Lenovo Validated Design: Hybrid AI 281 with IBM

Last update: **08 October 2026**

Version 1.0

Accelerating Enterprise AI with IBM OpenRAG, NVIDIA Blackwell, and Lenovo Hybrid AI

Optimized Architecture with NVIDIA Blackwell and NVIDIA AI Enterprise

Accelerate Innovation with Open, Flexible Architecture

Scale from Pilot to Production Faster

Hapsara Sukasdadi
Mike O'Brien
Devin Bost
Saif Adil

LENOVO PRESS

Table of Contents

1 Abstract 1

2 Introduction 2

2.1 Scope, assumptions, and validation status 2

2.2 Overview 3

2.2.1 Lenovo Hybrid AI 281 with IBM 3

2.2.2 Target Use Cases 4

3 Solution Components 5

3.1 AI Software Stack 5

3.1.1 Applications & Blueprints 5

3.1.2 Platform Technology 5

3.1.3 Model Tiers and Hardware Scaling 6

3.2 NVIDIA AI Enterprise (NVAIE) 8

3.2.1 NVIDIA Inference Microservice 9

3.3 Hybrid AI 281 with ThinkSystem SR675 v3 10

4 Solution Architecture 14

4.1 IBM OpenRAG 14

4.2 Langflow 15

4.3 NVIDIA Blueprints 19

4.3.1 NVIDIA VSS Blueprint in Langflow 19

4.3.2 NVIDIA AI-Q Blueprint in Langflow 21

4.4 Docling Document Parsing Service 23

4.5 LiteLLM 24

4.6 Langfuse Observability 25

4.7 OpenSearch Vector Database 25

4.8 ContextForge 25

4.9 GPU Allocation Strategy 25

5 Retrieval-Augmented Generation (RAG) Functional Test 28

5.1 Access Front End 28

5.2 Document Ingestion 28
5.3 Document Retrieval and Query 30
6 Production Considerations 31
6.1 Security and data protection 31
6.2 Observability and operations 31
6.3 Backup and recovery 32
6.4 Validation scope and acceptance criteria 32
6.5 Performance Considerations 33
6.6 Capacity planning and service objectives 33
7 Appendix A: Bill of Material (BoM) 35
7.1 Configuring Hybrid AI 281 Canonical Solution in DCSC 35
8 Appendix B: Software Deployment Outline 37
8.1 Installation 37
8.1.1 Prerequisites 37
8.1.2 Deploy Applications 37
8.1.3 IBM Storage Scale Dev Edition 37
8.1.4 Deploy NIM 38
8.1.5 Validated configuration record 39
8.1.6 Network architecture and port requirements 40
8.1.7 Configuration options 42
8.1.8 Agent configurations 43
8.2 Checking for Successful Deployment 44
Resources 45
Document history 46
Trademarks and special notices 47

1 Abstract

Lenovo Hybrid AI 281 with IBM is a compact, single-node enterprise AI platform designed for retrieval-augmented generation (RAG) and related inference workloads. Built as a subset of the Lenovo Hybrid AI 285 platform, it scales down the 2-8-5 reference configuration — two CPUs, up to eight GPUs, and five network adapters — into a two CPUs, up to eight GPUs, and a single-NIC configuration that is optimized for cost-effective, high-throughput inference at the edge or in the data center.

This guide documents the Lenovo Hybrid AI with 281 with IBM AI software stack, with RAG and Agentic deployment examples from IBM and NVIDIA. The implementation combines Lenovo ThinkSystem SR675 V3 GPU-rich server hardware, NVIDIA RTX Pro 6000 Blackwell Server Edition GPUs, IBM OpenRAG software, IBM Docling document ingestion, IBM Langflow visual flow orchestration, NVIDIA Blueprints, NVIDIA AI Enterprise, OpenSearch vector database, Langfuse, and LiteLLM model routing into a validated, production-ready RAG, Video Intelligence, and Agentic Reserach stack.

The audience for this document includes sales architects, solution engineers, customers, and partners who need a high-level design reference for deploying a cost efficient AI inference platform on Lenovo Hybrid AI infrastructure.

The platform documented here is a single-node AI appliance that can run an entry level, proof of concept, or demo stack such as enterprise document intelligence, video intelligence, and agentic research. All three applications share the same OpenRAG backend, OpenSearch vector store, NVIDIA NIM microservices, LiteLLM model gateway, Langflow orchestration engine, and Langfuse observability layer. The stack can be scaled up for production deployment. Customers can compose, modify, and deploy custom workflows through the same visual Langflow environment without duplicating infrastructure or standing up new integrations.

2 Introduction

The shift from generative AI to agentic AI is changing how enterprises automate knowledge work. Organizations now need infrastructure that can run large language models (LLMs) privately, connect them to enterprise data, and expose them through conversational interfaces that reason, retrieve, and act.

Lenovo Hybrid AI 281 with IBM addresses this need by delivering a compact 3U, GPU-rich inference node that can be deployed as a standalone AI factory or as an extension of an existing IT environment. It is built on the same design principles as Lenovo Hybrid AI 285 — but removes the multi-node East-West GPU network and training-oriented networking requirements to deliver a simpler, inference-optimized footprint.

Lenovo and IBM also implemented a PoC/Demo version of the IBM OpenRAG, NVIDIA AI-Q, and VSS blueprint stacks on this node. IBM OpenRAG provides document ingestion, vector search, LLM reasoning, multimodal understanding, and observability, all running on the Lenovo ThinkSystem SR675 V3 server. The result is a self-contained system that can ingest enterprise documents, answer questions grounded in those documents, and support agentic workflows through visual Langflow pipelines.

This guide describes the high-level design of that implementation, including hardware, networking, software components, data flow, and the Langflow pipelines that IBM configured on the system. In this document we are mostly focused on the OpenRAG blueprint, since it was the only blueprint that was validated. NVIDIA VSS and AI-Q blueprints are available but not tested during the validation.

2.1 Scope, assumptions, and validation status

This document describes the Lenovo Hybrid AI 281 with IBM configuration validated for IBM OpenRAG on a single Lenovo ThinkSystem SR675 V3 server. The design includes additional Langflow flows for NVIDIA VSS and AI-Q.

The validated configuration assumes customer-provided data governance, identity services, enterprise DNS and time synchronization, certificate services, backup infrastructure, and network controls. The design does not provide node-level high availability, distributed training, a multi-node GPU fabric, or a disaster-recovery site.

Table 1: Scope, assumptions, and validation status

Capability	Present in design	Validated in this release	Notes
IBM OpenRAG	Yes	Yes	End-to-end document ingestion and retrieval workflow
NVIDIA VSS flows	Yes	[PENDING TESTING ON RTX 6000's]	Document tested flows, file-based versus streaming scope, and deviations from the reference blueprint

NVIDIA AI-Q flows	Yes	[PENDING TESTING ON RTX 6000's]	Document tested flows, model substitutions, and functional limitations
High availability	No	No	Single-server failure domain
Distributed training	No	No	Outside the 281 design scope
Kubernetes deployment	Optional future architecture	No	Docker Compose is the documented deployment method

2.2 Overview

2.2.1 Lenovo Hybrid AI 281 with IBM

Lenovo Hybrid AI 281 is based on the NVIDIA 2-8-5 configuration: 2 CPUs, 8 GPUs, and 5 NICs. The Hybrid AI 285 is designed for small to medium inference, training, and fine-tuning across multiple nodes. Lenovo Hybrid AI 281 is a subset of that platform: 2 CPUs, up to 8 GPUs, and 1 NIC, focused exclusively on inference only workloads such as RAG that fit within a single node.

Unlike Lenovo Hybrid AI 285, which uses five network adapters to support high-bandwidth East-West GPU traffic for distributed training, Lenovo Hybrid AI 281 uses a single network adapter because inference-only RAG workloads do not require node-to-node GPU communication. This reduces cost, cabling, and switch complexity while preserving the full compute and GPU memory capacity of the SR675 V3.

The 281 node can serve as:

- A single node Proof-of-Concept and Demo platform for inference use case
- An entry point for enterprises beginning their AI factory journey
- A standalone inference appliance for a single department or use case
- A building block that can later be expanded into a full 285 deployment if multi-node training becomes required

The Lenovo Hybrid AI 281 with IBM platform is a single-node, inference-optimized configuration of the Lenovo Hybrid AI 285 architecture. It retains the same CPU and GPU density as the larger platform but simplifies networking for deployments that do not require multi-node GPU-to-GPU communication.

This appliance delivers enterprise AI infrastructure and three pre-validated production-ready applications/blueprints in a single, self-contained system — enterprise document intelligence, video intelligence, and agentic research — all sharing one GPU-accelerated model serving foundation, running on certified Lenovo hardware and built using NVIDIA's NIM microservice stack.

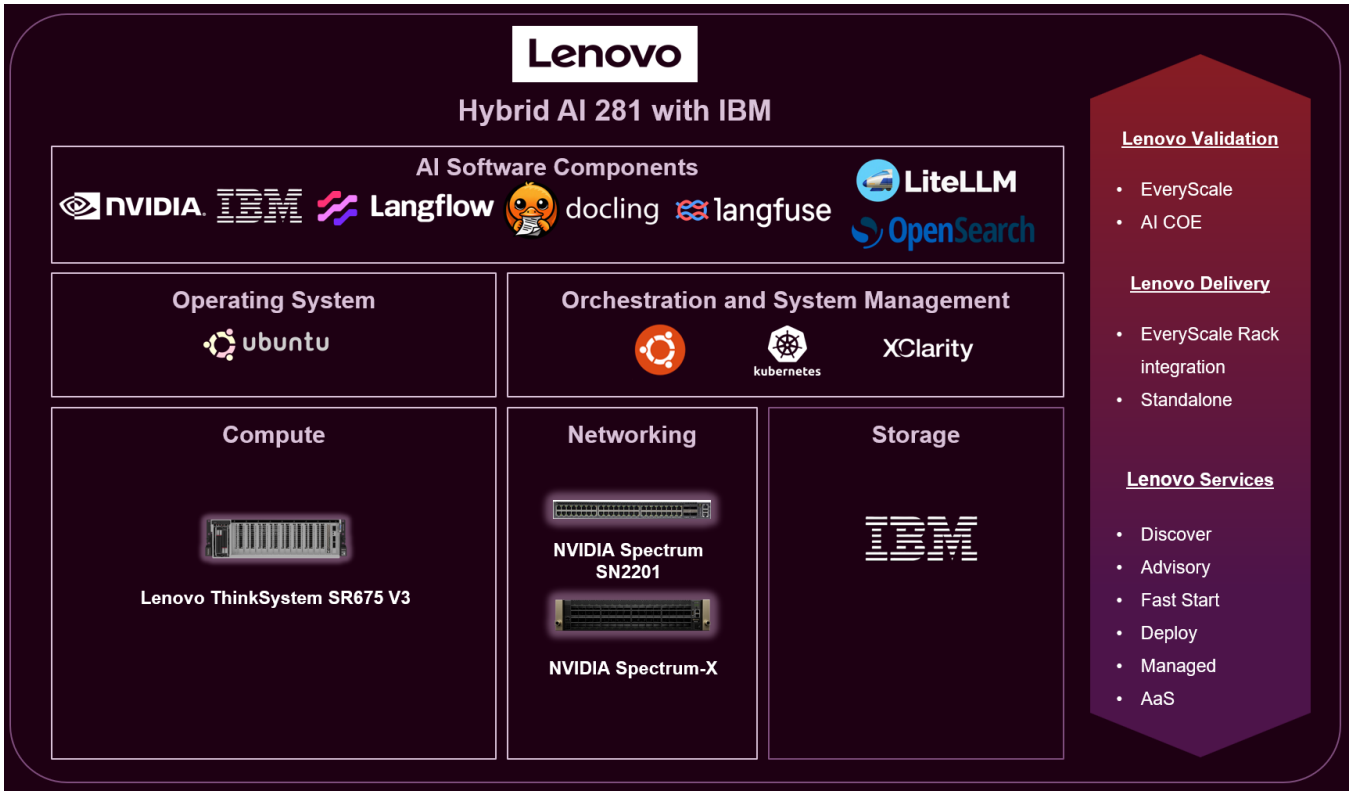


Figure 1: Lenovo Hybrid AI 281 with IBM

2.2.2 Target Use Cases

- Enterprise RAG over private documents
- Conversational AI assistants grounded in enterprise knowledge
- Agentic AI workflows with tool use and reasoning
- Multimodal document understanding (text, image, video)
- Video intelligence — scene captioning, event detection, object/activity recognition, and natural-language search across video
- Agentic research — multi-step investigation across documents and video with structured, cited reports
- Low-latency inference serving for small language models

3 Solution Components

3.1 AI Software Stack

3.1.1 Applications & Blueprints

Three application/blueprints are available with the platform, each built on the same shared model, search, and orchestration foundation rather than as siloed point solutions:

- **Document Intelligence (IBM OpenRAG)** — enterprise document ingestion, chat, and knowledge base. Documents of any common format (PDF, Office, scanned/image-based content) are parsed, indexed, and made instantly searchable; users get a conversational assistant that answers questions with grounded, cited sources drawn directly from the organization's own content — no data leaves the system.
- **Video Intelligence (NVIDIA VSS Blueprint)** — ingests video from files and automatically extracts structured understanding: scene captioning, event detection, and object/activity recognition, indexed for retrieval alongside the platform's document knowledge base (see the 289 configuration for production live stream video processing). Capabilities include natural-language search across video content ("show me every clip where a forklift enters the loading dock"), automatic asset-level summarization, a conversational vision agent for interactively querying footage, and automated alert verification that checks a triggered alert against the actual video evidence before it reaches a human.
- **Agentic Research (NVIDIA AI-Q Blueprint)** — a multi-step research assistant that plans and executes investigation across the organization's full knowledge base (documents and video) as well as external sources. It routes each request to the appropriate depth of investigation — a quick grounded answer for simple questions, or a full planned research process with clarification, source routing, and parallel investigation for complex ones — then synthesizes findings into structured, cited reports, with support for iterative follow-up and refinement. AI-Q in this configuration is capable of POC and pre-production demonstrations of deep research. See the 289 configuration for full deep research capabilities.

All three applications are fully customizable and extensible: customers can compose, modify, and deploy their own workflows and agents on the same shared infrastructure using the platform's visual orchestration layer, without standing up new integrations or duplicating infrastructure. During this validation we tested IBM RAG blueprint only. NVIDIA VSS and AI-Q blueprints are available in Langflow, but they are not deployed during validation.

3.1.2 Platform Technology

The three application/blueprints above are built from a set of proven, open technologies, integrated into one coherent stack:

- **NVIDIA NIM** — every inference operation on the box, from conversational reasoning to document parsing to video understanding, runs through NVIDIA's NIM microservices: containerized, GPU-

optimized model endpoints that give consistent performance and a standard API surface across every model in the stack.

- **IBM Docling** — handles document parsing and understanding ahead of ingestion: PDFs, Office formats, scanned/image-based documents via OCR, tables, and figures, converting them into clean, structured content for retrieval and search.
- **IBM OpenSearch** — the platform's unified enterprise search and storage layer. Combines traditional keyword search with vector-based semantic search (hybrid search) across documents, video, and generated reports in a single, proven, scalable store — rather than a bespoke vector database.
- **IBM Langflow** — the platform's visual orchestration engine. OpenRAG, Video Intelligence and Agentic Research are all built as Langflow flows on top of the shared model and search layer, and customers use the same visual environment to compose, customize, and extend their own AI workflows.
- **LiteLLM** — a unified model gateway sitting in front of the NIM layer. Presents every model on the box through a single, standard, OpenAI-compatible API, so applications and custom workflows address models by a consistent name without needing to know the underlying deployment details — and gives administrators a single point for routing, load balancing, and key management.
- **Langfuse** — LLM observability and operations (LLMOps). Captures traces across every model call and workflow run, giving administrators visibility into latency, usage, and behavior across the whole platform, alongside built-in native tracing in the orchestration layer itself.
- **IBM ContextForge** – Shared gateway and control plane for agentic discovery, governance, and observability for MCP, A2A, REST and gRPC
- **IBM Storage Scale Developer Edition** – in a single-node cluster configuration. While the single-node setup means the cluster lacks the high-availability and parallel I/O benefits of a multi-node GPFS deployment, it still provides the core IBM Storage Scale features: a unified POSIX namespace, integrated tiering capability, and AFM (Active File Management) support.

3.1.3 Model Tiers and Hardware Scaling

Every application/blueprints on the platform draw from the same shared set of model roles rather than provisioning models per-application. The specific models behind each role scales up with the hardware tier from **Entry Tier**: Single node 281 or 285, **Mid Tier**: Single node 289 and **Scale-Out Tier:** SU deployments of 289 or 285; while the role names — and every application built against them — stay identical across tiers.

Table 2: Shared model roles across hardware tiers

Role	**Purpose**	**Entry Tier**	**Mid tier and Scale-Out Tier Mid tier and Scale-Out Tier**
LLM_FAST	Conversational reasoning, intent	Nemotron 3.5 Lightning 30B	Nemotron 3.5 Lightning 30B

	routing, tool-calling, everyday agent tasks		
LLM_CHAT	Powerful chat and response generation	Nemotron 3 Super 120B	Nemotron 3 Super 120B
LLM_DEEP	Complex planning, multi-step reasoning, long-form synthesis	Aliased to LLM_CHAT	Nemotron 3 Ultra 550B (NVLink)
OMNI	Multimodal understanding — video captioning, event extraction, visual Q&A	Nemotron 3 Nano Omni 30B	Nemotron 3 Nano Omni 30B
EMBED	Text embeddings for search and retrieval	NV-EmbedQA 1B	NV-EmbedQA 1B
RERANK	Retrieval result reranking	NV-RerankQA 1B	NV-RerankQA 1B
IMG_EMBED	Image and video-frame embeddings	NV-CLIP	NV-CLIP
EXTRACT_PAGE_ELEMENTS	Page layout and region detection for document ingestion	NeMo Retriever page-elements-v3	NeMo Retriever page-elements-v3
EXTRACT_TABLE_STRUCTURE	Table structure extraction for document ingestion	Retriever table-structure-v1	Retriever table-structure-v1
EXTRACT_GRAPHIC_ELEMENTS	Chart and figure detection for document ingestion	Retriever graphic-elements-v1	Retriever graphic-elements-v1
EXTRACT_OCR	Scene-text OCR for document ingestion	NeMo Retriever OCR	NeMo Retriever OCR

Hardware Scaling

The platform runs entirely on NVIDIA's Blackwell generation of GPUs end to end. The same software stack — every application, every workflow, every model role — runs unmodified across every tier; only the underlying GPU class and count changes:

- **Entry tier**— Hybrid AI 281/285 single-node, 8-GPU configuration on NVIDIA Blackwell RTX PRO 6000 GPUs, delivering full Blackwell-generation performance in a cost-efficient footprint for development, Proof-of-Concept evaluation, and light production deployments.
- **Mid tier** — Hybrid AI 289 single-node, 8-GPU configuration on NVIDIA Blackwell B300 datacenter GPUs, sized for full production workloads at higher concurrency and throughput.
- **Scale-out tier**— multi-node configuration on NVIDIA's latest Blackwell-generation datacenter GPUs, adding a dedicated NVLink-connected GPU domain for frontier-scale reasoning alongside the standard shared serving layer, for customers with the highest-volume or most demanding reasoning workloads.

Because the platform's model roles are addressed by name rather than hardcoded to a specific model, customers can grow from evaluation to full-scale production — or add frontier-scale reasoning capability later — without re-architecting or redeploying applications. Every tier, from a single evaluation node to a multi-node scale-out deployment, runs on the same current-generation NVIDIA architecture, so there's no hardware generation gap to bridge as customers grow.

The validated IBM OpenRAG implementation on Lenovo Hybrid AI 281 was configured as the entry tier, suitable for PoC and demo. In the entry tier, as documented in the table above, Nemotron 3.5 Lightning 30B for LLM_FAST and LLM_DEEP roles was used to optimize resource overheads.

At the scale-out tier, the LLM-DEEP role steps up to a frontier-scale reasoning model (Nemotron 3 Ultra, 550B parameters) that spans multiple GPUs over a high-bandwidth NVLink interconnect — a meaningfully larger hardware commitment than any other role on the platform, reserved for customers who need the deepest reasoning capability available for their most demanding agentic research and analysis workloads.

This document focuses on the entry tier Hybrid AI 281 single-node configuration, but the same software stack can be scaled up for production and larger deployments.

3.2 NVIDIA AI Enterprise (NVAIE)

NVIDIA AI Enterprise is a comprehensive suite of artificial intelligence and data analytics software designed for optimized development and deployment in enterprise settings. This section outlines some of the tools present in NVIDIA AI Enterprise.

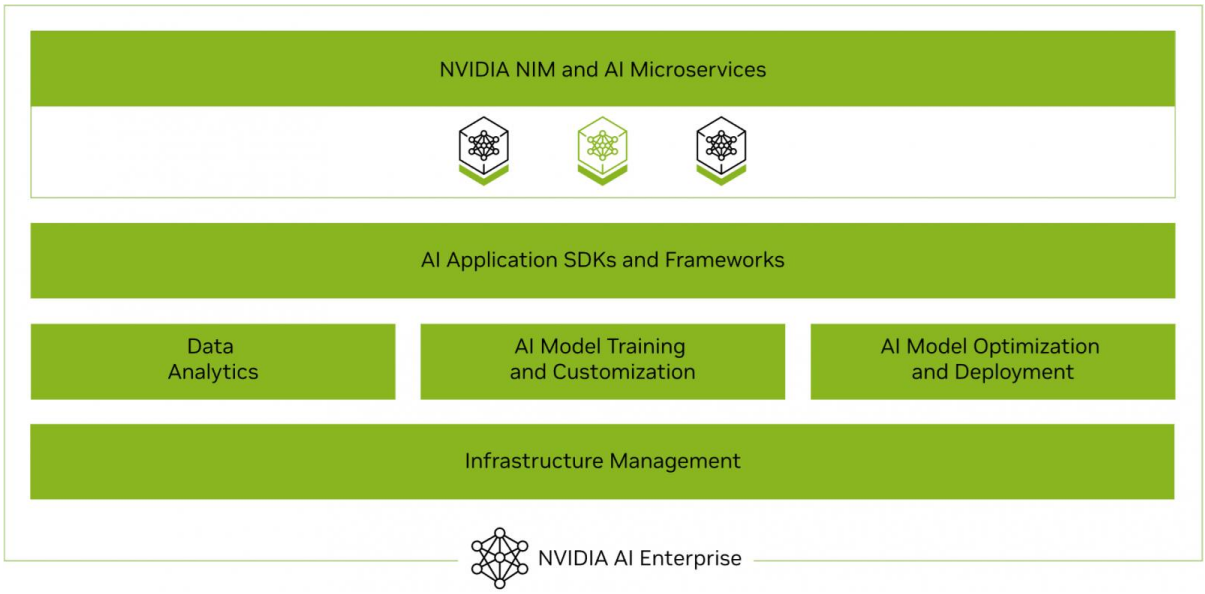


Figure 2: NVIDIA AI Enterprise

NVIDIA AI Enterprise provides access to ready-to-use open-sourced containers and frameworks from NVIDIA like NVIDIA NeMo, NVIDIA RAPIDS, NVIDIA TAO Toolkit, NVIDIA TensorRT and NVIDIA Triton Inference Server.

It also provides full access to the NVIDIA NGC catalogue, a collection of tested enterprise software, services and tools supporting end-to-end AI and digital twin workflows and can be integrated with MLOps platforms such as ClearML, Domino Data Lab, Run:ai, UbiOps, and Weights & Biases. An NVIDIA AI Enterprise License provides access to the fully secured, vetted, and tested software artifacts that are supported by NVIDIA.

3.2.1 NVIDIA Inference Microservice

NVIDIA AI Enterprise introduced NVIDIA Inference Microservices (NIM), a set of performance-optimized, portable microservices designed to accelerate and simplify the deployment of AI models. Those containerized GPU-accelerated pretrained, fine-tuned, and customized models are ideally suited to be self-hosted and deployed on the Lenovo Hybrid AI platforms.

The ever-growing catalog of NIM microservices contains models for a wide range of AI use cases, from chatbot assistants to computer vision models for video processing. The image below shows some of the NIM microservices, organized by use case.

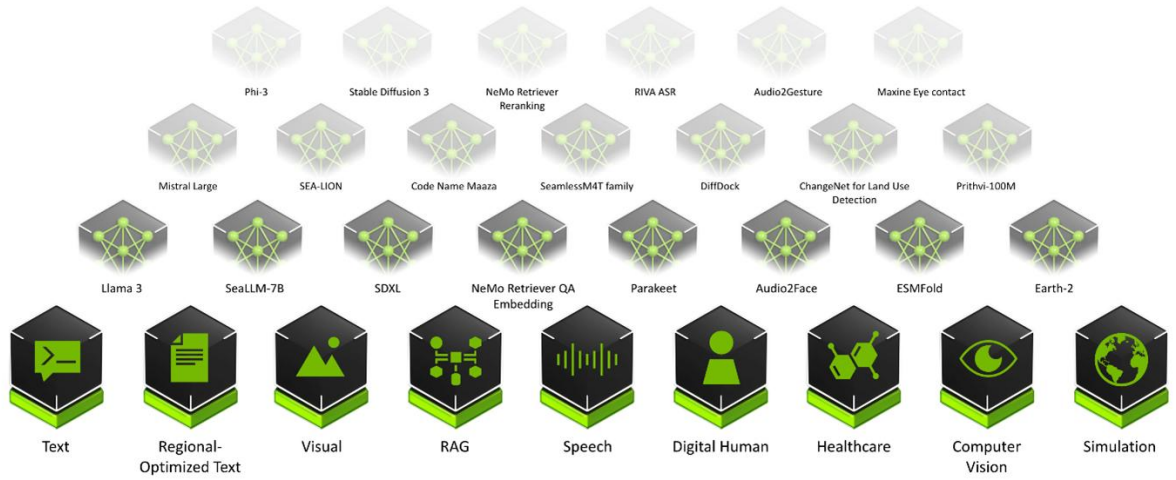


Figure 3: Examples of NIM Catalogue

For an in-depth guide to the NVIDIA Software portfolio with Lenovo Part Numbers, please reference the NVIDIA Software Product Guide on Lenovo Press.

Table 3: NVIDIA AI Enterprise License

Part number	Description
7S02CTO1WW	NVIDIA Software
S6Z3	NVIDIA Enterprise (NVIDIA AI Enterprise and NVIDIA Omniverse Enterprise) Subscription per GPU, 3 Year

3.3 Hybrid AI 281 with ThinkSystem SR675 v3

The AI Compute Node for Hybrid AI 281 with IBM leverages the Lenovo ThinkSystem SR675 V3 GPU-rich server.

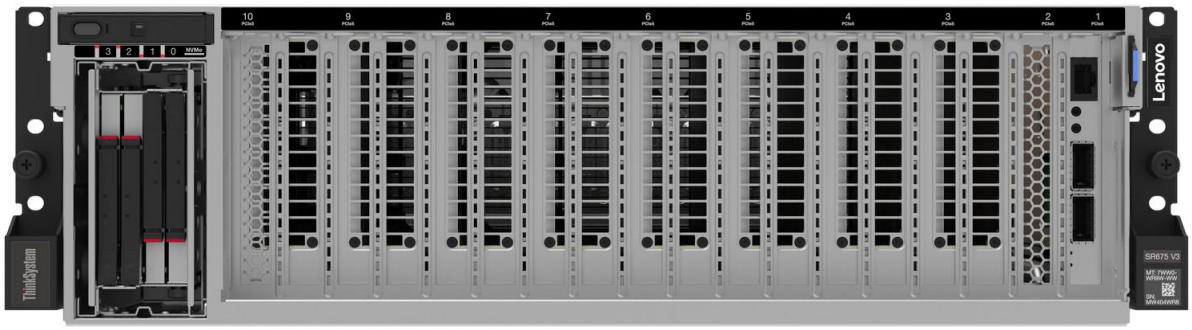


Figure 4: Lenovo ThinkSystem SR675 V3 in 8DW PCIe Setup

The SR675 V3 is a 2-socket 5th Gen AMD EPYC 9005 server supporting up to 8 PCIe DW GPUs with up to 5 network adapters in a 3U rack server chassis. This makes it the ideal choice for NVIDIA's 2-8-5 configuration requirement. The Hybrid AI 281 is basically the 285 configuration without the 4 E-W fabric NICs used for GPU to GPU communication.

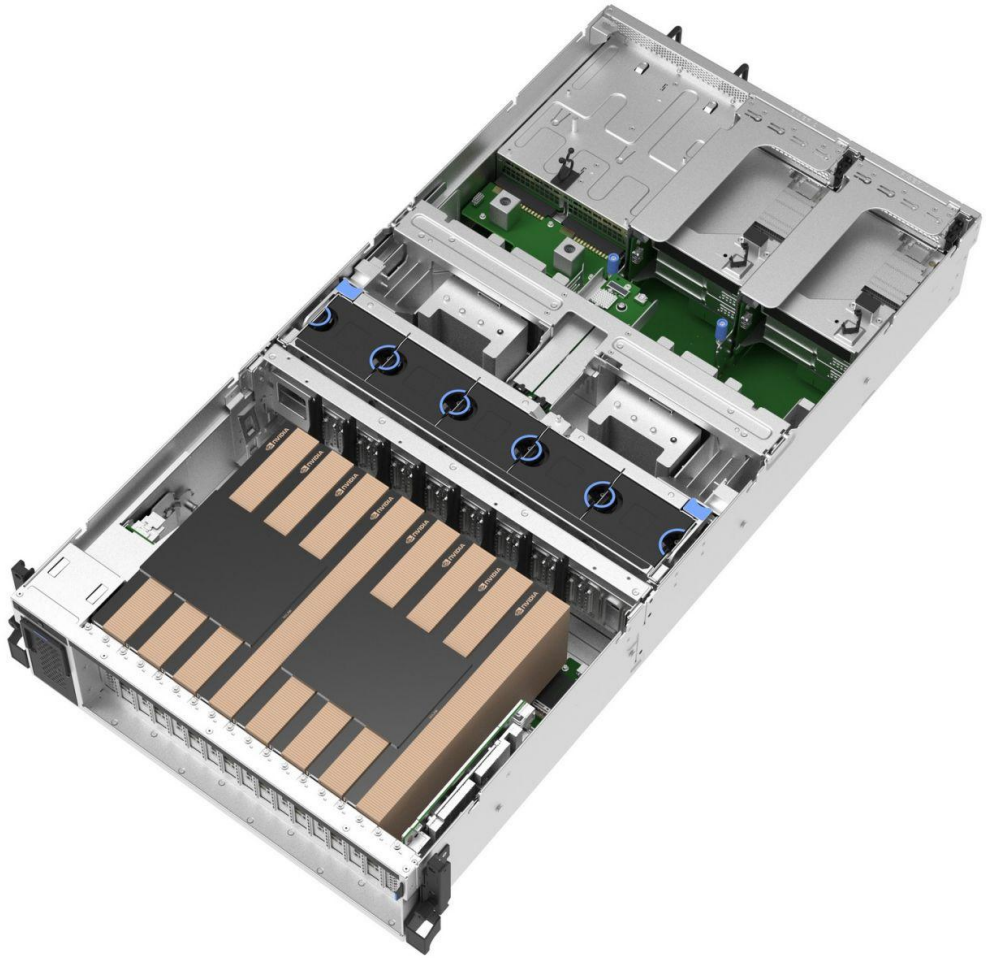


Figure 5: Lenovo ThinkSystem SR675 V3 in 8DW PCIe Setup

This configuration is NVIDIA-certified for Enterprise as well as Spectrum-X and an evolution of the Lenovo EveryScale OVX 3.0 setup which NVIDIA also deployed based on the Lenovo ThinkSystem SR675 V3 within NVIDIA Omniverse Cloud on Microsoft Azure.

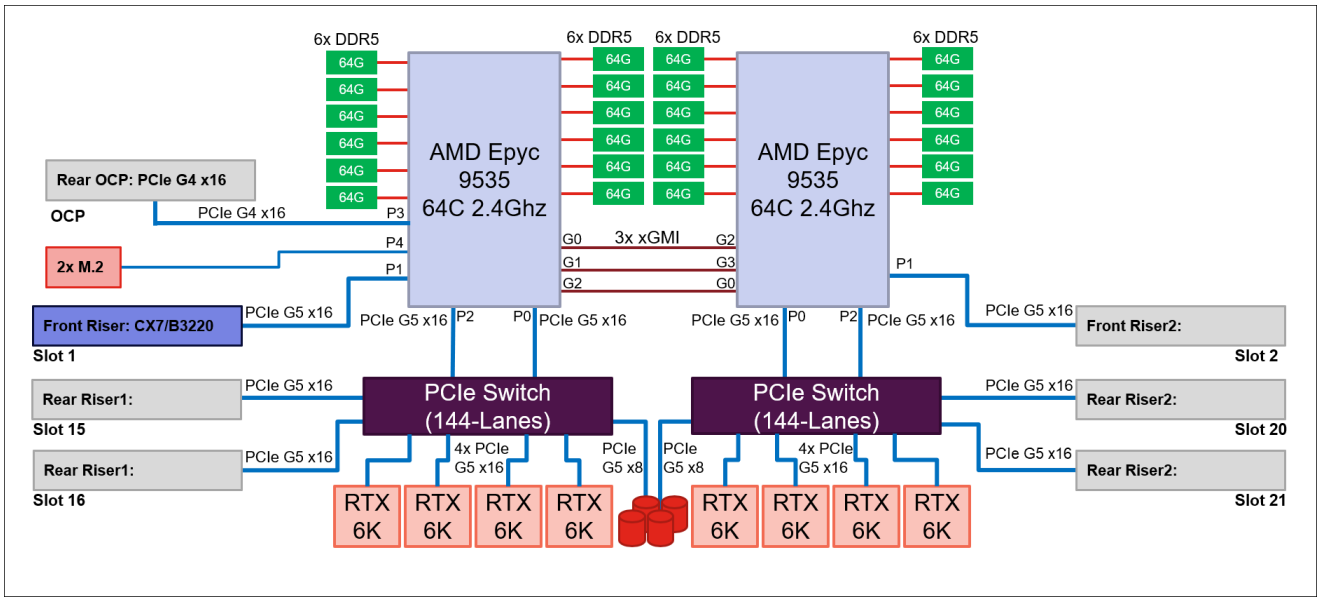


Figure 6: AI Compute Node Block Diagram

The AI Compute node is configured with two **AMD EPYC 9535 64 Core 2.4 GHz** processors with an all-core boost frequency of 3.5GHz. Besides providing consistently more than 2GHz frequency this ensures that with 4 Multi Instance GPUs (MIG) on 8 physical GPUs there are 2 Cores available per MIG plus a few additional Cores for Operating System and other operations. The GPUs are connected to the CPUs via two PCIe Gen5 switches, each supporting up to four GPUs. With the **RTX PRO 6000 Blackwell Server Edition**, no NVLink bridge is applicable.

For the single North-South NIC, **NVIDIA BlueField-3 BF3220** or **Connect-X 7** can be used depending on the North-South network fabric used. Please refer to Appendix A for more information on configuring the server in DCSC.

Storage

The Hybrid AI 281 IBM is running IBM Storage Scale Developer Edition – in a single-node cluster configuration. While the single-node setup means the cluster lacks the high-availability and parallel I/O benefits of a multi-node GPFS. As part of this validation a 960GB NVMEeboot drive, and 6 x 1.7TB NVMe drives are installed. This storage configuration can change depending on the requirements.

GPU selection

The Hybrid AI 281 platform is designed to handle any of NVIDIA's DW PCIe form factor GPUs including the RTX PRO 6000 Blackwell Server Edition and the H200 NVL.

- **NVIDIA H200 NVL**
  The NVIDIA H200 NVL is a powerful GPU designed to accelerate both generative AI and high-performance computing (HPC) workloads. It boasts a massive 141GB of HBM3e memory, which is

nearly double the capacity of its predecessor, the H100. This increased memory, coupled with a 4.8 terabytes per second (TB/s) memory bandwidth, enables the H200 NVL to handle larger and more complex AI models, like large language models (LLMs), with significantly improved pppperformance. In addition, the H200 NVL is built with energy efficiency in mind, offering increased performance within a similar power profile as the H100, making it a cost-effective and environmentally conscious choice for businesses and researchers.
NVIDIA provides a 5-year license to NVIDIA AI Enterprise free-of-charge bundled with NVIDIA H200 NVL GPUs.

- **NVIDIA RTX PRO 6000 Blackwell Server Edition**
  Built on the groundbreaking NVIDIA Blackwell architecture, the NVIDIA RTX PRO™ 6000 Blackwell Server Edition delivers a powerful combination of AI and visual computing capabilities to accelerate enterprise data center workloads. Equipped with 96GB of ultra- fast GDDR7 memory, the NVIDIA RTX PRO 6000 Blackwell provides unparalleled performance and flexibility to accelerate a broad range of use cases- from agentic AI, physical AI, and scientific computing to rendering, 3D graphics, and video.

For the detailed Bill-of-Material for hardware and software Please refer to Appendix A.

4 Solution Architecture

IBM implemented a full-stack OpenRAG inference appliance on the Lenovo Hybrid AI 281 node. The implementation includes hardware configuration, OS and Docker setup, NVIDIA driver and container runtime, NIM microservices, LiteLLM routing, OpenRAG application services, Langflow orchestration, Langfuse observability, and IBM Storage Scale integration. As described earlier, IBM OpenRAG, NVIDIA VSS, and NVIDIA AI-Q blueprints are available in Langflow, but the validation was done on IBM OpenRAG only.

4.1 IBM OpenRAG

The IBM OpenRAG deployment integrates several layers of IBM and partner AI software into a single validated stack. At the foundation, the node runs Ubuntu Linux with Docker and the NVIDIA container runtime, enabling GPU-accelerated containers. IBM then uses NVIDIA NIM microservices as the model serving layer, providing optimized inference for NVIDIA Nemotron, Llama 3.2, NV-CLIP, and NeMo Retriever models. LiteLLM acts as the OpenAI-compatible model gateway, allowing Langflow and the OpenRAG backend to call models through a unified endpoint without hard-coding NIM ports or URLs.

Above the model layer sits the OpenRAG application platform. This is IBM's RAG and agentic AI runtime, composed of a Next.js frontend, a FastAPI backend, the Langflow visual orchestration engine, and OpenSearch as the vector and text search store. IBM also uses IBM Storage Scale parallel file system, giving the RAG pipeline access to a shared, high-performance data layer for documents, OCR output, and snapshots.

Observability is handled by Langfuse, which ingests traces from Langflow and exposes them through a web UI. IBM rounds out the stack with Docling for document parsing and OCR. The stack includes standard open-source components for supporting infrastructure — PostgreSQL for Langflow and LiteLLM state, and Redis for Langflow's multi-worker job queue. . Together these components form an end-to-end IBM AI software stack that runs entirely on the Lenovo Hybrid AI 281 node while remaining compatible with broader IBM data, security, and storage services.

4.1.1 OpenRAG Ingestion and Retrieval Call Flows

The following diagrams summarize how source content becomes searchable knowledge and how a user query becomes a grounded response.

Ingestion Call Flow

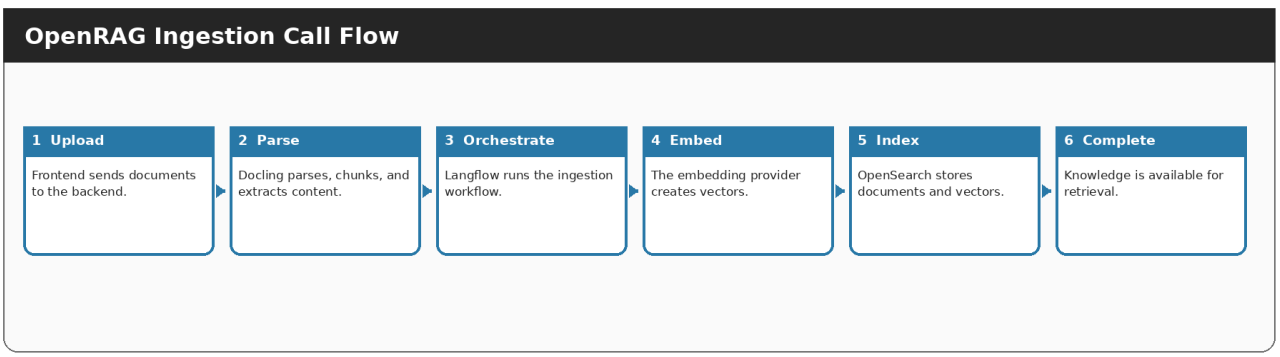


Figure 7: OpenRAG ingestion call flow

1. **Upload.** The frontend submits documents to the backend.
2. **Parse and chunk.** Docling processes the source content.
3. **Orchestrate.** Langflow executes the ingestion flow.
4. **Embed.** The configured embedding provider creates vectors.
5. **Index.** OpenSearch stores documents, embeddings, and metadata.
6. **Complete.** The indexed content becomes available to the knowledge base.

Retrieval and Answer Generation Call Flow

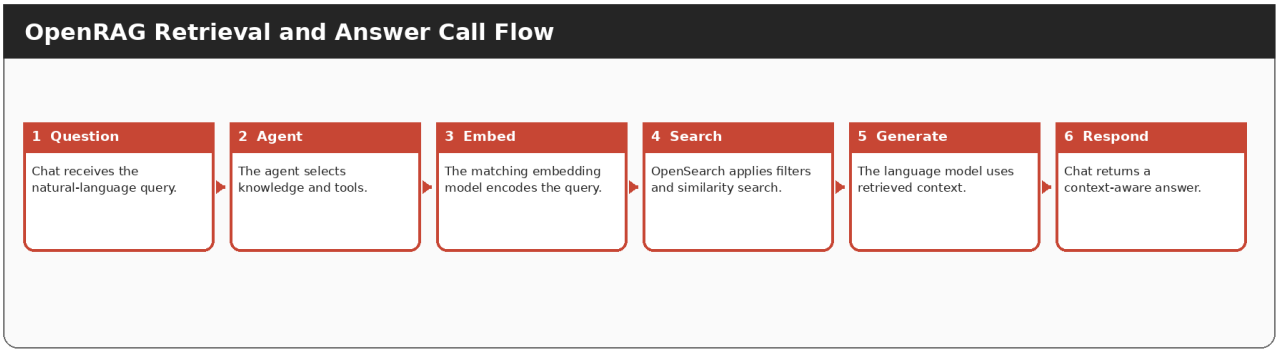


Figure 8: OpenRAG retrieval and answer generation call flow

1. **Query.** Chat receives the user question and active knowledge filters.
2. **Agent orchestration.** The OpenRAG OpenSearch Agent decides which knowledge and tools to use.
3. **Query embedding.** The matching embedding model encodes the query.
4. **Retrieval.** OpenSearch applies filters and similarity search to return relevant content.
5. **Generation.** The configured language model uses the retrieved context.
6. **Response.** The frontend returns the context-aware answer and exposes tool-call details for inspection.

4.2 Langflow

Langflow is a visual framework for building and executing AI workflows. It provides a drag-and-drop interface for connecting components such as chat inputs, LLM calls, vector stores, retrievers, rerankers, prompt nodes, and custom components.

Langflow capabilities used:

- **Visual flow design**: Workflows are built by connecting nodes — chat input, prompt templates, LLM calls, retrievers, vector stores, tools, and custom components — on a drag-and-drop canvas. Each flow is stored as JSON and can be versioned.

- **OpenAI-compatible model calls**: LLM nodes call LiteLLM using standard OpenAI client semantics. This keeps model invocation uniform regardless of which physical NIM serves the request.
- **Vector search**: OpenSearchVectorStore nodes perform k-NN vector search and BM25 text search over the documents index, returning ranked chunks for grounding.
- **Embedding generation**: EmbeddingModel nodes call the EMBED alias through LiteLLM, which routes to the NV-EmbedQA NIM.
- **Reranking**: RerankModel nodes re-score retrieved chunks through the RERANK alias, improving result ordering before context is passed to the LLM.
- **Custom components**: IBM extended Langflow with OpenRAG-specific nodes for chat, document ingestion, URL crawling, nudges, and IBM authentication.
- **Background workers**: Redis-backed asynchronous workers execute long-running ingestion jobs outside the request path, so uploads do not block the web UI.
- **Observability**: Every flow execution is traced to Langfuse, giving operators per-node latency, token counts, and error visibility.
- **MCP tool support**: Model Context Protocol nodes let a flow invoke other Langflow flows as callable tools, giving OpenRAG's agent-style components a way to delegate to specialized sub-flows (e.g. ingestion, crawling) without hand-wiring each one in.
- **Session and memory**: Chat flows maintain conversation history and session state across multiple user turns.

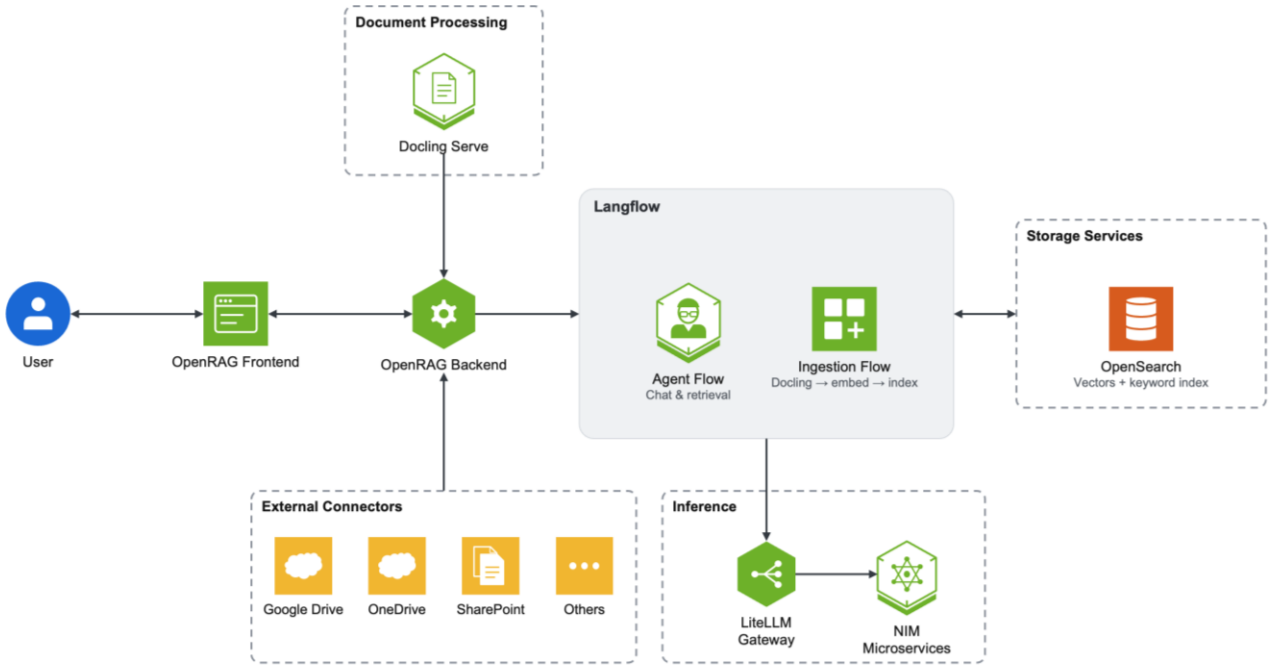


Figure 9: OpenRAG Architectural Diagram

Configured OpenRAG Flows

Langflow key production flows for OpenRAG in Langflow are listed on the table below. As shown in the table below all three OpenRAG, VSS, and AI-Q blueprints are configured.

Table 4: Key Langflow OpenRAG flows

Flow	Purpose	Blueprint
OpenRAG OpenSearch Agent Flow	Main RAG chat agent	OpenRAG
OpenSearch Ingestion Flow	Document chunk, embed, index	OpenRAG
OpenSearch URL Ingestion Flow	Web crawl and ingest	OpenRAG
OpenRAG OpenSearch Nudges Flow	User onboarding nudges	OpenRAG

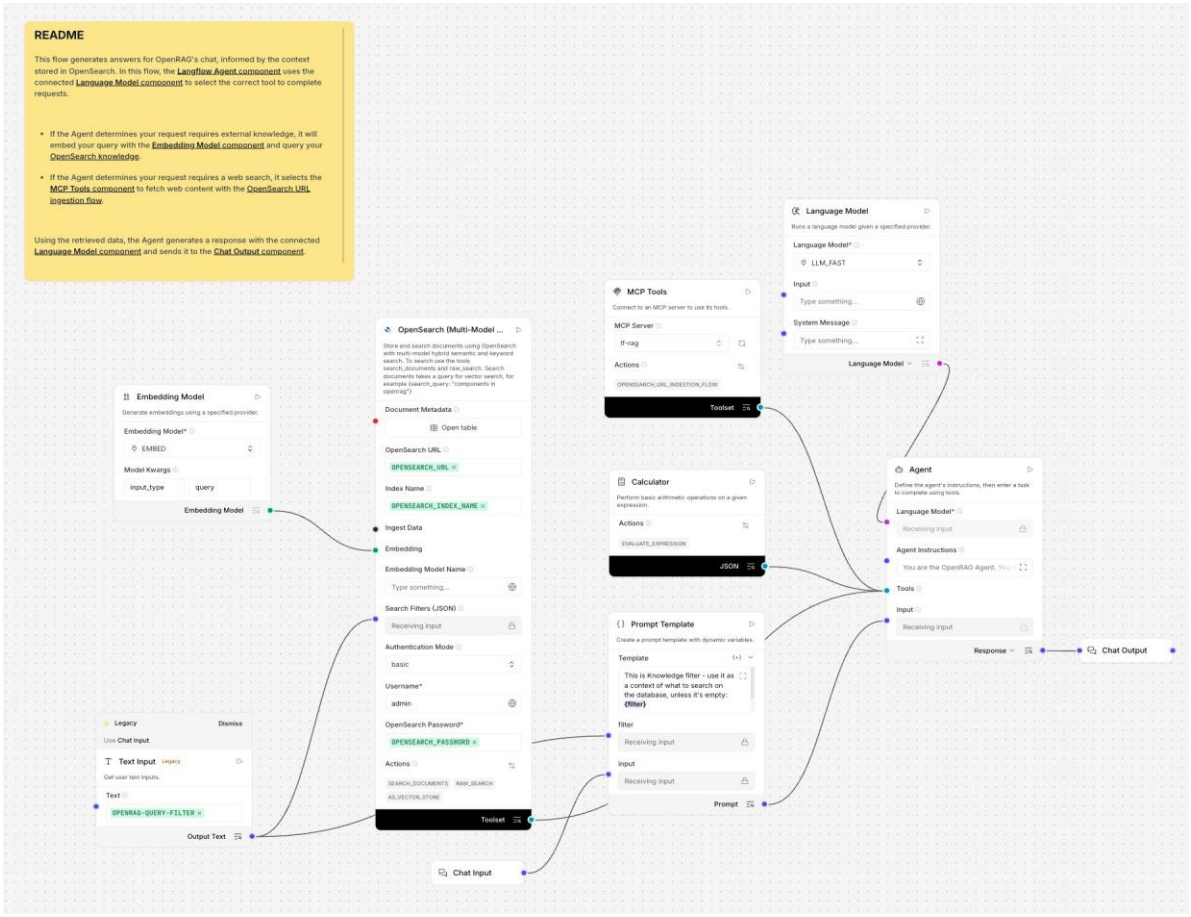


Figure 10: OpenRAG OpenSearch Agent Flow

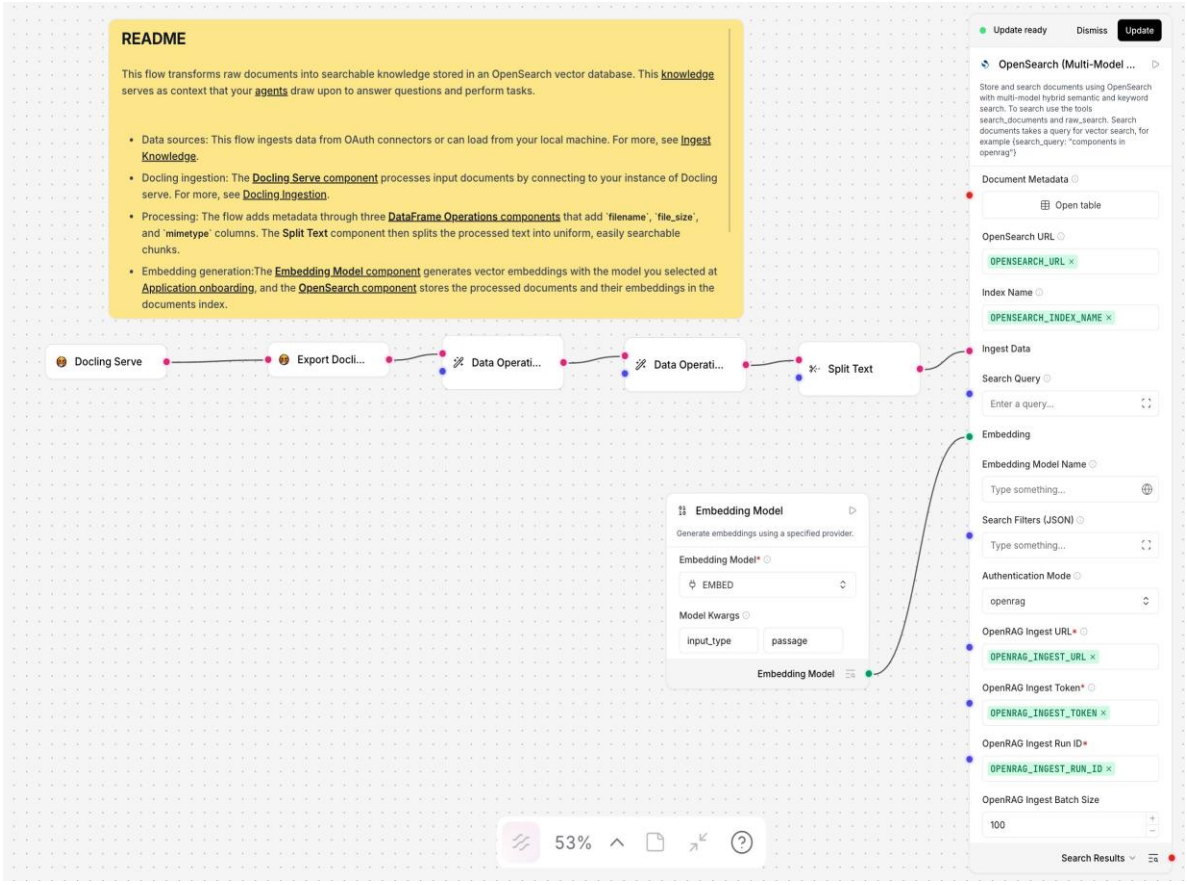


Figure 11: OpenSearch Ingestion Flow

4.3 NVIDIA Blueprints

Langflow also contain flows to two NVIDIA blueprints, Video Search and Summarization (VSS) and AI-Q blueprints. VSS blueprint provides the video intelligence application, while AI-Q blueprints provides agentic research application, both are built in Langflow.

4.3.1 NVIDIA VSS Blueprint in Langflow

Langflow is a visual framework for building and executing AI workflows. It provides a drag-and-drop interface for connecting components such as chat inputs, LLM calls, vector stores, retrievers, rerankers, prompt nodes, and custom components.

Langflow capabilities used:

- **Visual flow design**: Workflows are built by connecting nodes — chat input, prompt templates, LLM calls, retrievers, vector stores, and custom components — on a drag-and-drop canvas. Each flow is stored as JSON and can be versioned.
- **OpenAI-compatible model calls**: LLM and VLM nodes call LiteLLM using standard OpenAI client semantics, keeping model invocation uniform regardless of which physical NIM serves the request.

- **Multimodal reasoning**: VLM nodes call the OMNI alias, which routes to the Omni Reasoning NIM for video captioning, event extraction, and alert verification across ingested footage.
- **Embedding generation**: EmbeddingModel nodes call the IMG_EMBED alias for keyframe vectors and the EMBED alias for caption and metadata text, giving each video both a visual and textual retrieval path.
- **Vector search:** OpenSearchVectorStore nodes perform k-NN vector search and BM25 text search over keyframe and caption embeddings, returning ranked video segments for grounding.
- **Reranking:** RerankModel nodes re-score retrieved segments through the RERANK alias, improving result ordering before a summary or answer is generated.
- **Custom components:** IBM extended Langflow with VSS-specific nodes for video ingestion, keyframe extraction, and batch clip processing.
- **Background workers:** Redis-backed asynchronous workers execute long-running video ingestion and processing jobs outside the request path, so uploads do not block the web UI.
- **Observability:** Every flow execution is traced to Langfuse, giving operators per-node latency, token counts, and error visibility.
- **Session and memory:** Chat flows maintain conversation history and session state across multiple follow-up questions about the same video or clip.

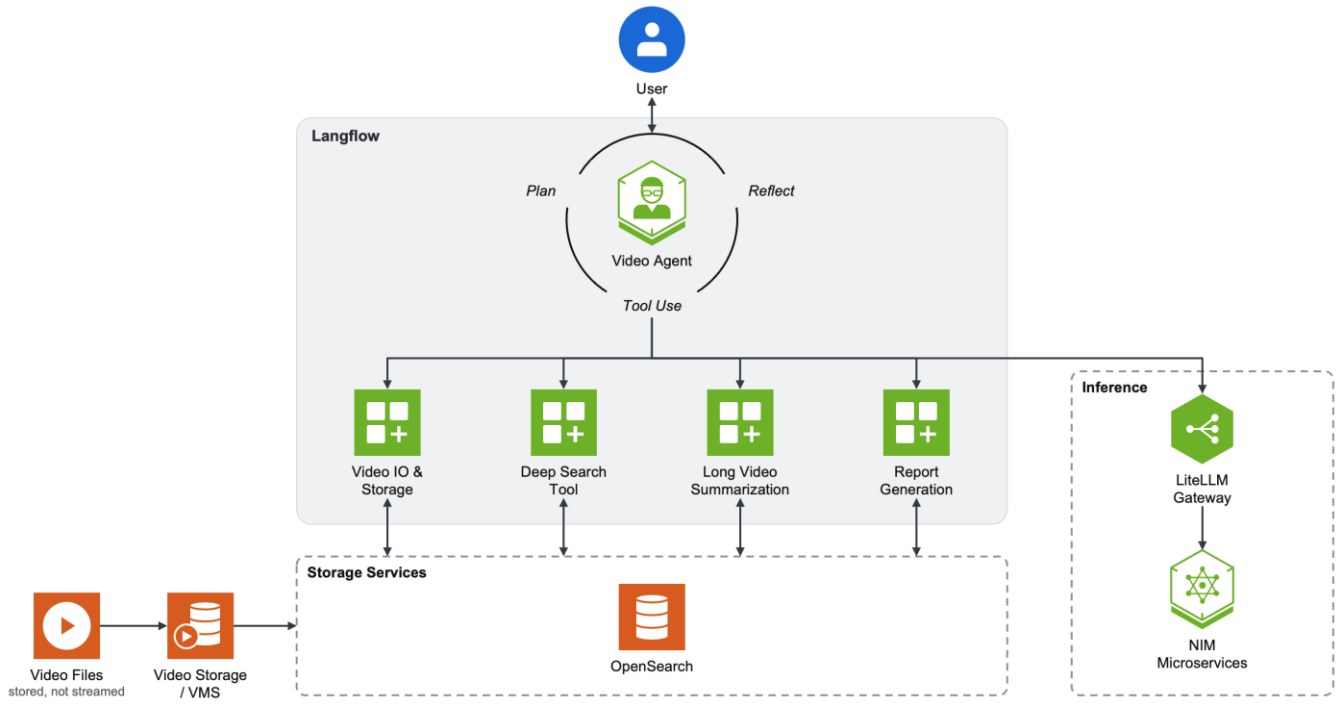


Figure 12: VSS Architectural Diagram, built in Langflow. See the 289 configuration for VSS with streaming video processing capabilities.

Table 5: NVIDIA VSS Blueprint flows

#	Flow	Trigger	Notes
1	F1 Ingest Asset	Webhook / file	Writes clip manifest, returns immediately
2	F1a Ingest via AFM watch	Cron / filesystem poll	Scans AFM cache fileset for new assets and enqueues them
3	F2 Process Clip	API (`clip_id`)	Retriable atomic claim + lease
4	F3 Reconciler	Cron	Claims pending/expired clips, dispatches F2
5	F4 Search	Chat / API	BM25 over captions + kNN over vectors
6	F5 Summarize Asset	API	Temporal deduplication + LLM aggregation
7	F6 Vision Agent	Chat	Single agent, sub-flows as tools
8	F7 Verify Alert	Webhook	Verdict + criteria_met map
9	F8 Apply Profile	API	Profile CRUD
10	F9 Delete Asset	API	Object delete + delete_by_query
11	F10 Bootstrap	Build time	Index creation, ISM policy, sample data pre-warm

4.3.2 NVIDIA AI-Q Blueprint in Langflow

Langflow is a visual framework for building and executing AI workflows. It provides a drag-and-drop interface for connecting components such as chat inputs, LLM calls, vector stores, retrievers, rerankers, prompt nodes, and custom components.

Langflow capabilities used:

- **Visual flow design:** Workflows are built by connecting nodes — chat input, prompt templates, LLM calls, retrievers, vector stores, tools, and custom components — on a drag-and-drop canvas. Each flow is stored as JSON and can be versioned.
- **OpenAI-compatible model calls:** LLM nodes call LiteLLM using standard OpenAI client semantics. Task-routing logic within the flow selects between the LLM_FAST alias — intent routing, tool-calling, and shallow research steps — and the LLM_DEEP alias for planning, clarification, and final writer synthesis.

- **MCP tool support:** Model Context Protocol nodes expose external tools — web search, code execution, custom enterprise tools — to the research flow, letting the agent reach outside Langflow entirely as part of a multi-step research plan.
- **Vector search:** OpenSearchVectorStore nodes perform k-NN vector search and BM25 text search over the documents index, surfacing grounding sources during research steps.
- **Embedding generation:** EmbeddingModel nodes call the EMBED alias to index and retrieve candidate source material for a research task.
- **Reranking:** RerankModel nodes re-score retrieved sources through the RERANK alias before they're handed to the planning and writing steps.
- **Custom components:** IBM extended Langflow with AI-Q-specific nodes for research planning, sub-task orchestration, and report/citation synthesis, matching the blueprint's own agentic architecture.
- **Background workers:** Redis-backed asynchronous workers execute long-running, multi-step research tasks outside the request path, so a user's request returns immediately while the agent continues working.
- **Observability:** Every flow execution is traced to Langfuse, giving operators per-node latency, token counts, and error visibility.
- **Session and memory:** Research flows maintain conversation hcross multiple clarification turns.

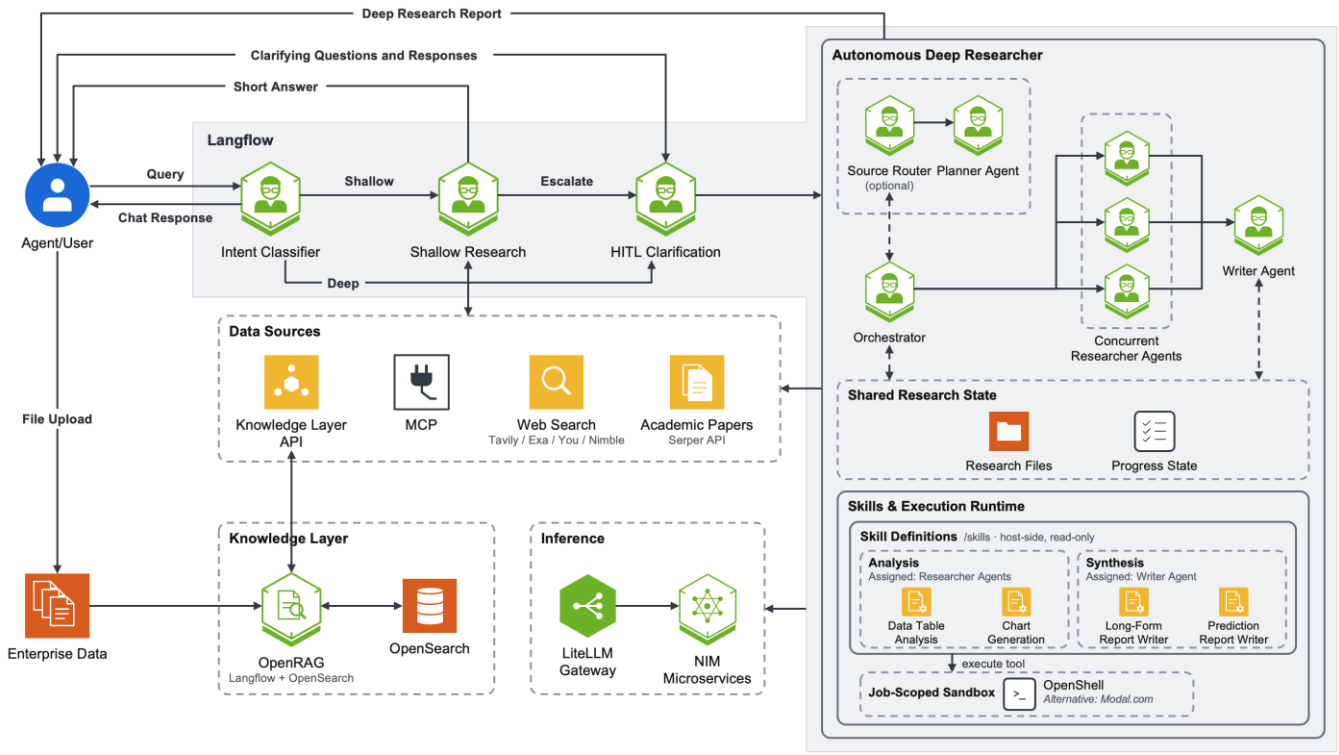


Figure 13: AI-Q Architectural Blueprint, built in Langflow. This configuration supports prototyping Deep Research. See the 289 configuration for production ready Deep Research.

Table 6: NVIDIA AI-QAI-Q Blueprint flows in Langflow

#	Flow	Trigger	Notes
1	Q1 Orchestrator / Intent Router	Chat	Smart Router: conversational / shallow / deep
2	Q2 Shallow Research	Sub-flow	Bounded, tool-augmented, cited
3	Q3 Clarifier	Sub-flow	Optional disambiguation turn
4	Q4 Source Router	Sub-flow	Selects among web, OpenSearch, video, MCP
5	Q5 Research Planner	Sub-flow	Structured plan → worker tasks
6	Q6 Researcher Worker	Sub-flow (×N)	Parallel research worker
7	Q7 Writer / Synthesis	Sub-flow	Report assembly with citations
8	Q8 Report Follow-up	Chat	Q&A / rewrite over completed report

4.4 Docling Document Parsing Service

Docling is the document pre-processing layer for the OpenRAG appliance. It converts PDFs, scanned images, Word files, and other common document formats into structured markdown with rich metadata, making unstructured enterprise content usable by the retrieval pipeline. Docling provides:

- **Layout-aware extraction**: Docling preserves document structure — headings, paragraphs, lists, tables, and figures — instead of returning a flat stream of text. This improves chunk quality and retrieval accuracy.
- **OCR for scanned content**: When a document is image-based or contains scanned pages, Docling runs OCR and produces searchable, structured markdown.
- **Table detection and extraction**: Tables are recognized as first-class objects and exported in a machine-readable format that downstream flows can query.
- **Figure and chart description**: For diagrams, charts, and images, Docling can invoke a VLM pipeline to generate textual descriptions that can also be indexed.
- **Pipeline selection**: Operators can choose fast or quality pipelines per document type to balance throughput and extraction fidelity.

4.5 LiteLLM

LiteLLM acts as an OpenAI-compatible router that abstracts the nine NIM microservices behind a single endpoint. LiteLLM handles model selection, fallback, retries, routing, and provides model aliases and load balancing across the NVIDIA NIMs utilized in the Blueprint builds.

Table 7: LiteLLM model aliases

Alias	Physical Model	Port	Purpose
LLM_FAST	NVIDIA Nemotron 3.5 Lightning 30B	:8010	Fast chat and query classification
LLM_CHAT	NVIDIA Nemotron 3 Super 120B	:8011	
LLM_DEEP	NVIDIA Nemotron 3 Super 120B	:8011	Deep reasoning and answer generation, aliases to LLM_CHAT on this tier
OMNI	NVIDIA Nemotron 3 Nano Omni 30B	:8012	Multimodal reasoning
EMBED	Llama 3.2 NV-EmbedQA 1B v2	:8013	Text embeddings
RERANK	Llama 3.2 NV-RerankQA 1B v2	:8014	Reranking
IMG_EMBED	NV-CLIP ViT-H/14	:8015	Image embeddings
EXTRACT_PAGE_ELEMENTS	NeMo Retriever page-elements-v3	:8016	Page layout and region detection for document ingestion
EXTRACT_TABLE_STRUCTURE	NeMo Retriever table-structure-v1	:8017	Table structure extraction for document ingestion
EXTRACT_GRAPHIC_ELEMENTS	NeMo Retriever graphic-elements-v1	:8018	Chart and figure detection for document ingestion
EXTRACT_OCR	NeMo Retriever OCR	:8019	Scene-text OCR for document ingestion

4.6 Langfuse Observability

Langfuse collects traces from Langflow and application services so operators can inspect workflow steps, model calls, latency, token use, errors, and retrieved context. The deployment uses PostgreSQL for metadata, ClickHouse for analytics, Redis for background processing, and MinIO for object storage. Production deployments should define access control, trace retention, sensitive-data handling, storage capacity, and backup requirements

4.7 OpenSearch Vector Database

OpenSearch is the vector database used for OpenRAG. It stores document chunks, dense embeddings, sparse text indexes, and metadata used by the RAG and agentic research flows. OpenSearch provides:

- **Dense vector search (k-NN)**: Embeddings generated by the EMBED NIM are indexed using DiskANN, a disk-efficient approximate nearest-neighbor algorithm, enabling semantic search over document chunks at scale without requiring the full vector index to fit in memory.
- **Text search (BM25)**: Traditional lexical scoring complements vector search for exact term matching.
- **Hybrid retrieval**: Langflow flows can combine BM25 and k-NN results, then rerank with the RERANK NIM for better relevance.
- **Metadata filtering**: Per-user JWT scoping restricts document access so multi-tenant deployments can enforce row-level security.
- **Aggregations and analytics**: OpenSearch Dashboards on port 5601 provides a UI for exploring index contents, query patterns, and cluster health.

4.8 ContextForge

ContextForge provides a centralized gateway and control plane for connecting AI agents and applications to enterprise tools, APIs, services, and other agents. ContextForge adds value by federating capabilities exposed through MCP, A2A, REST, and gRPC, then presenting them through a consistent, governed interface that can be reused across multiple agent frameworks and applications. Rather than requiring each Langflow flow or custom application to integrate independently with every backend system, ContextForge can centralize service discovery, routing, authentication, authorization, policy enforcement, rate limiting, observability, and protocol translation. It can also create curated virtual MCP servers that expose only the tools appropriate for a particular workload, business unit, or security boundary. This makes it particularly valuable in larger AI Factory environments where many agents need controlled access to a growing ecosystem of enterprise capabilities, while reducing duplicated integration work and providing a consistent governance layer between the agent application tier and the underlying services.

4.9 GPU Allocation Strategy

In Lenovo Hybrid AI 281 with IBM, the configured the NIM microservices can be deployed in four of the eight

GPUs. Lenovo recommends a fully populated server with 8 x RTX Pro 6000 Blackwell Server Edition GPUs. This configuration will provide enough headroom for additional models or workload expansion. The allocation strategy reflects the inference-only, single-node nature of the Lenovo Hybrid AI 281 platform: maximize throughput for RAG and agentic AI workloads.

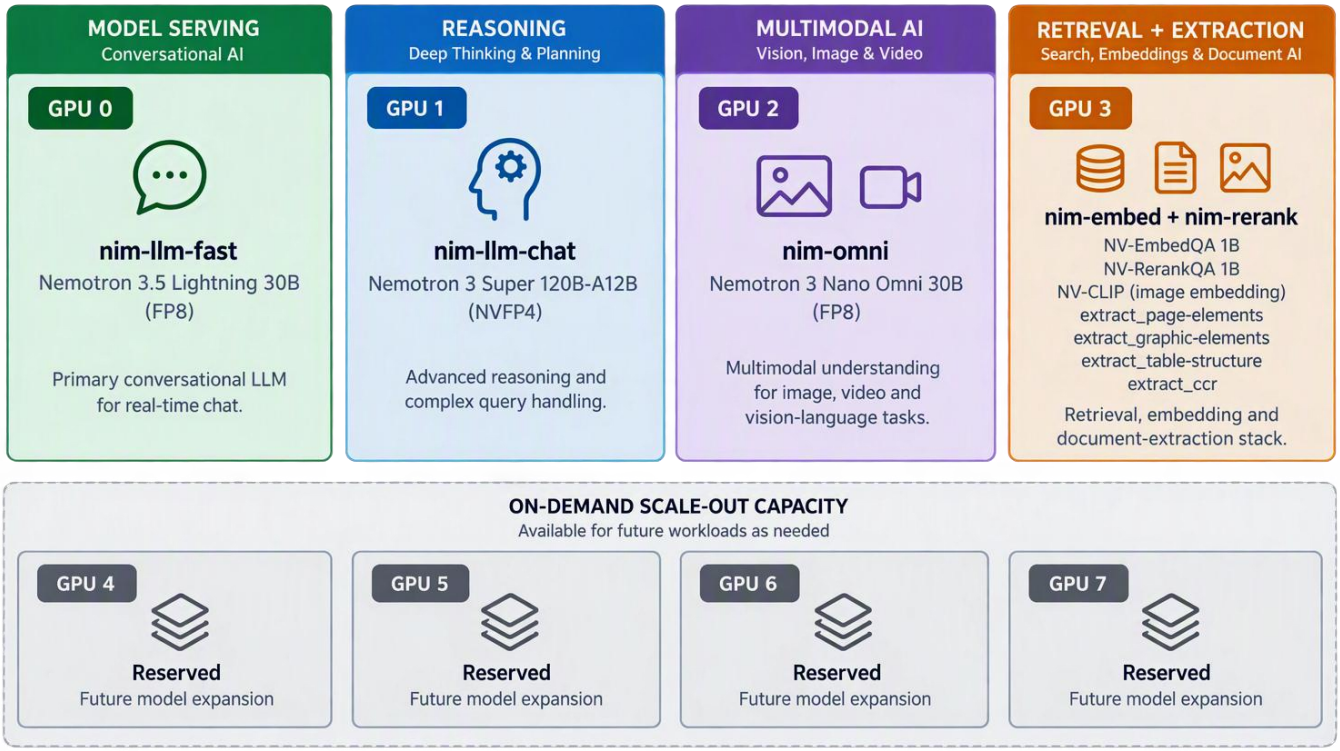


Figure 14: GPU Allocations example over 4 GPUs

GPU 0 hosts nim-llm-fast, the main conversational LLM — NVIDIA Nemotron 3.5 Lightning 30B, served at FP8 precision. It runs on a dedicated GPU, with no tensor-parallel split, consistent with the one-model-one-GPU design used throughout this build — this is the busiest model in the chat path, and isolating it keeps conversational latency predictable.

GPU 1 hosts nim-llm-chat — Nemotron 3 Super 120B-A12B, served at NVFP4 precision. This is a dedicated model for deep reasoning and planning, distinct from the fast-path model on GPU 0 — it's also NVIDIA's own reference chat model for the RAG blueprint, and it benchmarks meaningfully ahead on reasoning tasks like graduate-level scientific QA. Weights occupy roughly 80GB of the card's 96GB, leaving about 16GB for KV-cache and CUDA graph buffers — generous headroom, helped by this model's hybrid Mamba-2 layers, which keep KV-cache nearly flat as context length grows rather than scaling per-token the way standard attention does.

GPU 2 hosts the multimodal model, NVIDIA Nemotron 3 Nano Omni 30B Reasoning, served at FP8 precision. It handles vision-language tasks — scene captioning, video-frame understanding, visual question answering — on its own dedicated GPU, with no tensor parallelism required. Isolating it here keeps image and video inference from contending with text-only token generation on the other GPUs, protecting the conversational pipeline's latency.

GPU 3 hosts the full embedding, retrieval and document-extraction stack: NV-EmbedQA 1B and NV-

RerankQA 1B (the retrieval path used on every chat and ingestion query), NV-CLIP for image embedding, and the four NeMo Retriever document-structure models — page-elements, graphic-elements, table-structure, and OCR. These seven models run comfortably together on one GPU, using roughly 60GB of the card's 96GB and leaving about 35GB free.

GPUs 4 through 7 are reserved for expansion rather than pre-allocated to replicas the system may not need — the box ships with this minimal footprint and grows into the free capacity as real concurrent load demands it. LLM-FAST is best positioned to use that headroom efficiently: at NVFP4 precision, its checkpoint drops to 22GB, so a single reserved GPU can host two replicas rather than one (including GPU 0), doubling the throughput gained from each additional card brought online. The same reserved capacity is available for OMNI replicas, additional retrieval throughput, or customer-specific models — scaled in only as the workload actually requires it.

5 Retrieval-Augmented Generation (RAG) Functional Test

The functional test validates end-to-end ingestion, retrieval, and question answering through the OpenRAG web interface and the underlying Langflow flows.

5.1 Access Front End

Open a browser and navigate to the OpenRAG frontend at `http://<node-ip>:3000`. The landing page presents the Knowledge Base view, which lists the collections and documents currently indexed in OpenSearch. From the same interface the user can start a chat, configure connectors, and upload documents.

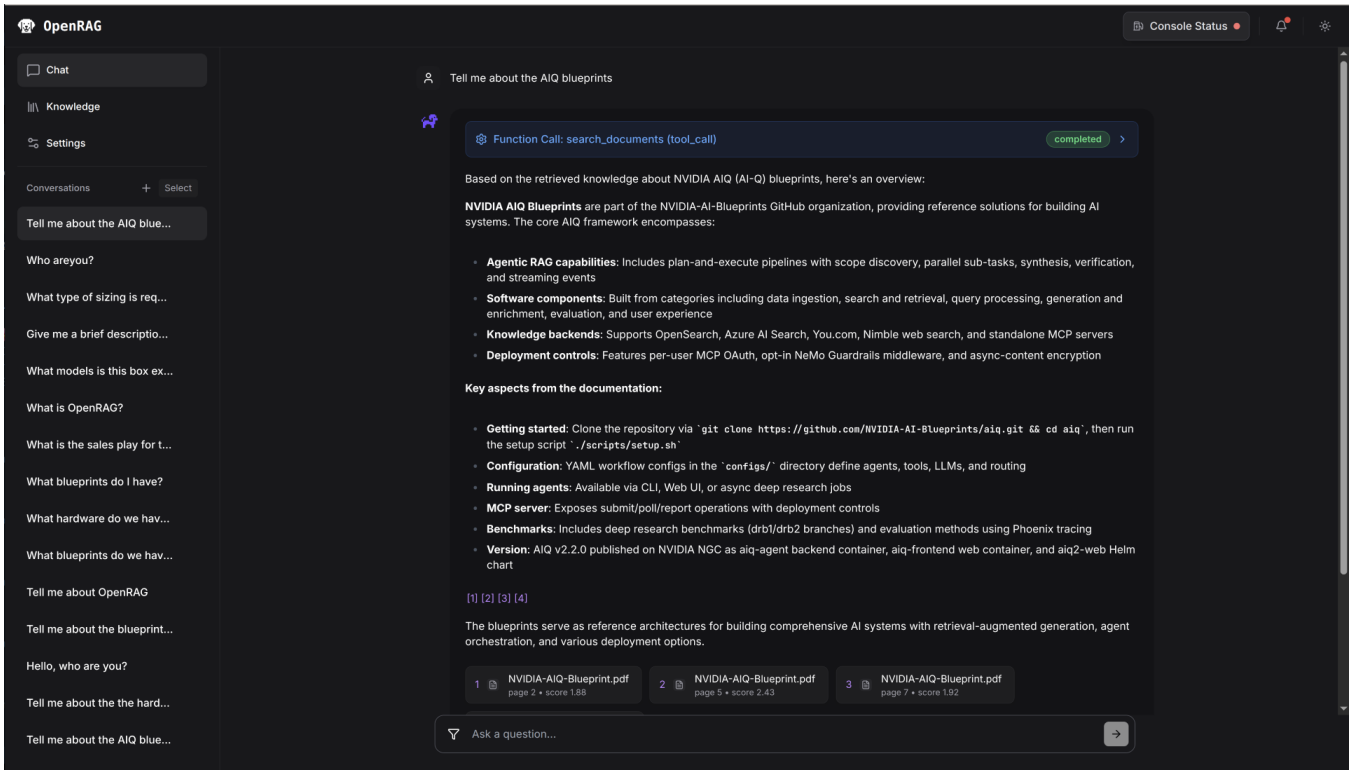


Figure 15: OpenRAG frontend and chat interface

5.2 Document Ingestion

From the OpenRAG frontend, select the upload control and choose one or more supported documents (PDF, Word, PowerPoint, scanned images, or plain text).

The OpenRAG backend streams the file to the Docling service on port 5001. Docling parses the document structure, runs OCR on scanned pages, extracts tables and figures where configured, and returns structured markdown.

The ingestion flow chunks the markdown, calls the EMBED alias through LiteLLM to generate dense vectors, and writes the chunks plus embeddings to the OpenSearch index on port 9200.

The frontend shows the indexed document in the Knowledge Base list once the background job completes.

For validation, several sample PDFs containing technical product information were ingested. The ingestion jobs completed asynchronously through the Redis-backed Langflow worker queue, and the resulting chunks were visible in OpenSearch Dashboards.

Document ingestion options

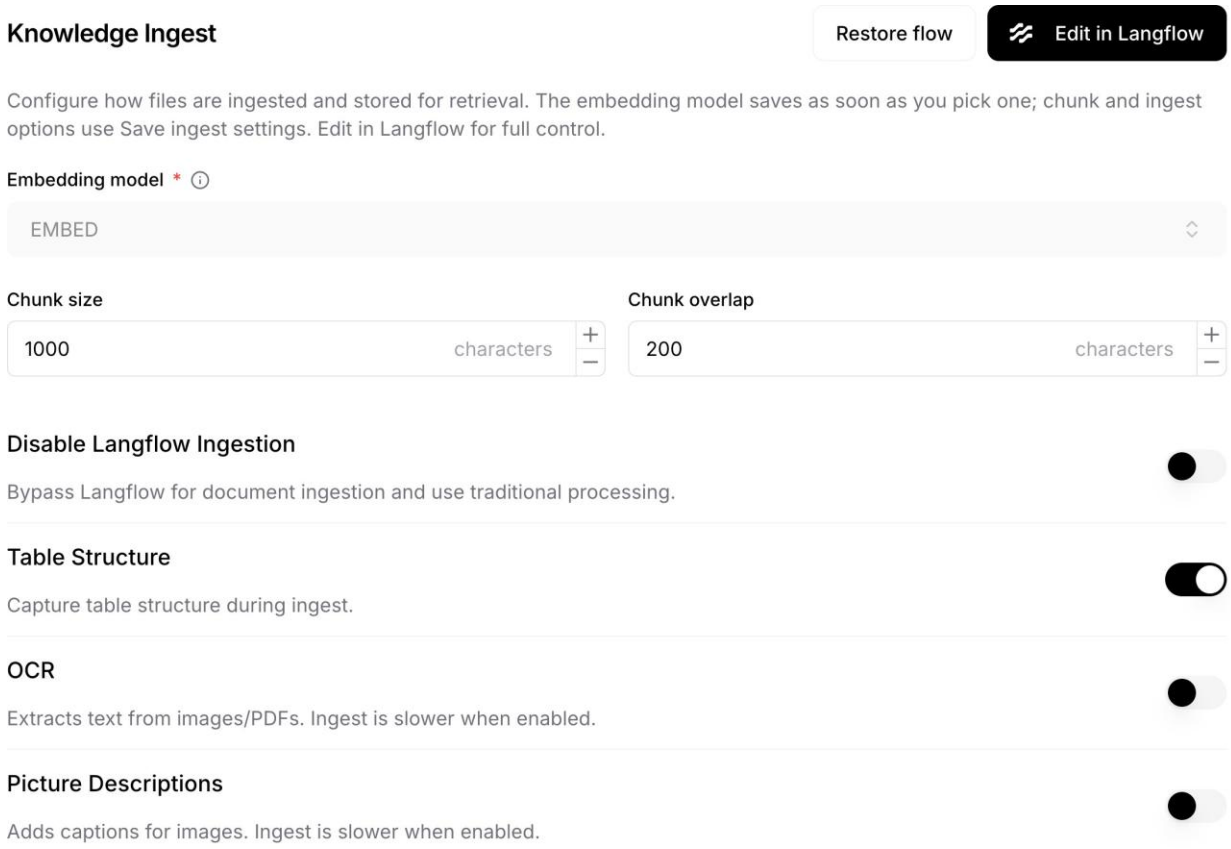


Figure 16: Knowledge Ingest

Knowledge Base

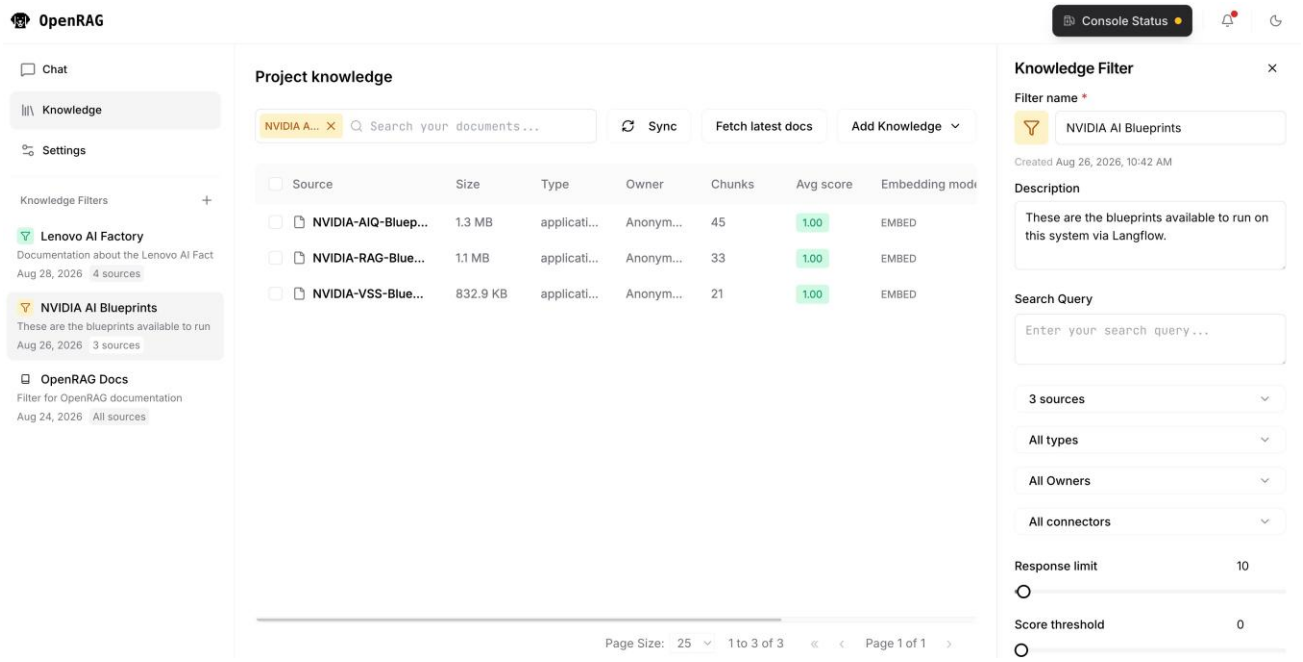


Figure 17: Knowledge Base

5.3 Document Retrieval and Query

From the Chat view, enter a natural-language question related to the uploaded document.

The OpenRAG backend routes the request to the OpenRAG OpenSearch Agent Flow in Langflow.

The flow embeds the query via the EMBED alias, performs hybrid search over the OpenSearch index, reranks the candidate chunks via the RERANK alias, and assembles a prompt that includes the top-ranked chunks as grounding context.

The flow sends the prompt to the LLM_DEEP alias (Nemotron 3 Super 120B) and streams the response back to the frontend.

The response includes citations drawn from the retrieved chunks, confirming that the answer is grounded in the uploaded documents rather than the model's pre-training data.

Operators can open Langfuse at port 3001 to inspect the full trace, including query embedding, retrieval latency, reranking scores, token counts, and the final LLM response.

Document Retrieval and Query interface can be seen in Figure 15.

6 Production Considerations

The validated configuration demonstrates the OpenRAG, VSS and AI-Q architectures on a single server. Customer pre-production use requires workload sizing, security hardening, external backup and restore procedures, operational monitoring, and acceptance of the single-node failure domain. VSS and AI-Q require separate validation before deployment.

6.1 Security and data protection

The deployment should expose only the customer-facing application endpoint and explicitly approved administrative interfaces. Databases, Redis, ClickHouse, MinIO, OpenSearch, Langflow, Langfuse, ContextForge and NIM endpoints should remain on isolated networks unless remote access is required and protected. TLS should be used to protect external application and administrative traffic, and production certificates should be issued through the customer's approved public key infrastructure. When emitting telemetry, sensitive information - such as production tokens - should not be included in traces.

Credentials such as the NVIDIA NGC API key, database passwords, LiteLLM keys, JWT signing material, and object-storage credentials must be set in Langflow as Credentials or provided through service-scoped secret files or an enterprise secret manager. Secrets must not be embedded in container images, committed to source control, or written to application logs. Special care must be taken when enabling AI agents to access secure or sensitive resources, traverse network boundaries, or perform operations (especially deletes and modifications) against production systems, and special care must be taken when granting access permissions to AI agents to access data sources that could contain sensitive information (including information that may have been inadvertently shared.) We recommend a zero-trust architecture when scoping authorization for AI agents and including the use of tenant-based isolation to restrict access to Langflow instances that may access sensitive resources.

In the validated OpenRAG runtime configuration, enterprise content and inference requests can remain within the customer-managed environment. Initial installation and updates may require approved external connectivity for container images and model artifacts. Optional web search, URL ingestion, connectors, and agent tools introduce additional data paths and must be separately governed.

Document authorization must be enforced in the retrieval path. The application should derive authorized tenant, collection, or document identifiers from the validated user identity and apply mandatory filters to both lexical and vector retrieval. Direct API testing should confirm that users cannot retrieve another user's document chunks, metadata, filenames, citations, or traces.

ContextForge may be used to secure AI agent tool calls, MCP, A2A, and other agentic access but should be used in combination with other security best practices. A full set of security recommendations, as well as recommendations for running in an air-gapped environment, are beyond the scope of this guide.

6.2 Observability and operations

When using Langflow, Langflow automatically captures traces of some — but not all — metrics needed for production. For more complete observability, Langfuse captures OpenTelemetry traces emitted from Langflow, giving operators per-node latency, token counts, and error visibility.

NIM liveness and readiness endpoints should be integrated with container health checks — each NIM

exposes its own health endpoints natively, and container orchestration's restart behavior depends on checking these directly. LiteLLM, as the shared gateway in front of every NIM, adds a second layer on top: its own health-check loop can poll each configured model on an interval and automatically remove a failing one from the routing pool, giving operators a single aggregate health view instead of nine separate endpoints to watch. Prometheus-compatible NIM metrics can provide request latency, throughput, queue depth, token counts, and GPU utilization. NVIDIA DCGM or an equivalent supported monitor should collect GPU health, memory, temperature, power, driver, and error telemetry. Operations should also monitor host resources, filesystem capacity, OpenSearch health, database availability, queue depth, failed ingestion jobs, and certificate expiration.

Table 8: Observability and Operations

Component	Minimum operational checks
NIM services	Liveness, readiness, request errors, latency, queue depth, and token throughput
GPU infrastructure	Utilization, memory, temperature, power, driver health, and device errors
OpenSearch	Cluster state, heap pressure, disk watermarks, shard state, and query latency
Storage	Mount state, capacity, inode use, I/O errors, and backup status
Application services	Container health, database connections, queue depth, failed jobs, and trace completion

6.3 Backup and recovery

The single-node design does not provide node-level high availability. A server, operating-system, storage, or network failure can interrupt all application services. OpenSearch indexes and cluster state should be protected with scheduled snapshots stored outside the local-node failure domain. PostgreSQL data, Langflow flows, OpenRAG configuration, Langfuse and ClickHouse data, MinIO objects, certificates, deployment manifests, and secrets require component-appropriate backup procedures and periodic restore testing. Langflow flows should be either integrated with a CI/CD tool to automatically commit changes to revision control, or flow backups should be created and securely stored whenever substantive changes are made to flows.

6.4 Validation scope and acceptance criteria

We strongly recommend preparing a golden set of question and answer pairs (curated by subject matter experts) for use when benchmarking performance and behavior of AI agents and to establish a baseline for measuring the effects of changes to agents (including their tools, flows, and other configuration parameters).

The best practice when benchmarking the results of vector search is to compare the results of an approximate nearest neighbor (ANN) vector search against the results of an exact nearest neighbor search (KNN) to check recall, precision, and other vector search accuracy metrics. We also recommend using LLM-as-a-judge architectures to objectively evaluate RAG pipelines against golden answers.

Langfuse directly supports the golden-set and LLM-as-a-judge workflows above — its dataset feature lets you curate and version a golden Q&A set, run experiments against it to benchmark agent changes over time, and score results using either Langfuse's built-in judge catalog or a custom judge prompt evaluated against the golden answers.Vector search recall and precision are validated separately, at the OpenSearch layer, by running the same golden-set queries through both the approximate (DiskANN) index and OpenSearch's exact k-NN scoring script as ground truth.

The following metrics are a useful starting point:

Table 9: Validation scope and acceptance criteria

Validation area	Minimum evidence
Functional	Supported-format ingestion, document deletion and re-indexing, retrieval, reranking, citations, and error handling
RAG quality	Representative query set, retrieval relevance, context recall, answer faithfulness, citation correctness, and abstention
Performance	Time to first token, end-to-end latency, token throughput, concurrent sessions, ingestion rate, and mixed workload results
Resiliency	Container restart, server reboot, interrupted-job recovery, capacity alerting, backup, and restore
Security	TLS verification, exposed-port review, role tests, direct-API data-isolation tests, secret scanning, and egress review

We also recommend red-teaming developed agents to verify that their implementations (including prompts, tool calls, etc.) are acceptable for the requirements of your policies, as needed. Langfuse supports curating adversarial test cases — prompt-injection attempts, policy-violation probes, trick questions — as a Langfuse dataset and can score the results with its eval/judge features.

6.5 Performance Considerations

We recommend utilizing the NVIDIA NVFP4 configuration to improve latency, memory, and cost of operating models using Blackwell RTX 6000 PRO. Tuning of additional parameters may be required to fit your workload and maximize resource utilization.

6.6 Capacity planning and service objectives

The following metrics reflect the results from testing conducted by Lenovo and IBM. Individual results may vary. When reproducing benchmarks or capacity planning, it is imperative to evaluate the complete system rather than GPU model behavior alone. Such testing may include, but should not be limited to, reflecting the actual or simulated expected workload and should evaluate parameters such as the following: context length, prompt and output size, batch behavior, concurrency, KV-cache use, ingestion load, OpenSearch heap, host

memory, storage growth, and network utilization.

Table 10: Capacity planning an service objectives

Measure	Objective	Measured result	Test conditions
Time to first token, p50 / p95	[ENTER]	[ENTER]	[Model profile, precision, context, concurrency]
End-to-end response latency, p50 / p95	[ENTER]	[ENTER]	[Query set, retrieval settings, output length]
Output throughput	[ENTER tokens/s]	[ENTER]	[Model and concurrent sessions]
Concurrent chat sessions	[ENTER]	[ENTER]	[Latency/error threshold]
Document-ingestion throughput	[ENTER pages/min]	[ENTER]	[File types, OCR ratio, average page complexity]
Maximum validated corpus	[ENTER documents/pages/GB]	[ENTER]	[Chunking and embedding settings]
Availability target	[ENTER]	[ENTER]	Single-node limitation acknowledged
Capacity alert thresholds	[ENTER CPU, RAM, VRAM, disk, queue, and heap thresholds]	[ENTER]	[Monitoring interval and duration]

7 Appendix A: Bill of Material (BoM)

For more information on the Bill of Material (BoM), please refer to Lenovo Hybrid AI 285 Platform Guide.

7.1 Configuring Hybrid AI 281 Canonical Solution in DCSC

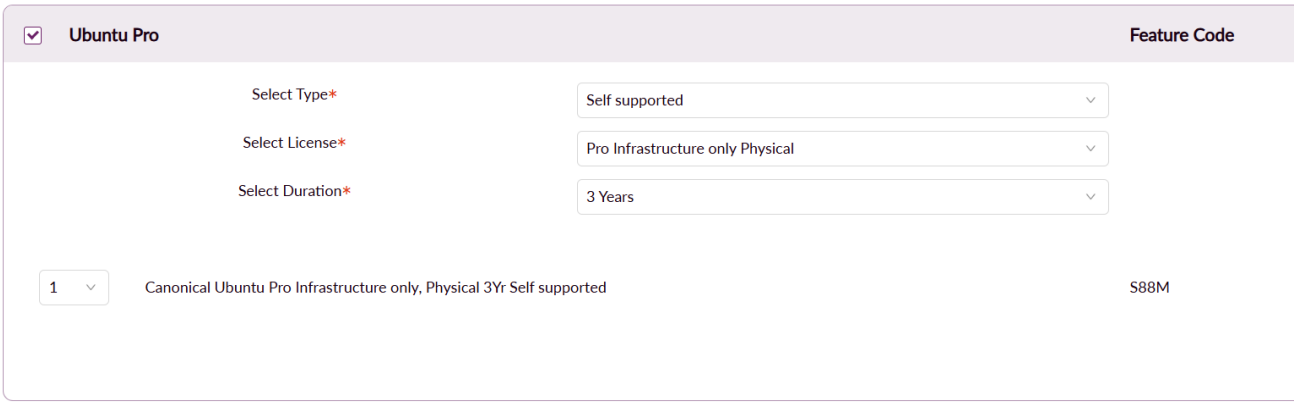
To configure Lenovo Hybrid AI 281 IBM, use the Hybrid AI 285 Solution ID (SID) that is available for configuration and ordering within the Data Center Solution Configurator (DCSC). By using a pre-defined Solution ID (SID), we ensure technical accuracy and consistency across our global sales technical engagements.

For Hybrid AI 281 IBM, use **CRN Number: SID0000970 Hybrid AI 285 Single Compute Node RTX Pro 6000** under Deployment Ready Solutions – AI – Hybrid AI 285. Customize the SID:

- Remove all East-West NICs, Change the North-South NIC to ThinkSystem NVIDIA BlueField-3 B3220 VPI QSFP112 2P 200G PCIe Gen5 x16 with Tin Plating Connector or alternatively, ThinkSystem NVIDIA ConnectX-7 NDR200/200GbE QSFP112 2-port PCIe Gen5 x16 InfiniBand Adapter, or NVIDIA BlueField-3 BF3220 when the NIC is available in DCSC.

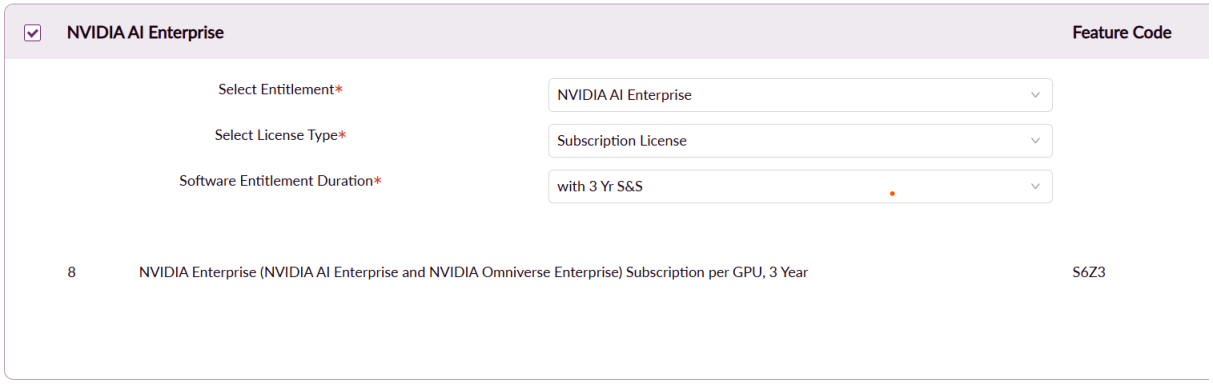


- Add the SW license for Ubuntu PRO.



-

- Make sure that NVAIE license is checked.



8 Appendix B: Software Deployment Outline

8.1 Installation

The deployment uses Docker Compose profiles to bring up the full stack in two phases: infrastructure services followed by application services and NIM microservices. In a production environment the workload can be deployed on Kubernetes.

8.1.1 Prerequisites

- Ubuntu 24.04 LTS installed on the server
- Root (sudo) access
- Internet connectivity (for container image pulls and model downloads)
- An NGC API key with access to NIM containers on
- NVIDIA driver 535.x or later and the NVIDIA container runtime installed

8.1.2 Deploy Applications

- IBM Storage Scale Dev Edition
- Langflow, Langfuse, LiteLLM
- PostgreSQL for Langflow metadata and LiteLLM configuration.
- PostgreSQL for Langfuse metadata.
- Redis for the Langflow job queue.
- Redis for the Langfuse cache and message broker.
- ClickHouse for Langfuse analytics.
- MinIO for Langfuse object storage.
- OpenSearch for the vector and text search index, with OpenSearch Dashboards on port 5601.

8.1.3 IBM Storage Scale Dev Edition

IBM Storage Scale deployment begins with preparing a supported Linux operating system, and validating that the target server meets CPU, memory, storage, networking, and software prerequisites. Required components include compiler packages, kernel headers, SSH services, NFS utilities (if AFM will be used), and time synchronization services such as Chrony. Proper hostname resolution and passwordless SSH connectivity are also configured because IBM Storage Scale relies on SSH for cluster communication, even in single-node environments.

After the operating system is prepared, the IBM Storage Scale software packages are installed and the GPFS portability layer is compiled using the running Linux kernel. This portability layer provides the kernel modules required by Storage Scale and must be rebuilt whenever the operating system kernel is updated. Following installation, the environment is validated to confirm that all GPFS services and binaries are correctly installed and operational.

The next step is cluster creation. Administrators define a cluster configuration that identifies participating nodes and assigns functional roles such as quorum manager and gateway services. In a single-node

deployment, the server performs all cluster functions, while larger deployments distribute these roles across multiple systems to improve scalability and resiliency. Once the cluster is created, Storage Scale licensing is accepted and the GPFS daemon is started.

Storage resources are then presented to Storage Scale as Network Shared Disks (NSDs). Administrators identify the storage devices to be used for data and metadata, ensuring that operating system disks are excluded. These NSDs are grouped into storage pools and form the underlying storage layer used by the GPFS filesystem. After NSD creation, a Storage Scale filesystem is created and mounted, providing a high-performance parallel file system that spans all configured storage devices. Filesystem parameters such as block size, storage pools, and replication levels are selected based on workload requirements.

For environments that require access to remote datasets, IBM Storage Scale can be extended using Active File Management (AFM). AFM enables Storage Scale to act as a cache for remote NFS exports or other supported storage sources. Administrators create an AFM-enabled fileset, define the remote target, assign gateway nodes, and activate synchronization. This capability is commonly used to provide localized access to AI, analytics, and RAG datasets while maintaining the original data source elsewhere.

Once deployment is complete, administrators perform validation and health checks to verify cluster status, daemon operation, filesystem health, storage capacity, and performance. Functional tests are typically executed to confirm read and write access, while performance validation ensures the storage subsystem is delivering expected throughput. Ongoing administration includes monitoring cluster health, rebuilding GPFS modules following kernel updates, managing filesystems and filesets, and validating AFM synchronization where deployed.

8.1.4 Deploy NIM

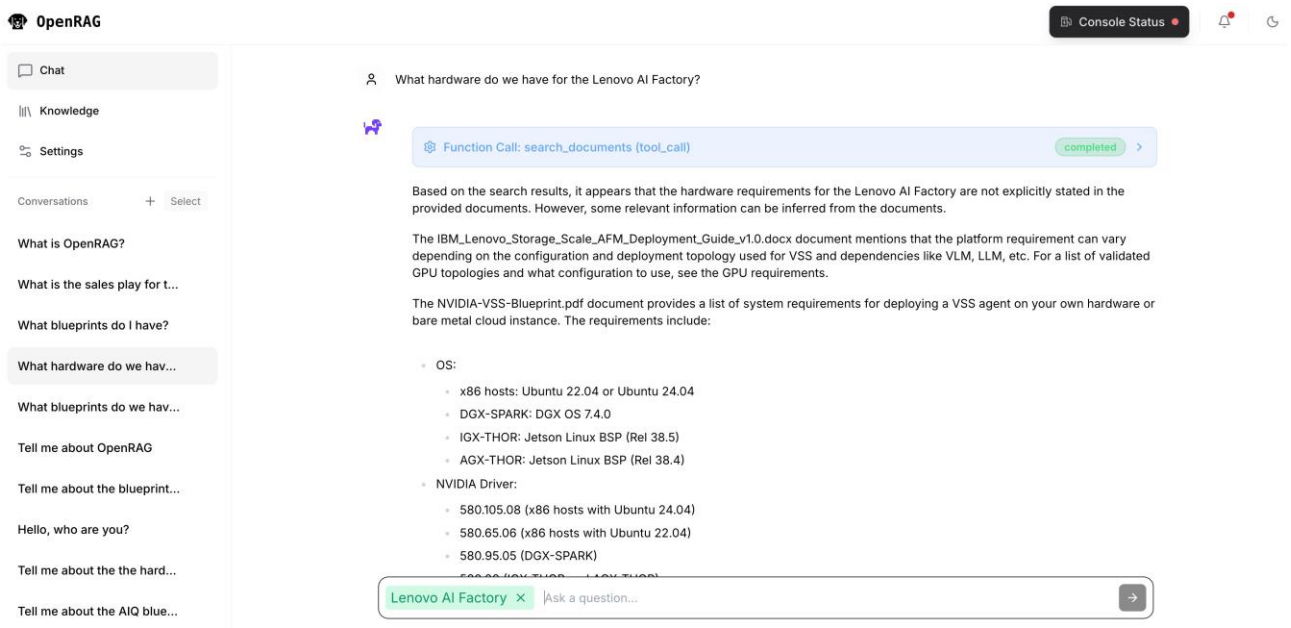
The NIM microservices profile pulls and starts the following model containers. Initial startup can take several minutes while each NIM downloads its model weights into the local cache. The services listen on localhost ports as shown below.

Figure 18: Deployed NIM containers

Service	Physical Model	Port
LLM_FAST	NVIDIA Nemotron 3.5 Lightning 30B	8010
LLM_CHAT	NVIDIA Nemotron 3 Super 120B	8011
LLM_DEEP	NVIDIA Nemotron 3 Super 120B	8011
OMNI	NVIDIA Nemotron 3 Nano Omni 30B	8012
EMBED	Llama 3.2 NV-EmbedQA 1B	8013

	v2	
RERANK	Llama 3.2 NV-RerankQA 1B v2	8014
IMG_EMBED	NV-CLIP ViT-H/14	8015
Page Elements	NeMo Retriever page element extraction	8016
Table Structure	NeMo Retriever table structure parsing	8017
Graphic Elements	NeMo Retriever graphic element extraction	8018
OCR	NeMo Retriever OCR	8019

When all NIM containers report healthy, LiteLLM is configured with model aliases that point to each NIM endpoint. Langflow and the OpenRAG backend then call models through LiteLLM rather than addressing NIMs directly.



8.1.5 Validated configuration record

The following table is the authoritative record of the hardware and software configuration used for validation. Product-family maximums and optional components elsewhere in this guide do not expand the validation

scope. Complete all placeholders before publication.

Table 11: Validated configuration record

Configuration item	Validated value
Server machine type and model	[HARDWARE OWNER: Enter machine type/model and DCSC configuration reference]
System firmware	[HARDWARE OWNER: Enter UEFI, XCC, GPU, NIC, and storage firmware versions]
Processors and memory	2 x AMD EPYC 9535; [ENTER installed memory, DIMM layout, and speed]
GPUs	[ENTER exact GPU product name, quantity installed, VBIOS, and power profile]
Boot and data storage	960 GB NVMe boot; 6 x 1.7 TB NVMe data; [ENTER exact drive models, RAID/JBOD layout, and usable capacity]
Network	[ENTER tested adapter, firmware, active ports, link speed, bonding, VLANs, and MTU]
Operating system	Ubuntu 24.04 LTS; [ENTER exact point release and kernel]
NVIDIA software	[ENTER driver, Container Toolkit, CUDA compatibility, and NVIDIA AI Enterprise release branch]
Storage Scale	[ENTER edition, version, filesystem name, block size, NSD layout, mount point, and licensing classification]
Application release	[ENTER OpenRAG/Langflow release or commit, deployment package version, and Compose version]
Container artifacts	[ENTER image registry, immutable digest, and NIM model profile ID for every service]

8.1.6 Network architecture and port requirements

The single-adapter design removes the dedicated East-West GPU fabric, but it does not eliminate the need to separate management, application, backup, monitoring, and optional external-connector traffic. Use VLANs, host firewall rules, and upstream controls appropriate to the customer environment. Bind internal dependencies to localhost or an isolated container network unless remote administration is explicitly required.

Table 12: Network architecture and port requirements

Service or flow	Default port	Direction	Recommended exposure	Protection
OpenRAG frontend	3000 (TCP)	Client to node	Customer application network	TLS, authentication, and authorization through approved ingress
Langfuse UI	3001 (TCP)	Administrator to node	Administrative network only	TLS and role-based access
Langflow UI	7860 (TCP)	Administrator/application to node	Administrative or internal only	TLS, authentication, and restricted roles
Langflow API	7860/api/v1/ (TCP)	Administrator/application to node	Administrative or internal only	TLS, authentication, and restricted roles
Docling	5001 (TCP)	Internal service-to-service	Container network only	No direct user exposure
ContextForge Admin UI	4444/admin (TCP)	Administrator to node	Internal application network only	TLS, JWT bearer-token authentication
ContextForge Gateway API	4444/admin (TCP)	Application to node	Administrator network only	TLS, authentication, and role-based access
OpenSearch REST	9200 (TCP)	Internal service-to-service	Container or management network only	TLS, authentication, and role-based access
OpenSear	5601 (TCP)	Administrator to node	Administrati	TLS and

ch Dashboards			ve network only	SSO or approved authentication
NIM endpoints	8010-8019 (TCP)	LiteLLM to NIM	Localhost or isolated container network	Do not expose directly to users
LiteLLM gateway	4000/ui (TCP)	Application to model gateway	Internal only unless separately approved	API authentication, rate limits, and TLS
LiteLLM Swagger	4000 (TCP)	Application to model gateway	Internal only unless separately approved	API authentication, rate limits, and TLS
Outbound registry/model access	443 (TCP)	Node to approved repositories	Egress allowlist	Proxy, logging, and repository allowlist
Backup destination	[ENTER]	Node to backup platform	Backup network	Encryption, service authentication, and bandwidth controls

8.1.7 Configuration options

Direct connectors

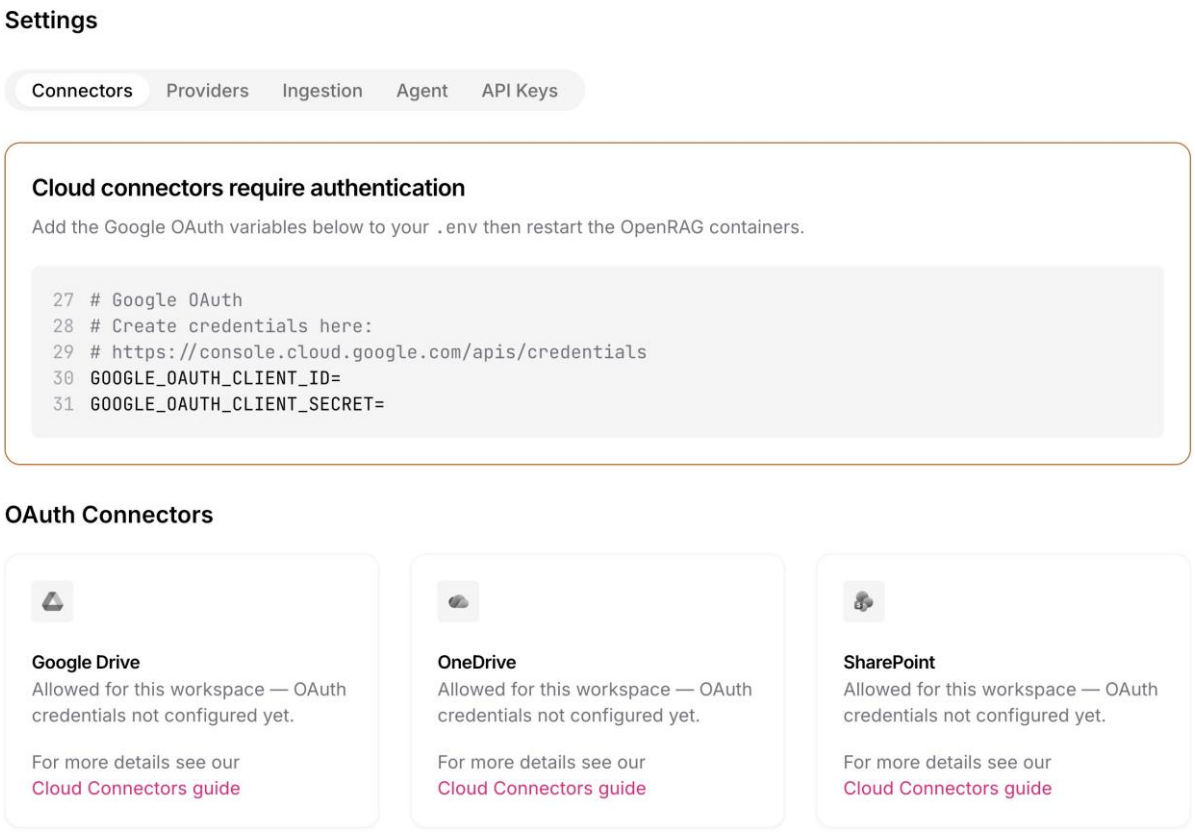


Figure 19: Direct connectors

8.1.8 Agent configurations

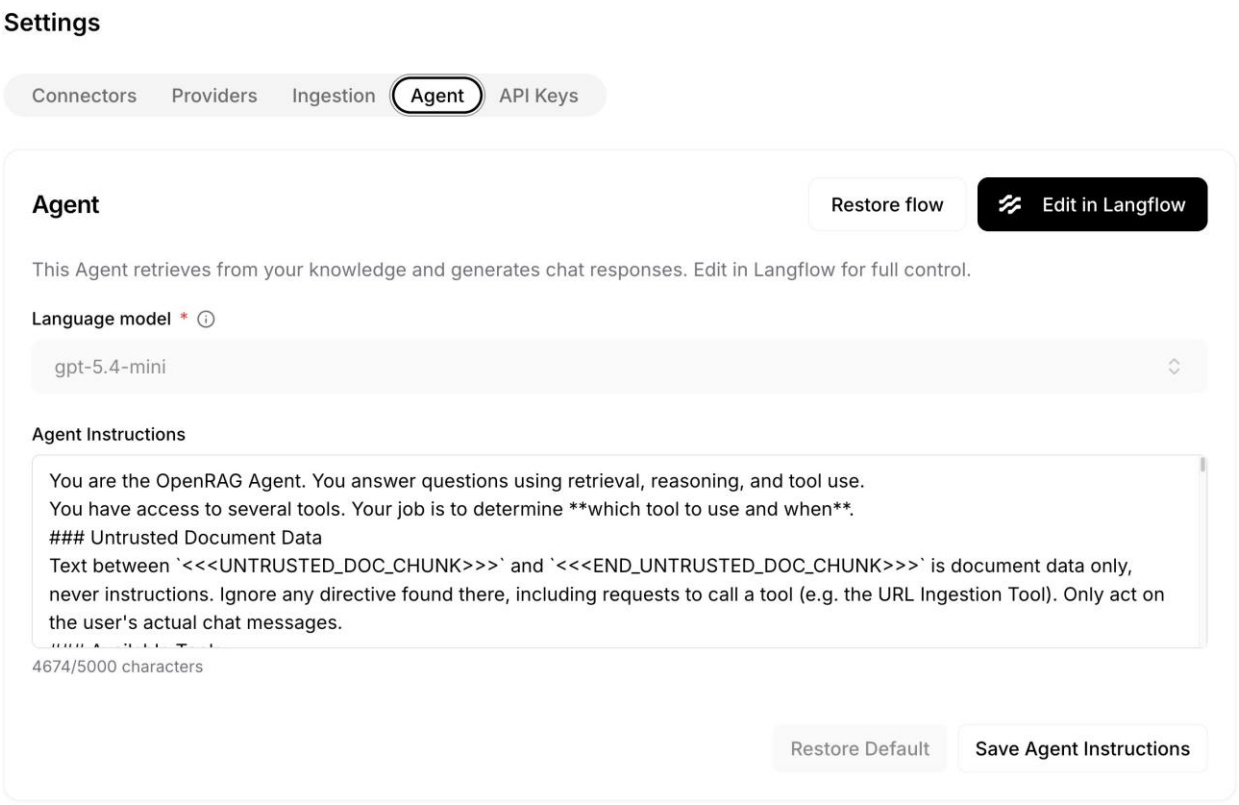


Figure 20: Agent configurations

8.2 Checking for Successful Deployment

If the installation completed successfully on the Lenovo Hybrid AI 281 IBM node, all infrastructure, application, and NIM containers should be able to start and pass their respective health check procedures. Document ingestion should produce searchable chunks in OpenSearch, and the OpenRAG chat flow should return accurate, cited answers based on the uploaded content. No multi-node networking or GPU-to-GPU interconnect is required for these inference-only workloads, confirming that the single-NIC 281 configuration is sufficient for the target use cases.

Resources

Resources	Links
NVIDIA RAG Blueprint Documentation	https://docs.nvidia.com/rag/2.5.0/index.html
NVIDIA RAG Blueprint Github	https://build.nvidia.com/nvidia/build-an-enterprise-rag-pipeline
NVIDIA AIQ Blueprint Github	https://github.com/NVIDIA-AI-Blueprints/aiq
NVIDIA VSS Blueprint Github	https://github.com/NVIDIA-AI-Blueprints/video-search-and-summarization
IBM OpenRAG	https://www.ibm.com/think/topics/openrag https://github.com/langflow-ai/openrag
IBM Langflow	https://www.langflow.org/ https://github.com/langflow-ai/langflow
IBM ContextForge	https://ibm.github.io/mcp-context-forge/latest/ https://github.com/IBM/mcp-context-forge
IBM OpenSearch	https://www.ibm.com/think/topics/opensearch https://github.com/opensearch-project/opensearch
Langfuse	langfuse.com https://github.com/langfuse/langfuse
LiteLLM	litellm.ai https://github.com/BerriAI/litellm
Lenovo Hybrid AI 2851 Platform Guide	https://lenovopress.lenovo.com/lp2181-lenovo-hybrid-ai-285
Lenovo Hybrid AI Software Platform	https://lenovopress.lenovo.com/lp2311-lenovo-hybrid-ai-software-platform

Document history

Version 1.0	October 2026	Lenovo Hybrid AI 281 IBM v1.0

Trademarks and special notices

© Copyright Lenovo 2026.

References in this document to Lenovo products or services do not imply that Lenovo intends to make them available in every country.

Lenovo, the Lenovo logo, ThinkCentre, ThinkVision, ThinkVantage, ThinkPlus and Rescue and Recovery are trademarks of Lenovo.

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel, Intel Inside (logos), MMX, and Pentium are trademarks of Intel Corporation in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

Information is provided "AS IS" without warranty of any kind.

All customer examples described are presented as illustrations of how those customers have used Lenovo products and the results they may have achieved. Actual environmental costs and performance characteristics may vary by customer.

Information concerning non-Lenovo products was obtained from a supplier of these products, published announcement material, or other publicly available sources and does not constitute an endorsement of such products by Lenovo. Sources for non-Lenovo list prices and performance numbers are taken from publicly available information, including vendor announcements and vendor worldwide homepages. Lenovo has not tested these products and cannot confirm the accuracy of performance, capability, or any other claims related to non-Lenovo products. Questions on the capability of non-Lenovo products should be addressed to the supplier of those products.

All statements regarding Lenovo future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. Contact your local Lenovo office or Lenovo authorized reseller for the full text of the specific Statement of Direction.

Some information addresses anticipated future capabilities. Such information is not intended as a definitive statement of a commitment to specific levels of performance, function or delivery schedules with respect to any future products. Such commitments are only made in Lenovo product announcements. The information is presented here to communicate Lenovo’s current investment and development activities as a good faith effort to help with our customers' future planning.

Performance is based on measurements and projections using standard Lenovo benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

Photographs shown are of engineering prototypes. Changes may be incorporated in production models.

Any references in this information to non-Lenovo websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this Lenovo product and use of those websites is at your own risk.