The National Research Platform for AI-Enabled Research

GPUs and AI for research and education

Daniel Diaz · San Diego Supercomputer Center, UC San Diego

UCLA Health IT Research Tech Talk · 10 November 2026



Agenda

- What is NRP?
- Getting access, and what a namespace is
- GPUs: what's available, and how to run a GPU job
- AI services: hosted LLMs and Jupyter AI
- JupyterHub for labs and classes
- Live demo
- Questions

Slides, the demo notebooks and the GPU job:

https://training.nrp-nautilus.io/ucla-health

The Platform

What is NRP?

- A national **Kubernetes** cluster, **Nautilus**, for research and education
- Run by SDSC (UC San Diego), Nebraska–Lincoln and MGHPCC; funded by NSF

https://dash.nrp-nautilus.io



GPUs and nodes: cluster snapshot, 9 Nov 2026

What is NRP?

- A national **Kubernetes** cluster, **Nautilus**, for research and education
- Run by SDSC (UC San Diego), Nebraska–Lincoln and MGHPCC; funded by NSF
- Hardware contributed by 70+ institutions, UCLA among them

https://dash.nrp-nautilus.io



GPUs and nodes: cluster snapshot, 9 Nov 2026

What is NRP?

- A national **Kubernetes** cluster, **Nautilus**, for research and education
- Run by SDSC (UC San Diego), Nebraska–Lincoln and MGHPCC; funded by NSF
- Hardware contributed by 70+ institutions, UCLA among them
- Scale today:
  - 536 nodes
  - 1,411 GPUs in 33 models
  - Petabytes of storage

https://dash.nrp-nautilus.io



Site locations from the NRP dashboard, 10 Nov 2026
Not drawn: 8 sites in Hawaii, Guam, Korea, Brazil and Europe

GPUs and nodes: cluster snapshot, 9 Nov 2026

What is NRP?

- A national **Kubernetes** cluster, **Nautilus**, for research and education
- Run by SDSC (UC San Diego), Nebraska–Lincoln and MGHPCC; funded by NSF
- Hardware contributed by 70+ institutions, UCLA among them
- Scale today:
  - 536 nodes
  - 1,411 GPUs in 33 models
  - Petabytes of storage
- Free for non-commercial research and education

https://dash.nrp-nautilus.io

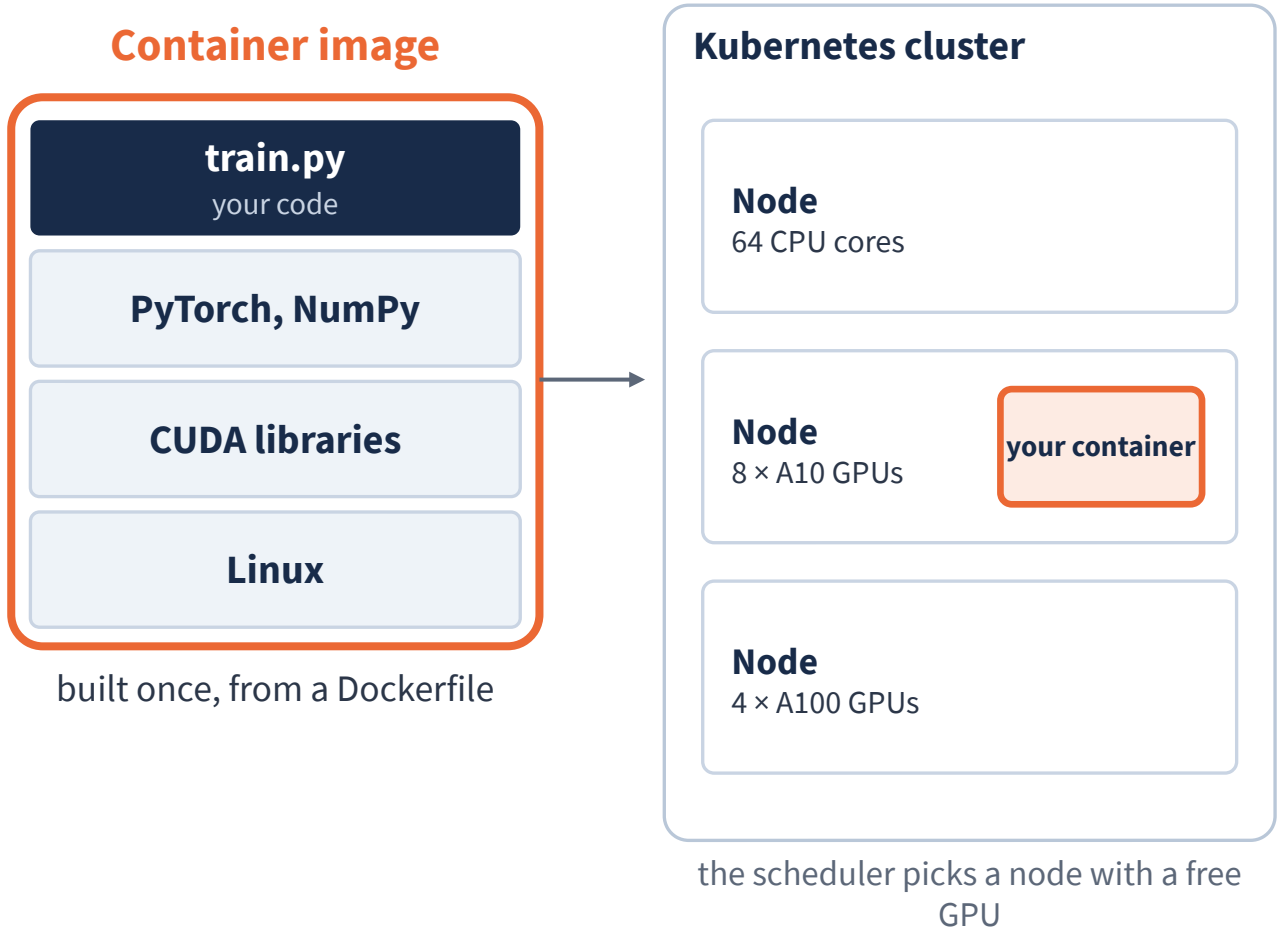


Site locations from the NRP dashboard, 10 Nov 2026
Not drawn: 8 sites in Hawaii, Guam, Korea, Brazil and Europe

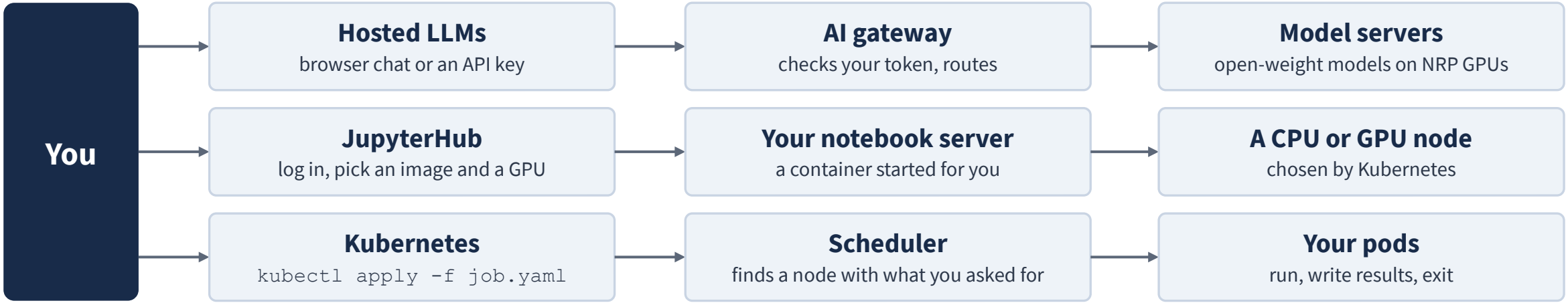
GPUs and nodes: cluster snapshot, 9 Nov 2026

Kubernetes and Containers

- **Container image:** your code plus everything it needs (Python, PyTorch, CUDA), in one package
- Runs the same on a laptop, a cluster or the cloud
- **Kubernetes** runs containers across many machines
- You say what you need (CPUs, memory, a GPU); it finds a machine that has it
- NRP is one Kubernetes cluster: 536 nodes



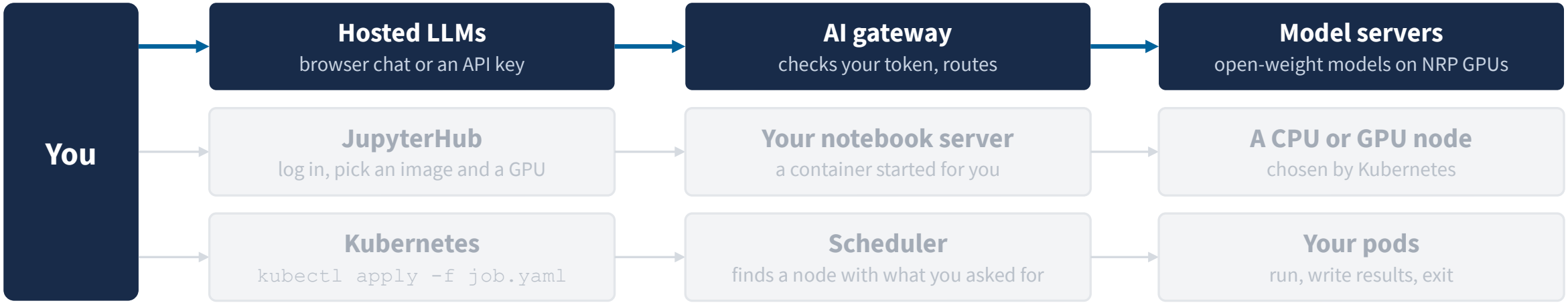
Access Methods

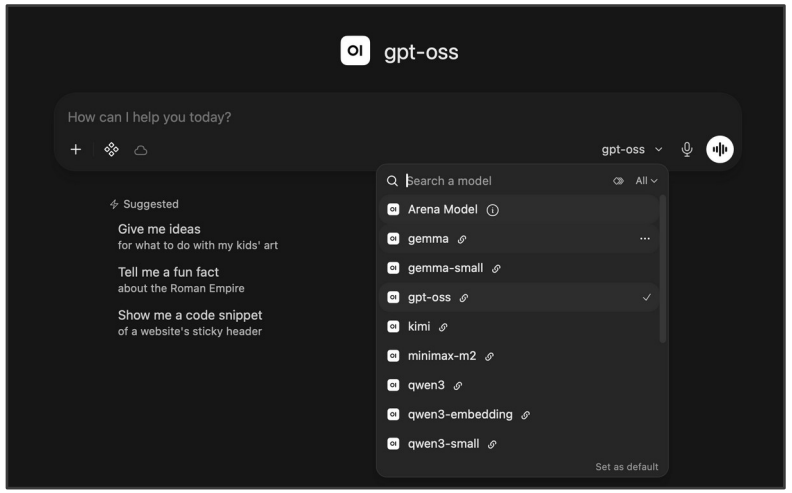


No login node and no batch queue: pick the door that matches how much infrastructure you want to touch.

Most people start with JupyterHub or the hosted LLMs.

Access Methods

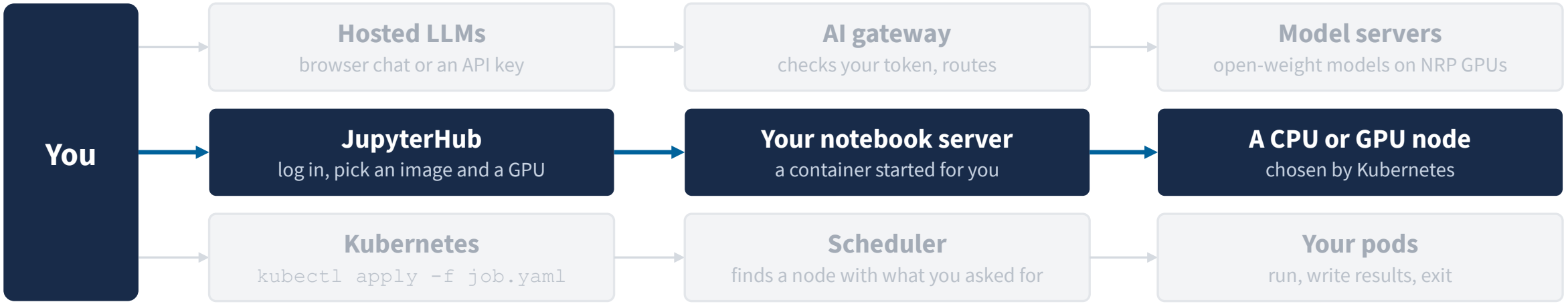


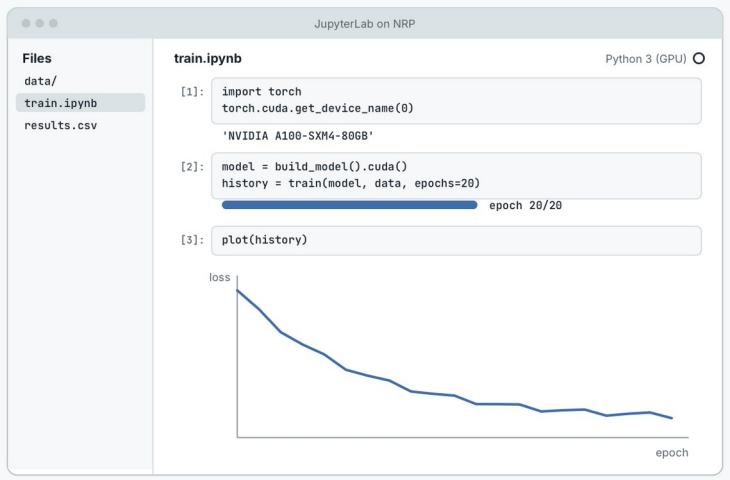



```
from openai import OpenAI
client = OpenAI(base_url="https://ellm.nrp-nautilus.io/v1",
                api_key=TOKEN)         # from nrp.ai/llmtoken
client.chat.completions.create(model="gpt-oss",
    messages=[{"role": "user", "content": "Hi"}])
```

In the browser at nrp-openwebui.nrp-nautilus.io, or from any code that speaks the OpenAI API

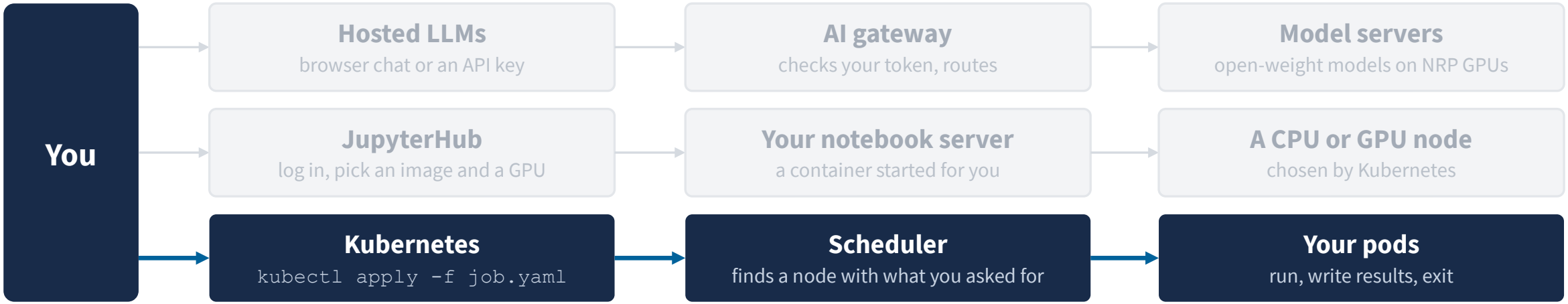
Access Methods





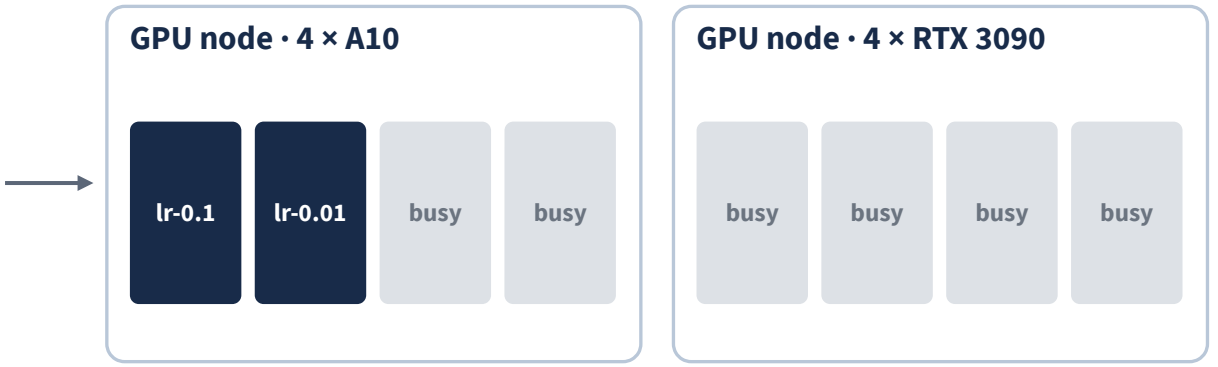
- Log in with your campus account through CILogon
- Choose an image, CPUs, memory and a GPU type
- NRP's hubs: jupyterhub-west.nrp-nautilus.io and jupyterhub-east.nrp-nautilus.io
- Today's demo: jh-training.nrp-nautilus.io

Access Methods



Your jobs

```
$ kubectl apply -f sweep/
job.batch/train-lr-0.1 created
job.batch/train-lr-0.01 created
job.batch/train-lr-0.001 created
$ kubectl get pods
NAME                    READY   STATUS    RESTARTS   AGE
train-lr-0.1-7xkqp      1/1     Running   0          40s
train-lr-0.01-m4q9d     1/1     Running   0          40s
train-lr-0.001-c8w3n    0/1     Pending   0          40s
```



lr-0.001 waits in Pending until a GPU frees up; a finished job releases its GPU

What NRP Is For

OK on NRP		Not on NRP
Public or synthetic data	**De-identified data**	**Protected health information**
public datasets, simulations, benchmark and synthetic data	if your IRB and data use agreement allow it	PHI or identifiable patient data, including in LLM prompts

Good uses

model development and training · image analysis · simulation · teaching · LLMs on non-sensitive text

Also not for: commercial work, or services that need guaranteed uptime (it is a shared, best-effort cluster)

NRP is **not HIPAA compliant**. Keep PHI in UCLA Health's approved environments.

Getting Access

Getting Access

1. Log in at **nrp.ai**

Log In → CILogon → choose **UCLA** → your UCLA login

2. Get into a namespace

Request one at **get-started.nrp-nautilus.io**

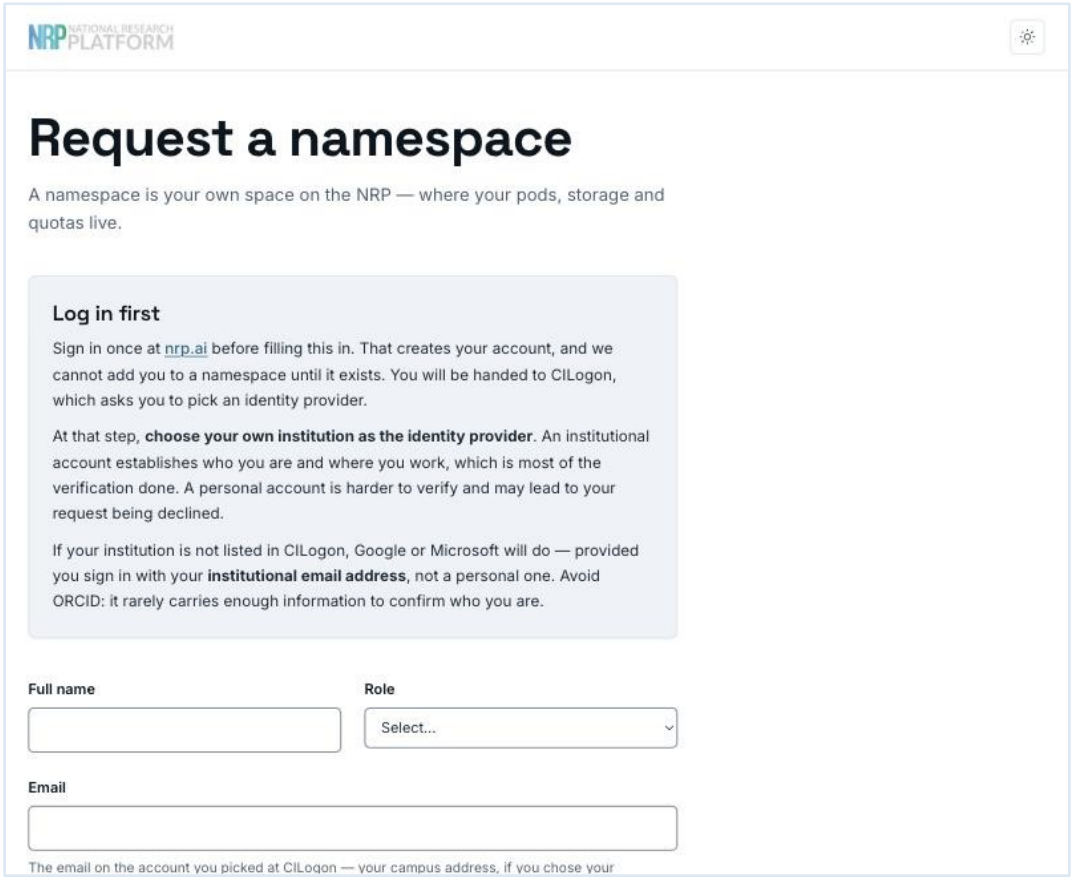
Or ask an existing namespace admin to add you

3. Start working

JupyterHub, kubectl, or an LLM token from **nrp.ai/llmtoken**

Who can request a namespace: faculty, staff, postdocs and research scientists. **Students:** your PI or instructor requests it and adds you.

https://get-started.nrp-nautilus.io



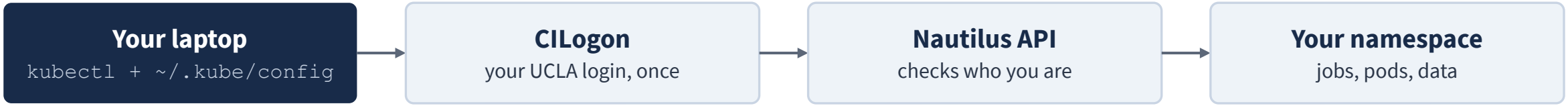
What is a Namespace?

- Your group's workspace on the cluster
- One per lab, project or course; the PI or instructor is its admin and adds members
- Needed for GPU jobs, storage, your own JupyterHub, and **API access** to the hosted LLMs
- Not needed to chat with the LLMs in the browser
- What's inside is visible only to its members

Nautilus cluster

`lab-smith`	**Dr. Smith (admin) and 3 students** GPU jobs · a data volume · LLM tokens
`bio-course`	**the instructor (admin) and 40 students** a course JupyterHub · shared datasets
`another-lab`	**their members only** you can't see their jobs or data

Talking to the Cluster



1. Install **kubectl** and the **kubelogin** plugin
2. Download your config from **nrp.ai/config**, save it as ~/.kube/config
3. Run any kubectl command: a browser opens for the UCLA login
4. Work in your namespace

On the training JupyterHub this is already set up.

```
# one-time setup
mkdir -p ~/.kube
curl -o ~/.kube/config -fSL https://nrp.ai/config

# first command: a browser opens for CILogon
kubectl get pods -n my-lab
No resources found in my-lab namespace.

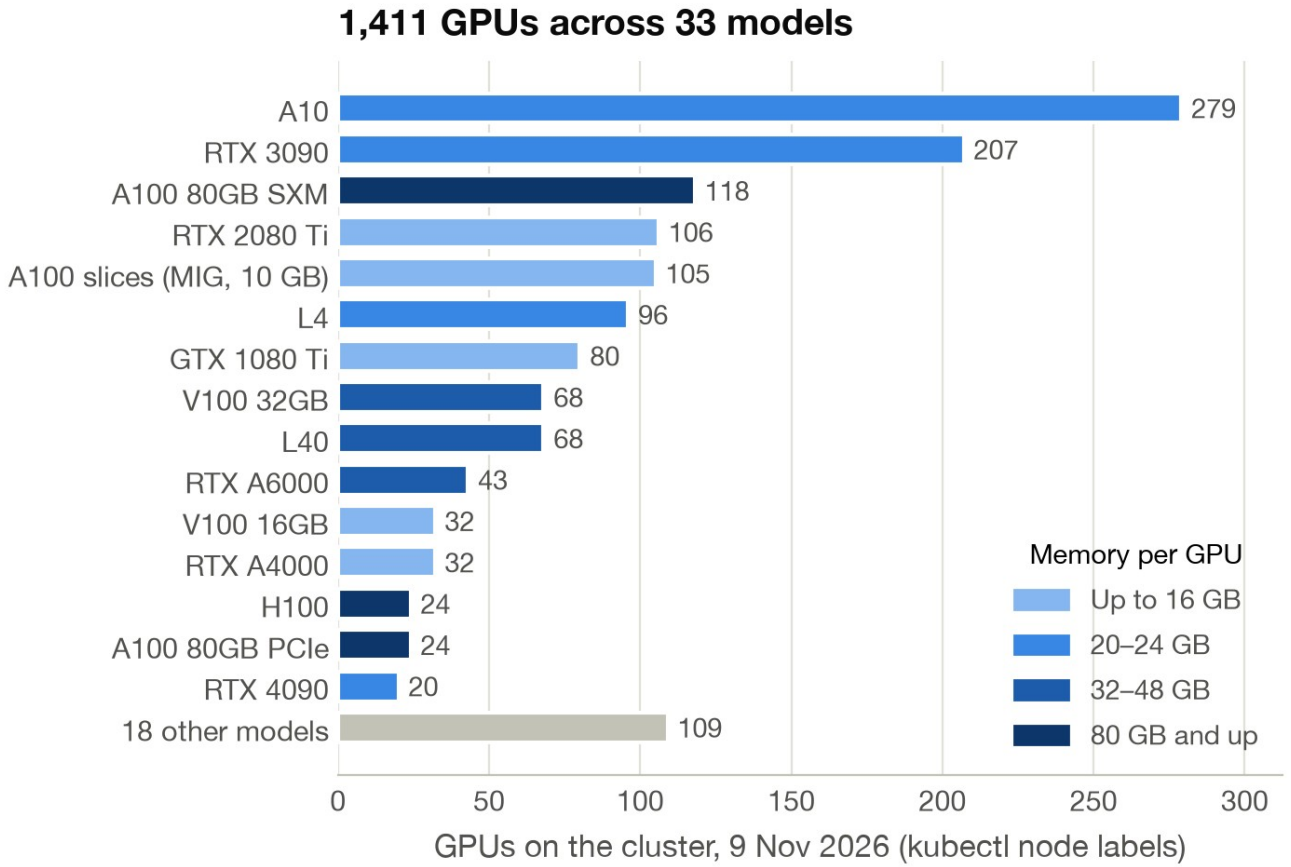
# make my-lab your default namespace
kubectl config set contexts.nautilus.namespace my-lab
```

https://nrp.ai/documentation/userdocs/start/getting-started/

GPUs

GPUs on NRP

- 1,411 GPUs in 33 models, contributed by campuses
- Most common: A10 and RTX 3090, 24 GB each
- About 140 A100s with 80 GB, plus H100s and H200s
- Ask for the GPU memory your job needs, not a brand name
- What's free right now: **nrp.ai/viz/resources**



Choosing a GPU

GPU memory	Examples	Good for	How to ask
Up to 16 GB	RTX 2080 Ti, GTX 1080 Ti, A4000, A100 slices	Teaching, debugging, classic ML, small CNNs	nvidia.com/gpu: 1
20–24 GB	A10, RTX 3090, L4, RTX 4090	Most deep learning; fine-tuning and serving models up to ~8B parameters	nvidia.com/gpu: 1 + memory or model
32–48 GB	L40, V100 32 GB, RTX A6000, A40	Bigger models and batches; serving models up to ~30B	nvidia.com/gpu, or rtxa6000 / a40
80 GB and up	A100 80 GB, H100, H200	Large training runs and large models	A100: access request; H100/H200 reserved

Current PyTorch builds need **Turing or newer**: on a GTX 1080 Ti or V100, use an older image.

https://nrp.ai/documentation/userdocs/running/gpu-pods/

Ways to Run Work

Interactive: JupyterHub

- Pick a GPU from the start menu; explore, debug, teach
- Idle servers stop: 1 hour after you close the browser on NRP's hub, 6 hours at most on any hub

Interactive: a pod

- Start a container with kubectl, then open a shell in it
- Deleted after 6 hours; at most 2 GPUs

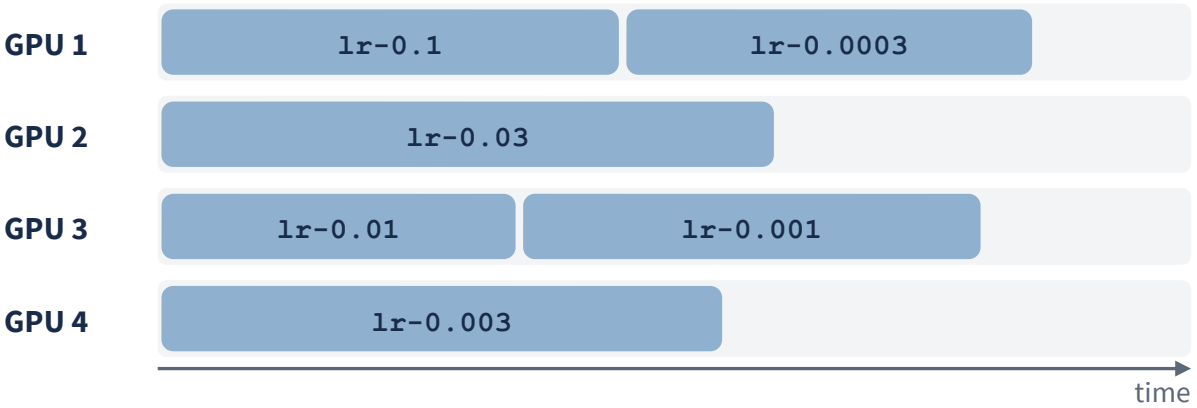
Long-running: a Deployment

- Keeps a service up (a web app, a database) and restarts it if it crashes
- Removed after 2 weeks unless admins grant an exception; no GPU if it sits mostly idle

Batch: a Job

- Runs your script to the end, then gives the GPU back
- Submit many at once; the rest wait in Pending for a free GPU

```
$ kubectl apply -f sweep/        # six training jobs
```



Six jobs on four GPUs: two wait in Pending and take the first GPUs that free up. When the last one ends, all four GPUs are free again.

A GPU Job

```
apiVersion: batch/v1
kind: Job
metadata: {name: gpu-train-daniel, namespace: my-lab}     # (1)
spec:
  template:
    spec:
      restartPolicy: Never
      containers:
      - name: train
        image: pytorch/pytorch:2.9.0-cuda13.0-cudnn9-runtime  # (2)
        command: ["python", "train.py"]
        resources:                                          # (3)
          requests: {cpu: "2", memory: 6Gi, nvidia.com/gpu: 1}
          limits:   {cpu: "2", memory: 6Gi, nvidia.com/gpu: 1}
      affinity:                                             # (4)
        nodeAffinity:
          requiredDuringSchedulingIgnoredDuringExecution:
            nodeSelectorTerms:
            - matchExpressions:
              - {key: nvidia.com/gpu.memory, operator: Gt, values: ["20000"]}
              - {key: nvidia.com/gpu.compute.major, operator: Gt, values: ["7"]}
```

(1) **Where:** your namespace

(2) **What:** a container image, and the command to run

(3) **Resources:** 2 CPUs, 6 GB of memory, one GPU

(4) **Which GPU:** more than 20 GB, Ampere or newer; or name exact models

Running the Job

```
# submit
kubectl apply -f gpu-job.yaml

# watch: Pending → Running → Completed
kubectl get pods -l job-name=gpu-train-daniel

# read the output
kubectl logs -f job/gpu-train-daniel

# clean up
kubectl delete job gpu-train-daniel
```

Output, 9 Nov 2026

```
Running on: NVIDIA A10
Matrix multiply, 4096 x 4096:
  CPU (2 cores)  0.09 TFLOP/s
  GPU            14.0 TFLOP/s  (154x)
Training on 60,000 images, testing on 10,000
epoch 1: loss 0.090, accuracy 98.4%, 1.6 s
epoch 2: loss 0.009, accuracy 98.7%, 1.0 s
epoch 3: loss 0.033, accuracy 98.7%, 1.1 s
Done: the job exits and frees the GPU.
```

Stuck in **Pending**? `kubectl describe pod` says why, usually that no matching GPU is free.

Our test: under 30 s from submit to done on an A10 that had the image; several minutes in Pending when the cluster was full.

Using GPUs Fairly

Do

- Run batch work as a **Job** that exits
- Request what you'll use; keep limits within 20% of requests
- Keep data on a persistent volume or in S3
- Run a sweep as many one-GPU jobs
- Borrow idle A100/H100s with `priorityClassName: opportunistic`, and checkpoint

Don't

- Leave a notebook holding a GPU overnight
- Put `sleep` in a Job to keep a GPU (bannable)
- Ask for 4 GPUs "just in case": use must stay above 40%
- Keep results on the pod's own disk; it vanishes with the pod

https://nrp.ai/documentation/userdocs/start/policies/

AI Services

Hosted LLMs

- Open-weight models on NRP GPUs, behind one OpenAI-compatible endpoint:
  - ellm.nrp-nautilus.io/v1
- Free for non-commercial research and education
- Prompts are processed on NRP hardware, not sent to a commercial provider
- Still not for PHI
- About 7.1 billion tokens a day (September 2026)

Model	Good for
`qwen3, qwen3-small`	Hard reasoning, long documents, images
`gpt-oss`	General chat, reproducible research
`glm-5, kimi, minimax`	Agentic coding
`deepseek-v4-flash`	Whole codebases, very long documents
`gemma, gemma-small`	Images, video, audio transcription
`qwen3-embedding`	Search and RAG (embeddings)

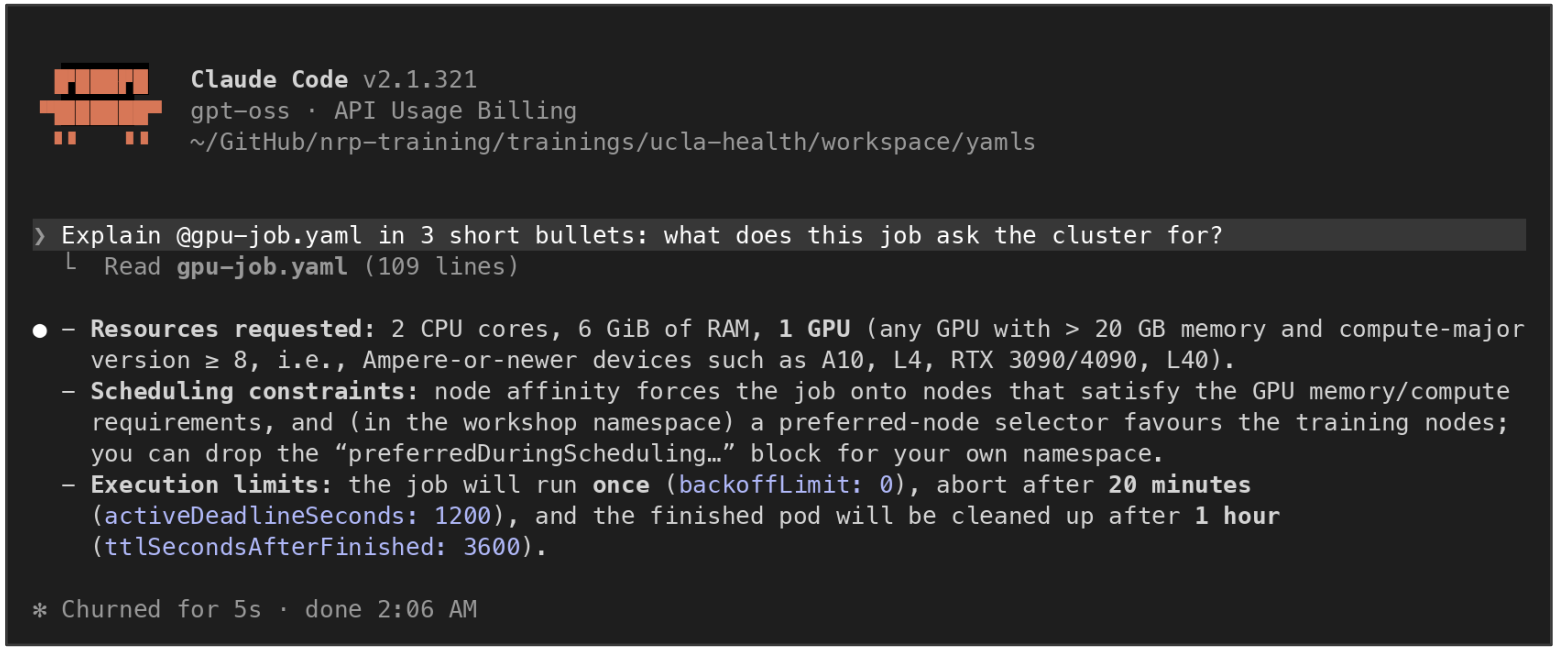
https://nrp.ai/documentation/userdocs/ai/llm-managed/models/

Uses from the models page, 26 Oct 2026. The catalog rotates.

Using the LLMs

- **In the browser:** Open WebUI
  - nrp-openwebui.nrp-nautilus.io
- **From code:** the standard OpenAI library; token from nrp.ai/llmtoken
- **In JupyterHub:** Jupyter AI (in the demo)
- **Coding agents:** Claude Code, opencode, VS Code and others
  - tested configurations in the NRP docs

Claude Code on NRP's gpt-oss, asked about the demo's GPU job



Using the LLMs

- **In the browser:** Open WebUI
  - nrp-openwebui.nrp-nautilus.io
- **From code:** the standard OpenAI library; token from nrp.ai/llmtoken
- **In JupyterHub:** Jupyter AI (in the demo)
- **Coding agents:** Claude Code, opencode, VS Code and others
  - tested configurations in the NRP docs

opencode on NRP's qwen3, same question

```
Explain gpu-job.yaml in 3 short bullets: what does this job ask the cluster for?

+ Thought: 773ms

✱ Glob "**/gpu-job.yaml" (1 match)

+ Thought: 432ms

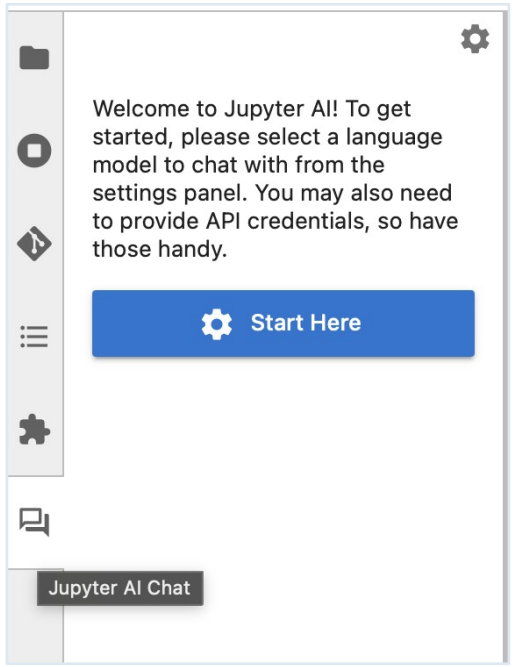
→ Read gpu-job.yaml

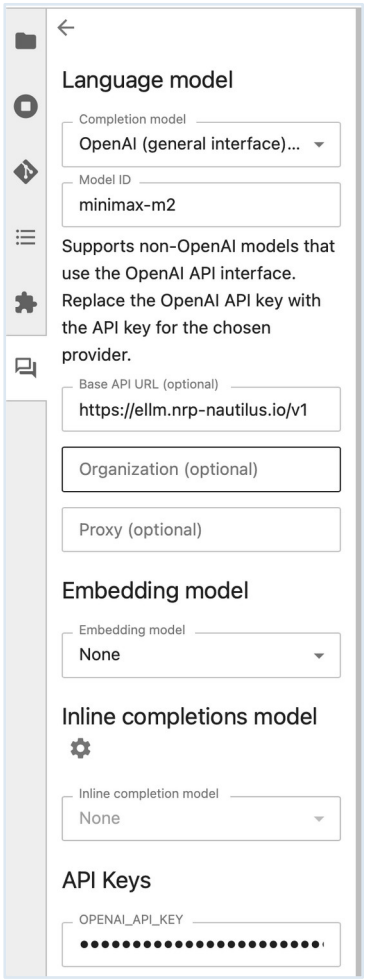
+ Thought: 2.9s

- 1 GPU, 2 CPUs, 6 GiB RAM — requests equal limits, so the scheduler reserves a Guaranteed-QoS
  pod using the pytorch/pytorch:2.9.0-cuda13.0-cudnn9-runtime image.
- A specific class of GPU node — hard affinity for >20000 MiB memory and compute.major > 7 (
  Ampere/Ada+: A10, L4, RTX 3090/4090, L40…), plus a soft preference for nrp-training=true nodes.
- A short, self-cleaning run — restartPolicy: Never, backoffLimit: 0 (no retry), 20-min deadline,
   and TTL-garbage-collect 1 hour after it trains MNIST and exits.

▣  Build · qwen3 · 13.4s
```

Jupyter AI





- Already pointed at NRP's models on the training hub, and on any course hub you run
- **Chat panel:** ask about code, an error or a concept
- **/fix:** select a cell that crashed; it explains and proposes a fix
- **%%ai magics:** {variables} in a prompt become their live values, so the model sees your data

```
%%ai openai-chat:gpt-oss
Here is a summary of my dataset: {summary}
Which columns would you check first, and why?
```

Running Your Own Models

- **Serve your own model:** vLLM or SGLang on a GPU in your namespace, your weights or your fine-tune behind the same OpenAI API; ask the admins before running one as a long-lived service
- **Train and fine-tune:** PyTorch and Hugging Face in GPU jobs, from one GPU to eight on a node
- **Search your documents (RAG):** qwen3-embedding and NRP's managed Milvus vector database
- **Inference accelerators:** Qualcomm Cloud AI 100 cards, requested in a job like a GPU

Worked examples: the PEARC26 tutorial (serving, RAG), and a physics neural network trained in GPU jobs (CMS HATS)

https://training.nrp-nautilus.io

Labs and Classes

JupyterHub for Classes

Your class needs	A hub on NRP gives it
The same environment for every student	A container image you choose
No laptop setup on day one	JupyterLab in the browser, with campus login
GPUs for a neural-network unit	GPU options on the start menu, with limits
An AI assistant, without per-seat fees	Jupyter AI and the hosted LLMs, preconfigured
Shared datasets and notebooks	One shared volume in every server

- **Instructors:** contact us; we'll help you set up a hub for your lab or course
- It's one configuration file in a namespace you own
- **Students:** if your course needs GPUs, ask your professor to get in touch

Getting Started

1. Log in: **nrp.ai** (CILogon → UCLA)

2. Request a namespace: **get-started.nrp-nautilus.io**
PIs, faculty, staff, postdocs; students ask their PI

3. Try the LLMs: **nrp-openwebui.nrp-nautilus.io**

4. Run your first GPU job:
training.nrp-nautilus.io/ucla-health

Help

- Documentation: nrp.ai/documentation
- Support chat and email: nrp.ai/contact
- Labs, courses or a UCLA campus group: d4diaz@ucsd.edu

Live Demo

Acknowledgments

- NRP is supported by NSF awards OAC-1541349, OAC-1826967, OAC-2030508, OAC-1841530, OAC-2005369, OAC-2112167, CISE-1713149, CISE-2100237 and CISE-2120019
- Networks: CENIC, Pacific Wave, Internet2, MREN, GPN, NYSERNet, FLR, NEREN, Sun Corridor, OARnet, SCLR and the Albuquerque GigaPoP
- The NRP operations teams at SDSC, the University of Nebraska–Lincoln and MGHPCC, and the campuses that host and share their hardware, UCLA among them
- Clifford and UCLA Health IT's Research Tech Talks, for hosting

Questions?

Today's materials	`training.nrp-nautilus.io/ucla-health`
Portal and login	`nrp.ai`
Request a namespace	`get-started.nrp-nautilus.io`
Chat with the LLMs	`nrp-openwebui.nrp-nautilus.io`
LLM token	`nrp.ai/llmtoken`
GPUs free right now	`nrp.ai/viz/resources`
GPU jobs, docs	`nrp.ai/documentation/userdocs/running/gpu-pods`
Support	`nrp.ai/contact`

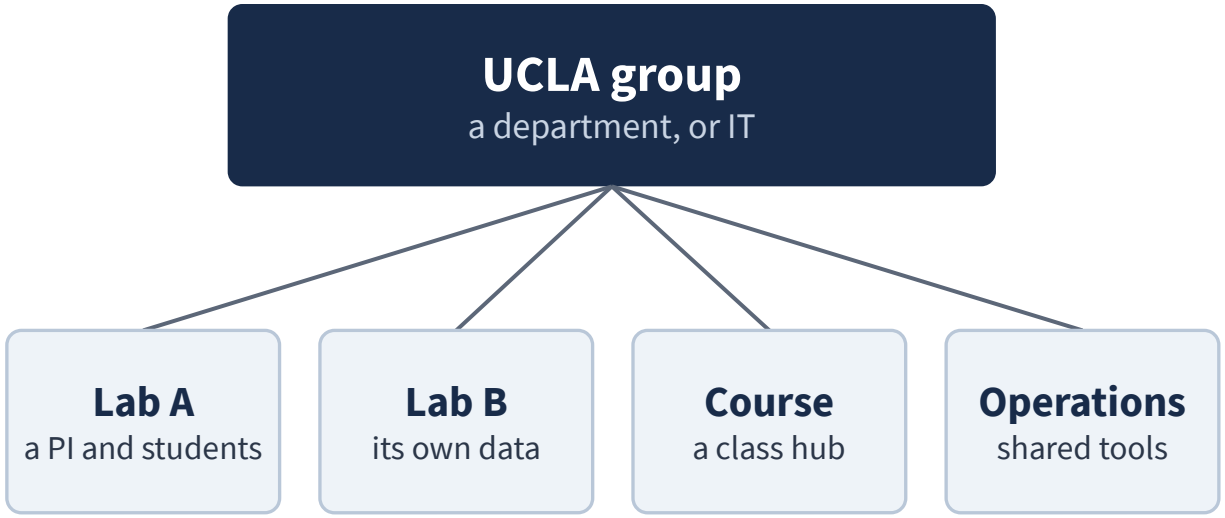
Daniel Diaz · d4diaz@ucsd.edu

Backup

Glossary

Term	Meaning
Kubernetes	Software that runs containers across many machines; NRP's cluster is one
Container	Your software and everything it needs, packaged to run anywhere
Pod	One or more containers running together on one machine
Job	A pod that runs to completion, then releases its resources
Namespace	Your group's workspace on the cluster: members, jobs, data
Volume (PVC)	Storage in your namespace that outlives any one pod
kubectl	The command-line tool you use to talk to the cluster
GPU memory	How big a model or batch fits on one GPU; the main number to ask for
Open-weight model	A model whose weights are published for anyone to run
Token (LLM)	A piece of a word, about ¾ of an English word
Embedding	A vector that captures meaning, for search and RAG
RAG	Look up relevant text first, then ask the model to answer from it

Organizing Groups at UCLA



Data lives on volumes inside a namespace, so each lab's data is visible only to its members.

- Namespaces can be nested under a campus-level group
- Its admins create namespaces for labs and courses, and name their admins
- Fresno State does this through central IT and HPC staff
- UCLA has no campus-level admin yet: a department or group could take that role

GPU Resource Keys

GPU	Request it as	Notes
Most GPUs (A10, 3090, L4, 2080 Ti, L40, V100 ...)	`nvidia.com/gpu: 1`	Pick a model or memory with node affinity
NVIDIA RTX A6000	`nvidia.com/rtxa6000: 1`	48 GB
NVIDIA A40	`nvidia.com/a40: 1`	48 GB
NVIDIA A100 (80 GB)	`nvidia.com/a100: 1`	Quota is zero by default: access request at nrp.ai/reservations
A100 slice (MIG 1g.10gb)	`nvidia.com/mig-small: 1`	10 GB slice of an A100
NVIDIA H100 / H200 / GH200	`nvidia.com/h100 · h200 · gh200`	Reserved for contributors and LLM serving
Any of the big ones, preemptibly	`priorityClassName: opportunistic`	Runs on idle capacity; can be preempted

Check live with kubectl get nodes -L nvidia.com/gpu.product

%%ai Magics

```
%load_ext jupyter_ai_magics

%%ai openai-chat:gpt-oss
What is the National Research Platform, in two sentences?

# explain the most recent traceback, no copy-paste
%ai error openai-chat:gpt-oss

# {variables} are replaced by their live values
%%ai openai-chat:gpt-oss
My readings are {readings} and my code printed {average}.
Is that the mean of every reading after the first?
```

- The chat panel can't see the kernel; the magics can
- %ai error reads the last traceback
- The silent bug: code that runs and prints the wrong number has no traceback, so you have to say what you meant
- %ai list is Jupyter AI's built-in list, not NRP's catalog; the name you type passes through to NRP