COMPUTER
VISION
AFRICA

# LLM Agents From Scratch to Project (Including live demo)

Speakers

**Daniel Etukudo**(Nigeria)

***Host:*** *CEO @ DalensAI*

*Founder Computer Vision Africa*

**Tariq Jamil**(pakistan)

*Founder & Principal ML Engineer @TensorDot*

tensordot
Solutions

**Facilities**

- Interactive Sessions
  Questions and Answers
  Networking Opportunities

**Date: Sat 26th Oct, 2024**

**Time: 5pm WAT, 9PM PKT**

Meeting Link
**https://meet.google.com/qio-vwxq-wxj**

Made with Gamma

## speaker Introduction

Tariq Jamil (https://www.linkedin.com/in/tj-yazman/)

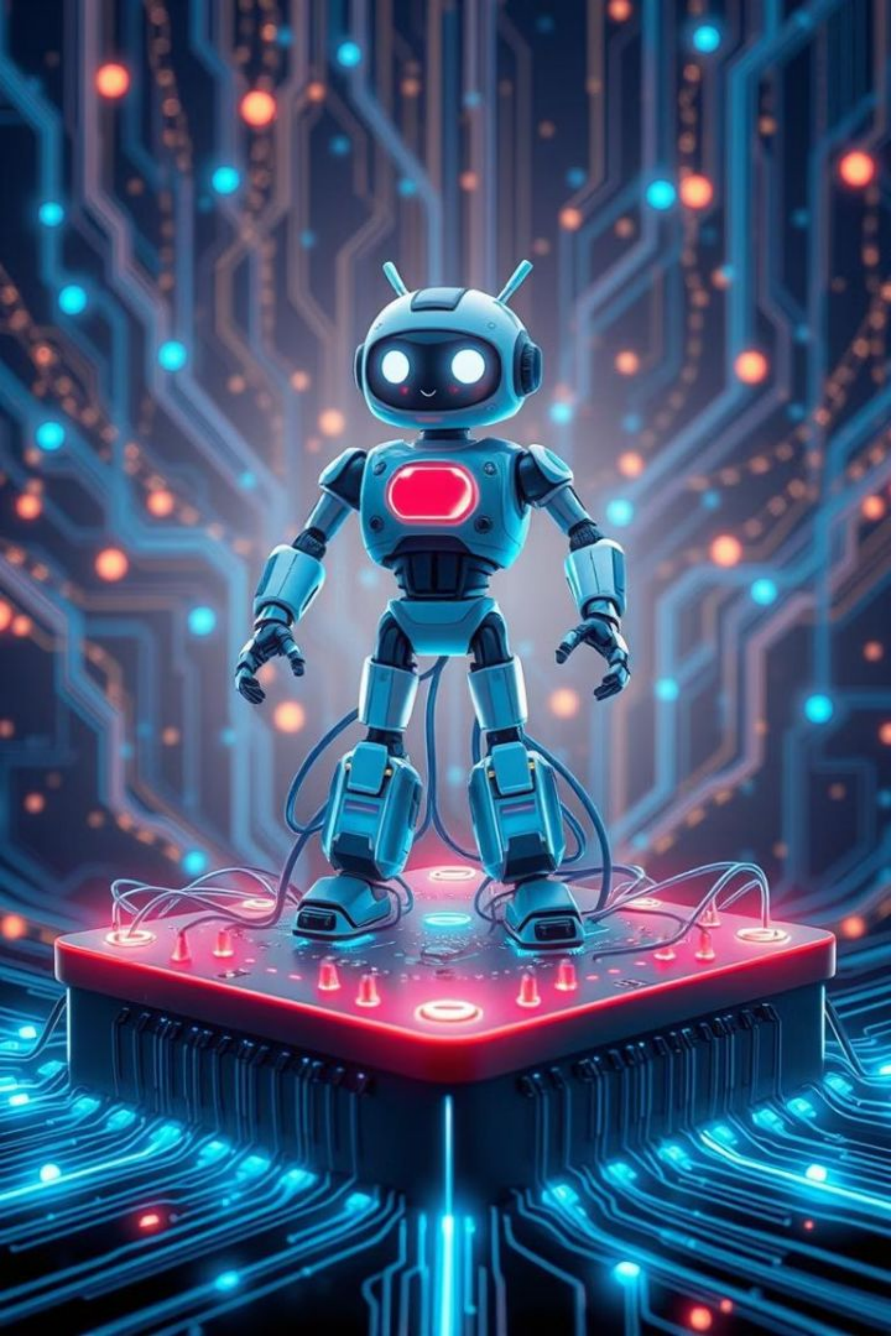
Founder & Principal MLE (TensorDot Solutions)

Education

- BS Engineering (Aviation Electronics) & Post Graduate Diploma (Data Science with AI)

Professional Engagement (Current)

- Research Assistant (Contract): Research Project, Missouri University of Sciences and Technology (USA), Sep 2024 - Jun 2025
- Projects Advisor (Part-Time): Independent Study Projects (PGD students), NED University, Pakistan, Mar 2024 - todate
- Freelance AI / Data Science Projects (Upwork)
- Omdena UAE Chapter Lead



# LLM Agents

## Hands-on from Scratch

LLM Agents are revolutionizing AI, empowering machines with greater autonomy and intelligence. This workshop will dive into their capabilities, applications, and the underlying technologies that make them possible.

by Tariq Jamil

# What are LM Agents?

LLM Agents are esentially a framework comprising an LLM, external tools, and a feedback mechanism that enables the agent autonomously perform tasks within a given environment.

Agents leverag LLMs for to understand, reason, and reflect.

- Reason and Plan: Analyze information, solve problems, and make decisions.
- Take Action: Using Tools and APIs, able to interact with external wprld.
- Reflect: improve performance by going over again on the results adjust to new situations.

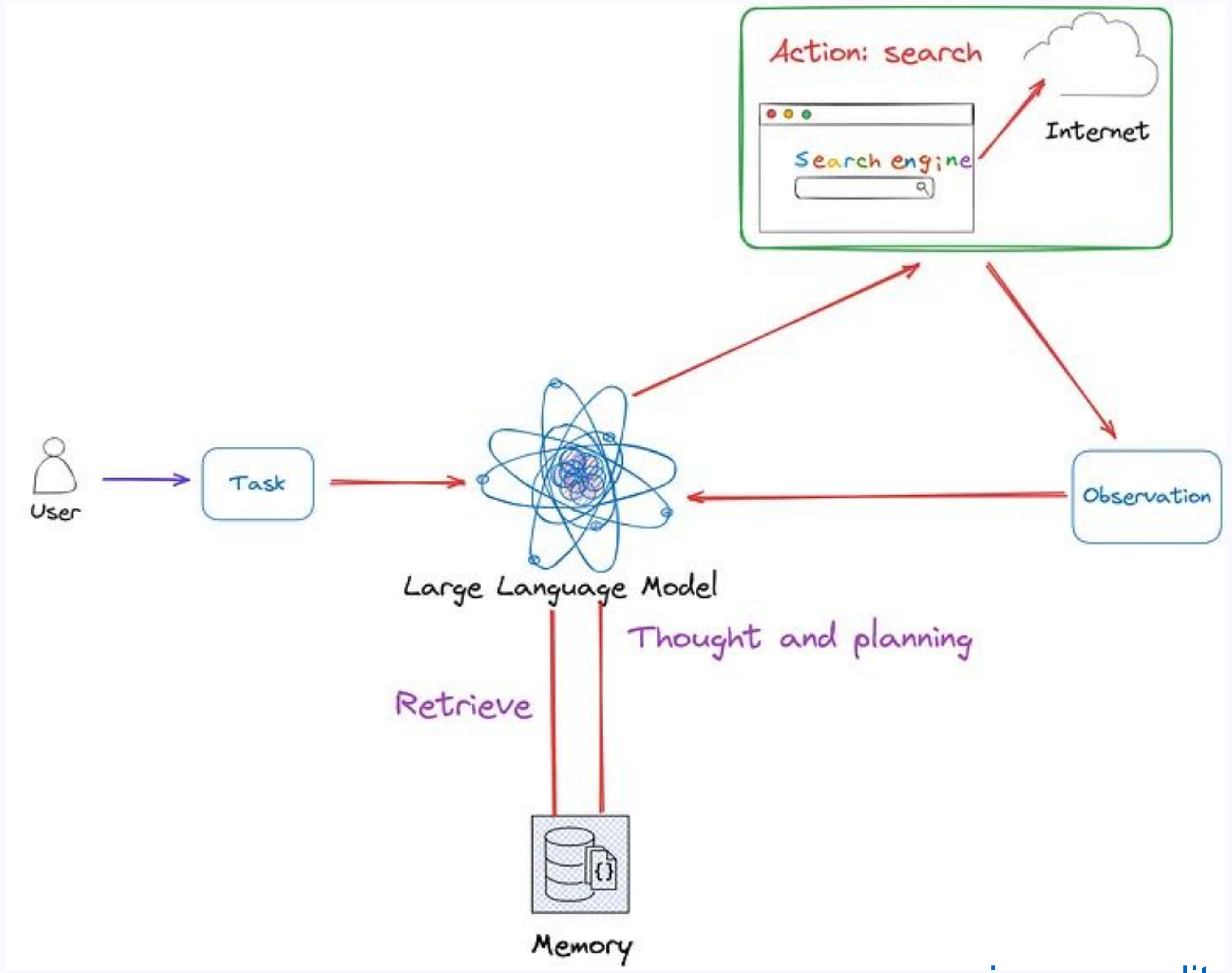


image credits

Made with Gamma

## Why LLM Agents are gaining fame

**because;**

- they overcome the limitations of traditional LLMs
  - infromation cut-off date,
  - no capability to externally interact with environment
- address the need for more intelligent and versatile AI systems.
  - reasoning, and adaptability,
  - many llms are gaing momentum in this area, like Openai o1

## Agents Core Constituents

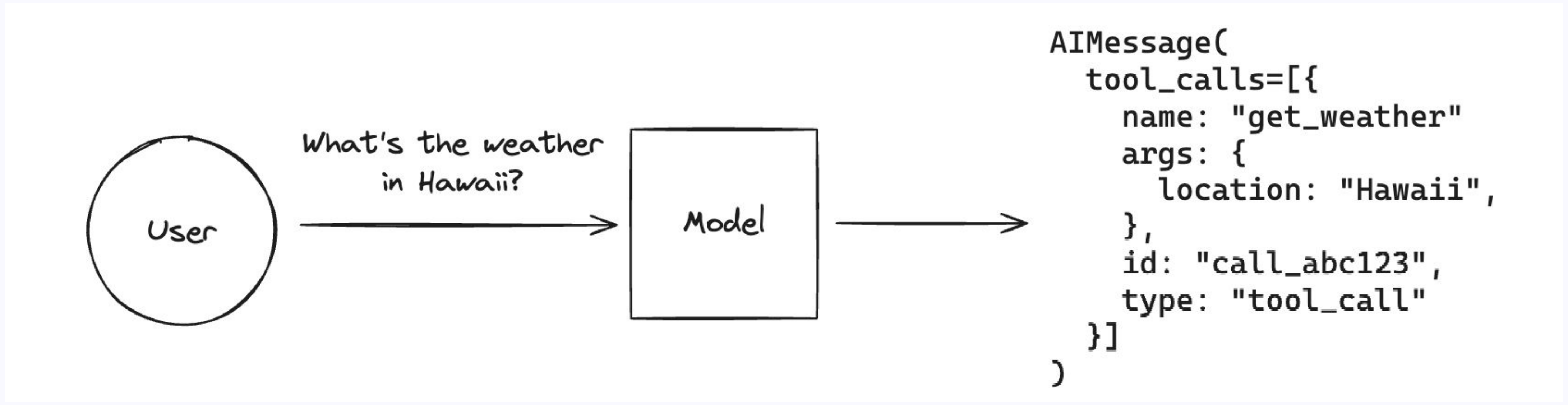
- LLM
- System Prompt
  - Persona [recommended]
  - ReAct & Reflection Instruction [Essential for traditional approach]
  - Tools information [Essential when using tools]
  - CoT Examples [recommended]

- External Tools / API access [well formatted with doc-string]
- Agent State (Messsages Record) [essential]
- Agentic Loop [core coodenator for various parts]

# A Brief on LLMs with Tool call capability

**Most LLMs released today have ReAct capability builtin, i.e.;**

- if given tools information on a prescribed format, they can correlate user query with appropriate tool and return;
  - a tool call structure with tool-name and arguments formatted as JSON.
  - Can give all the tools in one go for a specific query.
- The Agent Construction becomes much easy with minimal errors.
- Problme is solved more quickily and efficiently (less overhead on LLM)
- Popular Open source models supporting tool-call: Llama3.1 onwards, Gemma2 onwards



```
AIMessage(
  tool_calls=[{
    name: "get_weather"
    args: {
      location: "Hawaii",
    },
    id: "call_abc123",
    type: "tool_call"
  }]
)
```

Made with Gamma

## Model Hosting Preferences (personal recommendations)

OpenAI’s clinet-chat-completion has become the de-facto standard and most popular. And most tool calling models (as per my knowledge) support openai’s tools-specifications as input for tool-call output.

Fortunately following frameworks support OpenAI client chat completion compatibility.

- Groq: https://console.groq.com/docs/openai
- Ollama: https://ollama.com/blog/openai-compatibility

You can seamlessly integrate these frameworks at most use cases (text only) requiring tool call, to mimic the openai compatibility.

# Major LLM Agents Frameworks

**AutoGen**:
**The oldest platform.** AutoGen shines when it comes to autonomous code generation. Agents can self-correct, re-write, execute, and produce impressive code, especially when it comes to solving programming challenges. [Going through transformation to version 0.4, now development preview]

**crewAI**:
If you're looking to get started quickly, CrewAI is probably the easiest. Great documentation, tons of examples, and a solid community. Abstraction is challenging

**LangGraph**:
LangGraph, to me, offers more control and I feel that it's best suited for more complicated workflows, especially if you need RAG, or are juggling multiple tools and scenarios.

**OpenAI Agent Swarm**:
OpenAI just released Swarm a few days an  they've said, it's experimental. It's the simplest, cleanest, and most lightweight of the bunch. it's more for prototyping. Things could change quickly, though, since this space moves fast.

**Haystack Agents**:
Its similar to Langgraph -- is graph theory based -- neat interface - and is the company is in the business since 2019.

**Phidata**:
I am experimenting with it and looks quite decent to work. Sometimes challenging due errors but easier to fix as github code is quute easy to follow.

## Future Workshop

Developing an Agentic Application based upon our own framework.

## Learning Resources

- Functions, Tools and Agents with LangChain
- Multi AI Agent Systems with crewAI
- AI Agentic Design Patterns with AutoGen
- AI Agents in LangGraph [*]
- Introduction to LangGraph [*]

# Let’s Code Along

Github Repo:
https://github.com/tariqjamil-bwp/Workshop-Agents-CV-Africa-26Oct2024.git