

Request for Proposal: GPU Cloud and AI Infrastructure

Template from RFPhub.com. Replace everything in square brackets, adjust the Must-have/Nice-to-have priorities, and delete this note and the instructions section before you issue the RFP.

Instructions for the buyer (delete before issuing)

A GPU cloud evaluation comes down to two questions: can the provider deliver the GPUs you need, where and when you need them, and will that capacity perform and stay healthy under your workloads. The example schedule below runs about 10 weeks from kickoff to contract award; if you need a large reserved cluster with a future delivery date, start capacity conversations earlier and allow for acceptance testing after signature.

- 1. Characterize your workloads and size the demand (weeks 1-2).**
Separate training, fine-tuning and inference, and for each one record the model sizes, the largest single job in GPUs, expected GPU-hours per month, data volumes and latency targets. Write down the numbers vendors will price against: [GPU models required], GPU counts, regions, start date, term, storage capacity and expected data transfer out.
- 2. Assemble the buying team.** Include ML research and engineering leads, the platform or infrastructure team that will run the clusters, networking, security, privacy and data protection, finance, procurement and legal. Bring finance in early: decide whether you can commit to reserved capacity for a term, and how GPU costs will be charged back to teams.
- 3. Choose the capacity model, and decide whether you need an RFI (weeks 2-3).** Decide how much demand should be reserved and how much can run on demand or on interruptible capacity, and whether you need bare-metal nodes, managed Kubernetes, managed Slurm or several of these. If you haven't compared renting with buying your own cluster, do that first. If you don't know which providers can deliver your GPU models in your regions by your start date, issue a short RFI asking for available capacity, lead times, access models and pricing structure, and use the answers to build a shortlist.

4. **Tailor and issue the RFP (weeks 3-6).** Delete requirements that don't apply, adjust priorities, and add your workload details to Section 1. Describe a reference training job and an inference workload you will use in the proof of concept, so vendors can plan for it. Give vendors three weeks to respond, with a written question-and-answer window in the first week.
5. **Score the responses (weeks 6-7).** Have each evaluator score independently against the rubric before comparing notes, and set aside any vendor that misses a Must-have. Check every 'yes' for whether the capacity is installed and running today or still to be delivered, and whether each feature is generally available or still on the roadmap.
6. **Run a proof of concept with two or three vendors (weeks 7-9).** Run your reference training job across several nodes on the same GPU models and network fabric you would be buying, not a demo environment. Measure how throughput scales as nodes are added, data loading and checkpoint times, and inference latency under realistic load. Ask the vendor to show what happens when a node fails mid-job, and log how quickly support answers your questions.
7. **Negotiate and award (weeks 9-10).** Use the pricing table to compare like for like. Settle capacity delivery dates and remedies, acceptance testing before billing starts, node replacement times, commitment flexibility, renewal caps and exit terms before you sign, not after.

Contents

1. Introduction and Background
2. Project Objectives
3. Scope of Work
4. Requirements
5. Vendor Questions
6. Evaluation Criteria and Scoring
7. Pricing Response
8. Submission Guidelines
9. Timeline

1. Introduction and Background

[Company Name] is seeking proposals for GPU cloud capacity and supporting AI infrastructure for our model training, fine-tuning and inference workloads, including cluster networking, storage, scheduling and operational support. This RFP describes our requirements, the questions we need answered, and how proposals will be evaluated.

Organization background

- [Describe your organization: industry, size, locations]
- [Regulatory or contractual requirements that apply]
- [Current AI workloads: training, fine-tuning and inference, with model sizes and the largest single job in GPUs]
- [GPU models required, GPU counts, regions and required start date]
- [Existing GPU capacity on premises or in the cloud, and current monthly GPU-hours]
- [Schedulers, frameworks and MLOps tools in use]
- [Training data volumes, where the data is stored, and data residency requirements]

2. Project Objectives

- Secure guaranteed access to [GPU models required] in [regions] from [date], with burst capacity for peak demand
- Run multi-node training jobs of up to [number] GPUs with predictable performance
- Serve models in production with autoscaling and defined availability
- Keep data, checkpoints and model weights in approved regions, with tenant isolation that meets our security requirements
- Make GPU utilization and cost visible by team and project

3. Scope of Work

The selected vendor will be responsible for:

- Provision of [number] [GPU models required] as reserved capacity in [regions], plus on-demand or burst capacity
- Cluster networking, shared file storage and object storage sized for our workloads

- Managed scheduling (Kubernetes, Slurm or both) and validated software images
- Acceptance testing of reserved capacity before handover
- Onboarding, data transfer and migration of initial workloads, in phases agreed with [Company Name]
- Monitoring, support, node replacement and service management for the contract term

4. Requirements

For each requirement, enter a response code and a short explanation. Y = available today in the proposed edition; P = partially met or needs configuration/services (explain); R = on the roadmap (give a date); N = not supported. Must-have requirements are pass/fail; Nice-to-have requirements add points.

4.1 Functional

ID	Requirement	Priority	Response	Vendor comments
F1	Guaranteed capacity of [number] [GPU models required] in [regions] for the committed term, available from a committed delivery date. State whether reserved capacity is dedicated to us or allocated from a shared pool.	Must		
F2	Burst capacity above our reservation, on demand or reserved at short notice. State how on-demand capacity is allocated when demand is high, and the usual lead time for each GPU model.	Must		
F3	Spot or preemptible capacity for interruptible jobs, with a documented interruption notice period.	Nice		
F4	Access to GPU nodes as bare metal or virtual machines. State which options are available for each GPU model, and any documented virtualization overhead.	Must		
F5	A managed scheduler for our workloads: Kubernetes, Slurm or both [delete as appropriate]. State which parts the provider operates and upgrades, and which parts we operate.	Must		
F6	Gang scheduling for distributed jobs, so a multi-node job starts only when all of its GPUs are available, with job queueing, priority-based preemption and topology-aware placement of the nodes in a job.	Must		

ID	Requirement	Priority	Response	Vendor comments
F7	Multi-tenancy for our internal teams: separate projects or namespaces, per-team GPU quotas, priorities and fair-share policies, with GPU usage and cost reported by team and project.	Must		
F8	GPU partitioning or fractional GPU allocation for development, notebooks and small inference workloads.	Nice		
F9	Validated machine and container images with GPU drivers, communication libraries and [frameworks we use], kept current by the provider.	Must		
F10	Automatic health checks that detect failing GPUs, memory errors, overheating and network interconnect faults, take the affected node out of scheduling, and alert us.	Must		
F11	Automatic restart of interrupted training jobs from the last checkpoint once a failed node has been replaced.	Nice		
F12	GPU observability: utilization, memory use, power draw, temperature and error counts per GPU, node, job and team, in the console and through an API, with alerts for idle GPUs, failed jobs and hardware errors.	Must		
F13	Managed inference endpoints for our own models and open-weight models, with autoscaling on request rate, latency or queue depth, including scale to zero.	Nice		
F14	Inference serving features such as dynamic batching, quantized models, several models sharing one GPU, and canary or blue-green rollout of new model versions with rollback.	Nice		

4.2 Technical and architecture

ID	Requirement	Priority	Response	Vendor comments
T1	Published specifications for each proposed node type: GPU model and count, GPU memory, CPU, system memory, local NVMe capacity, and the GPU-to-GPU interconnect topology within the node.	Must		
T2	A dedicated RDMA network fabric (InfiniBand or RoCE) for multi-node training. For a cluster of [number] GPUs, state the per-GPU fabric bandwidth, the topology, and any oversubscription between switch tiers.	Must		

ID	Requirement	Priority	Response	Vendor comments
T3	A non-blocking fabric across our full cluster size. If the fabric is oversubscribed at any tier, state where and by how much.	Nice		
T4	All nodes in a reserved cluster located in one data center on one fabric. State the largest cluster of [GPU models required] you can provide on a single fabric, and whether we can add nodes to the same fabric later.	Must		
T5	Shared file storage for training data and checkpoints (a parallel or distributed file system) sized for our cluster. State sustained read throughput per node and in aggregate, and checkpoint write throughput, to be verified in the proof of concept.	Must		
T6	Object storage for datasets, artifacts and model weights with a widely used object storage API, and stated throughput between object storage and GPU nodes.	Must		
T7	GPU capacity, storage and backups in [regions]. State which GPU models are available in each region today, and which are planned, with dates.	Must		
T8	Private connectivity to our data centers and cloud environments (dedicated links or IPsec VPN), and an option for moving [data volume] of training data in bulk at onboarding.	Must		
T9	Control over GPU driver, firmware and system software versions, including version pinning, advance notice of upgrades, and maintenance scheduled around our long training runs.	Must		

4.3 Integration

ID	Requirement	Priority	Response	Vendor comments
I1	Single sign-on to the console and API via SAML 2.0 or OIDC, with role-based access control. State whether user and group provisioning via SCIM is supported.	Must		
I2	A documented REST API and CLI for all provisioning and management tasks, plus infrastructure-as-code support (for example, a Terraform provider).	Must		
I3	Export of GPU, node and job metrics to our monitoring tools through a Prometheus-compatible endpoint or OpenTelemetry,	Must		

ID	Requirement	Priority	Response	Vendor comments
	and of audit and security logs to our SIEM.			
I4	Use of our own container images and private container registry, and integration with our existing MLOps, experiment tracking and model registry tools [list tools], without requiring the provider's own tooling.	Must		
I5	Export of billing and usage data, by API or scheduled file, to our cost management tools.	Nice		

4.4 Security and compliance

ID	Requirement	Priority	Response	Vendor comments
S1	A current SOC 2 Type II report and ISO/IEC 27001 certification covering the proposed services and data center locations, available under NDA.	Must		
S2	Attestations required for our data, such as [HIPAA, PCI DSS or FedRAMP at the required level]. Make this Must-have if these apply to your training or inference data.	Nice		
S3	Isolation between tenants at the compute, network fabric and storage layers, with documentation of how it is enforced wherever nodes, fabric or storage are shared.	Must		
S4	Dedicated single-tenant nodes or clusters for sensitive workloads. Make this Must-have if your training data or models require it.	Nice		
S5	Secure wiping of local storage, clearing of GPU memory and firmware integrity checks before any node we used is reassigned to another customer.	Must		
S6	Encryption in transit and at rest for all storage, including datasets, checkpoints and model weights. State whether customer-managed keys are supported for each storage type.	Must		
S7	Confidential computing options that protect data in use on the GPU, with remote attestation.	Nice		
S8	A data processing agreement and a contractual commitment that our data, model weights, logs and backups stay in [regions or jurisdictions], with a current list of data center locations and sub-	Must		