KubeCon | CloudNativeCon
North America 2024

CLOUD NATIVE & KUBERNETES
AI DAY
NORTH AMERICA

LLM Powered Agents with Kubernetes

Hema Veeradhi
Shrey Anand

Red Hat
Emerging Technologies (OCTO)

Hi! 👋

Hema Veeradhi
Principal Data Scientist
Emerging Technology | Red Hat

California, United States

hemaveeradhi

Shrey Anand
Principal Data Scientist
Emerging Technology | Red Hat

Toronto, Canada

shrey-anand

Agenda

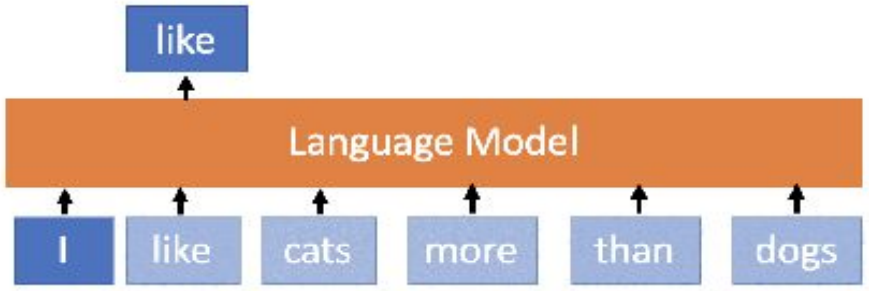
- **Introduction to LLMs**
- **Introduction to RAG**
- **What are Agents**
    - Components of LLM agents
- **Types of Agents**
    - Router based
    - ReAct
- **Agent Use Case**
- **Demo**
- **Concluding Remarks**

Introduction to LLMs

Large Language Models are **computational models that are capable of modeling and generating human language**. LLMs have the transformative ability to predict the likelihood of word sequences or generate new text based on a given input.

What are LLMs good at?

- Text generation/code generation
- Chatbots and Conversational AI
- Information retrieval
- Sentiment analysis

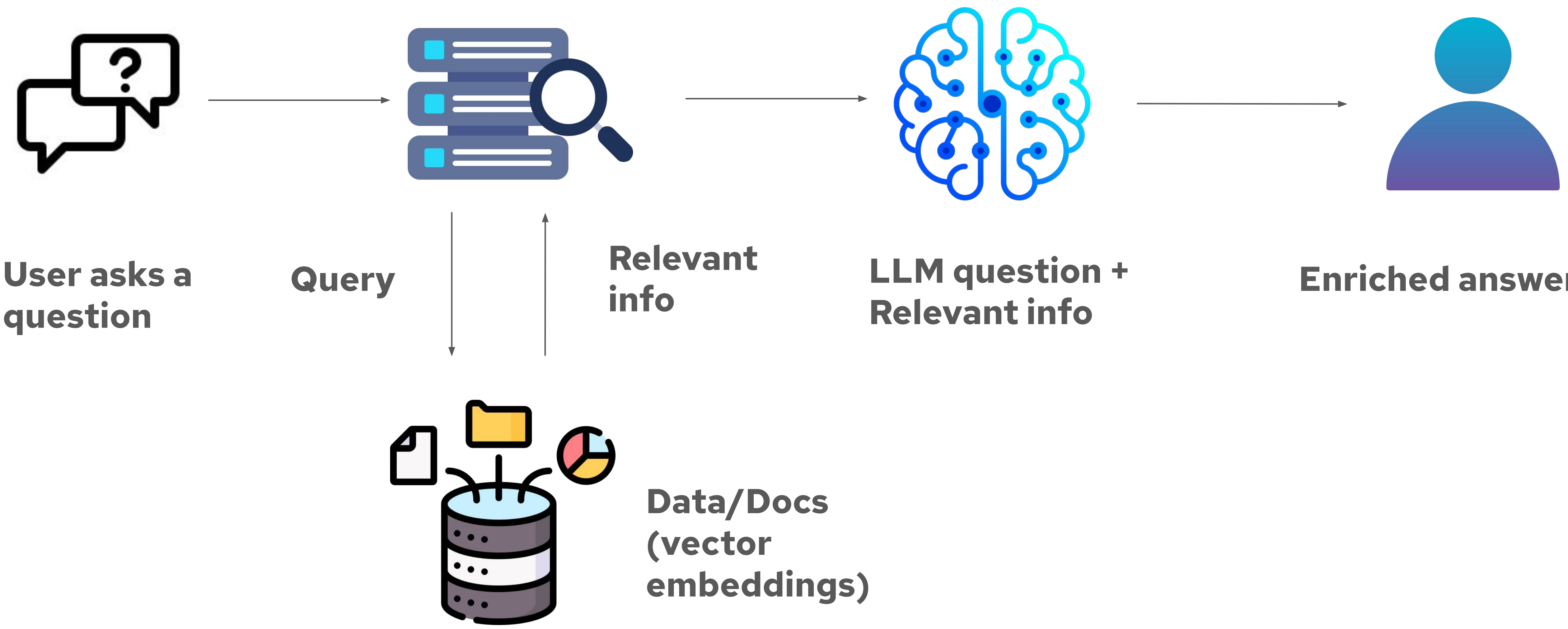


Input: n “tokens” -> I, like, cats, more, than
Output: 1 “token” -> dogs

When can LLMs fail?

- **Complex reasoning tasks** - LLMs have limited reasoning capability, LLMs are good knowledge retrievers but not good reasoners
- **No Dynamicity** - LLMs are static and unable to access real-time information
- **Limited Knowledge (hallucination)** - While trained on vast data, LLMs lack up-to-date world knowledge

RAG - Retrieval Augmented Generation



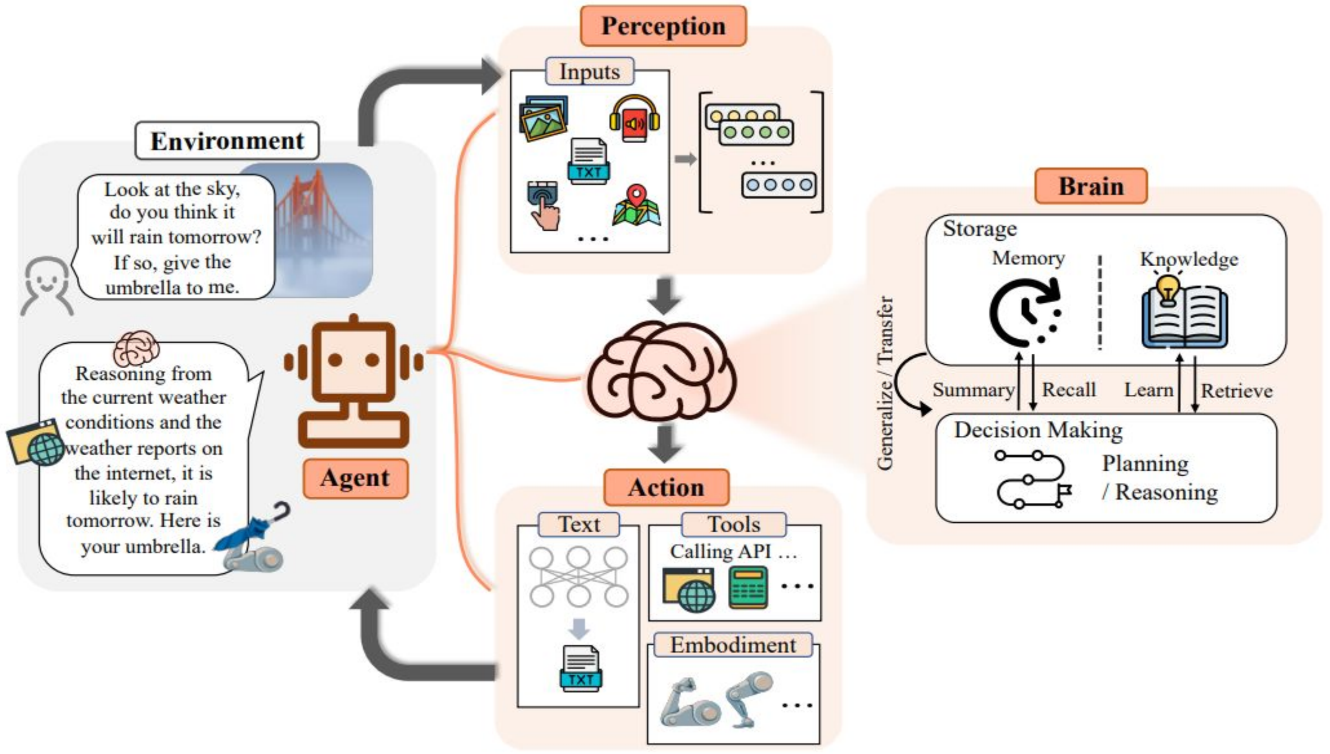
LLM Agents

LLM Agents, also known as Large Language Model agents, **leverage LLMs to execute complex tasks by integrating them with essential components like planning and memory**.

LLM Agent = LLM + Tools + State

- **LLM** - Computational engine i.e "brain"
- **Tools** - Agents ability to interact with the external world
- **Memory/State** - Agent's memory of previous messages and results from used tools

Components of an LLM Agent


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

- **LLM -** Computational engine i.e. "brain"
- **Planning** - Chain of thought process (CoT) to create a plan for executing the tasks
- **Tools** -Executable functions, APIs, other services that complete various tasks
- **Memory** - Short term memory to retain the agent's train of thought and long term memory to retain context/conversation history
- **Actions** - Agents perform actions based on their environment and reasoning, adapting and solving tasks iteratively through feedback.

Types of Agents

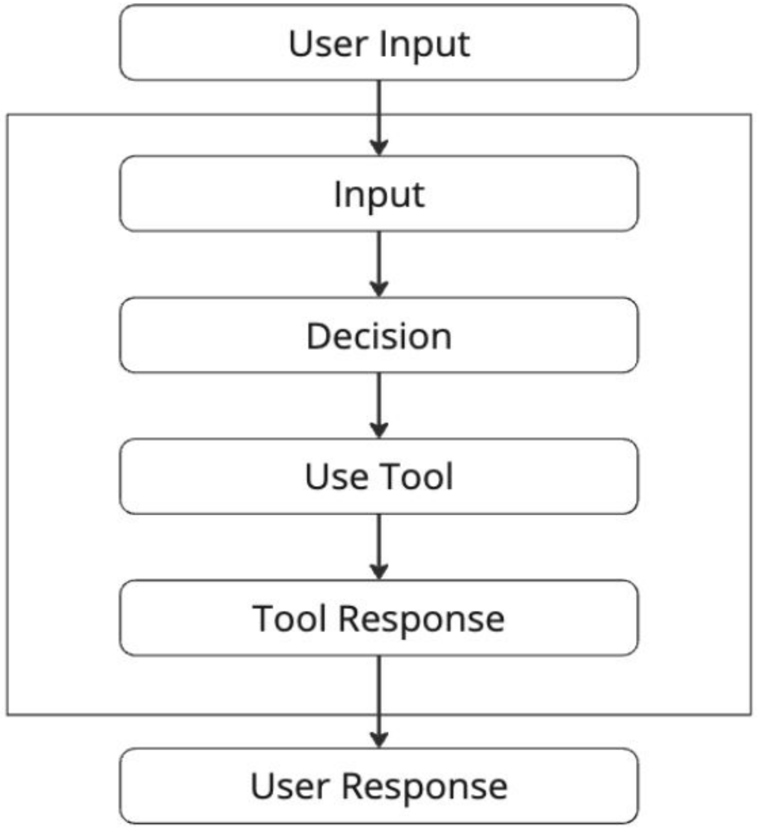
There are many different types of LLM agents to choose from, depending on the nature of your use case, including:

- React Agent
- Rule-based Agent
- Single/Multi Agent

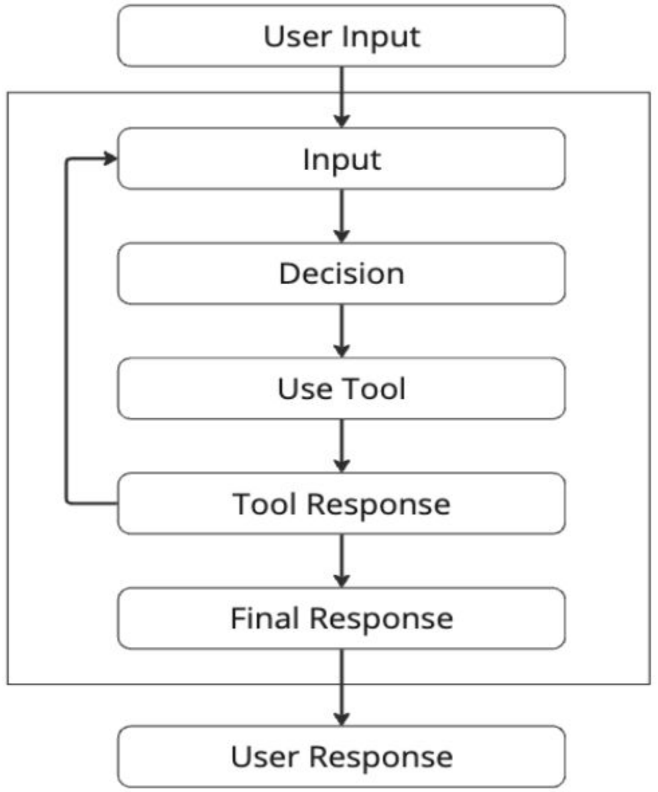
Tools and frameworks to build LLM agents

- LangChain
- Bee Agent
- OpenAgents
- CrewAI
- Arch

Router Agent



ReAct Agent



Use Case

- Imagine a **fictional cloud company CloudForge Dynamics that has several departments and is looking to have a unified chat assistant** for its employees to answer queries across these departments

- Let’s say we have access to Products, HR, and Customer accounts documents

- How can we answer questions by gathering/combining information from all of these different sources?
  - A LLM agent can decide which department’s sources should be used for answering the query

KubeCon CloudNativeCon
North America 2024

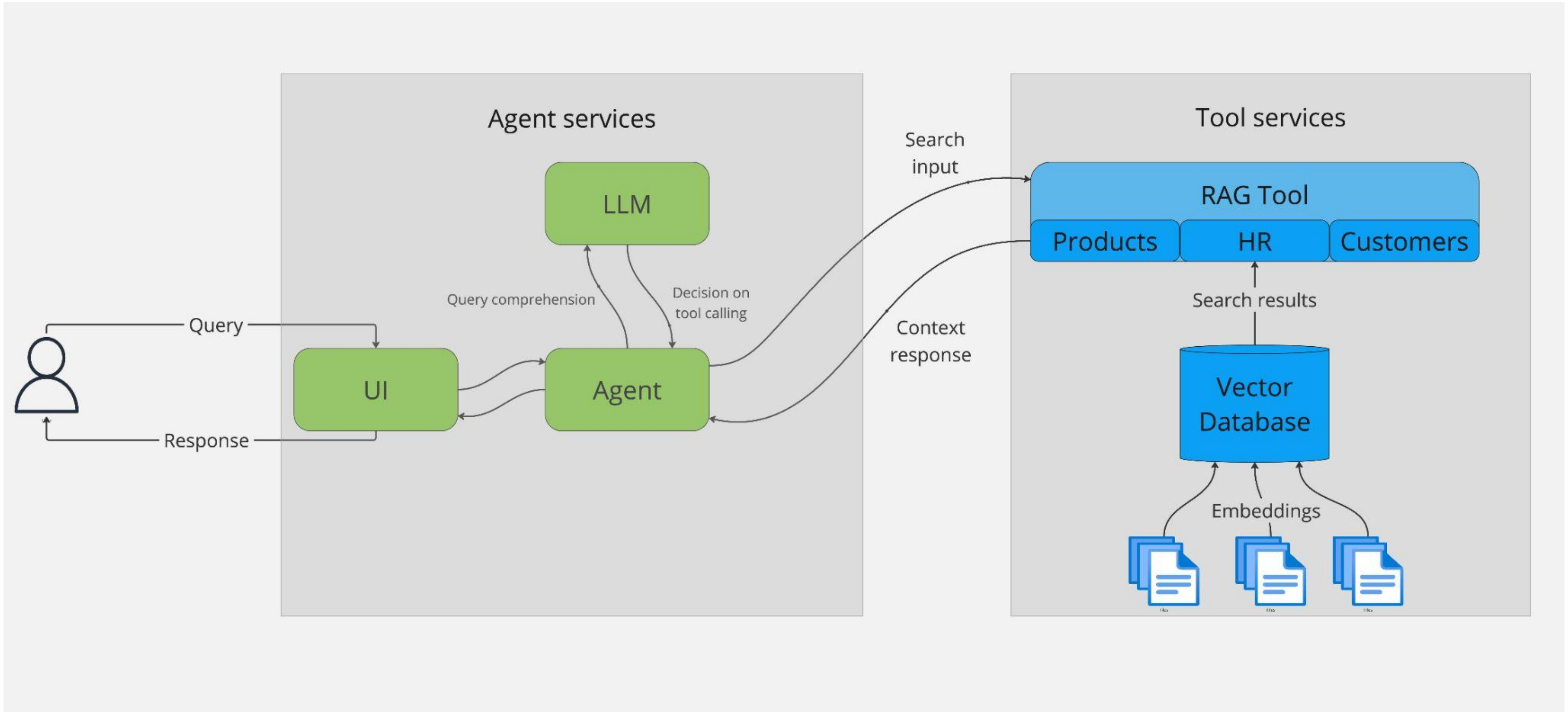
CLOUD NATIVE & KUBERNETES
AI DAY
NORTH AMERICA

DEMO

Example Queries

- What does the CloudForge Migrate product do?
- What HR things should I do before the start date?
- Total Payments Received from FinNova Bank?

Architecture



Resources

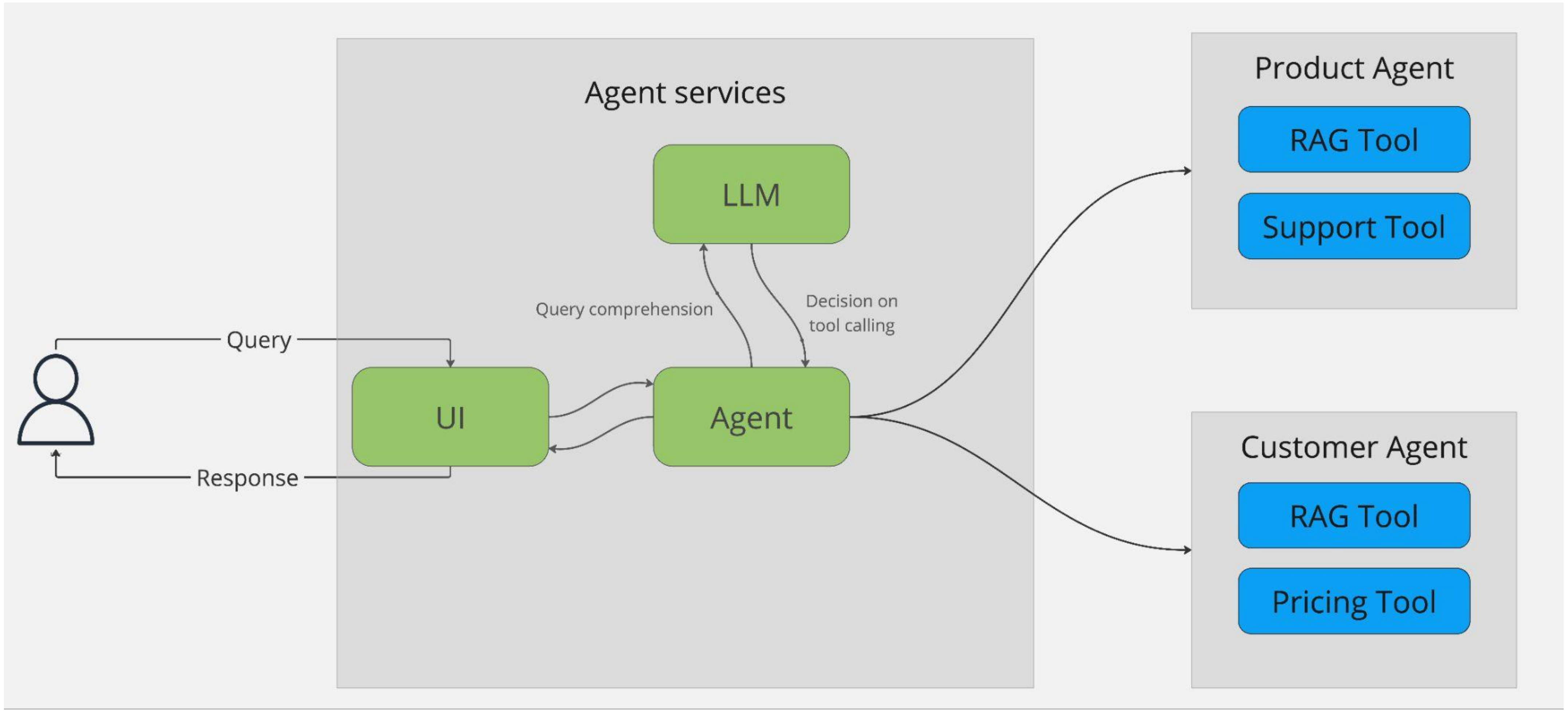
- **GitHub Repository** - https://github.com/redhat-et/llm-agents

- **Agent Services** (UI, Agent, LLM deployment files)
  - **Streamlit UI** - https://github.com/redhat-et/llm-agents/tree/main/streamlit/openshift
  - **ReAct Agent** - https://github.com/redhat-et/llm-agents/tree/main/react_agent/openshift

- **RAG Tool Services**
  - **Milvus vector database** - https://github.com/redhat-et/llm-agents/blob/main/rag/vector_db
  - **RAG service** - https://github.com/redhat-et/llm-agents/blob/main/rag

SCAN ME

Discussion


Agent services
LLM
Query comprehension
Decision on tool calling
Query
UI
Agent
Response
Product Agent
RAG Tool
Support Tool
Customer Agent
RAG Tool
Pricing Tool

Concluding Remarks

- LLMs are awesome, but they need RAG for grounding

- LLMs + RAG pattern is awesome, but it needs Agent's framework for planning and interacting with external sources

- **Challenges with the agents approach**:
  - Debugging is difficult as there can be many fault points such as selection of the right tool and input parameters
  - Latency of response is high if there are many API calls involved in creating the response
  - Load balancing and security concerns for external tools

KubeCon | CloudNativeCon

North America 2024

SCAN ME

Thank You!

Q?

hveeradh@redhat.com
shanand@redhat.com