

From Models to Agents

Running LLM-Powered AI Applications on Amazon EKS Auto Mode

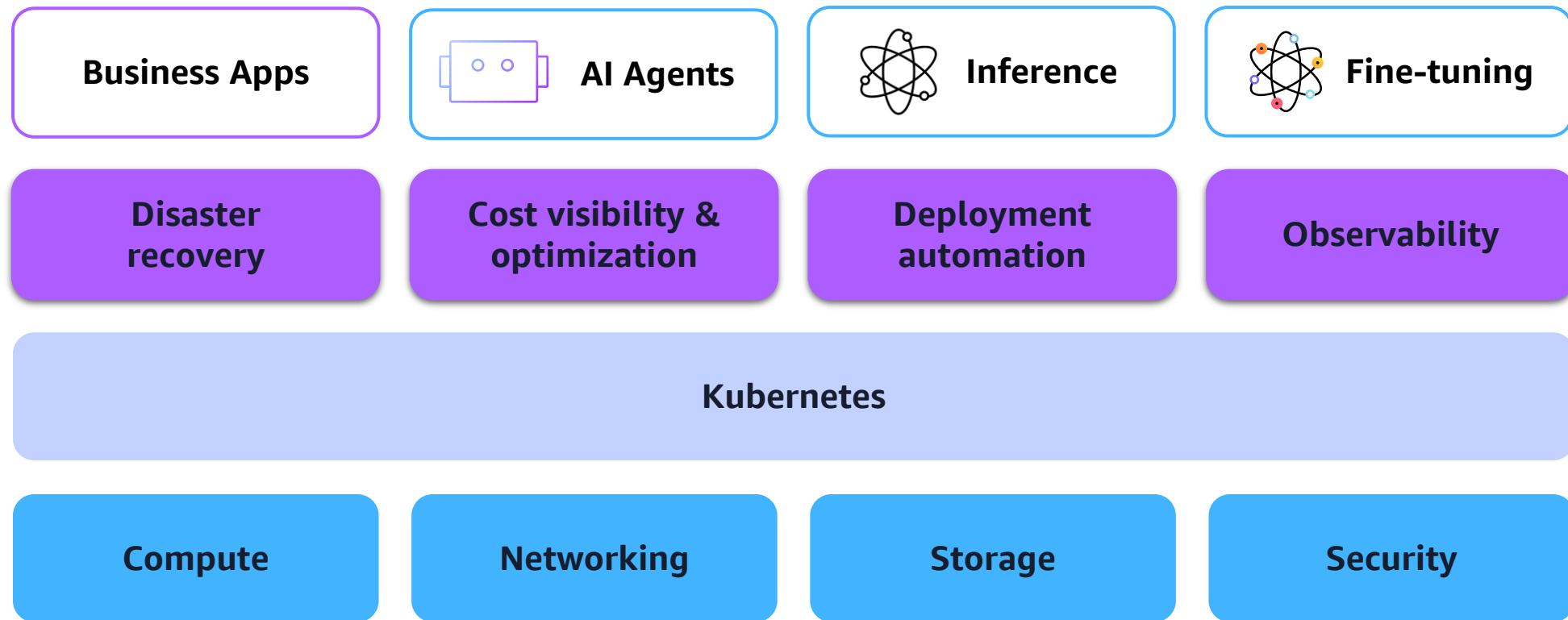
Sai Vennam

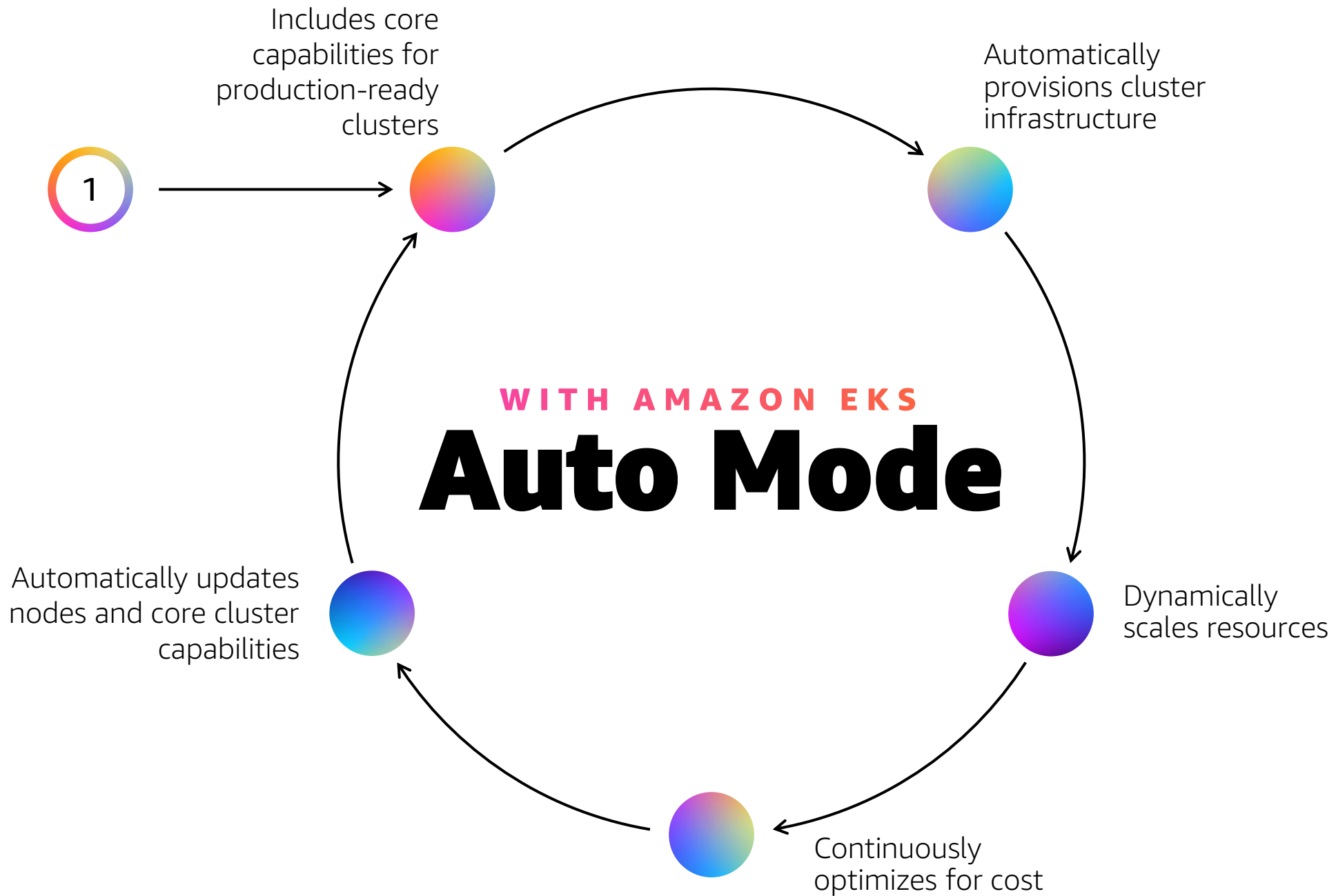
Prin. WW Specialist SA, Containers

AWS



Standardizing on Kubernetes





Enabling AI on EKS Auto Mode

Cluster Add-ons

AWS EFA Kubernetes Device Plugin

inter-node high-throughput networking
distributed inference • training

NVIDIA/Neuron Device Plugin

expose accelerator (GPU or NeuronCores) to a K8s scheduler
exposing hardware features via labels

GPU Metrics with Amazon CloudWatch Agent

collect GPU or Neuron metrics
NVML/DCGM-compatible API • sub-minute resolution • sdd

EBS CSI Driver

k8s interface to EBS service
models and data storage

AMI Configuration

CUDA Kernel Driver

hardware access • memory & other hardware management
can be swapped for NVIDIA GPU operator for flexibility/experiments/dev

Neuron Kernel Driver

hardware access • memory & other hardware management
must be (currently) baked into the AMI

CUDA/NeuronX Runtime

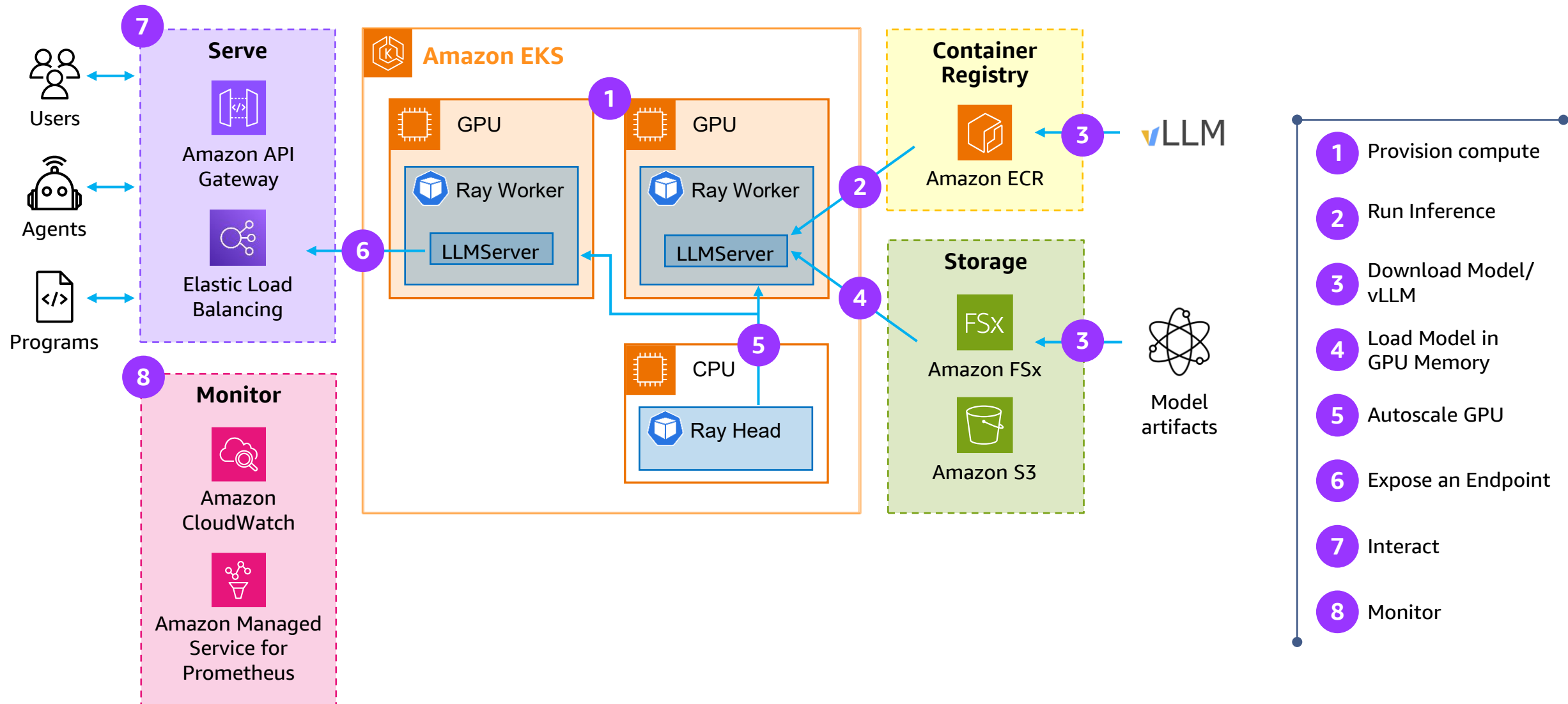
frameworks integration libs: pyTorch, vLLM etc.
resources abstraction • metrics • multi-GPU support

SOCI Containerd Snapshotter Plugin

containerd snapshotter plugin
lazy loading • parallel pull • multi-GPU support

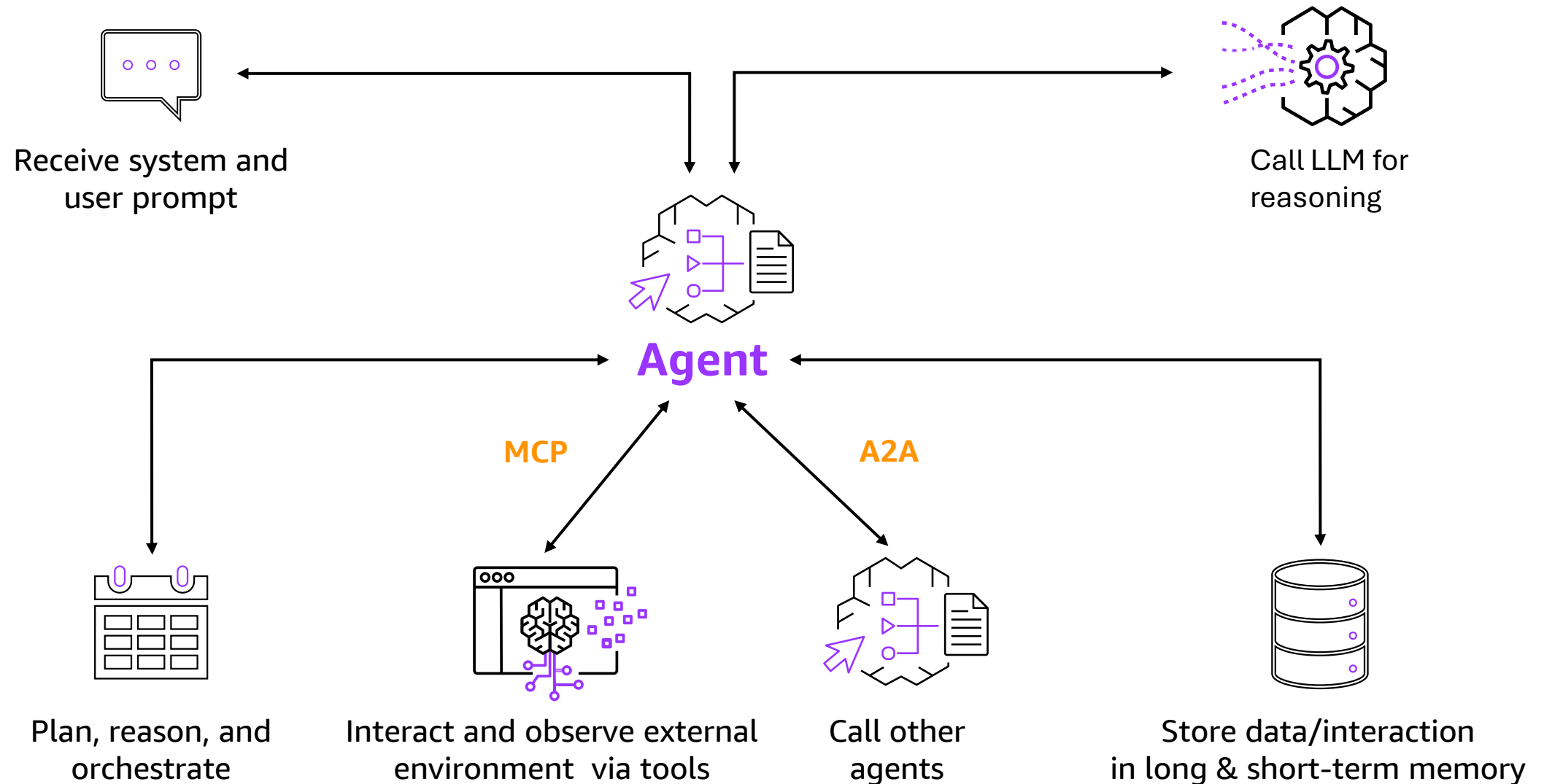
Deploying LLMs for Inference on Amazon EKS Auto Mode

Inference architecture on Amazon EKS Auto Mode



Deploying AI Agents on Amazon EKS Auto Mode

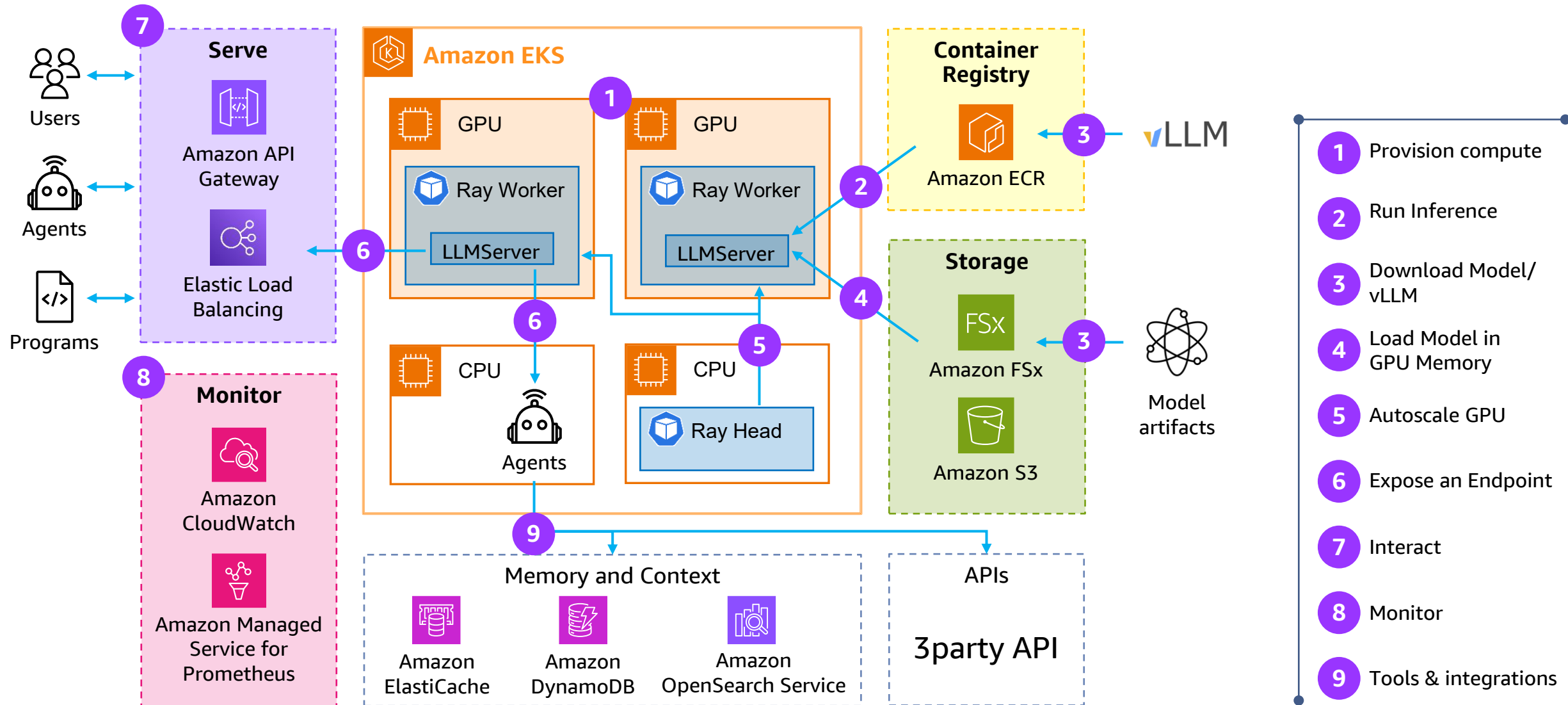
Agents interactions



How to build an Agent with Strands Agents

```
1  from fastapi import FastAPI, Body
2  from strands import Agent
3  from strands.models.openai import OpenAIModel
4  from strands.tools.mcp import MCPClient
5  from openai import OpenAI
6
7  model = OpenAIModel(
8      client_args={"base_url": "http://<vllm-host>:8000/v1", "api_key": "EMPTY"},
9      model_id="<served-model-name>",
10     params={"temperature": 0.7, "top_p": 0.9, "max_tokens": 4096},
11 )
12 weather_mcp = MCPClient(url="http://<weather-mcp-host>:8000/mcp")
13
14 ...
15
16 agent = Agent(model=model, tools=[..., *weather_mcp.list_tools_sync()])
17
18 app = FastAPI(lifespan=lifespan)
19
20 @app.post("/chat")
21 async def chat(message: str = Body(embed=True)):
22     result = await agent.invoke_async(message)
23     return {"response": str(result)}
```

Agents Deployment on Amazon EKS Auto Mode



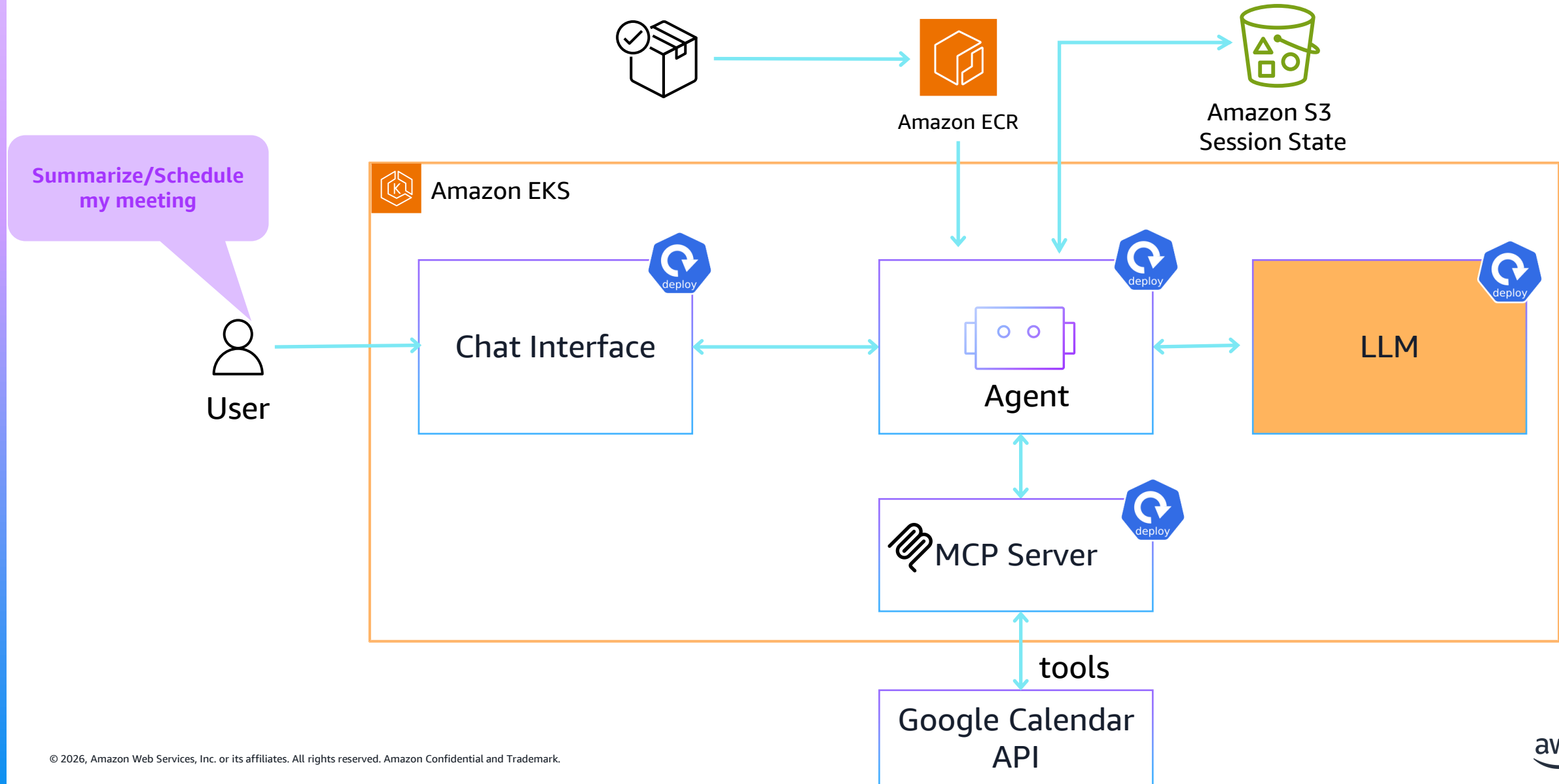
Kiro and MCP Server Integration with Amazon EKS



- Agentic AI: Features agents that work independently to build software, including writing code, running tests, and creating pull requests.
- Provide pre-approved tool permissions, context files and prompts
- Automate workflows in seconds. Analyze errors and trace bugs with precision.

Demo

Demo Architecture



←

→

↺

🏠

us-east-2.console.aws.amazon.com/eks/clusters?region=us-east-2

☆

📄

👤¹⁰

⚙️

☰

aws



Search

[Option+S]



🔔²³

?

⚙️

United States (Ohio) ▾

☰

Amazon Elastic Kubernetes Service

>

Clusters

ⓘ

Amazon Elastic Kubernetes Service

<

Dashboard

Clusters

▼ Settings

Dashboard settings

Console settings

▼ Amazon EKS Anywhere

Enterprise Subscriptions

▼ Related services

Amazon ECR

AWS Batch

Documentation ↗

EKS workshops ↗ [New](#)

Clusters (0) [Info](#)

 [Delete](#) [Create cluster](#) ▾

🔍 Filter clusters

< 1 >

⚙️

Cluster name ▲ | Status ▾ | Kubernetes version ▾ | Support period ▾ | Upgrade policy ▾ | Created ▾ | Provider ▾

No clusters
You do not have any clusters.
[Create cluster](#)

Thank you!

セッションをご視聴いただき有難うございました。
お近くのブーススタッフまで QR コードのご提示をお願いいたします。
Thank you for attending this session. Please show your QR code to the nearest booth staff.

Sai Vennam