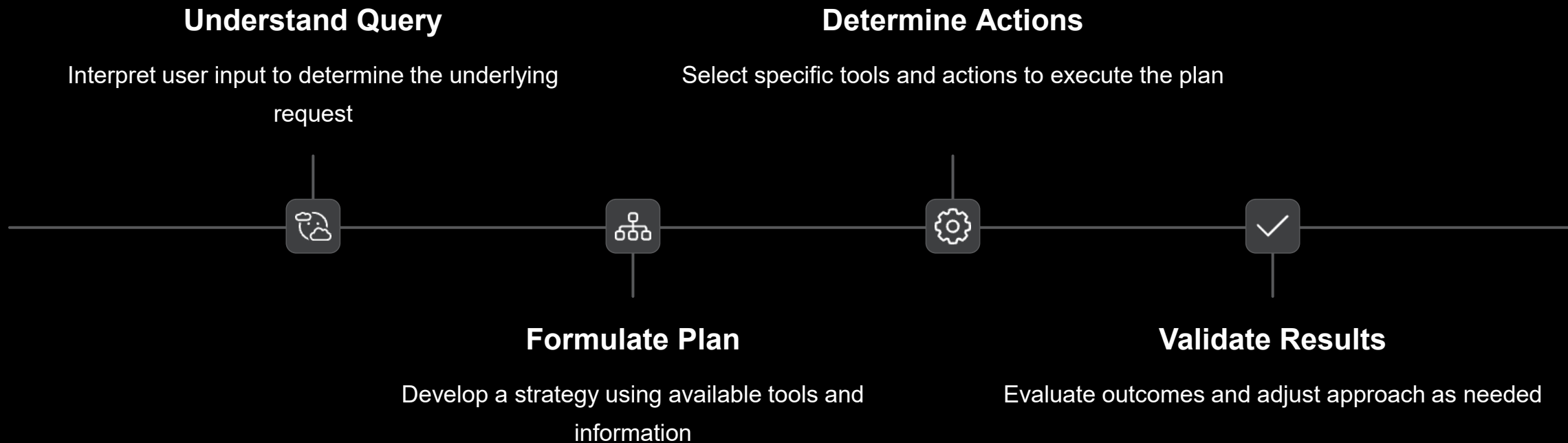
Combines LTM retrieval with STM generation, ensuring relevant memory is injected into prompts or agent reasoning loops.

- **Reflection and Update Module**

Performs post-task analysis to update LTM with key learnings, used in agents with planning capabilities.

Models: The Brain of AI Agents

Model



The model functions as the agent's decision-maker, typically implemented as a Large Language Model. It processes queries using reasoning frameworks like ReAct, Chain-of-Thought, and Tree-of-Thoughts.

These frameworks enable reasoning through intermediate steps, exploring multiple solution paths, and ensuring deliberate actions rather than simple reactions. Based on this process, the model selects the appropriate tools to accomplish tasks.

Tools: The Hands of AI Agents

Computational Tools

Handle mathematical operations and data analysis beyond the model's native capabilities.

API Connections

Link to external services to retrieve data, update systems, or trigger actions across platforms.

Information Retrieval

Access current information through searches and databases to complement the model's knowledge.

Physical Interfaces

Control physical devices, allowing agents to interact with the real world through sensors and actuators.

Tools extend AI agents beyond their built-in capabilities, functioning as specialized extensions for interacting with the external world. They enable agents to perform specific actions, access real-time information, and complete tasks otherwise impossible.

By combining LLM reasoning with specialized tools, agents can tackle complex problems requiring both intelligence and specific functional capabilities.



The Problem with Traditional AI Integrations

Data Isolation

Sophisticated AI models are often isolated from critical data. They are trapped behind information silos.

Custom Implementations

Each new data source requires custom implementations. This creates fragmented tool interactions.

Lack of Standardization

No standard interface exists for tool execution. There is a lack of a unified communication protocol.

Traditional AI integrations are constrained by fragmentation. This isolation limits their potential and efficiency.

A2A vs MCP: Complementary Protocols for AI Agents

Two distinct approaches working together to revolutionize the AI agent ecosystem

Agent-to-Agent Protocol (A2A)

Google's protocol enabling direct communication between AI agents.

Focuses on agent-to-agent interactions in multi-agent systems.

Model Context Protocol (MCP)

Universal protocol for AI tools and external systems integration.

Emphasizes connecting AI to real-world applications and services.

Rather than competing, these protocols serve complementary roles in the evolving AI ecosystem.

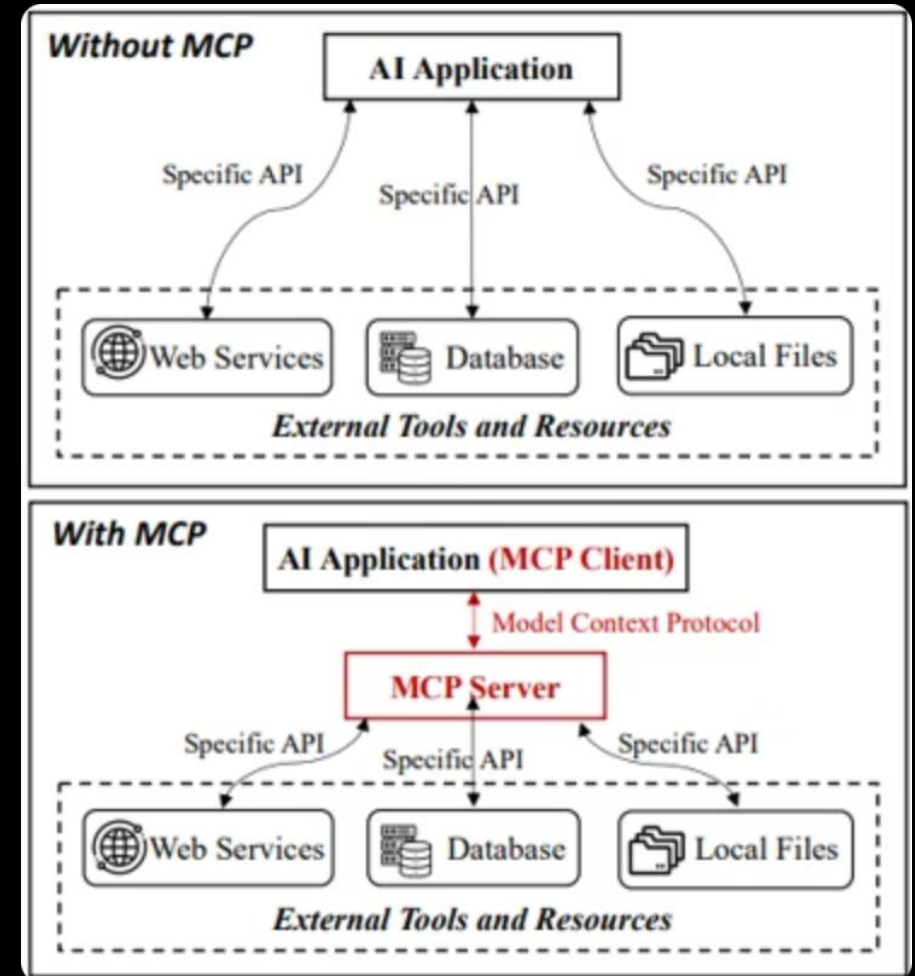
Tool invocation with MCP

MCP eliminates traditional manual API wiring that required custom connections for each external service.

1

Without MCP

AI applications needed specific APIs for each external tool, creating complex maintenance challenges.



Source: Arxiv Paper 6. April 2025: Model Context Protocol (MCP): Landscape, Security Threats, and Future Research Directions

2

With MCP

A standardized protocol creates a unified interface between AI models and external tools.

3

MCP Server

Acts as intermediary, handling authentication, data transformation, and error management across services.

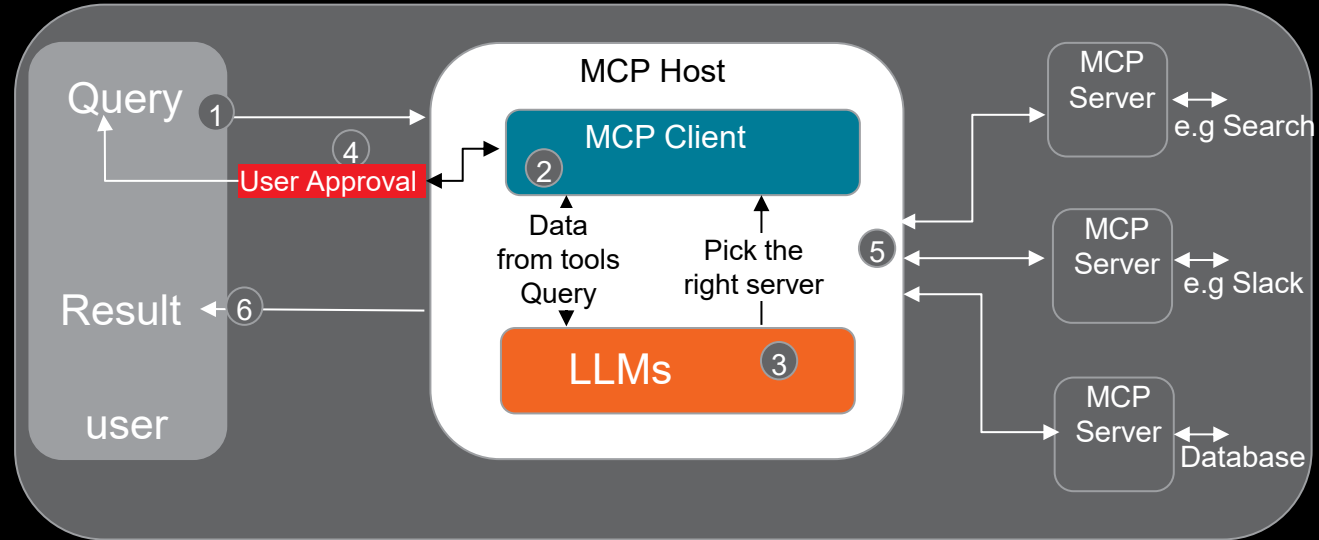
4

Dynamic Connectivity

Enables AI applications to access multiple resources without custom API development.

Workflow behind MCP

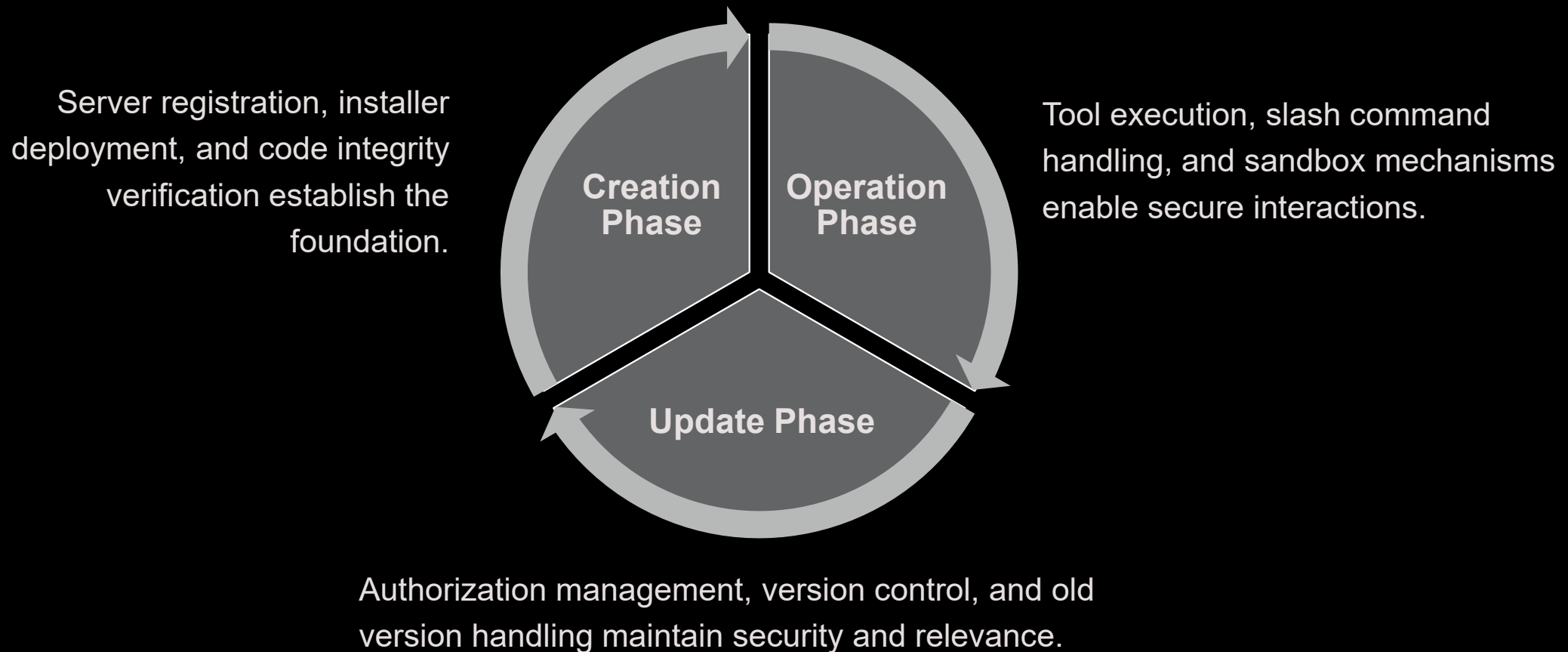
- 1. User Query** – The user asks for a task to be completed.
- 2. MCP Client → LLM** – The client sends the request to the language model.
- 3. LLM Selects Tool/Server** – The model decides which MCP server can handle the task.
- 4. (Optional) User Approval** – The user approves access to the selected server or tools.
- 5. MCP Server Executes Task** – The server runs tools or accesses data to complete the task.
- 6. Result Returned to User** – The final result is sent back through the client to the user.



MCP turns LLMs into safe, tool-using agents by standardizing how models discover, select, and interact with external services.

MCP Server Lifecycle

The MCP server lifecycle consists of three interconnected phases that ensure secure and efficient operations.



MCP Architecture advantages

The Model Context Protocol structures AI interactions through organized contextual information. It creates a standardized foundation for complex agent operations.

- **Structured Format**
MCP organizes inputs into interpretable, machine-readable contexts for LLMs.
- **Decoupled Architecture**
Standardized input types separate model logic from prompt engineering
- **Modular Schema**
Different input types are explicitly categorized for reasoning, planning and remembering.
- **Integration Ready**
Facilitates agent modularity, tool implementation, and memory management.

Challenges and Opportunities with MCP

Tool Misuse

AI does not always use tools correctly. This requires understanding complex tool descriptions.

Task Confusion

AI can get confused with complex tasks. It needs better guidance and feedback.

Ongoing Research

Current research focuses on improving prompt chaining, feedback loops, and fine-tuning tool usage.

AI faces challenges in tool understanding. Current research aims to improve AI's ability to handle complex tasks.

A2A Protocol – Agent-to-Agent Communication

A2A is a protocol introduced by Google DeepMind to enable **structured, modular, and open communication between AI agents**. It is designed to support **interoperability** across multi-agent systems, particularly those composed of **specialized, autonomous agents** developed independently.

Motivation

In real-world applications of agentic AI (e.g. multi-agent workflows, LLM+tool agents, planners + executors), it's essential to:

- Have **standardized messaging** between agents
- Enable **task delegation, capability negotiation, and status reporting**
- Avoid brittle, ad hoc prompt chaining

A2A provides a protocol to **formalize these agent interactions**, inspired by the success of web protocols like HTTP and RPC in distributed systems.

A2A vs MCP: Main Differences

These complementary protocols serve different functions in the AI agent ecosystem. Understanding their distinctions helps implement the right solution.

Feature	A2A Protocol	MCP Protocol
Focus	Communication between agents	Internal context structure for reasoning
Main Use	Inter-agent messaging & task exchange	Structuring prompt inputs and memory
Analogy	Like HTTP for agents	Like a note-taking system for agents
Format	JSON-based messages	Modular prompt blocks

While A2A enables agent-to-agent communication, MCP enhances an individual agent's cognitive abilities. Together, they create a powerful ecosystem.

AI Agent Types - Comparing Capabilities

Different types of AI agents excel at various tasks. Understanding these differences helps match the right agent to your needs.

Agent Type	Characteristics	Examples	Use Cases
Fixed Automation	No intelligence, predictable behavior, limited scope	RPA, email autoresponders, basic scripts	Repetitive tasks, structured data
LLM-Enhanced	Context-aware, rule-constrained, stateless	Email filters, content moderation	Flexible tasks, high-volume/low-stakes scenarios
ReAct	Multi-step workflows, dynamic planning, problem-solving	Travel planners, AI dungeon masters	Strategic planning, multi-stage queries
ReAct + RAG	External knowledge access, low hallucinations	Legal research tools, medical assistants	High-stakes decisions, domain-specific tasks

AI Agent Types - Comparing Capabilities (Continued)

Agent Type Characteristics Examples Use Cases Tool-Enhanced Multi-tool integration, dynamic execution Code generation tools, data analysis bots Complex...

AI Agent Types - Comparing Capabilities

(Continued)

Agent Type	Characteristics	Examples	Use Cases
Tool-Enhanced	Multi-tool integration, dynamic execution	Code generation tools, data analysis bots	Complex workflows requiring multiple tools
Self-Reflecting	Meta-cognition, explainability, self-improvement	Self-evaluating systems, QA agents	Tasks requiring accountability
Memory-Enhanced	Long-term memory, personalization, adaptive	Project management AI, personal assistants	Individualized experiences, long-term interactions
Self-Learning	Autonomous learning, adaptive/scalable, evolutionary	Neural networks, swarm AI, prediction models	Cutting-edge research, autonomous systems

Industry Applications (1- 4)

1. Retail

Personalized Shopping Experience: AI agents recommend products

Inventory Management: AI can autonomously track stock

2. Manufacturing

Predictive Maintenance: AI agents monitor equipment

Supply Chain Optimization: AI manages inventory

3. Education

Personalized Learning: AI adapts learning materials

Automated Grading: AI agents handle grading of assignments

4. Telecommunications

Network Optimization: AI monitors network performance in real-time

Customer Support Automation: AI chatbots support

Pictures Source from AMD Studio Stock Image Library



Industry Applications (5 – 8)

5. Healthcare

Patient Monitoring: AI agents track patient vitals and alert healthcare providers

Personalized Treatment Plans: AI analyzes patient data

6. Biosciences

Drug Discovery: AI agents autonomously sift through vast datasets

Gene Editing: AI simulates the effects of gene edits

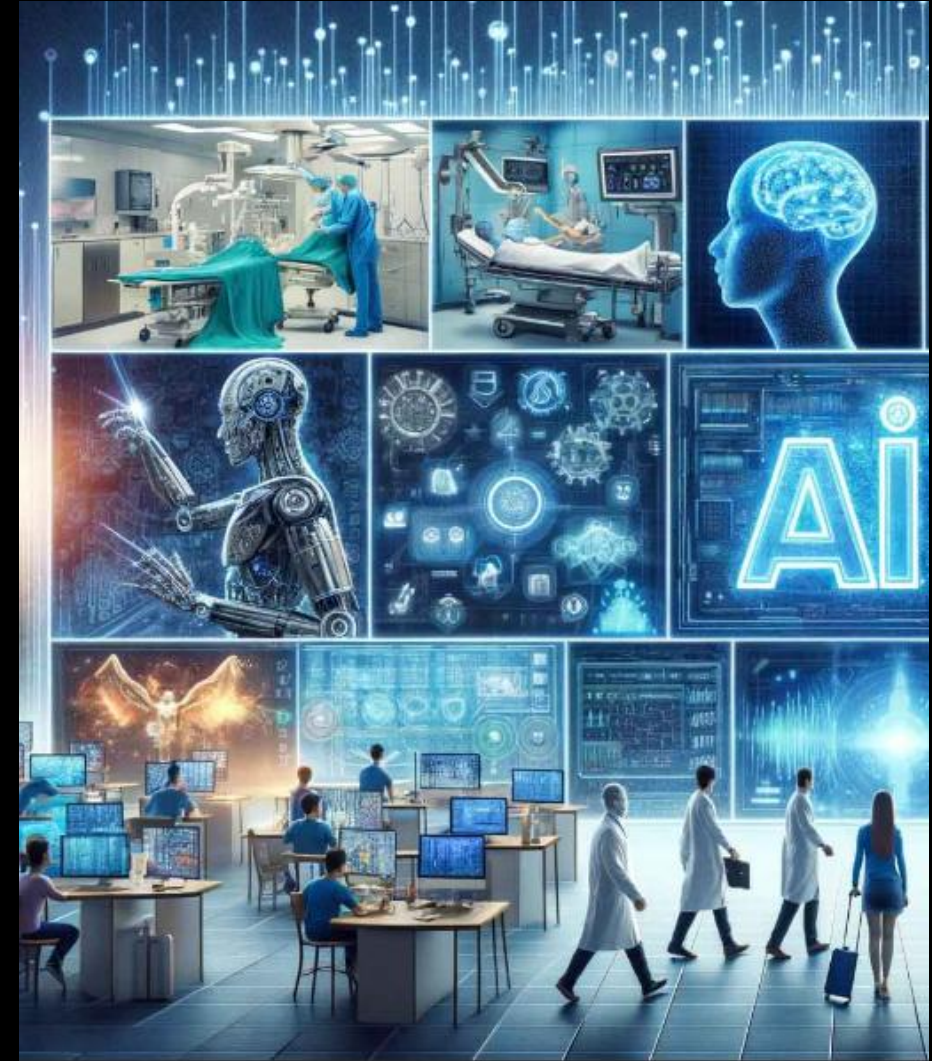
7. Finance & Insurance

Fraud Prevention & Risk Assessment: AI detects fraud in real-time and automates risk analysis, enhancing financial security and credit evaluations

8. Construction

Predictive Maintenance: AI agents monitor equipment

Supply Chain Optimization: AI manages inventory



Most used AI Agents in the industry

Computer Using Agents (CUA):

These general-purpose AI agents combine vision and language to handle complex multi-agent workflows.

Examples include e.g. **Anthropic's** Claude and **OpenAI's** Operator, with others like e.g. **Llama**, **Google** Gemini and Manus AI pushing boundaries.

Coding Agents:

Coding agents are booming. They help developers speed up coding, reviewing, testing, and deployment.

Cursor has hit \$200M ARR in 2025, and open-source projects like e.g. **Windsurf** and **GitHub** Code have gained massive attention.

Workflow Automation:

Think of this as the evolution of no-code AI automation, letting businesses build AI workflows easily.

In 2025 **n8n** and **OneReach**, is a key player here, enabling drag-and-drop AI agent building, alongside **Zapier**

Agentic RAG:

One of the most popular enterprise AI tools since the start of LLM.

Next-gen retrieval systems, like e.g. **Perplexity** AI and Glean, are already powering big enterprise solutions.

Voice Agents:

Voice automation is heating up. Automating sales calls and customer support with smart voice AI is becoming a must-have for large companies.

Look at players like e.g. **ElevenLabs** and **Retell AI** leading the change.

Tool-based Agents:

These agents focus on specific tasks by using specialized tools to get results.

Though niche, they're vital in their spaces like Clay's agent or Breeze by **HubSpot**.

Choosing the Right Agent

Key Takeaway

No one-size-fits-all solution exists. Each agent type has its sweet spot for specific use cases. Consider the complexity of tasks, required reasoning abilities, and necessary memory capabilities when selecting an agent.

- Fixed Automation agents excel at routine, predictable tasks
- ReAct agents shine in problem-solving scenarios
- Memory-Enhanced agents are ideal for personalized interactions

Next Steps

Match agent type to your needs. Consider reliability for routine tasks versus adaptability for research. Evaluate the tradeoffs between simplicity and capability.

Begin with a clear definition of your requirements:

- Task complexity and variability
- Need for historical context retention
- Importance of self-improvement
- Integration with existing tools and systems

The most successful implementations often combine multiple agent types, creating hybrid solutions that leverage the strengths of each approach while mitigating their individual limitations. Start with simpler implementations and gradually increase complexity as you better understand your specific needs.

When to Use AI Agents - And When to Avoid Them

✅ Use Agents When:

- Workflows can't be predetermined
- Tasks need flexible decisions
- Complex queries require multiple tools
- Context understanding matters
- Tasks don't fit linear processes

Example: "Coming Monday but forgot passport. Might arrive Wednesday. Can I surf Tuesday with cancellation insurance?"

❌ Avoid Agents When:

- Fixed workflows work well
- Simplicity is a priority
- Tasks fit clear categories
- Process consistency is vital
- Error tolerance is minimal

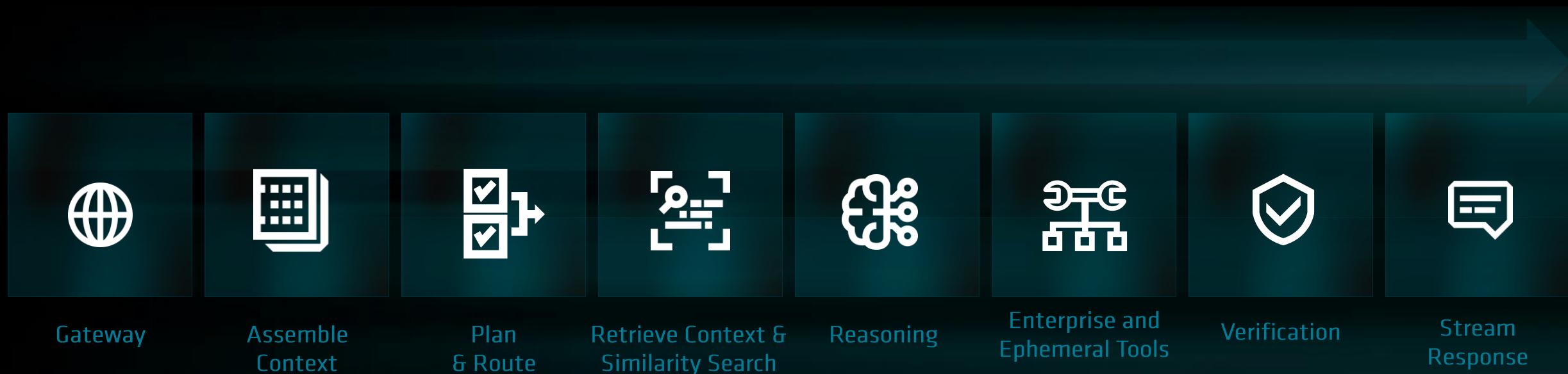
Example: Basic customer service: "Need trip information? → Search knowledge base" or "Want to talk to sales? → Contact form"

Ask yourself: "Do I need workflow flexibility to solve this task?" If predefined processes consistently fail, agents are likely needed. For reliability, default to simpler approaches when possible.

Agentic systems handle real-world tasks that don't fit predetermined workflows - advancing beyond traditional programming's reliance on extensive if/else conditions for complexity.

Agentic AI Is a Pipeline of Distinct Compute Stages

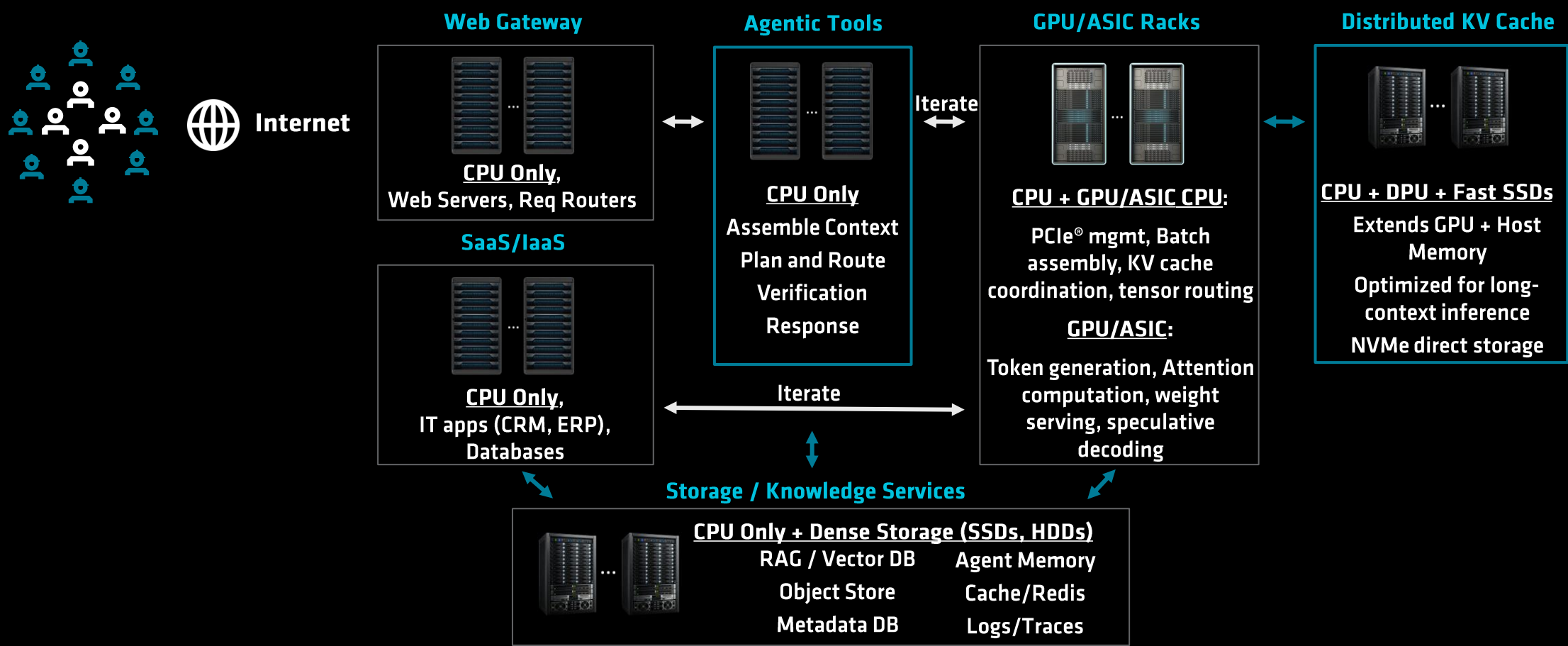
From Gateway to Stream Response, Each Stage Places Unique Demands on the CPU



Agentic AI continuously moves through compute, memory, I/O, security, and acceleration-intensive stages.

Agentic AI Workflow Creating New Data Center Tiers

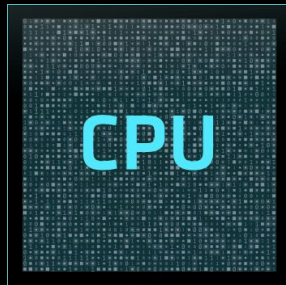
- ↔ Compute/Tools flows
- ↔ Storage flows
- New Design Points



Agentic AI Drives Multiple CPU Clusters – General Purpose, Agentic Tools, Storage Nodes, AI Host for GPU Clusters and More

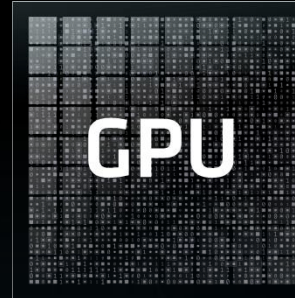
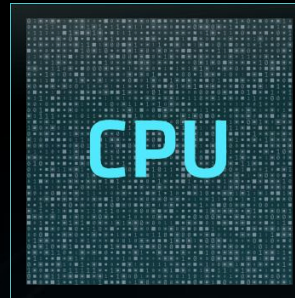
GENERAL PURPOSE CPU SERVERS

Broad portfolio trusted by
Enterprise & Hyperscale leaders



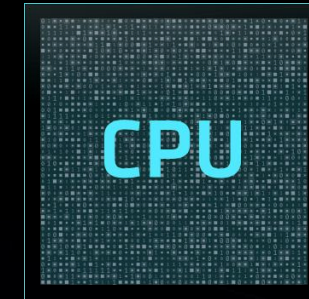
GPU SERVERS “HOST NODE”

Maximize GPU Utilization
and token generation



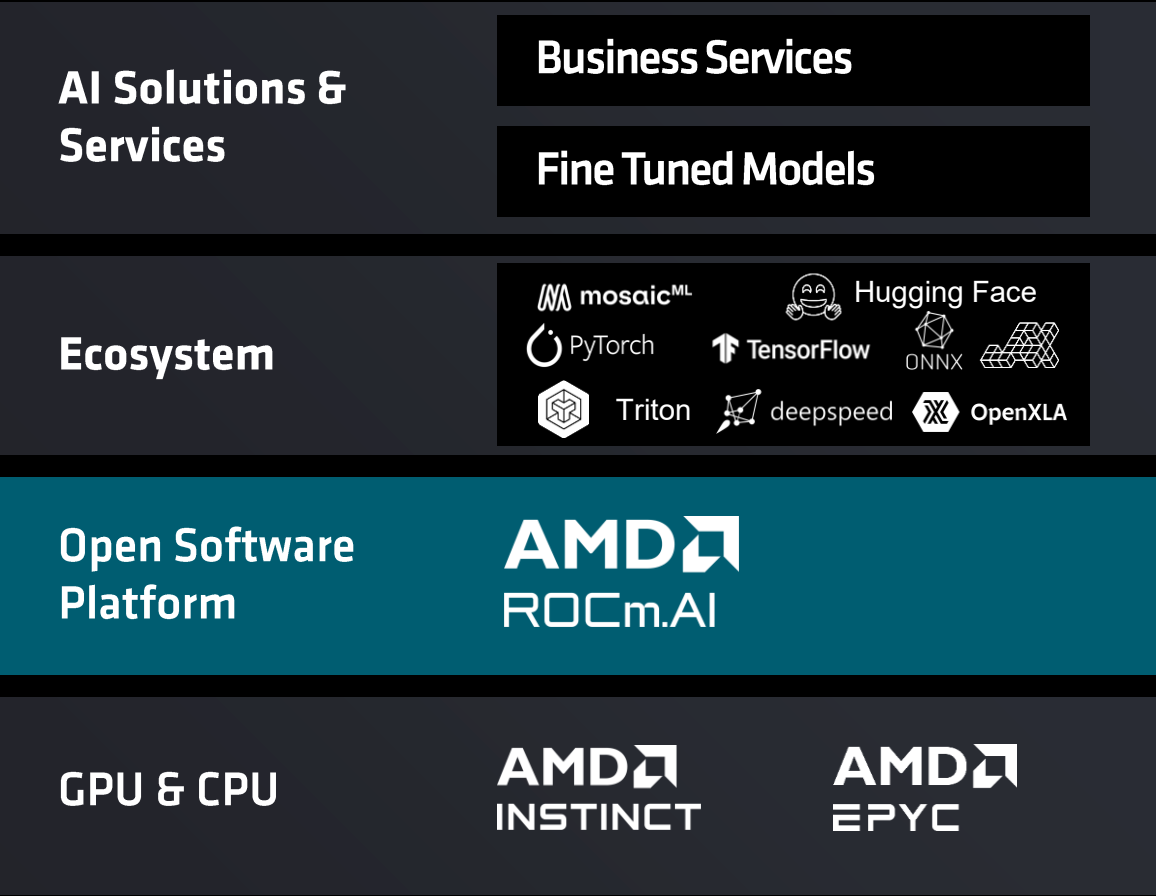
CPU SERVERS “SANDBOXES”

Scale to Max
Agent Capacity

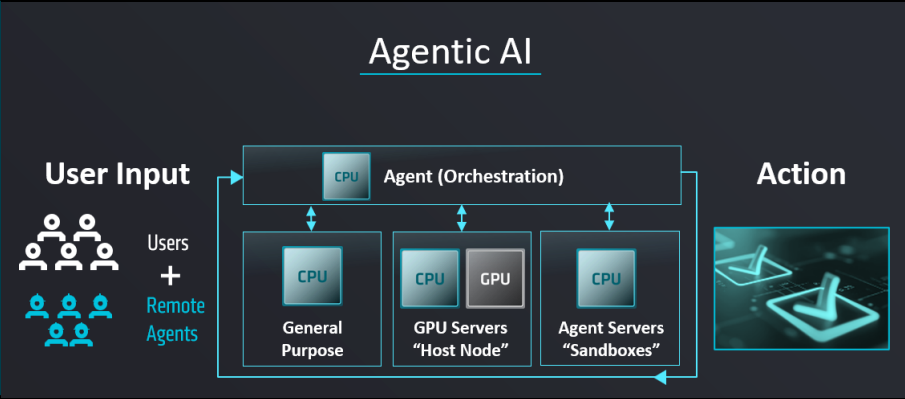


World's Best Server CPU Portfolio for the Agentic Era

AMD ROCm.AI™ Software Stack for CPUs



AMD ROCm.AI for the end-to-end agentic AI pipeline across both AMD Instinct™ GPUs and AMD EPYC™ CPUs



Summary and Evolution of AI Agents

1. The Foundation: Basic LLMs

- Simple Input/Output text processing
- Trained on vast datasets
- The building block of everything that followed

2. Enhanced Context Windows

- Breaking past 8k token limits
- Processing entire documents
- Deeper understanding of complex inputs

3. RAGs & Tool Integration

- Real-time information access
- Integration with external tools
- Enhanced response accuracy through APIs

4. Multi-Modal Revolution

- Processing diverse input types
- Real-time data retrieval
- Complex task handling with minimal supervision

5. Current Architecture (Where We Are (end of 2025))

- Advanced memory systems (Short-term, Long-term, Episodic)
- Sophisticated tool-calling capabilities
- ReACT-based decision making
- Semantic knowledge processing

We are moving from **language models as tools** to **language models as controllers of intelligent systems**.

References and further reading

1. Shafran, I., Cao, Y. et al., 2022, 'ReAct: Synergizing Reasoning and Acting in Language Models'. Available at: <https://arxiv.org/abs/2210.036292>.

Wei, J., Wang, X. et al., 2023, 'Chain-of-Thought Prompting Elicits Reasoning in Large Language Models'. Available at: <https://arxiv.org/pdf/2201.11903.pdf>.3.

Wang, X. et al., 2022, 'Self-Consistency Improves Chain of Thought Reasoning in Language Models'. Available at: <https://arxiv.org/abs/2203.11171>.4.

Diao, S. et al., 2023, 'Active Prompting with Chain-of-Thought for Large Language Models'. Available at: <https://arxiv.org/pdf/2302.12246.pdf>.5.

Zhang, H. et al., 2023, 'Multimodal Chain-of-Thought Reasoning in Language Models'. Available at: <https://arxiv.org/abs/2302.00923>.6. Yao, S. et al., 2023, 'Tree of Thoughts: Deliberate Problem Solving with Large Language Models'. Available at: <https://arxiv.org/abs/2305.10601>.7.

Long, X., 2023, 'Large Language Model Guided Tree-of-Thought'. Available at: <https://arxiv.org/abs/2305.08291>.8.

Google. 'Google Gemini Application'. Available at: <http://gemini.google.com>.9.

Xie, M., 2022, 'How does in-context learning work? A framework for understanding the differences from traditional supervised learning'. Available at: <https://ai.stanford.edu/blog/understanding-incontext/>.

Q&A

Thank you for your attention! Questions or discussion?



Break until 12:10

Hands-On

AMD AI Developer Program

Build Faster Learn Deeper Scale Smarter

AMD AI Developer Program Benefits

Cloud Credits

Start building with free cloud credits from **AMD Developer Cloud credits** or **Fireworks AI credits** when you join.

Community Access

Join our **Discord** and **forums** to get help, share ideas, and dive into technical conversations.

Learning Resources

- ✓ Explore **AI Academy** to access expert-led and self-paced AI courses.
- ✓ Enjoy one-month complimentary DeepLearning.AI pro membership.*

Showcase Opportunities

Share your projects for a chance to be featured on AMD Developer channels and recognized at AMD events

AI Developer Program provides the compute, expertise, and community developers need to accelerate AI innovation

*Terms and conditions apply see <https://www.amd.com/en/developer/ai-dev-program.html#faqs>.

AMD AI Developer Program

Who Should Join?

The AMD AI Developer program is built for developers who:

- Build or train AI models
- Optimize compute-intensive workloads
- Work with open-source frameworks
- Explore AI acceleration on AMD hardware



How to Join?

- Create your free AMD account and complete the AI Developer Program form
- Verify your email using the activation code and access the AMD AI Developer program portal
- Join the community channels to start connecting with fellow developers
- Submit the form to request your cloud credits to choose between AMD Developer Cloud credits or Fireworks AI Credits
- Stay tuned for member emails about events, training, and new resources

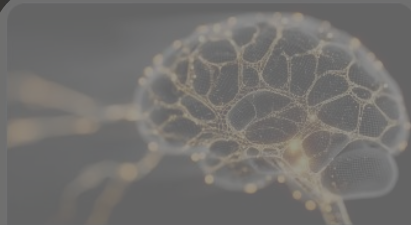
[Join Now](#)

AMD AI Academy

Self-learning courses

Learning Paths

[View All >](#)



AI Developer

Coursework developed to help AI developers design, build, and deploy intelligent systems.

📖 9 Courses ⌚ 11 Hour(s)

[View Pathway →](#)



Enterprise AI And Infrastructure Developer

Learn how to deploy, scale, and manage AI workloads across enterprise infrastructure.

📖 1 Courses ⌚ 1 Hour(s)

[View Pathway →](#)



GPU Kernel Developer

Learn how to leverage advanced tools and libraries simplify GPU kernel development.

📖 1 Courses ⌚ 1 Hour(s)

[View Pathway →](#)



Academic Developers

Learn how to empower teaching and research with AI through hands-on tutorials and modern techniques that enhance coursework, support student...

📖 6 Courses ⌚ 13 Hour(s)

[View Pathway →](#)

<https://www.amd.com/en/developer/resources/training/amd-ai-academy.html>

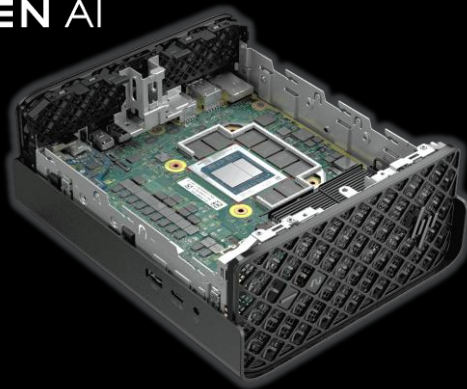
AUP Learning Cloud: Example Configuration

AMD
RYZEN AI

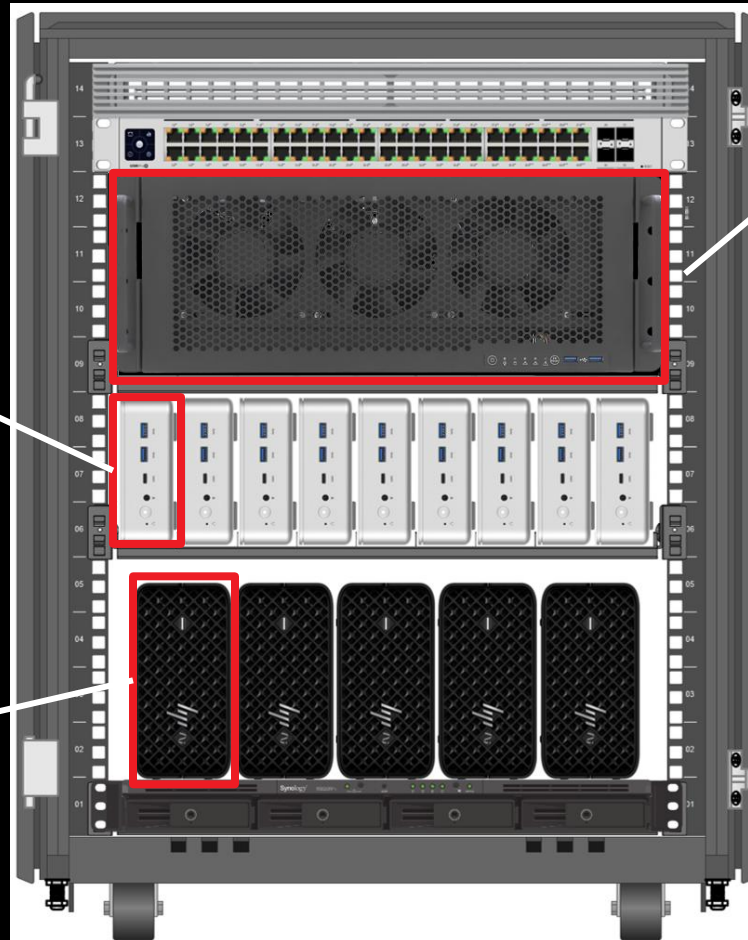


Minisforum™ A370¹

AMD
RYZEN AI



HP® Z2 Mini G1a²



ASRock® GAI4G-R9700 Radeon AI Pro Workstation³



Apply for Access

AMD
RYZEN AI
PRO

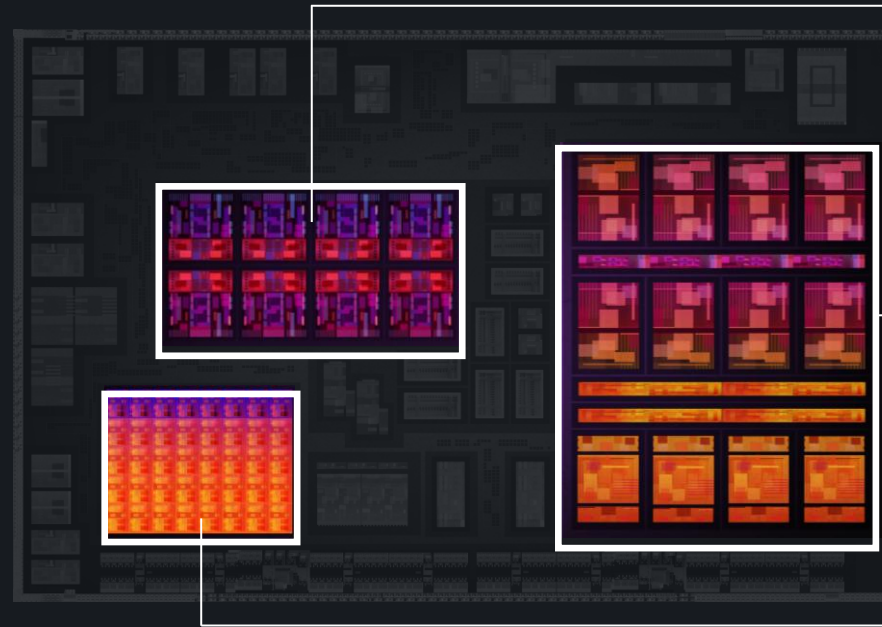
[1] <https://www.minisforum.com/products/elitemini-ai370-a>

[2] <https://www.hp.com/us-en/workstations/z2-mini-a.html>

[3] <https://www.asrock.com/Graphics-Card/spec/product.asp?Model=GAI4G-R9700>

Providing the Next Level of NPU, CPU, and GPU Architectures for Next-Gen AI PC Experiences

3rd Generation
AMD Ryzen™ AI
Best in class AI platform



AMD
RDNA 3.5

Next-Gen GPU
Up to 40 Compute Units



Next-Gen CPU
Up to 12 Cores, 24 Threads

AMD
XDNA 2

Next-Gen NPU
Industry-leading 50+ NPU TOPS

**The demand
for tokens is
growing
exponentially**

Agent harnesses
can use more than a
million tokens per day

*“Uber burned through it’s entire
2026 AI budget in four months.”*

FORTUNE

*“Microsoft cut its engineers off from
AI because the bill got too big”*

yahoo!
finance

*“Amazon is the latest tech giant to face
the consequences of AI ‘Tokenmaxing’”*

CNET

Small Models are Becoming Incredibly Efficient

GPT-OSS 	
Number of Parameters	120B
GPQA Score* Graduate-Level Google-Proof Q&A	80.1

August 2025

Qwen 3.5 	
Number of Parameters	9B
GPQA Score* Graduate-Level Google-Proof Q&A	81.7

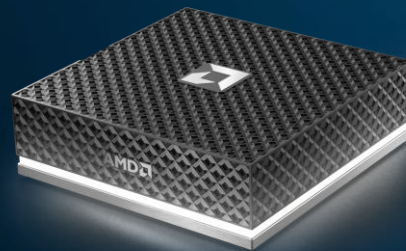
March 2026

One Portfolio for Every Client AI Workload

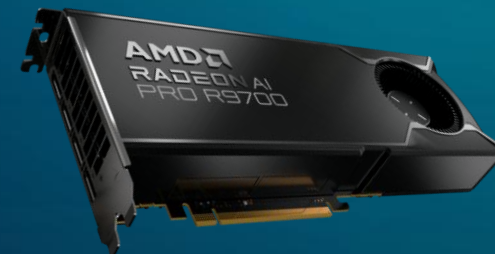
Personal agent



Agent team



Agent swarms



XS to M

Local Knowledge

Model Size : 8B - 24B

Context up to 16K

Min Memory 16 GB

Target TTFT, TPS 2 sec, 20+ tok/sec

8Qwn 3 8B 4K context

S to L

Multi-Step Agents

Model Size up to 300B

Context up to 32K-128K

Min Memory 64GB

Target TTFT, TPS 10 sec, 60 tok/sec

Qwen 3.5 35B 10K context

M to XL

Autonomous

Model Run concurrent SOTA Models

Context up to 128K

Memory 32GB-128GB

Target TTFT, TPS <1 sec, 120+ tok/sec

GPT OSS 120B 2x R9700 10K Context

AMD ROCm Support

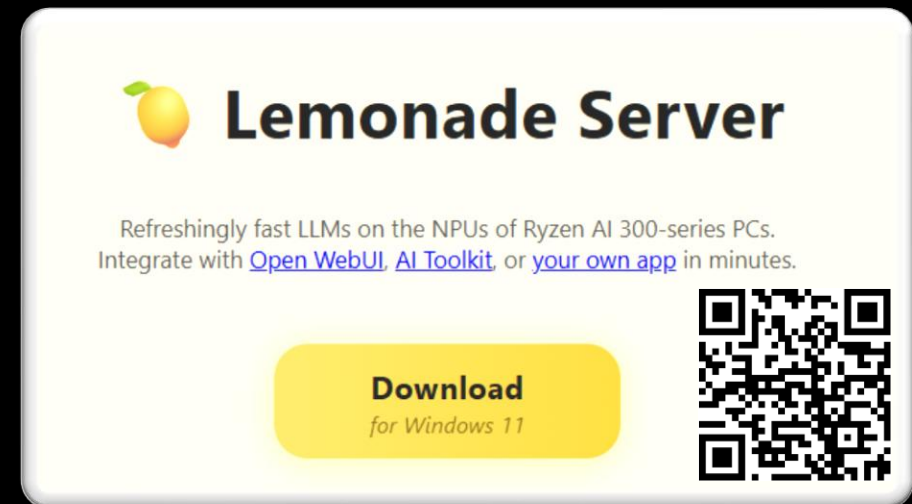
Running LLMs locally on Ryzen™ AI

Why?

- Privacy 🗝️
- Cost (avoid LLM subscriptions) 📉
- Control of your own model 🎮
- Predictable Latency 🎯
- Sometimes you are just on an airplane ✈️

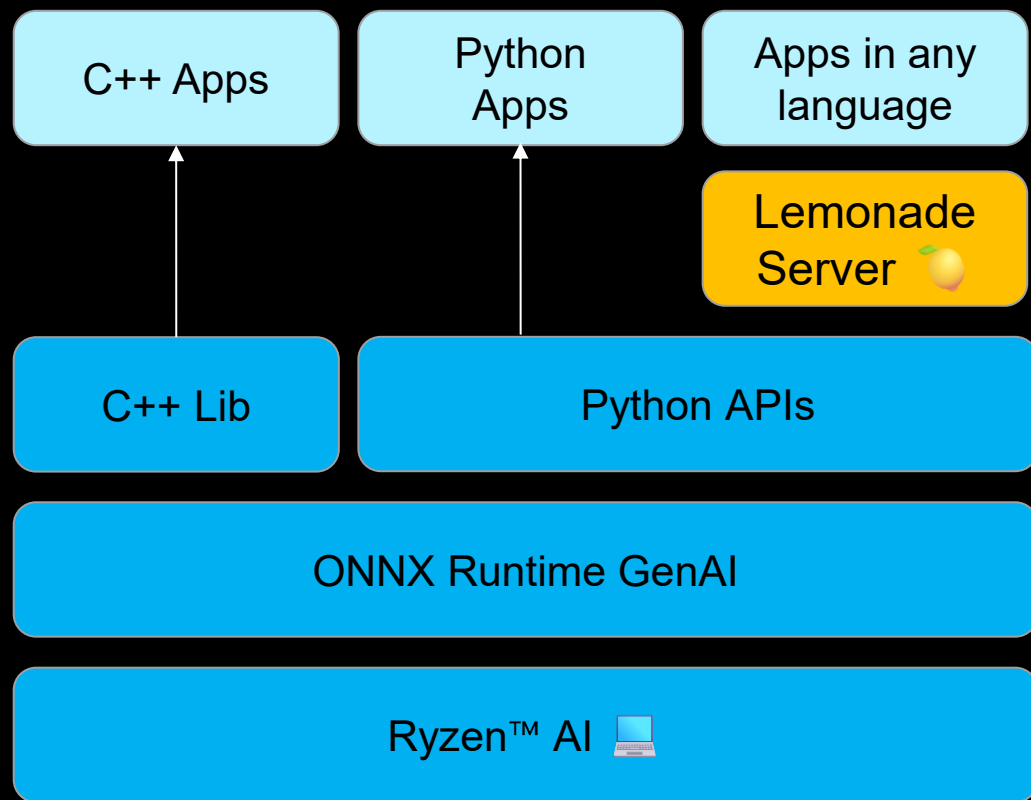
How?

- Many options available (Ollama, LlamaCPP, ...)
- 🍋 **Lemonade Server**
 - Focus: NPU + iGPU acceleration



Lemonade Server Overview

SW Stack



Open Source!

github.com \ **lemonade-sdk** \ **lemonade**

OpenAI Compatible!

- Develop using OpenAI Standard

```
from openai import OpenAI

client = OpenAI(base_url=lemonade_url)

completion = client.chat.completions.create(
    model=my_model,
    messages=input_message
)
```

- Connect to dozens of apps without changing a single line of code 🧩



Microsoft
AI Toolkit



Continue.dev



LM Eval
Harness



Open WebUI

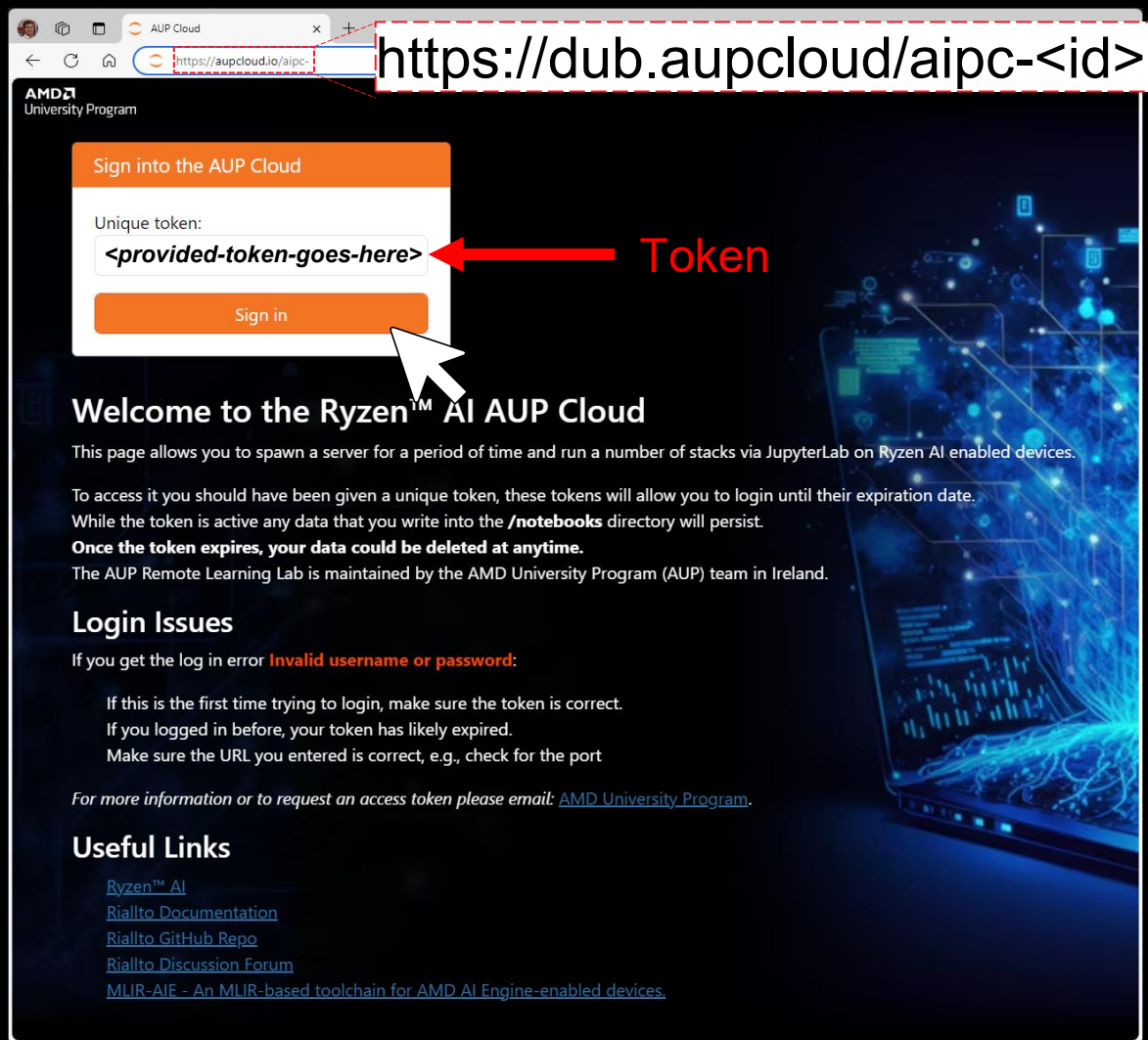


CodeGPT

AUP Cloud Ryzen AI cluster Login Information

- Login to JupyterHub via web browser
 - [https://dub.aupcloud.io/aipc-*<id>*](https://dub.aupcloud.io/aipc-<i><id></i>) – provided separately
 - Token: provided separately

JupyterHub login page



Sign into the AUP Cloud

Unique token:
<provided-token-goes-here> **Token**

Sign in

Welcome to the Ryzen™ AI AUP Cloud

This page allows you to spawn a server for a period of time and run a number of stacks via JupyterLab on Ryzen AI enabled devices.

To access it you should have been given a unique token, these tokens will allow you to login until their expiration date. While the token is active any data that you write into the **/notebooks** directory will persist. **Once the token expires, your data could be deleted at anytime.**

The AUP Remote Learning Lab is maintained by the AMD University Program (AUP) team in Ireland.

Login Issues

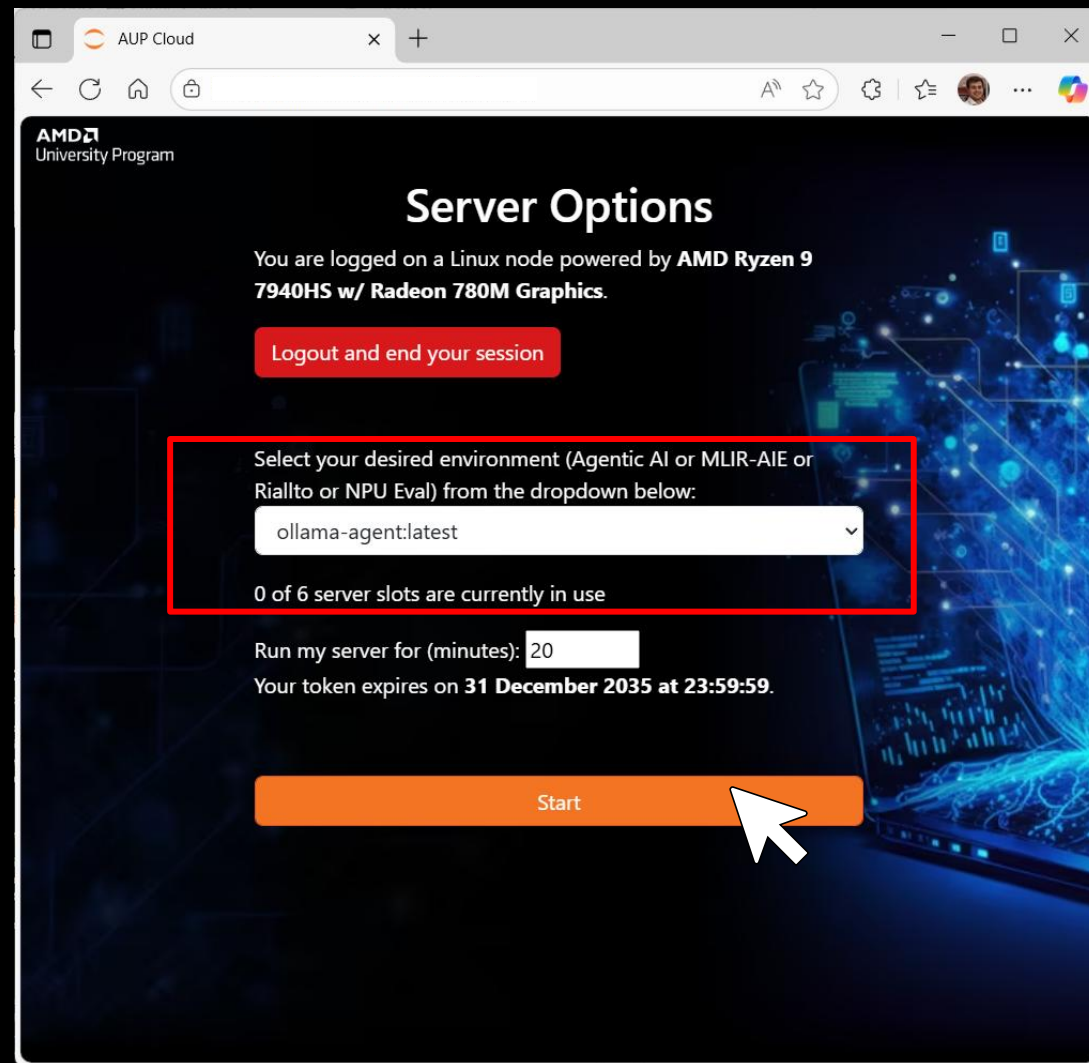
If you get the log in error **Invalid username or password:**

- If this is the first time trying to login, make sure the token is correct.
- If you logged in before, your token has likely expired.
- Make sure the URL you entered is correct, e.g., check for the port

For more information or to request an access token please email: [AMD University Program](#).

Useful Links

- [Ryzen™ AI](#)
- [Rialto Documentation](#)
- [Rialto GitHub Repo](#)
- [Rialto Discussion Forum](#)
- [MLIR-AIE - An MLIR-based toolchain for AMD AI Engine-enabled devices.](#)



Server Options

You are logged on a Linux node powered by **AMD Ryzen 9 7940HS w/ Radeon 780M Graphics.**

Logout and end your session

Select your desired environment (Agentic AI or MLIR-AIE or Rialto or NPU Eval) from the dropdown below:

ollama-agent:latest

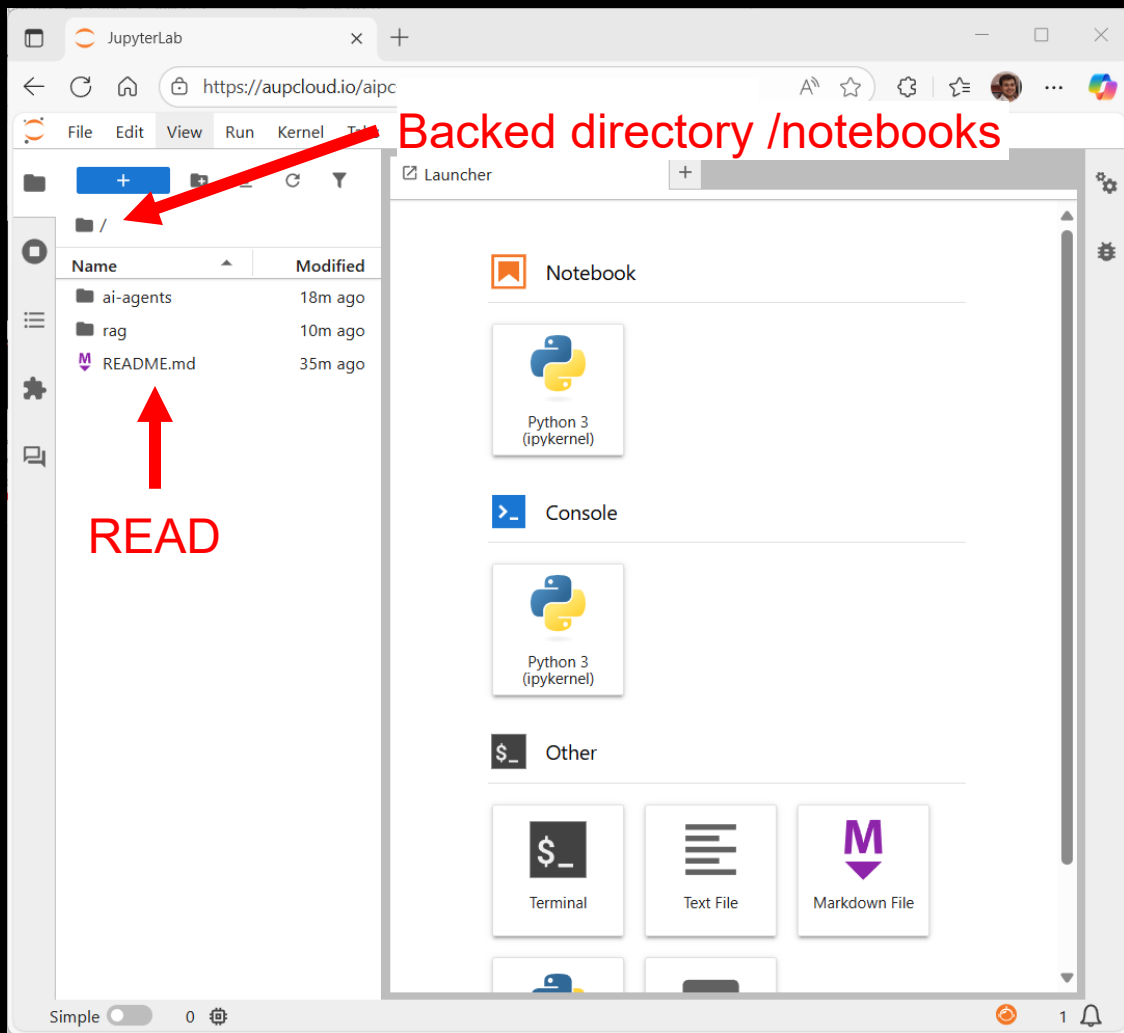
0 of 6 server slots are currently in use

Run my server for (minutes): 20

Your token expires on **31 December 2035 at 23:59:59.**

Start

JupyterHub working with Agents



Backed directory /notebooks

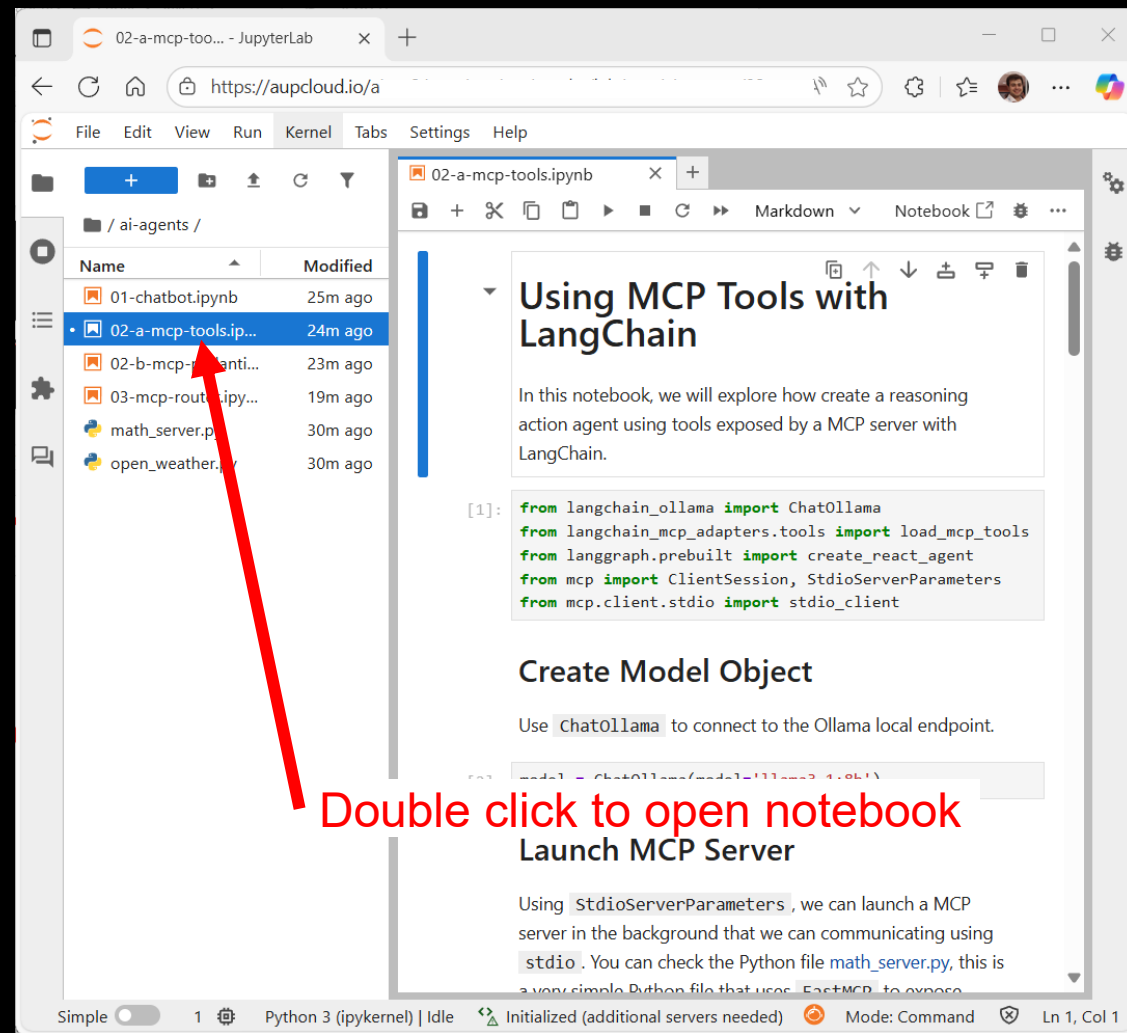
READ

File browser contents:

Name	Modified
ai-agents	18m ago
rag	10m ago
README.md	35m ago

Launcher options:

- Notebook (Python 3 (ipykernel))
- Console (Python 3 (ipykernel))
- Other (Terminal, Text File, Markdown File)



Double click to open notebook
Launch MCP Server

File browser contents:

Name	Modified
01-chatbot.ipynb	25m ago
02-a-mcp-tools.ip...	24m ago
02-b-mcp-...anti...	23m ago
03-mcp-rout...ipy...	19m ago
math_server.p...	30m ago
open_weather...	30m ago

Notebook content:

Using MCP Tools with LangChain

In this notebook, we will explore how create a reasoning action agent using tools exposed by a MCP server with LangChain.

```
[1]: from langchain_ollama import ChatOllama
from langchain_mcp_adapters.tools import load_mcp_tools
from langgraph.prebuilt import create_react_agent
from mcp import ClientSession, StdioServerParameters
from mcp.client.stdio import stdio_client
```

Create Model Object

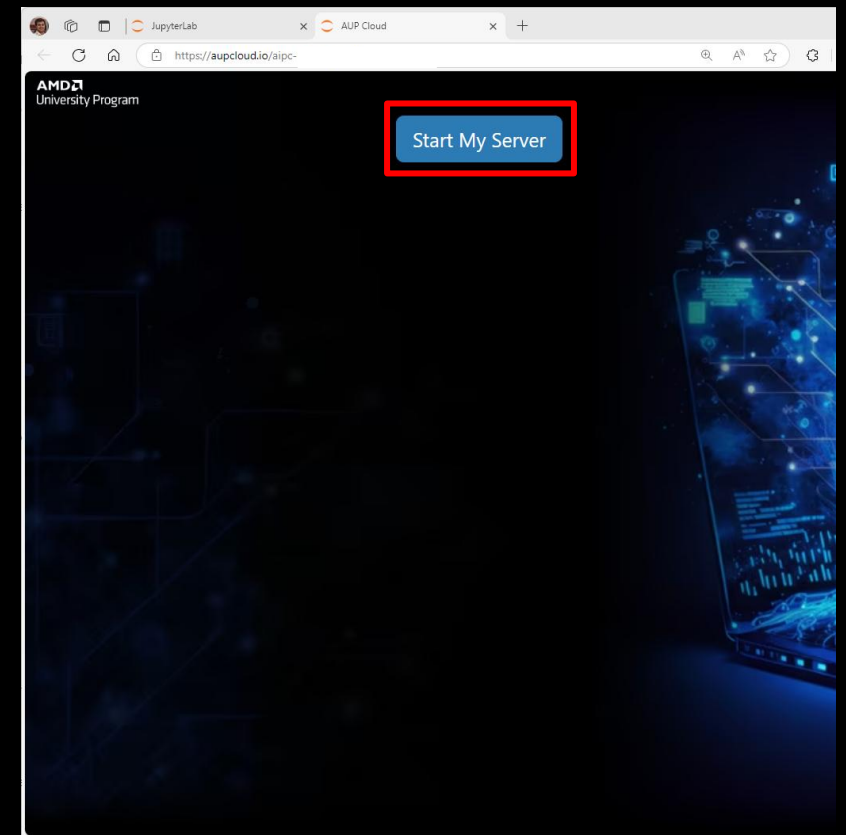
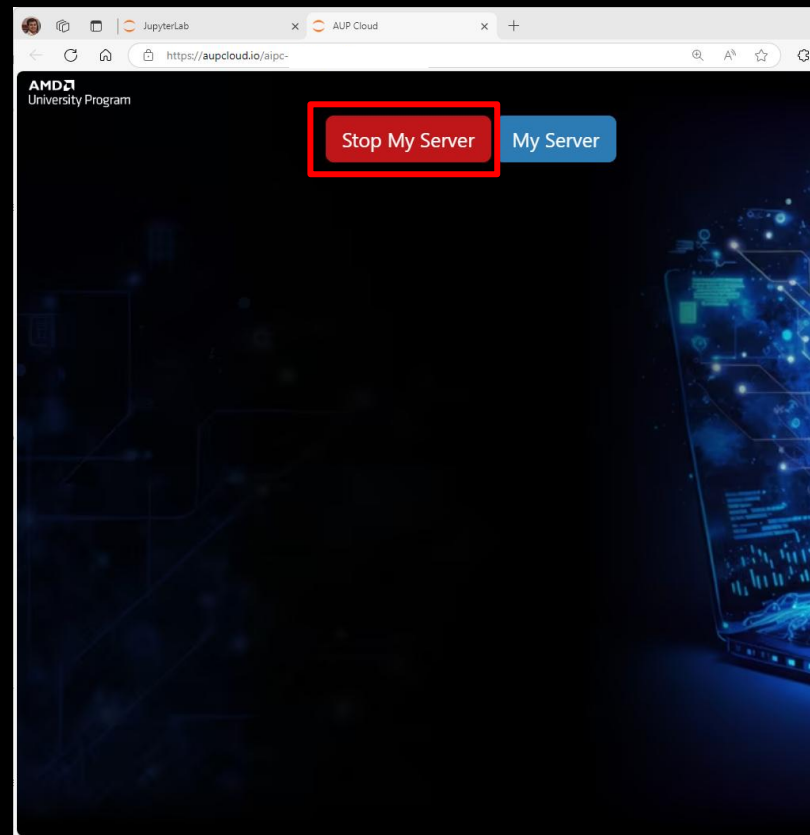
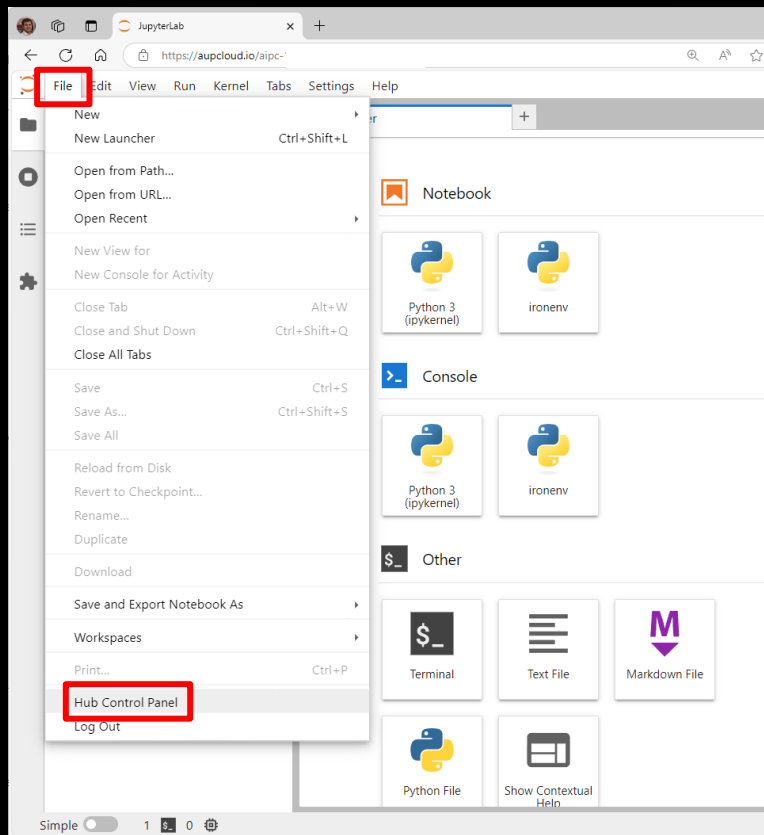
Use `ChatOllama` to connect to the Ollama local endpoint.

```
model = ChatOllama(ollama_base_url="http://localhost:11434")
```

Launch MCP Server

Using `StdioServerParameters`, we can launch a MCP server in the background that we can communicating using `stdio`. You can check the Python file `math_server.py`, this is a very simple Python file that uses `FastMCP` to expose

Stop the Server



Contact me/us

- mruiznog@amd.com
- aup@amd.com



Mario Ruiz
AMD University Program



AUP Teaching Resources

AMD University Program

Hub for Educators, researchers and students to access AMD resources, programs and news



AUP Hardware Donations

Apply to our donation program to request AMD hardware and software for teaching and research



AUP Learning Clouds

Access step-by-step guides to run AI workloads on AMD hardware



AUP AI Tutorials

Build real-world AI skills with free, hands-on learning.



Disclaimer & Attribution

Timelines, roadmaps, and/or product release dates shown in these slides are plans only and subject to change.

The information contained herein is for informational purposes only and is subject to change without notice. While every precaution has been taken in the preparation of this document, it may contain technical inaccuracies, omissions and typographical errors, and AMD is under no obligation to update or otherwise correct this. Advanced Micro Devices, Inc. makes no representations or warranties with respect to the accuracy or completeness of the contents of this document, and assumes no liability of any kind, including the implied warranties of noninfringement, merchantability or fitness for particular purposes, with respect to the operation or use of AMD hardware, software or other products described herein. No license, including implied or arising by estoppel, to any intellectual property rights is granted by this document. Terms and limitations applicable to the purchase or use of AMD's products are as set forth in a signed agreement between the parties or in AMD's Standard Terms and Conditions of Sale.

©2026 Advanced Micro Devices, Inc. All rights reserved. AMD, the AMD Arrow logo, Athlon, CDNA, EPYC, Infinity Fabric Radeon, RDNA, ROCm, Ryzen, Ryzen AI, Ryzen Threadripper, Xilinx, the Xilinx logo, Alveo, Artix, Kintex, Spartan, Versal, Vitis, Virtex, and Zynq and combinations thereof are trademarks of Advanced Micro Devices, Inc. Microsoft is registered trademark of Microsoft Corporation in the US and other jurisdictions. SPEC®, SPECrate®, SPECint and SPEC CPU® are registered trademarks of the Standard Performance Evaluation Corporation. See www.spec.org for more information. Other product names used in this publication are for identification purposes only and may be trademarks of their respective companies.