KubeCon | CloudNativeCon

North America 2024

Best Practices for Deploying LLM Inference, RAG and Fine Tuning Pipelines in K8s

Shiva Krishna Merla, NVIDIA
Meenakshi Kaushik, NVIDIA

Agenda

- Overview
- Inference and Fine-Tuning
- Looking forward
- Conclusion

Overview

Generative AI Fuels Rise of Sovereign Clouds

Data Sovereignty and Compliance

Tailored AI Infrastructure

Enhanced Security and Privacy

Enterprise Benefits from Customer LLM

Custom Models Provide Insightful Responses

Base Models Generate Generic Responses

Banking
Chatbot

Hi. How can I help you?

How do I apply for a business loan? U

To apply for a business loan, you'll typically need to provide financial statements, a business plan, and personal identification.

Custom Models Provide Business-Specific Answers

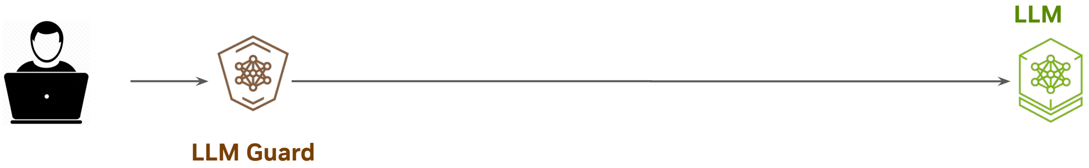
Banking
Chatbot

Hi. How can I help you?

How do I apply for a business loan? U

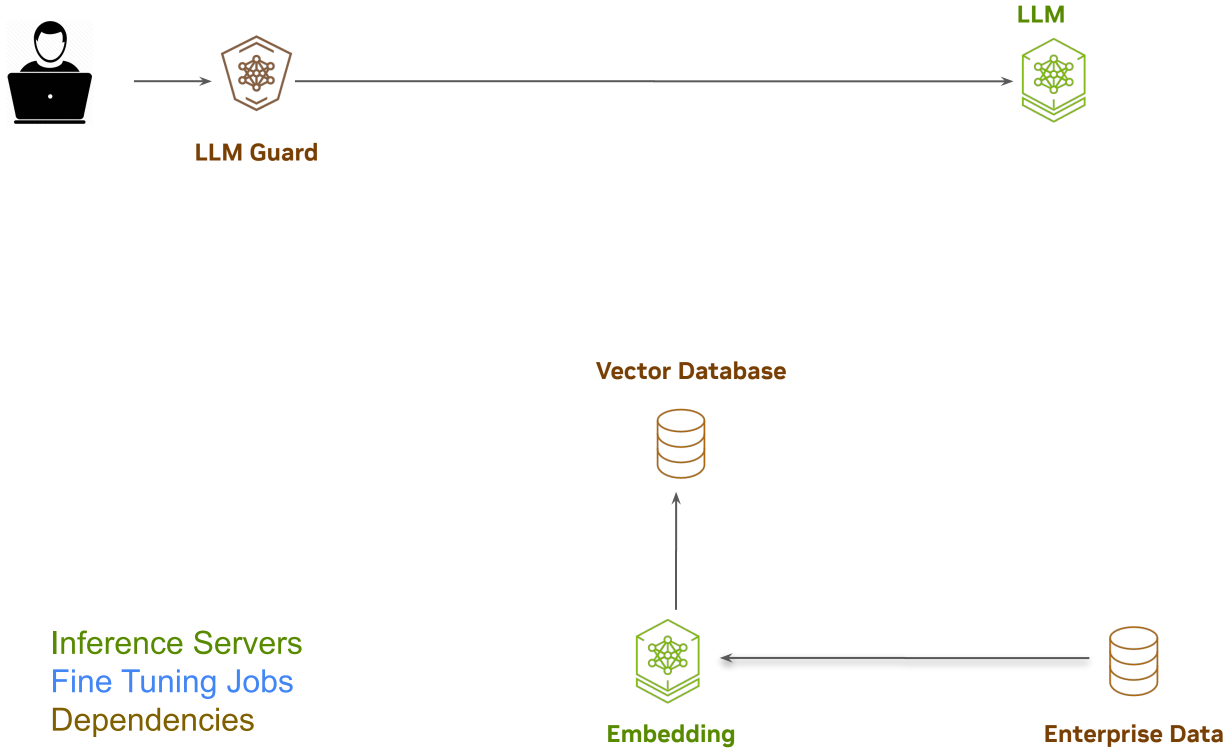
To apply for a business loan, visit our website's Business Banking section and fill out the application form. You'll need two years of financial statements, a business plan, and tax returns, with additional requirements for loans over $500,000.

Inference



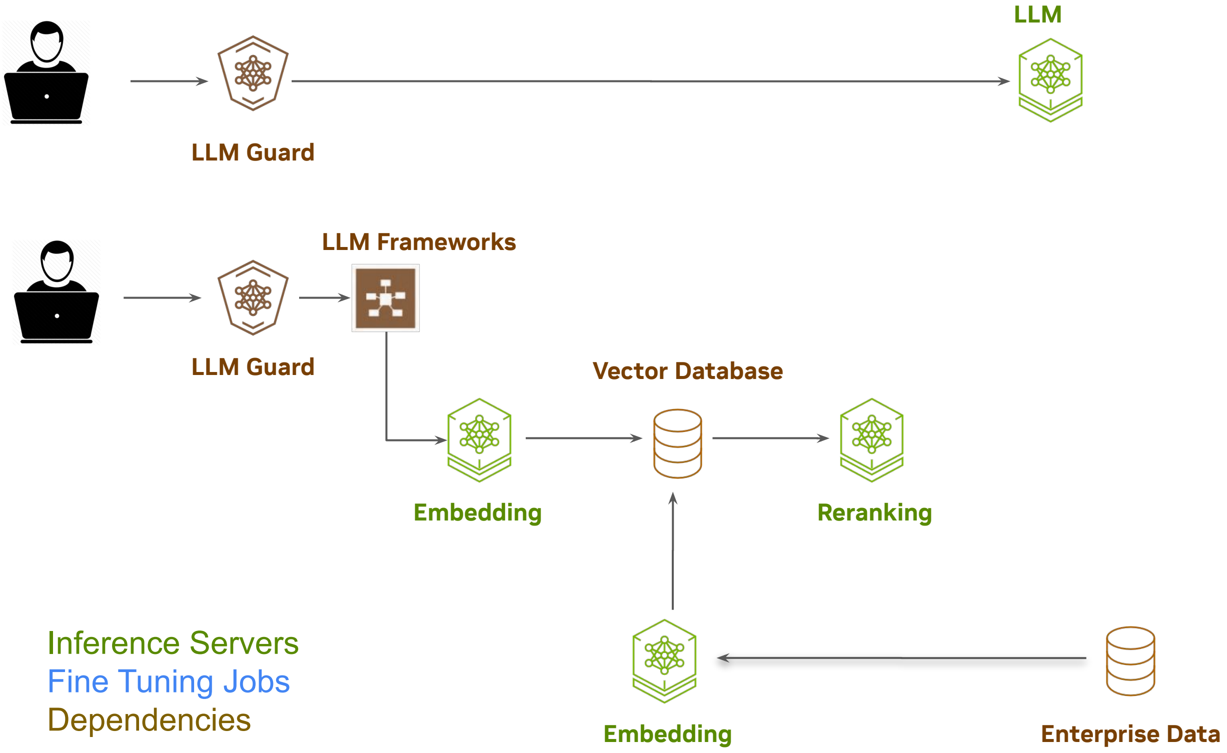
Inference Servers
Fine Tuning Jobs
Dependencies

Inference, RAG



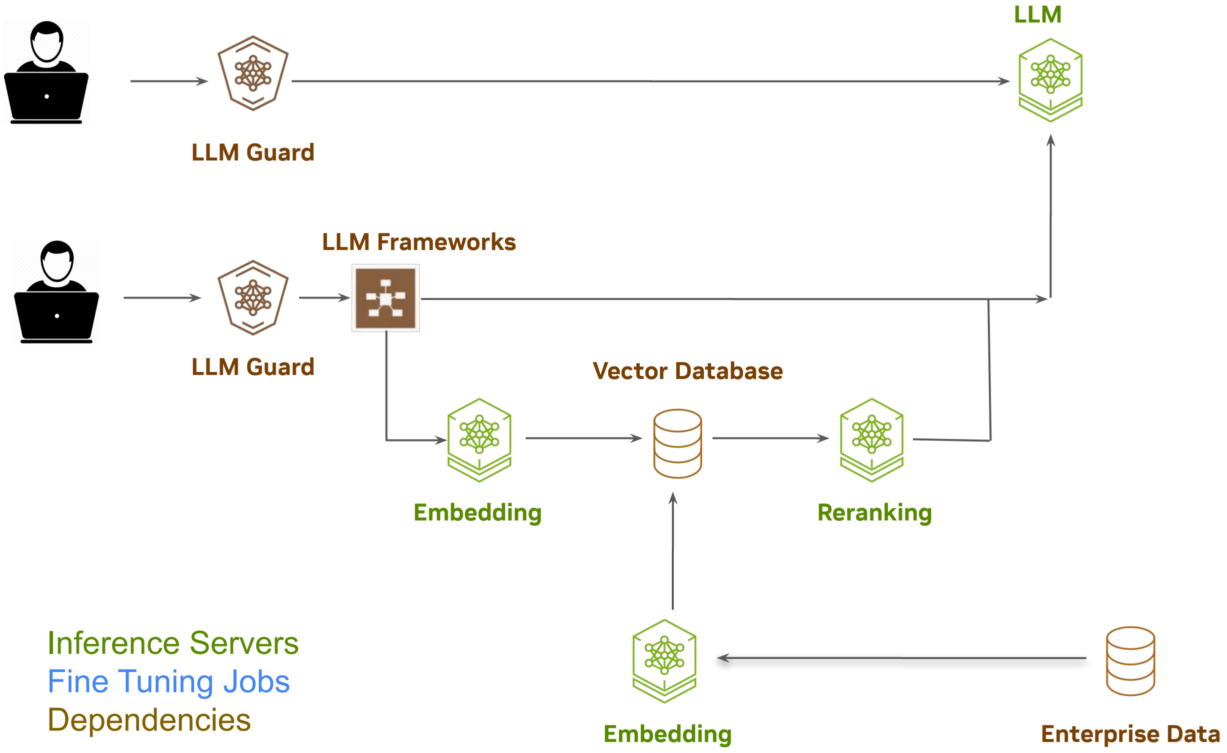
Inference Servers
Fine Tuning Jobs
Dependencies

Inference, RAG



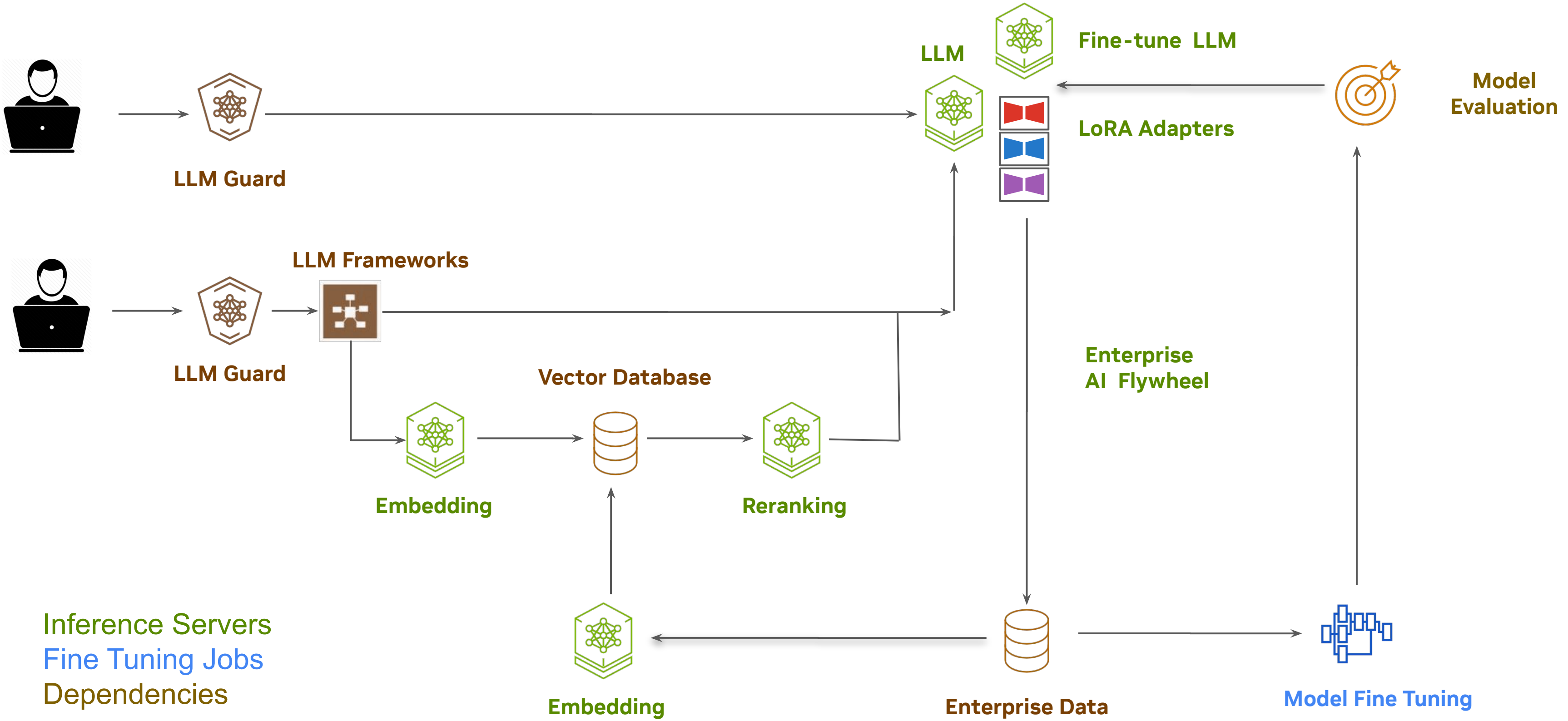
Inference Servers
Fine Tuning Jobs
Dependencies

Inference, RAG



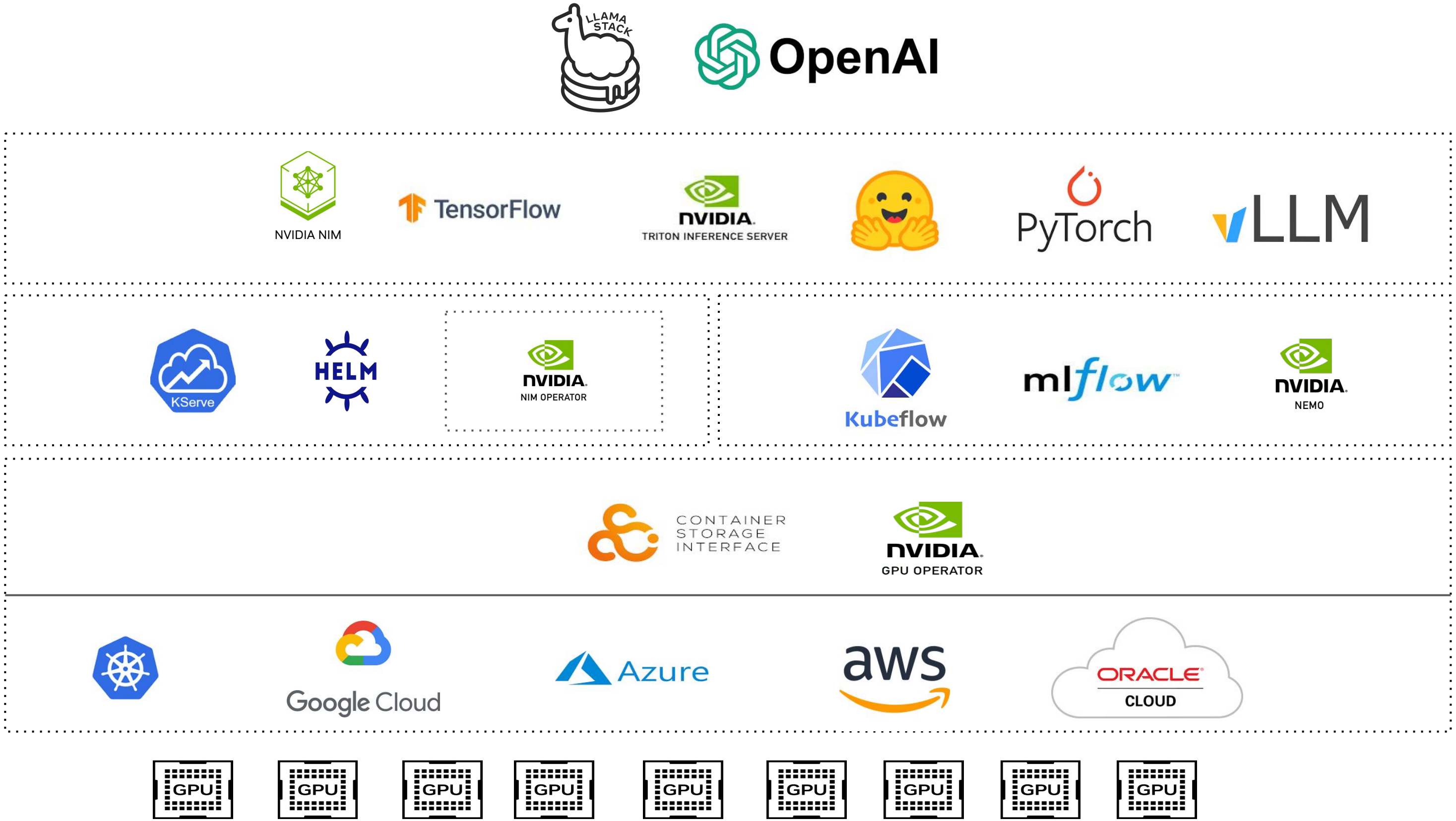
Inference Servers
Fine Tuning Jobs
Dependencies

Inference, RAG and Fine Tuning Pipelines



Inference Servers
Fine Tuning Jobs
Dependencies

Current Landscape



APIs

Inference Servers

Open-Source Inference Platforms

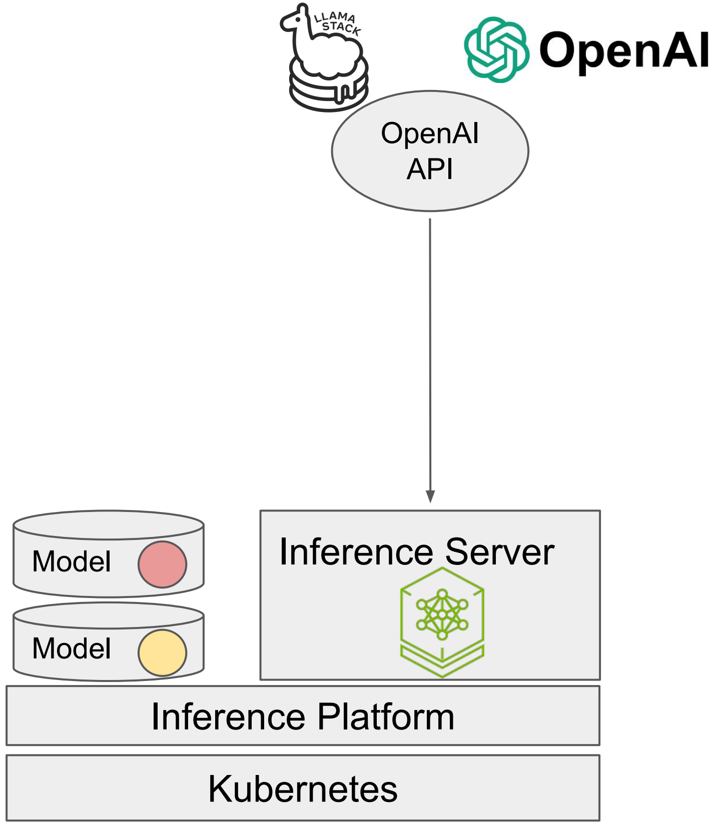
Open-Source Training Platforms

infrastructure

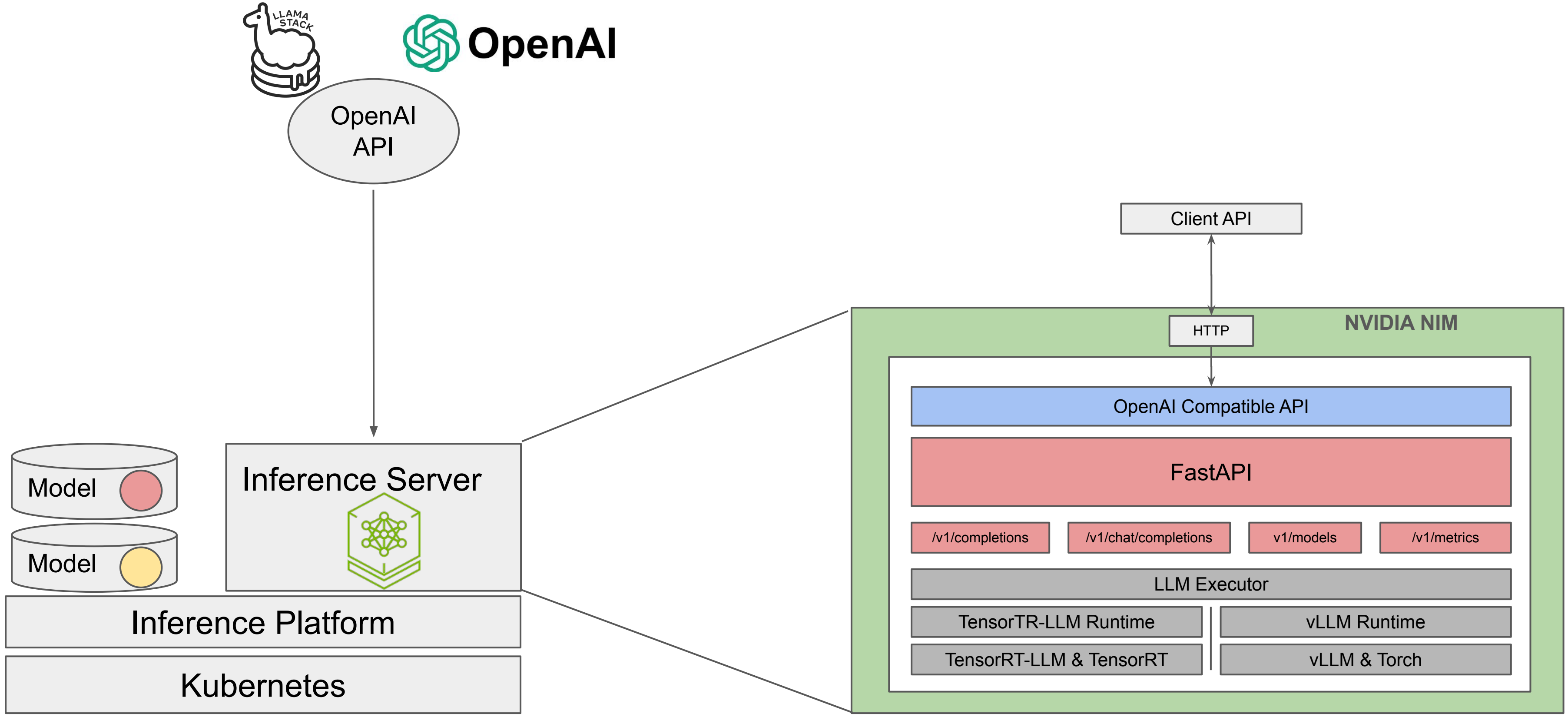
Hardware

Inference and Fine Tuning

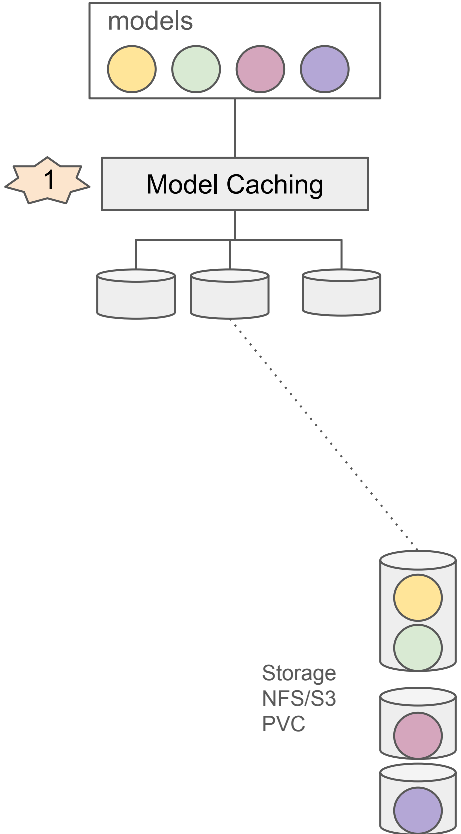
An Example Inference Server



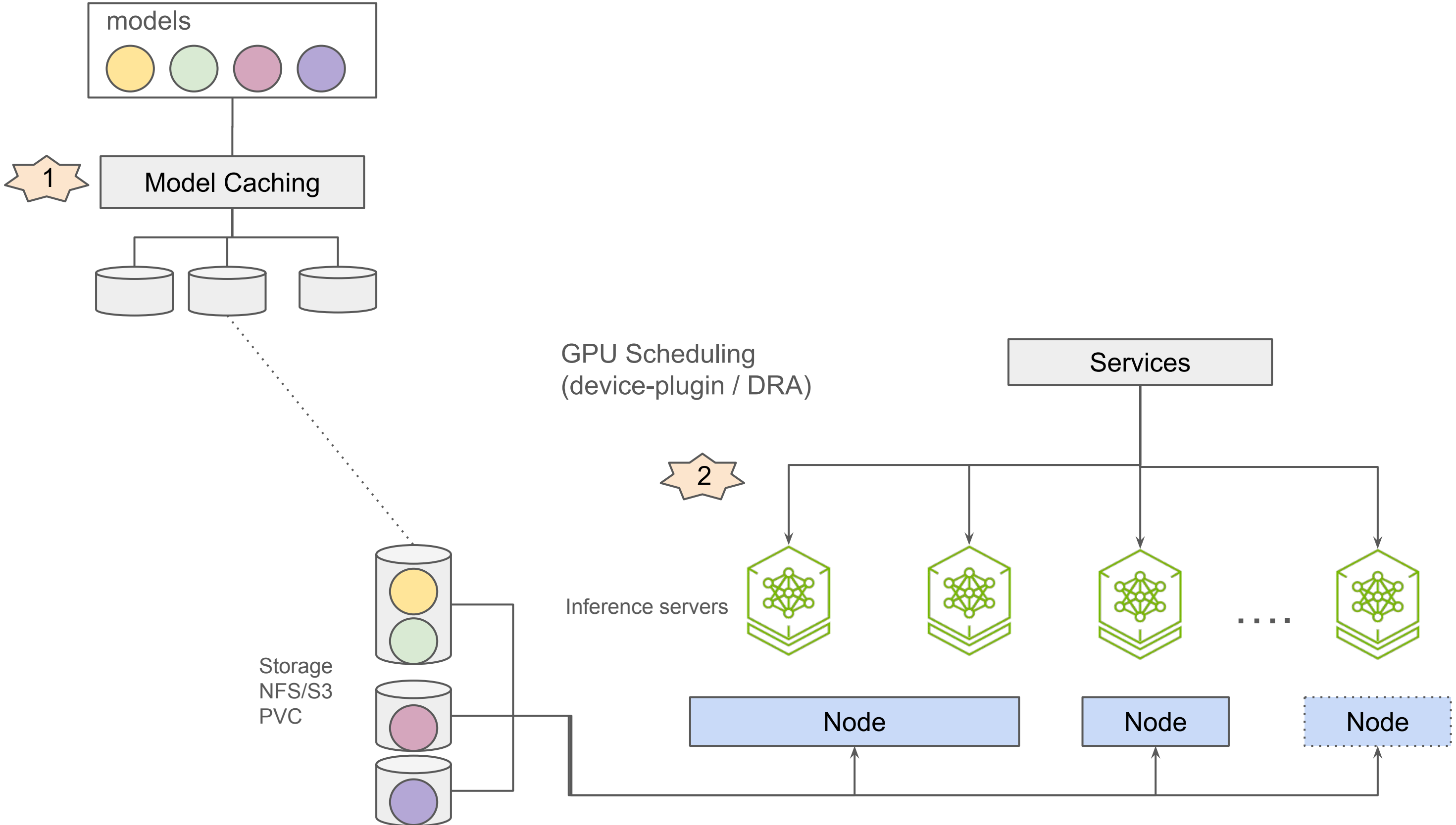
An Example Inference Server



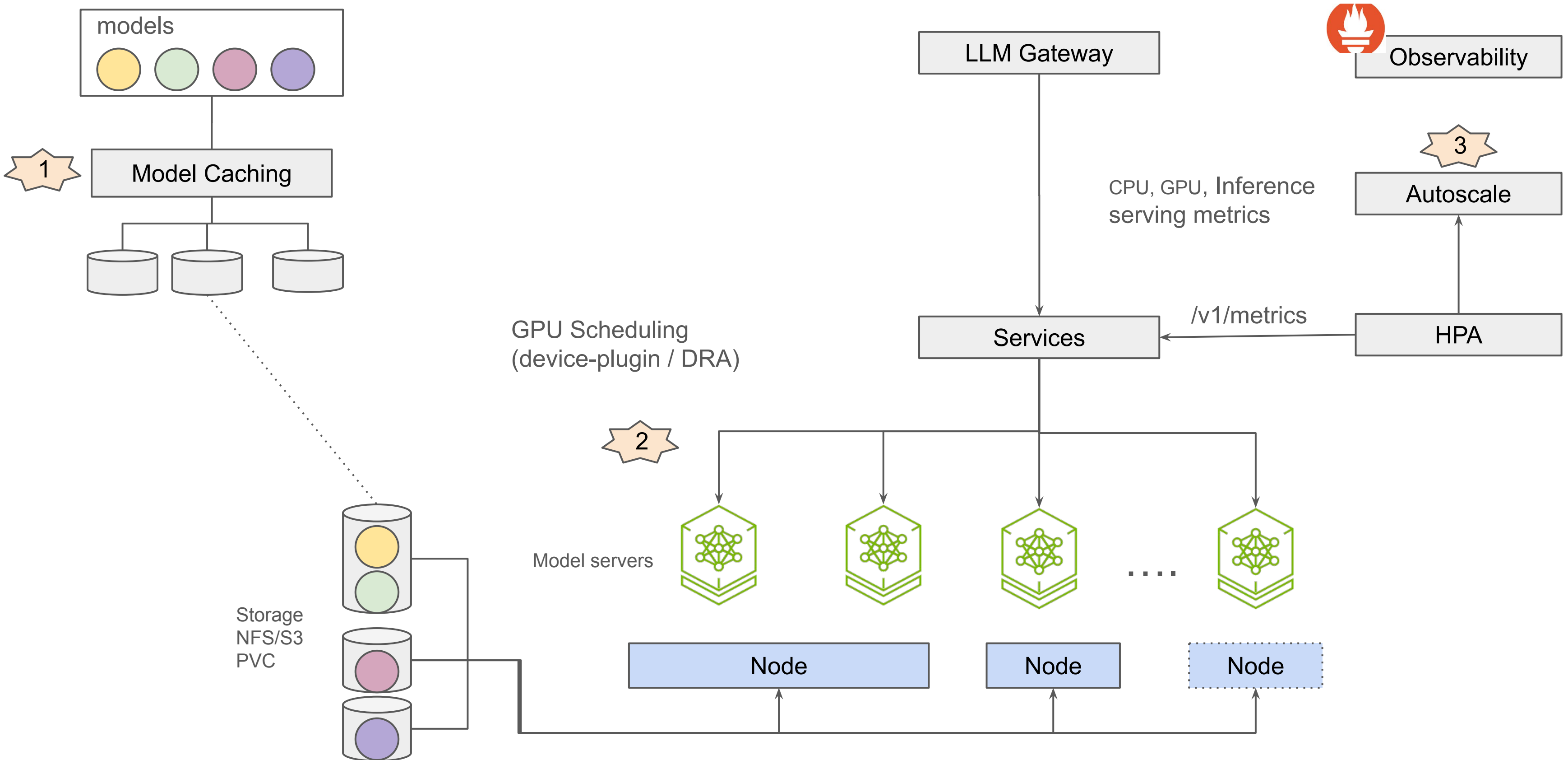
Typical Model Serving Pipeline



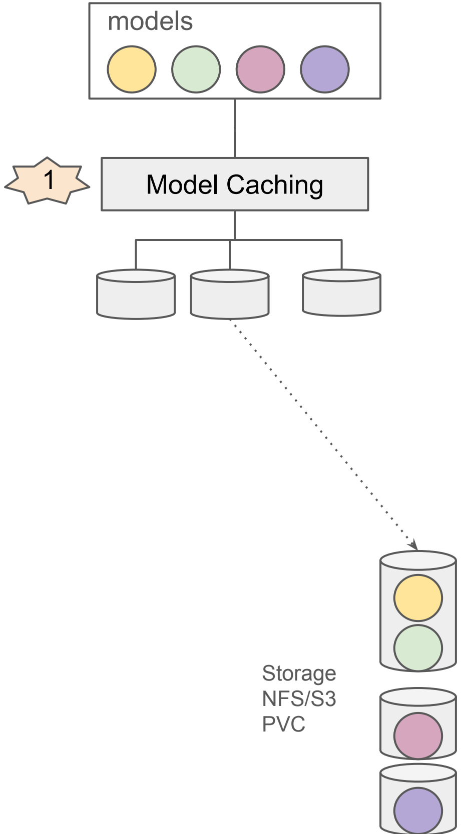
Typical Model Serving Pipeline



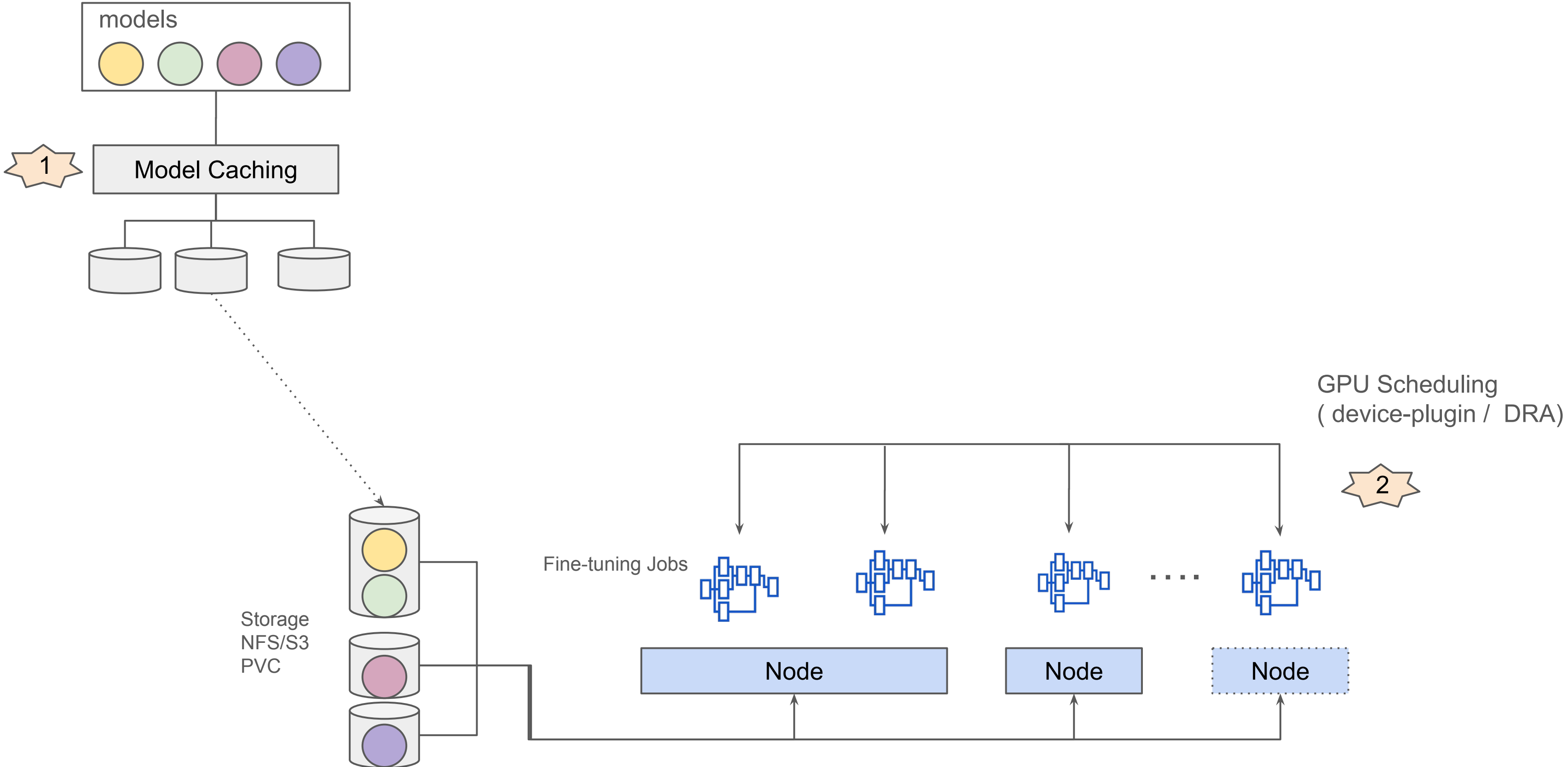
Typical Model Serving Pipeline



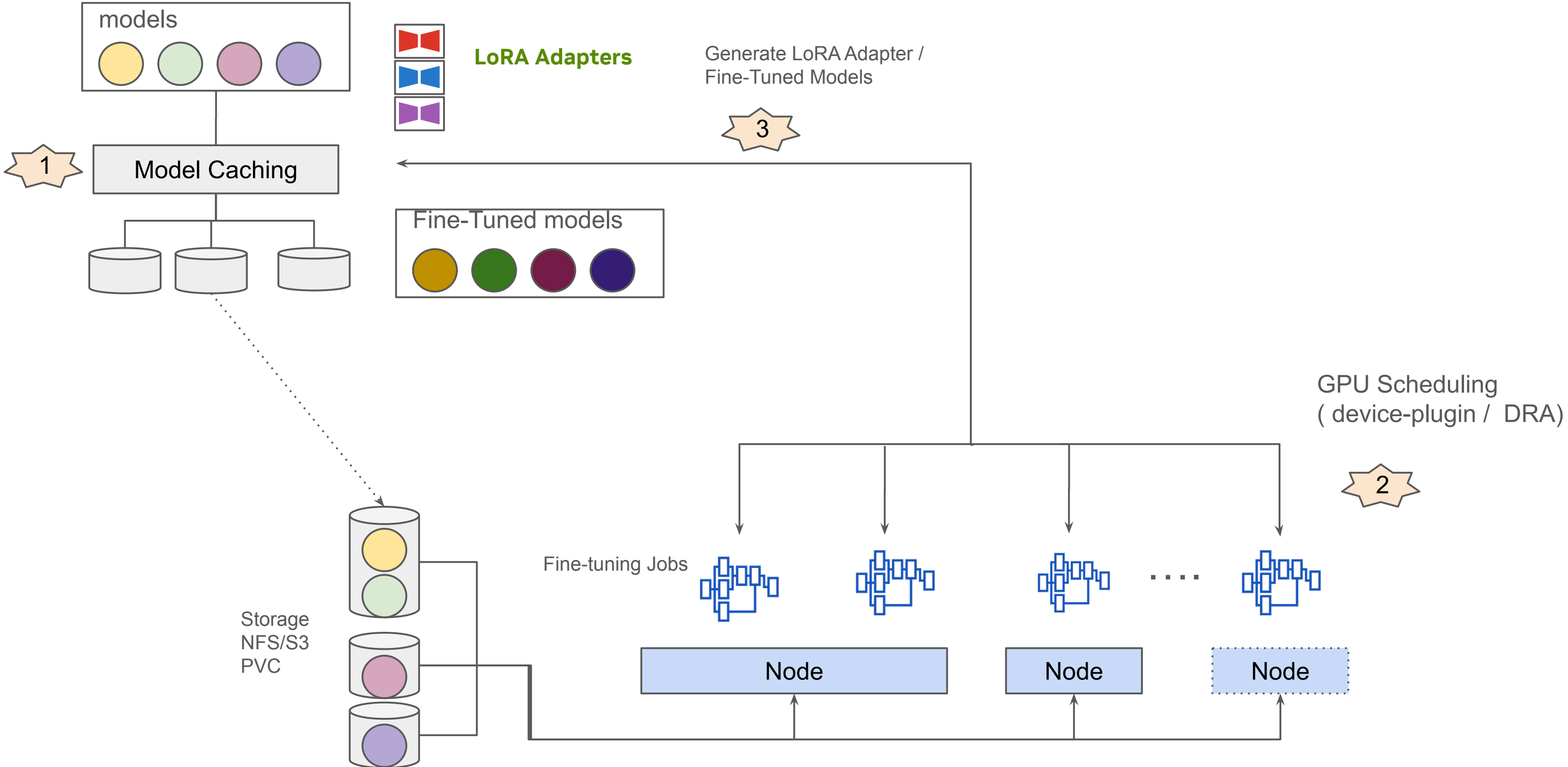
Typical Fine Tuning Pipeline



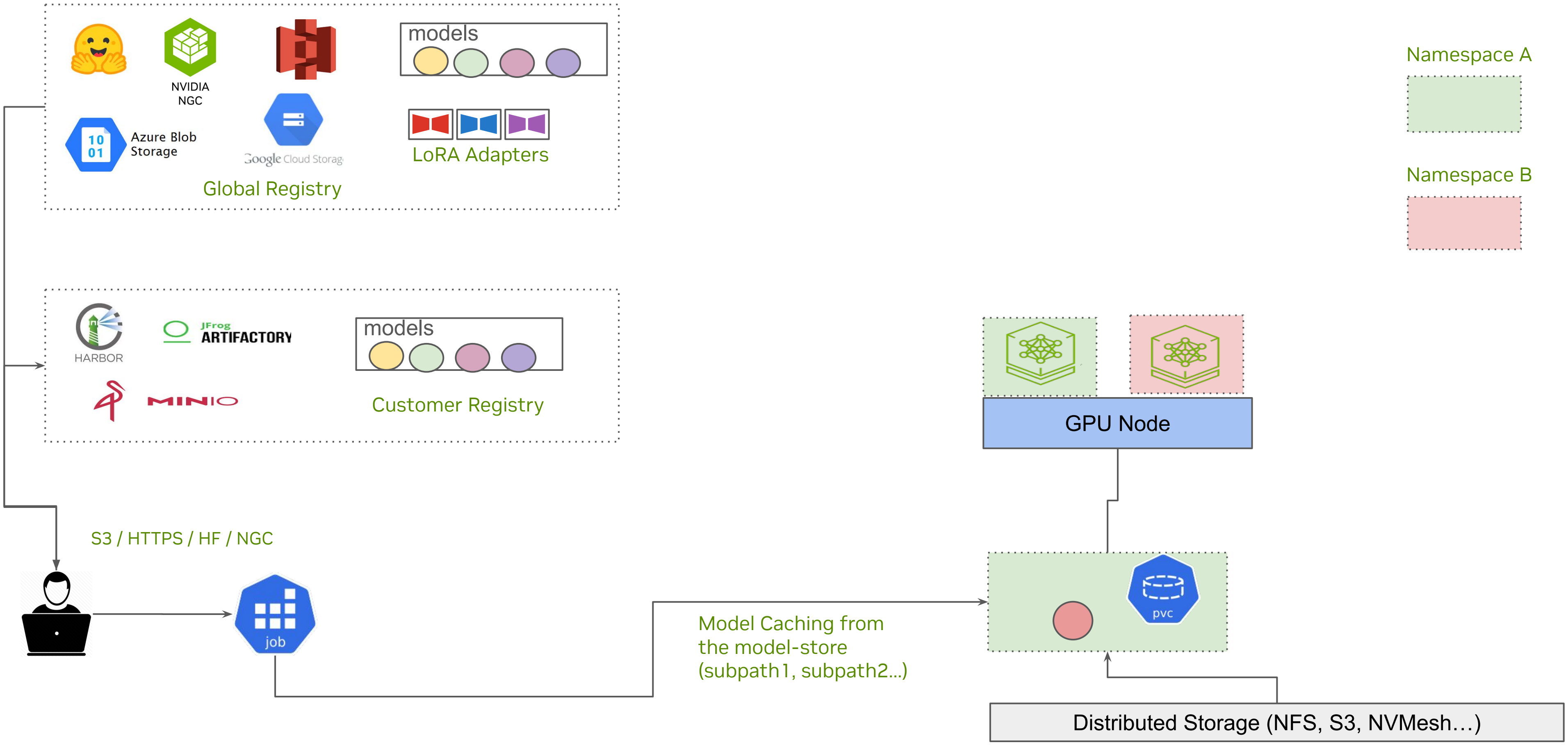
Typical Fine Tuning Pipeline



Typical Fine Tuning Pipeline



Model Management



NVIDIA NGC

Azure Blob Storage

Google Cloud Storage

models

LoRA Adapters

Global Registry

Namespace A

Namespace B

HARBOR

JFrog ARTIFACTORY

MINIO

models

Customer Registry

GPU Node

S3 / HTTPS / HF / NGC

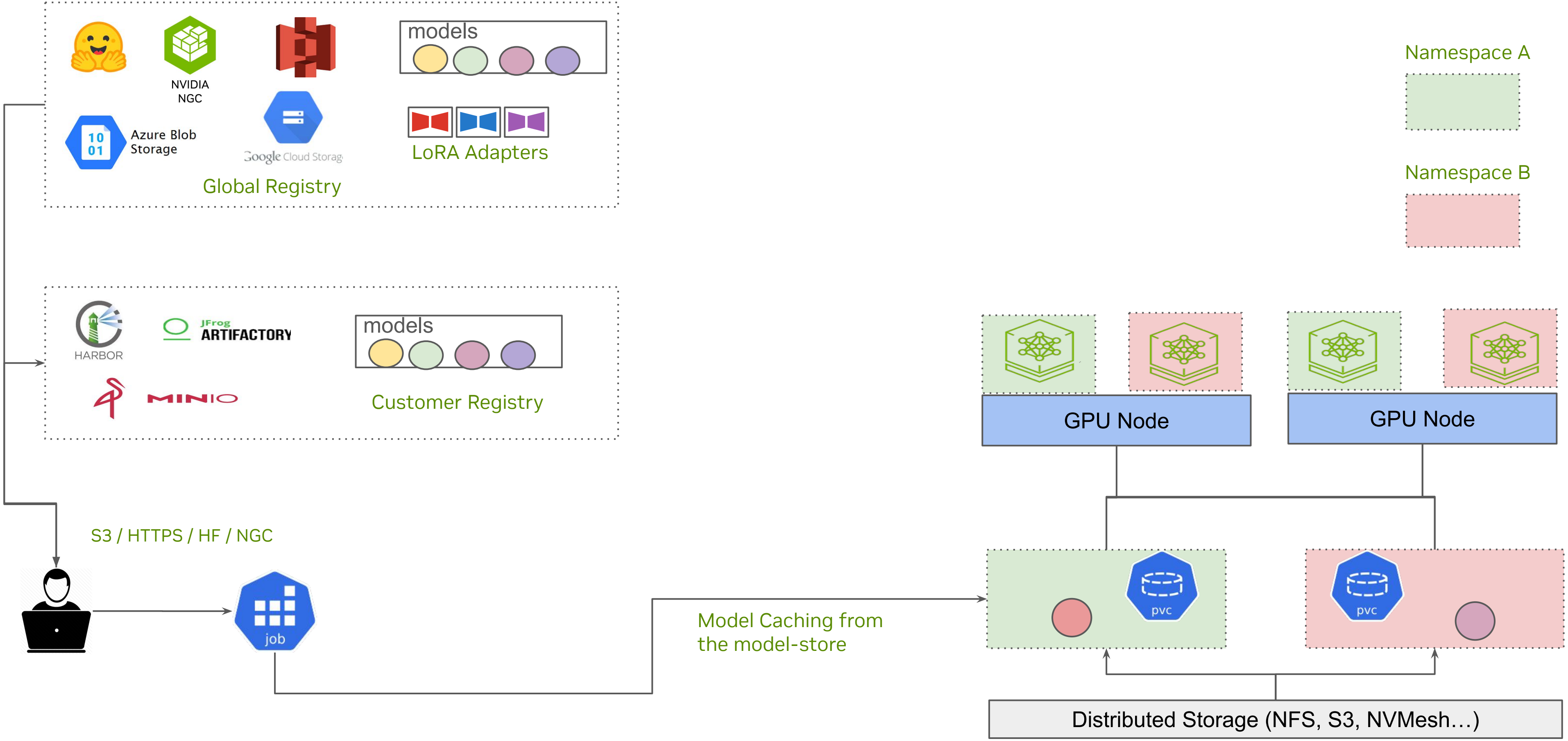
job

Model Caching from the model-store (subpath1, subpath2...)

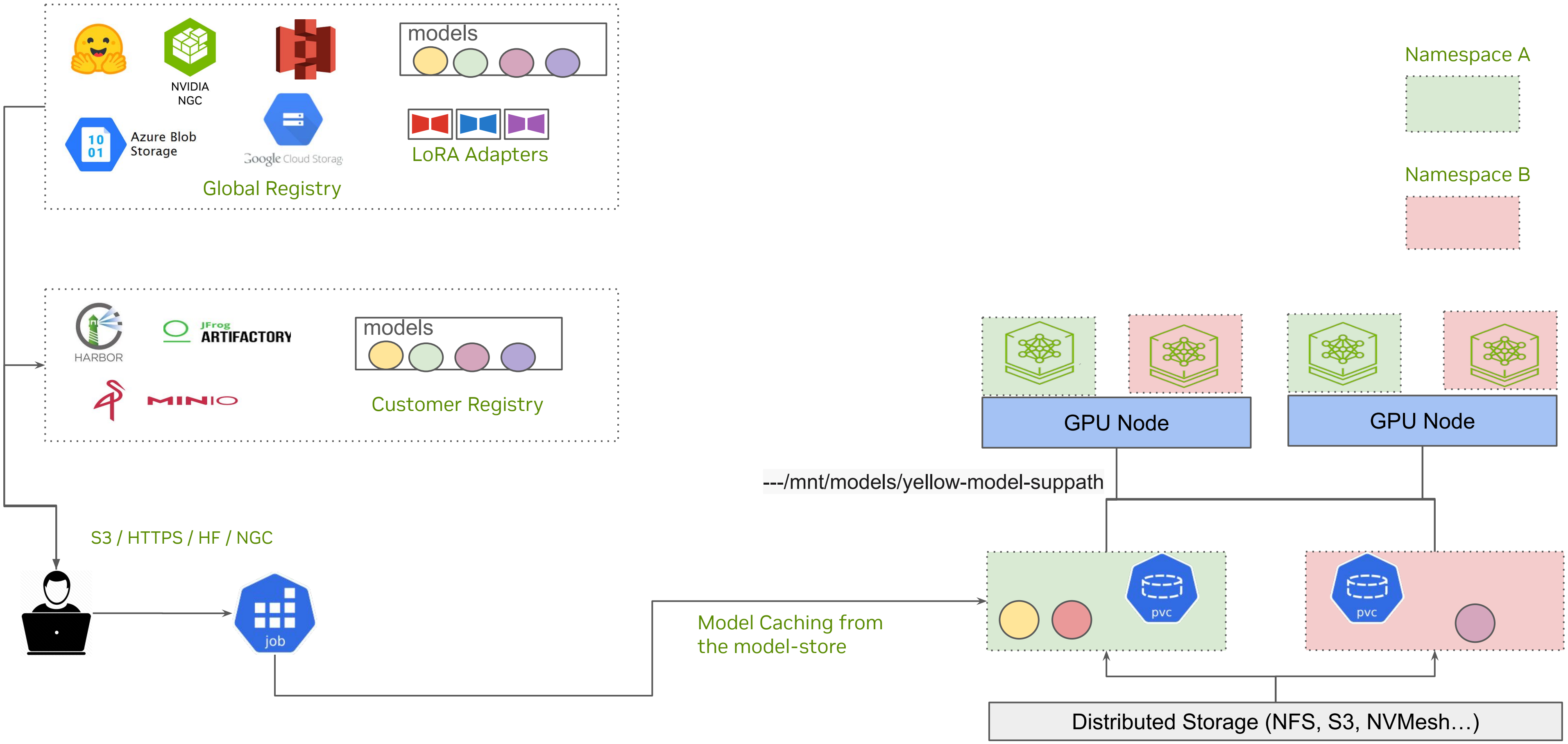
pvc

Distributed Storage (NFS, S3, NVMesh…)

Model Management



Model Management



NVIDIA NGC

Azure Blob Storage

Google Cloud Storage

models

LoRA Adapters

Global Registry

HARBOR

JFrog ARTIFACTORY

MINIO

models

Customer Registry

S3 / HTTPS / HF / NGC

job

Model Caching from the model-store

Namespace A

Namespace B

GPU Node

GPU Node

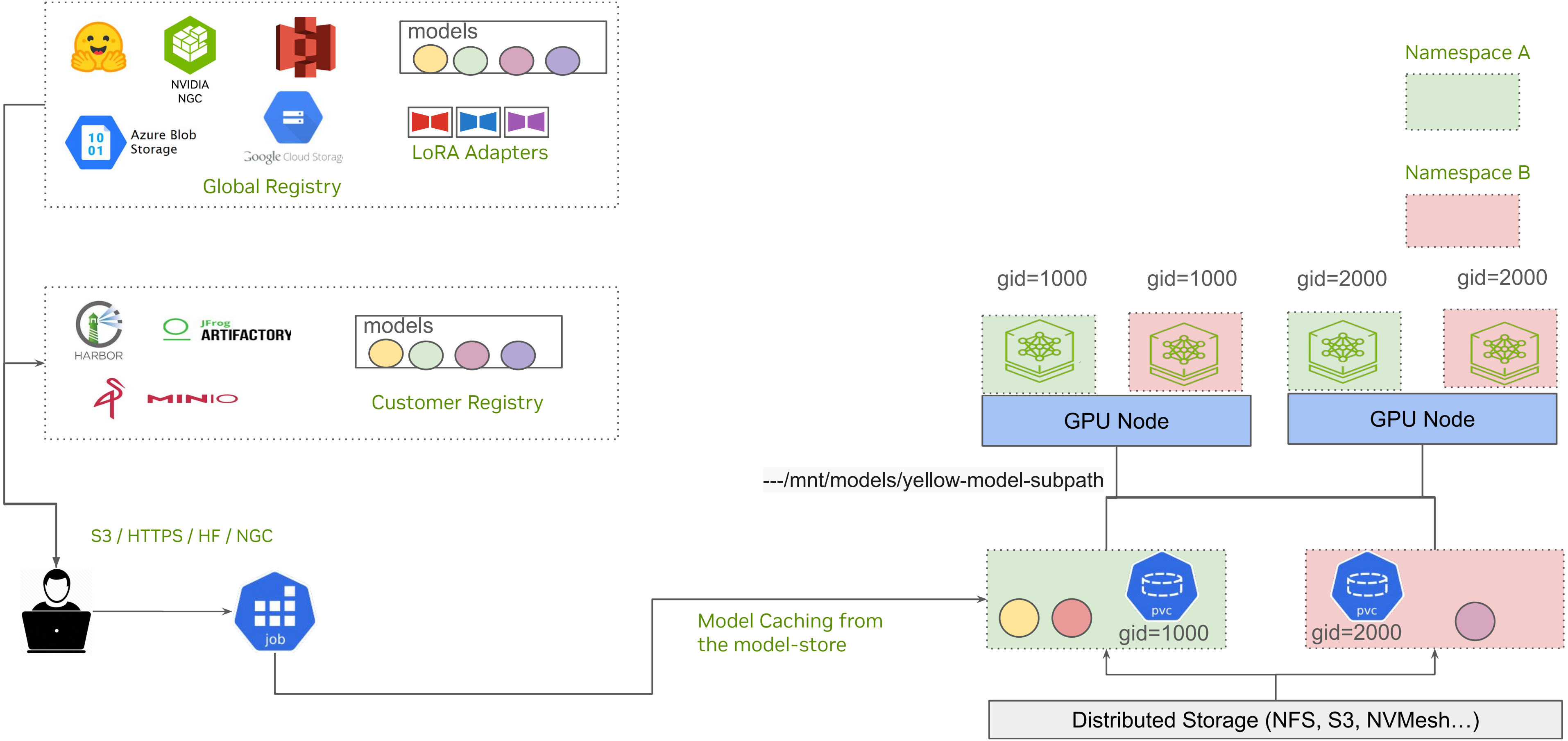
---/mnt/models/yellow-model-suppath

pvc

pvc

Distributed Storage (NFS, S3, NVMesh…)

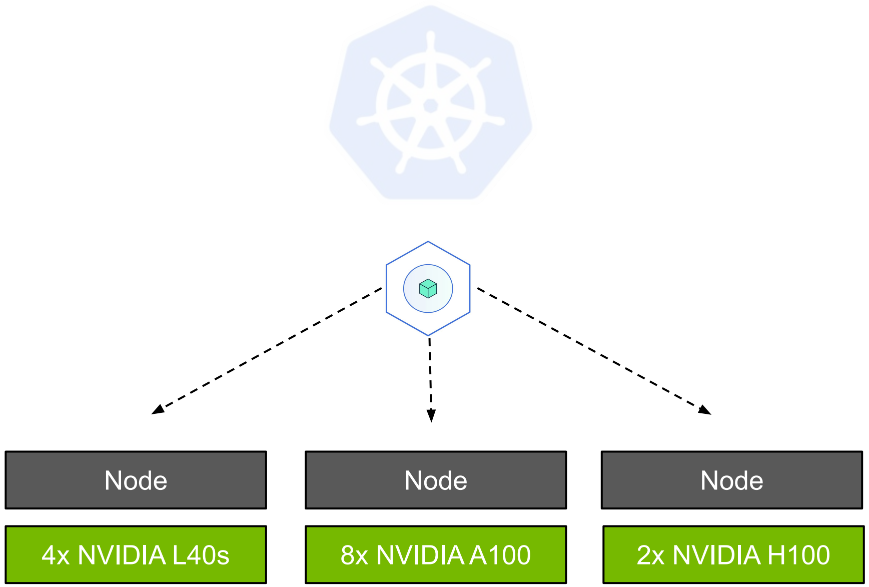
Model Management



GPU Scheduling

Allocate GPUs to Inference workloads in a Kubernetes Cluster

```
Model Selection
```



Node

4x NVIDIA L40s

Node

8x NVIDIA A100

Node

2x NVIDIA H100

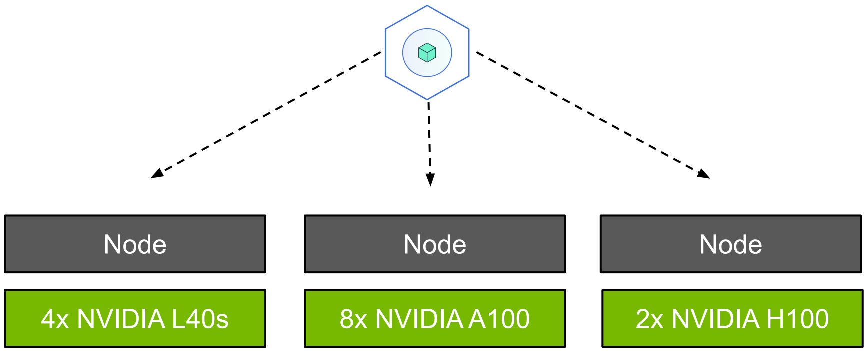
GPU Scheduling

Allocate GPUs to Inference workloads in a Kubernetes Cluster

```
Model Selection -> GPU Scheduling
```

```
apiVersion: v1
kind: Pod
metadata:
  name: model-server
spec:
  containers:
    - name: my-model-server
      image:
nvcr.io/nim/meta/llama3-70b-instruct:1.0.3
      resources:
        limits:
          nvidia.com/gpu: 1
```

- **Device Plugin**



Node

Node

Node

4x NVIDIA L40s

8x NVIDIA A100

2x NVIDIA H100

https://github.com/NVIDIA/gpu-operator

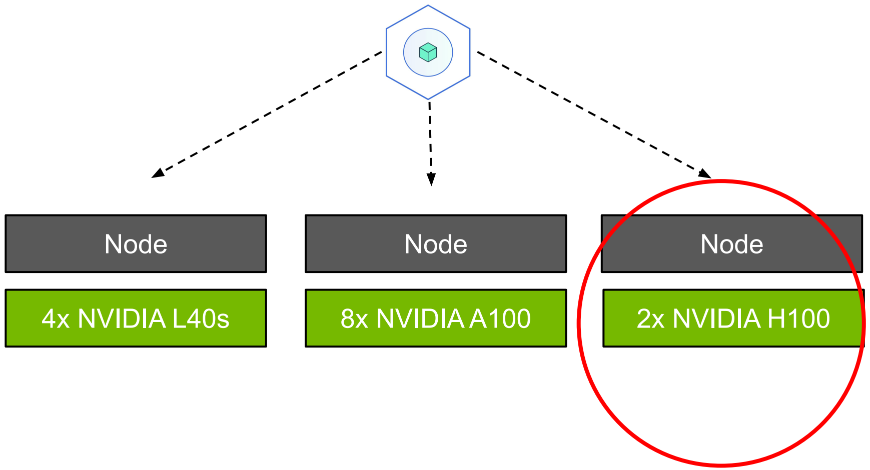
GPU Scheduling

Allocate GPUs to Inference workloads in a Kubernetes Cluster

```
Model Selection -> GPU Scheduling -> GPU
Allocation
```

```
apiVersion: v1
kind: Pod
metadata:
  name: model-server
spec:
  containers:
    - name: my-model-server
      image:
nvcr.io/nim/meta/llama3-70b-instruct:1.0.3
      resources:
        limits:
          nvidia.com/gpu: 1
  nodeSelector:
    nvidia.com/gpu.product: H100-PCIE-80GB
    nvidia.com/cuda.runtime: 12.7
    nvidia.com/cuda.driver: 565.57.01
```

- Device Plugin
- GPU Feature Discovery



https://github.com/NVIDIA/gpu-operator

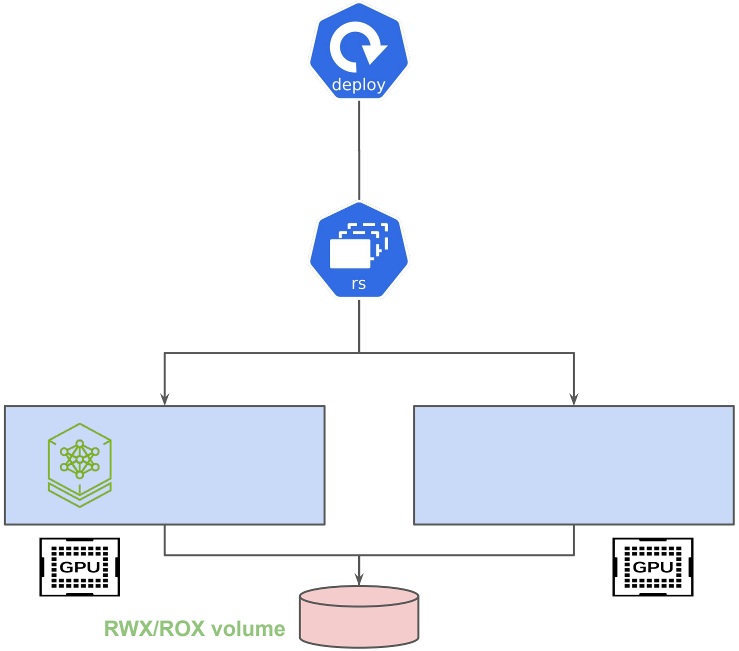
GPU Scheduling - With DRA

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: my-model-server
spec:
  containers:
  - name: model-server-ctr
    image: nvcr.io/nim/meta/llama3-70b-instruct:1.0.3
    command: ["nvidia-smi", "-L"]
    resources:
      limits:
        nvidia.com/gpu: 1
```

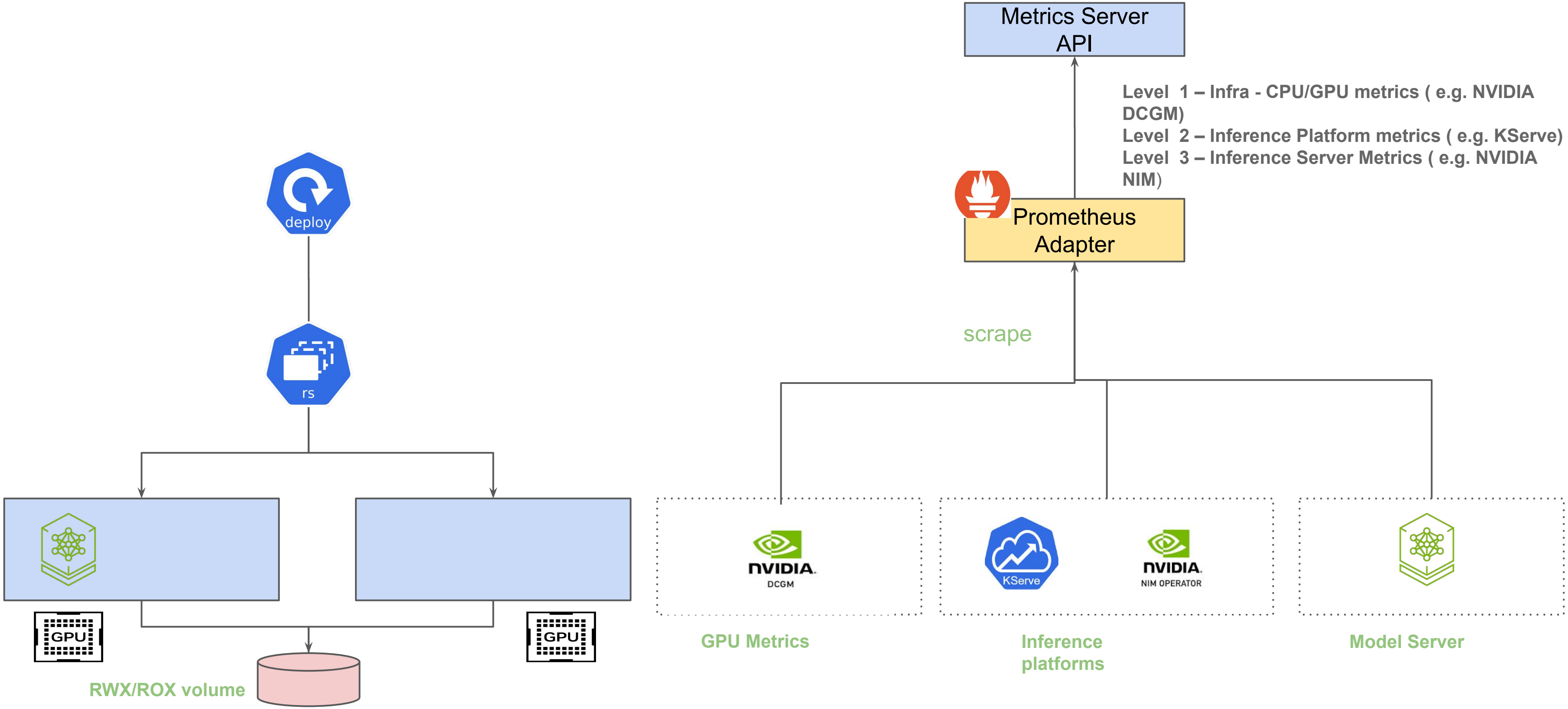
```yaml
apiVersion: resource.k8s.io/v1alpha2
kind: ResourceClaimTemplate
metadata:
  name: gpu-template
spec:
   devices:
     requests:
       - name: h100
         deviceClassName: gpu.nvidia.com
         selectors:
         - cel:
             expression: |
device.attributes['gpu.nvidia.com'].productName.lowerAscii().matches('^.*h100.*$')
         count: 1
```

```yaml
---
apiVersion: v1
kind: Pod
metadata:
  name: gpu-example
spec:
  containers:
  - name: ctr
    image: nvidia/cuda
    command: ["nvidia-smi" "-L"]
    resources:
      claims:
      - name: gpu
  resourceClaims:
  - name: gpu
    source:
      resourceClaimTemplateName: gpu-template
```

Observability and Autoscaling

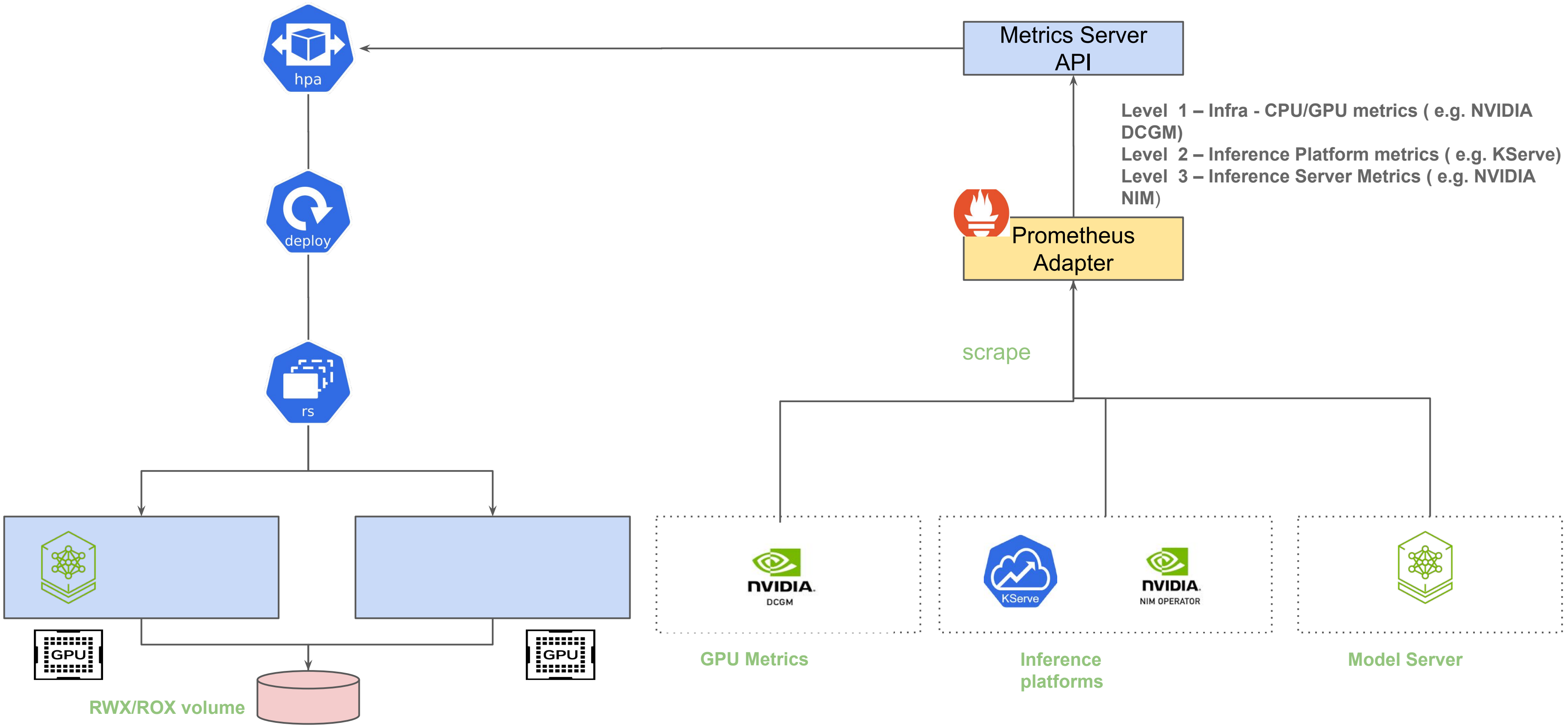


Observability and Autoscaling



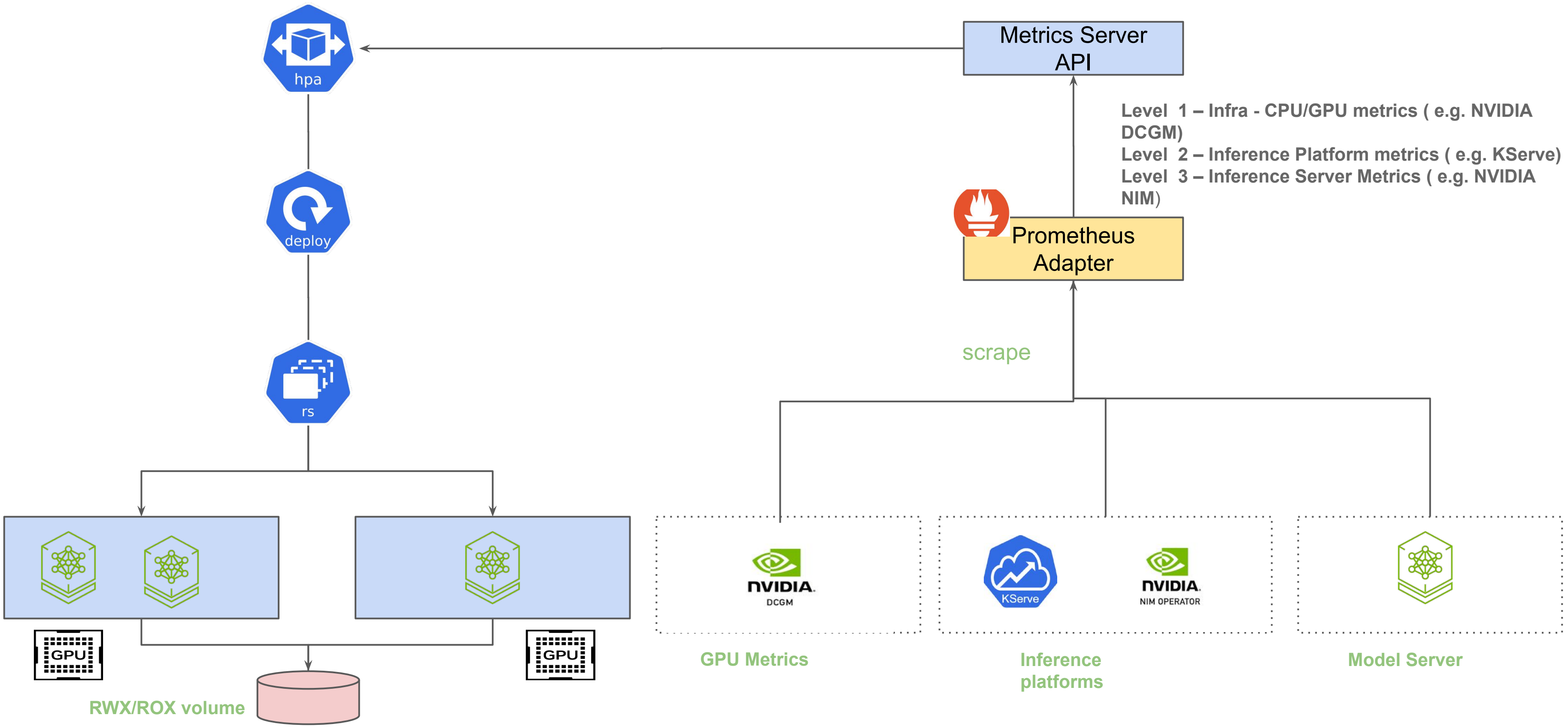
Level 1 – Infra - CPU/GPU metrics (e.g. NVIDIA DCGM)
Level 2 – Inference Platform metrics (e.g. KServe)
Level 3 – Inference Server Metrics (e.g. NVIDIA NIM)

Observability and Autoscaling



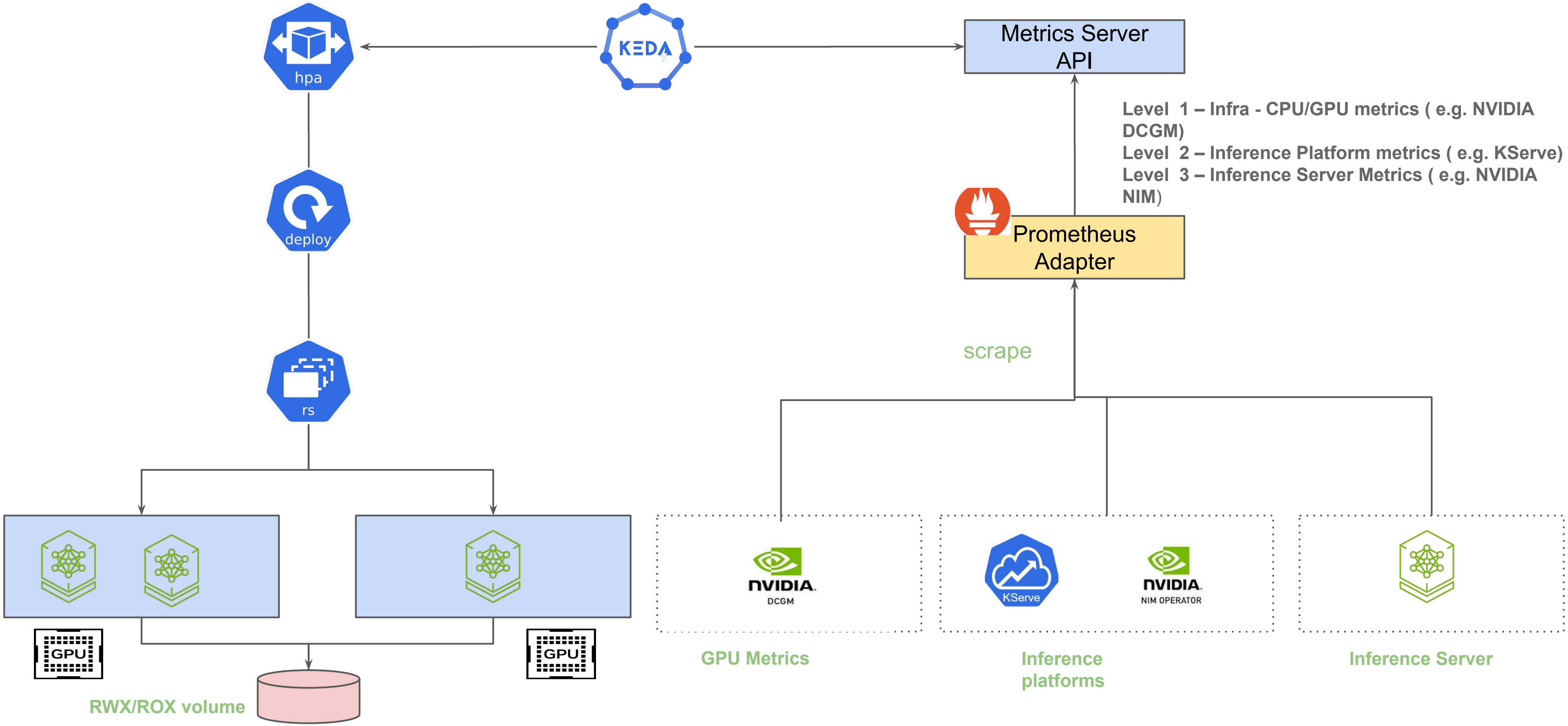
Level 1 – Infra - CPU/GPU metrics (e.g. NVIDIA DCGM)
Level 2 – Inference Platform metrics (e.g. KServe)
Level 3 – Inference Server Metrics (e.g. NVIDIA NIM)

Observability and Autoscaling



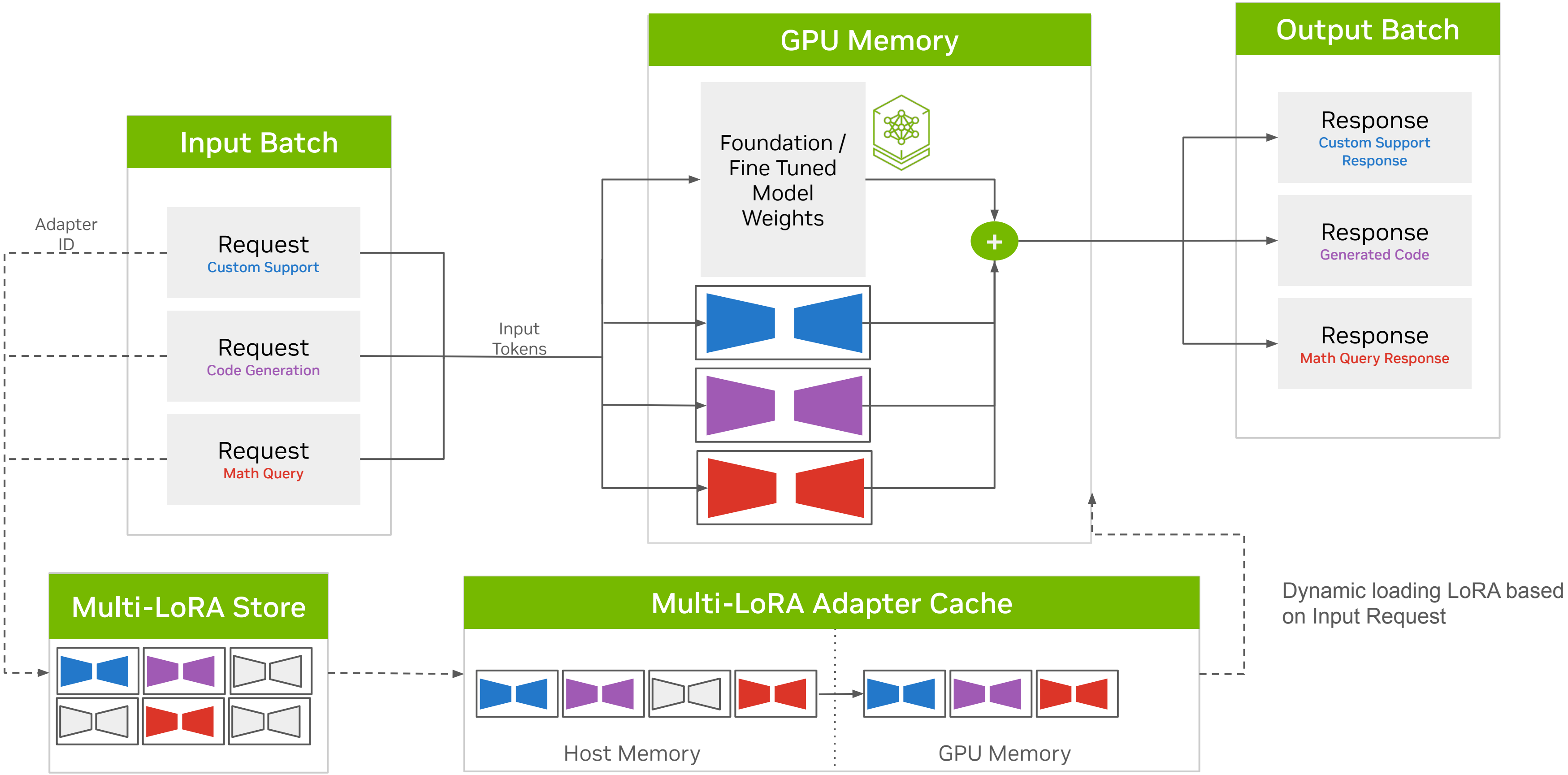
Level 1 – Infra - CPU/GPU metrics (e.g. NVIDIA DCGM)
Level 2 – Inference Platform metrics (e.g. KServe)
Level 3 – Inference Server Metrics (e.g. NVIDIA NIM)

Observability and Autoscaling

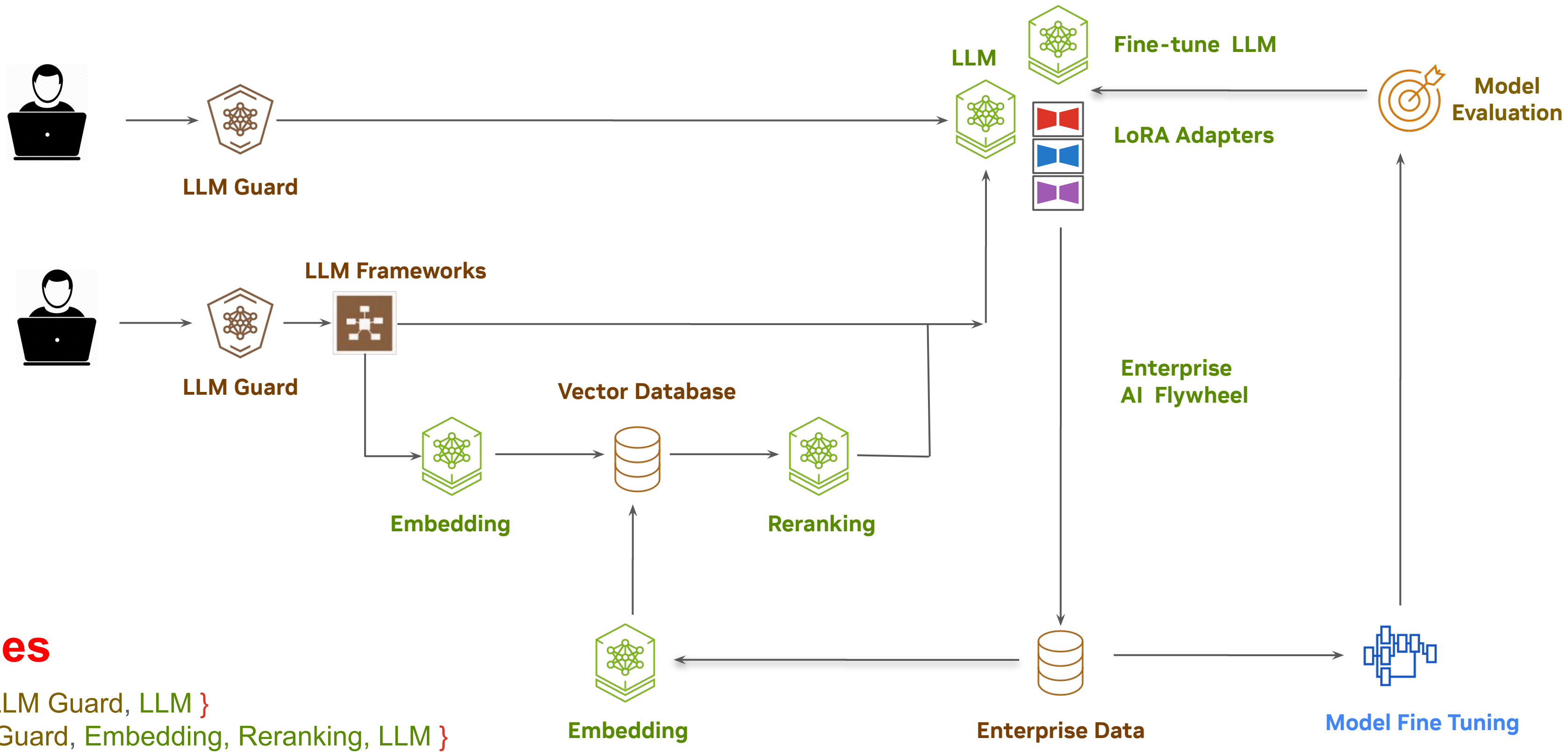


Level 1 – Infra - CPU/GPU metrics (e.g. NVIDIA DCGM)
Level 2 – Inference Platform metrics (e.g. KServe)
Level 3 – Inference Server Metrics (e.g. NVIDIA NIM)

Model and Multi-LoRA Serving



AI Pipelines

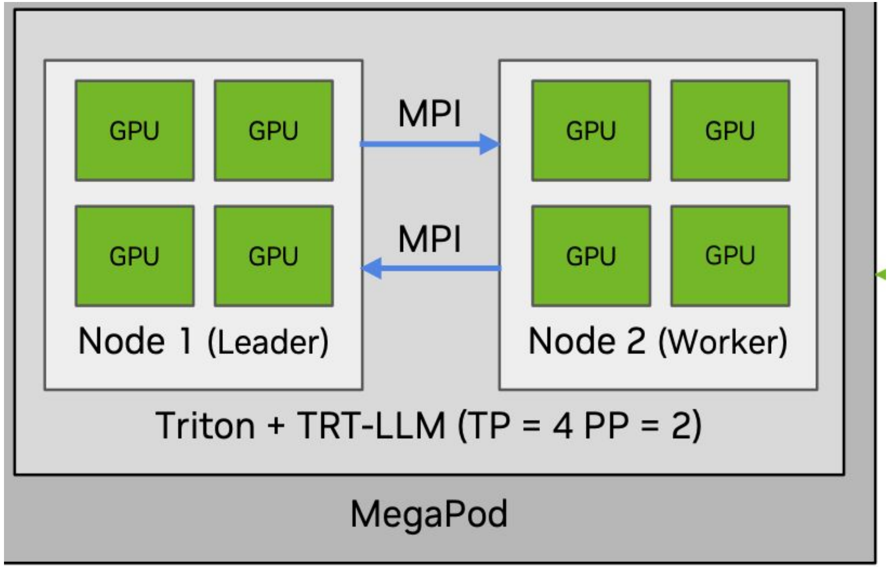


AI Pipelines

- Inference { LLM Guard, LLM }
- RAG { LLM Guard, Embedding, Reranking, LLM }
- Fine Tuning { Model Fine Tuning, Model Evaluation }
- …

Looking Forward…

Multi-node Inference



Use Case

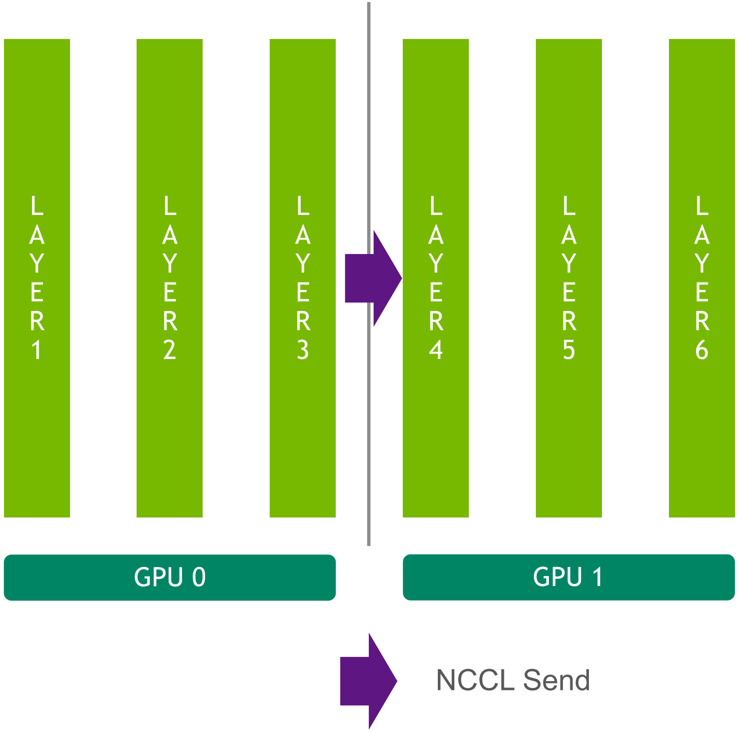
- Deploy Massive LLMs
- Automatically Scale and Load Balance

https://github.com/kserve/kserve/pull/3871

Multi-node Inference: Model Sharding

Pipeline Parallelism 2 (PP 2)

Split Into Sets of Layers



LAYER 1

LAYER 2

LAYER 3

LAYER 4

LAYER 5

LAYER 6

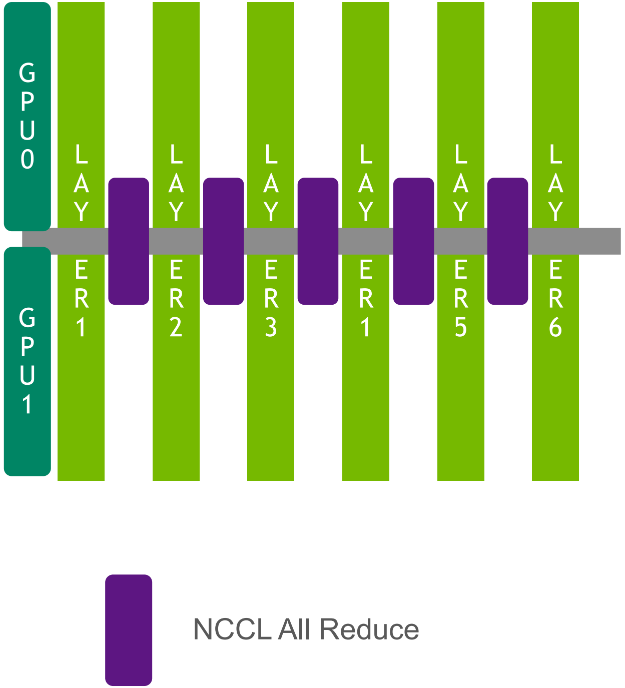
GPU 0

GPU 1

NCCL Send

Tensor Parallelism 2 (TP 2)

Split individual layers across GPUs



GPU 0

GPU 1

LAYER 1

LAYER 2

LAYER 3

LAYER 1

LAYER 5

LAYER 6

NCCL All Reduce

Multi-node Inference: Key Challenges

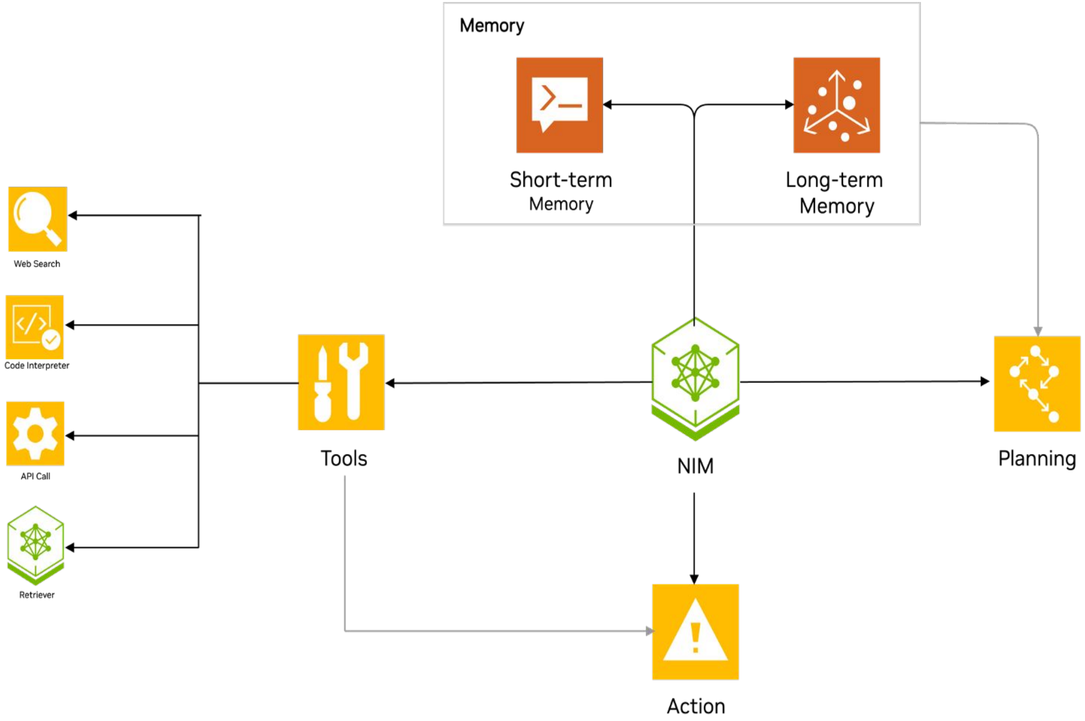
- Schedule group of nodes
  - Gang scheduling, Binpacking
- Operations
  - Deploy/Scale group of Nodes (LWS, ..)
  - Multi-node communication (MPI, …)
  - Leader Aware Load Balancing
- Optimizations
  - Accelerate initial loading (caching on shared storage)
  - Accelerate cross-node communication (RoCE / RDMA)

AI Agents

Enable advanced problem solving and automation for improved user experience

AI agents

- **Reasoning and Planning**: Decomposing complex tasks into manageable subgoals through reasoning
- **Memory**
  - Short-term memory in an LLM-powered agent acts as a record of actions and thoughts during a single query
  - Long-term memory logs interactions between the user and agent over extended periods
- **Tools:** Defined executable workflows that agents use to perform tasks



Conclusion

- K8s is a great platform for AI Pipelines
  - Strong GPU and Storage Integrations
  - Advanced Inference and Fine Tuning Platforms
  - Ease of Management and Monitoring

- NVIDIA NIM Operator leverages all these to simplify deployment of AI pipeilnes in Kubernetes

- Community is working on addressing gaps
  - Auto Scaling
  - Model Cache Management
  - LLM Gateway

Thank you and Feedback