MCP Server Service : User Guide

❖ How to connect

Each user receives a private, fully isolated workspace that is separated from all other users. You can connect to the service using an SSH client such as macOS Terminal, PuTTY, Windows PowerShell. Use the connection details provided by the system administrator.

Connection command format: # SSH *Username@Server* -p *Port*

Example connection command: # SSH user01@ai-g3.sit.kmutt.ac.th -p 2201

❖ Workspace Structure (Node Profiles)

- Base Linux: A minimal node with immediate access to the GPU accelerator. It is suitable for users who prefer to install additional software or specialized frameworks, such as C++ HIP or JAX, by themselves.
- PyTorch: A node with PyTorch for AMD ROCm preinstalled and ready to use, allowing you to begin writing model-training scripts immediately.
- vLLM: A node for providing an OpenAI-compatible API service, with the AMD ROCm build of the vLLM engine ready to use.
- Ollama: A node for application-oriented workloads. It allows you to load conversational models quickly and supports “ollama run” immediately.

❖ Service Rules and Precautions

- Because the service has only one AMD MI210 GPU, the GPU is shared fairly among all users. A Resource Monitor checks for usage beyond the defined limits. Each user is limited to 5 GB of GPU VRAM. If your process exceeds this limit, it will be terminated immediately. You may see a “Killed” message or an error message from your process. Your SSH terminal will remain available for continued use.

- Each user is allocated 16 GB of memory. You may load CSV files, arrays, or other data up to a maximum of 16 GB per user. If the limit is exceeded, the system will terminate your program immediately.

❖ Usage Tips (Best Practices)

To ensure smooth operation, always configure your tool to limit VRAM usage before running a workload, as described below:

- **For PyTorch users: Configure PyTorch to reserve only your GPU quota before loading a model. Always place the following line near the beginning of your Python code:**

  python

  import torch

  # The MI210 GPU has 64 GB of VRAM. Your quota is 5 GB, so set the fraction to 0.078.

  torch.cuda.set_per_process_memory_fraction(0.078)

- **For vLLM users: Configure the engine so that it does not reserve excessive VRAM. Its default behavior is to reserve 90%. Add the following option when running the command:**

  bash

  python3 -m vllm.entrypoints.openai.api_server --model facebook/opt-125m --gpu-memory-utilization 0.07

- **For Ollama users: Ollama loads model weights into VRAM according to the size of the model.**

Rule: Do not run models larger than the 7B-8B parameter range.

Supported models: “ollama run llama3:8b” (uses approximately 4.5 GB) or “ollama run gemma:2b”

Unsupported model: “ollama run llama3:70b” (uses more than 40 GB and will be killed while loading into GPU VRAM)

** Model sizes are based on the versions available when this guide was written. Check the current model size again before use.

❖ Safe Storage Area (Data Preservation)

Data stored within the Docker container may be lost when the container is rebooted. A dedicated 100 GB persistent storage area has been created for you. Save your work, source code, and models only in the “/workspace” folder. Data stored outside this area, such as in your user “/home” directory, may be lost during system maintenance.