

GPU erőforrások használata konténer alapon a Slurm szolgáltatásban a HUN-REN Cloudon



Integrált szolgáltatások - Singularity

- Konténer futtatókörnyezet, hasonló a Dockerhez
- HPC környezetekhez tervezve
- Docker image-ek letöltése és építése
- Nincs szükség root jogosultságra



Singularity

- Konténer letöltés
 - **singularity pull**
 - image fájl → *.sif* formátom
 - forrásként a Dockerhub-ot kell megjelölni



```
User x Slurm_PaaS x + v
konrad@slurm-master:~$ singularity pull pytorch.sif docker://pytorch/pytorch:2.1.2-cuda11.8-cudnn8-runtime
INFO:      Converting OCI blobs to SIF format
INFO:      Starting build...
INFO:      Fetching OCI image...
27.3MiB / 27.3MiB [=====] 100 % 20.6 MiB/s 0s
9.6MiB / 9.6MiB [=====] 100 % 20.5 MiB/s 0s
173.4MiB / 3.5GiB [==>-----] 5 % 20.6 MiB/s 2m44s
█
```



Singularity

- *.def* fájlok
 - saját személyre szabott könyvtárak letöltése vagy definiálása
 - konténeren belül környezeti változók beállítása
 - script-ek futtatása



```
Slurm_PaaS User
GNU nano 7.2
Bootstrap: docker
From: pytorch/pytorch:2.1.2-cuda11.8-cudnn8-runtime

%post
# Update package lists
apt-get update

# Install system dependencies
apt-get install -y \
    git \
    wget \
    curl \
    vim \
    build-essential \
    python3-pip

# Install Python packages
pip3 install --no-cache-dir \
    numpy \
    pandas \
    matplotlib \
    scikit-learn \
    jupyter

# Clean up
apt-get clean
rm -rf /var/lib/apt/lists/*

%environment
# Set environment variables
export LC_ALL=C
export PYTHONPATH=/workspace:$PYTHONPATH

%runscript
# Default command when container is run
exec /bin/bash "$@"
```

Singularity

- Konténer build-elés
 - **singularity build**
 - **--fakeroot** flag → nincs szükség root jogosultságra
 - image fájl → *.sif* formátum



```
User x Slurm_PaaS x + v
konrad@slurm-master:~$ nano pytorch.def
konrad@slurm-master:~$ singularity build --fakeroot pytorch.sif pytorch.def
INFO: Starting build...
INFO: Fetching OCI image...
9.6MiB / 9.6MiB [=====] 100 % 36.7 MiB/s 0s
27.3MiB / 27.3MiB [=====] 100 % 36.7 MiB/s 0s
3.5GiB / 3.5GiB [=====] 100 % 36.7 MiB/s 0s
INFO: Extracting OCI image...
```



Singularity –Shell mode

- Interaktív, emulált shell a worker node-on

```
[Slurm_PaaS] x + v
konrad@slurm-master:~/demo_singularity$ srun -p batch_gpu_g2.large_8 --gres=gpu:1 --pty singularity shell pytorch.sif
INFO: Setting 'NVIDIA_VISIBLE_DEVICES=all' to emulate legacy GPU binding.
INFO: Setting --writable-tmpfs (required by nvidia-container-cli)
Singularity> nvidia-smi
Thu Mar 27 13:04:57 2025
```

NVIDIA-SMI 550.144.03			Driver Version: 550.144.03			CUDA Version: 12.4		
GPU	Name	Perf	Persistence-M	Bus-Id	Disp.A	Volatile	Uncorr. ECC	
Fan	Temp		Pwr:Usage/Cap		Memory-Usage	GPU-Util	Compute M.	
							MIG M.	
0	GRID V100DX-8C		On	00000000:06:00.0	Off			0
N/A	N/A	P0	N/A / N/A	1MiB / 8192MiB		0%	Default	N/A

```
Singularity>
```

```
Processes:
```

GPU	GI	CI	PID	Type	Process name	GPU Memory Usage
	ID	ID				
No running processes found						

```
Singularity> █
```



Singularity

- Pytorch konténer futtatása
 - terminálon keresztül
 - **srun singularity exec**

```
User      Slurm_PaaS
konrad@slurm-master:~$ srun singularity exec pytorch.sif python3 pytorch_code.py
INFO:      Setting 'NVIDIA_VISIBLE_DEVICES=all' to emulate legacy GPU binding.
INFO:      Setting --writable-tmpfs (required by nvidia-container-cli)
Epoch [10/100], Loss: 1.4294
Epoch [20/100], Loss: 1.3638
Epoch [30/100], Loss: 1.3253
Epoch [40/100], Loss: 1.3016
Epoch [50/100], Loss: 1.2862
Epoch [60/100], Loss: 1.2756
Epoch [70/100], Loss: 1.2674
Epoch [80/100], Loss: 1.2611
Epoch [90/100], Loss: 1.2570
Epoch [100/100], Loss: 1.2534
Prediction for test input: 0.3572
Training complete!
konrad@slurm-master:~$
```

```
User      Slurm_PaaS
GNU nano 7.2                                pytorch.batch *
#!/bin/bash
#SBATCH --job-name=pytorch_code              # Job name
#SBATCH --output=pytorch_job.out             # Standard output log file
#SBATCH --error=pytorch_job.err              # Standard error log file
#SBATCH --gres=gpu:nvidia:1

# Run a script inside the Singularity container:
srun singularity exec pytorch.sif python3 pytorch_code.py
```



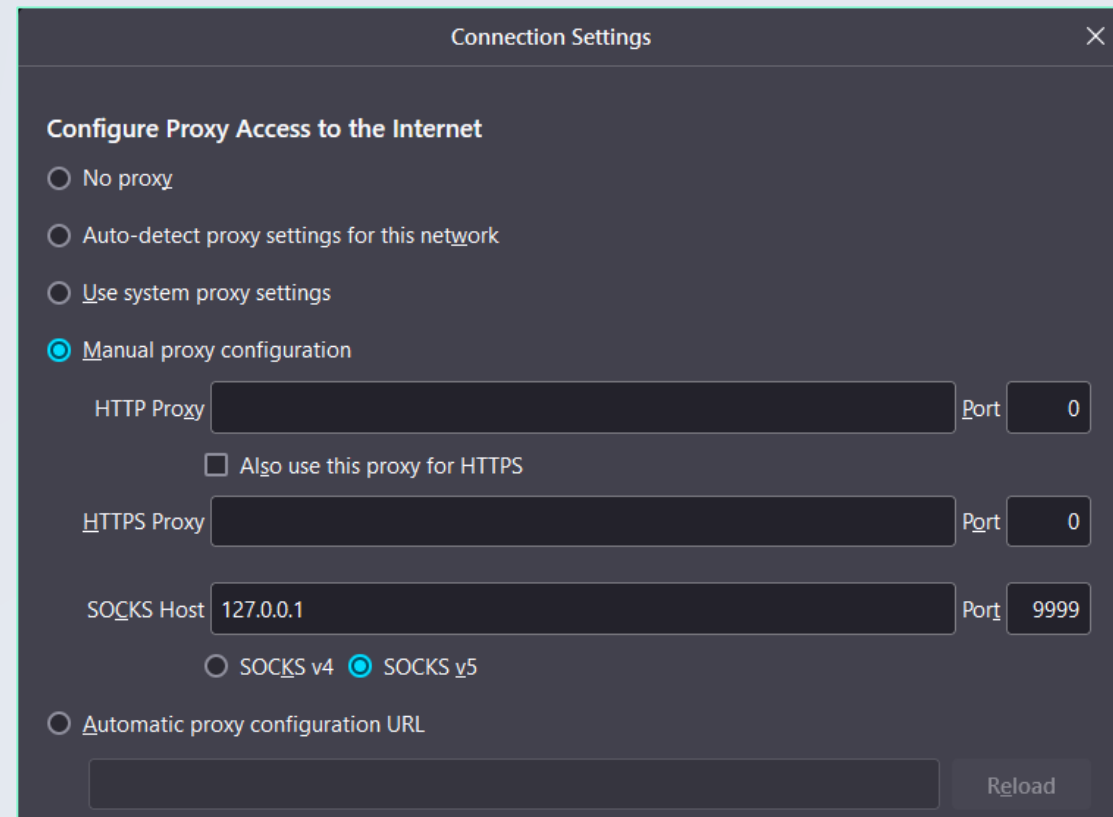
Interaktív job

- Lépések
 - `.sif` konténer fájl `/home` könyvtár alá másolása
 - `.batch` fájl létrehozása, paraméterezése és ellenőrzése
 - `.batch` fájl beküldése a feladatütemezőn keresztül az **sbatch** **<feladat_név>.batch** paranccsal
 - Adott feladatot futtató gép IP címének lekérdezése
 - Webes fejlesztői környezet elérése



Interaktív job

- Proxy csatlakozás
 - interaktív job-ok esetén
- Javasolt böngésző
 - Mozilla Firefox
- Proxy port
 - opcionális non-standard port



Connection Settings

Configure Proxy Access to the Internet

☐ No proxy

☐ Auto-detect proxy settings for this network

☐ Use system proxy settings

☒ Manual proxy configuration

HTTP Proxy Port

☐ Also use this proxy for HTTPS

HTTPS Proxy Port

SOCKS Host Port

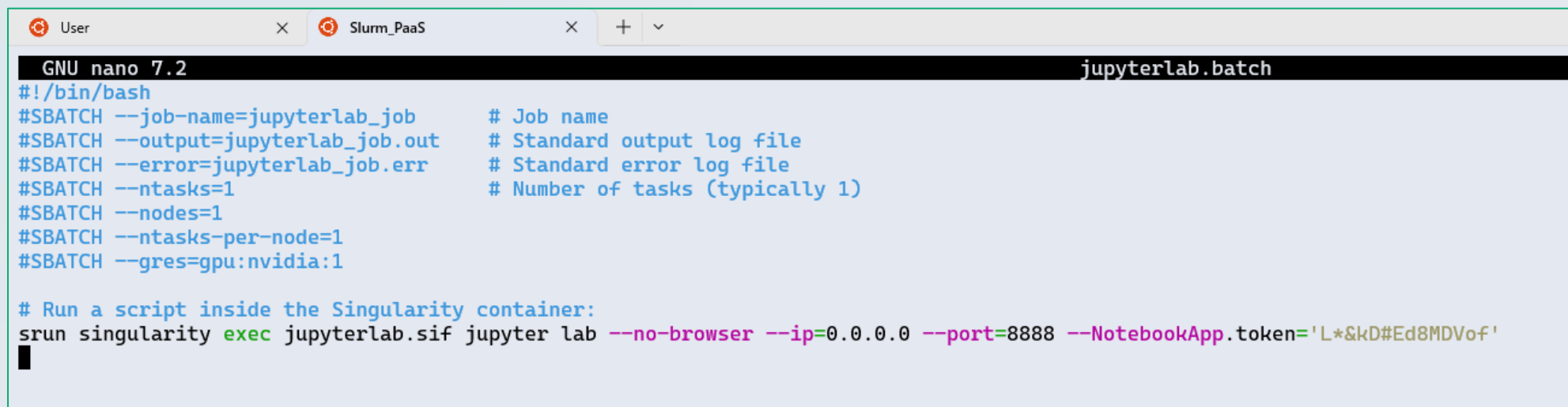
☐ SOCKS v4 ☒ SOCKS v5

☐ Automatic proxy configuration URL

```
User x Slurm_PaaS + v
banfikonrad@Wubalubadubdub:~$ ssh konrad@slurm.science-cloud.hu -i OpenStack -D 9999
Enter passphrase for key 'OpenStack':
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 5.15.0-131-generic x86_64)
```

Interaktív job

- Jupyterlab - példa *.batch* fájl
 - **#SBATCH --gres=gpu:nvidia:1** → job szintű GPU hozzáférés
 - **srun** → futtató parancs
 - **singularity exec** → konténerizált függőségek
 - Jupyterlab paraméterek



```
GNU nano 7.2 jupyterlab.batch
#!/bin/bash
#SBATCH --job-name=jupyterlab_job      # Job name
#SBATCH --output=jupyterlab_job.out    # Standard output log file
#SBATCH --error=jupyterlab_job.err     # Standard error log file
#SBATCH --ntasks=1                    # Number of tasks (typically 1)
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=1
#SBATCH --gres=gpu:nvidia:1

# Run a script inside the Singularity container:
srun singularity exec jupyterlab.sif jupyter lab --no-browser --ip=0.0.0.0 --port=8888 --NotebookApp.token='L*&kD#Ed8MDVof'
```



Interaktív job

- Interaktív partíció kiválasztása a futtatáshoz → **-p** paraméter



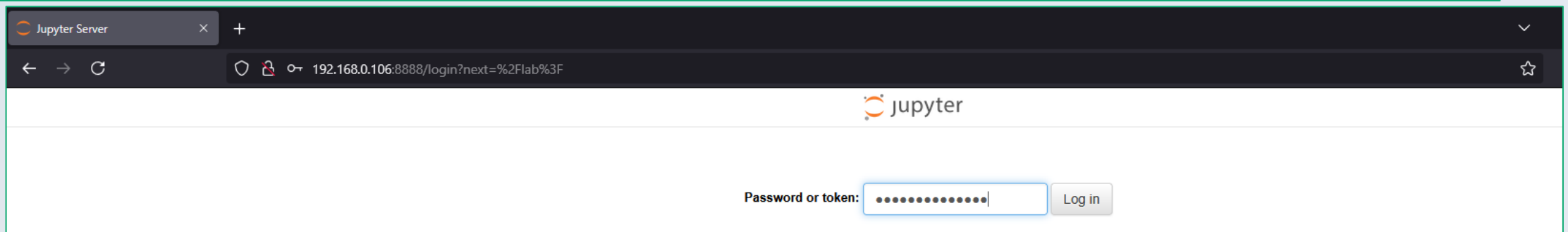
```
Slurm_PaaS x User x + v
konrad@slurm-master:~$ cp /storage/shared_batch_examples/example.batch ~
konrad@slurm-master:~$ pv /storage/shared_singularity_images/jupyterlab.sif > ~/jupyterlab.sif
6.49GiB 0:00:43 [ 152MiB/s] [=====>] 100%
konrad@slurm-master:~$ nano jupyterlab.batch
konrad@slurm-master:~$ sinfo
PARTITION      AVAIL  TIMELIMIT  NODES  STATE NODELIST
batch_cpu_m2.large      up 1-00:00:00      4   idle slurm-master,slurm-worker-cpu-m2-large-[1-3]
batch_gpu_g2.large_8*    up 1-00:00:00      7   idle slurm-worker-gpu-g2-large-[1-7]
batch_gpu_g2.xlarge_16   up 1-00:00:00      2   idle slurm-worker-gpu-g2-xlarge-[1-2]
batch_gpu_g2.2xlarge_32  up 1-00:00:00      1   idle slurm-worker-gpu-g2-2xlarge-1
interactive_cpu_m2.large  up 7-00:00:00      4   idle slurm-master,slurm-worker-cpu-m2-large-[1-3]
interactive_gpu_g2.large_8 up 7-00:00:00      7   idle slurm-worker-gpu-g2-large-[1-7]
konrad@slurm-master:~$ sbatch -p interactive_gpu_g2.large_8 jupyterlab.batch
Submitted batch job 3
konrad@slurm-master:~$ squeue -u $USER
      JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
        3 interacti jupyterl   konrad  R        0:06      1 slurm-worker-gpu-g2-large-1
konrad@slurm-master:~$ █
```



Interaktív job

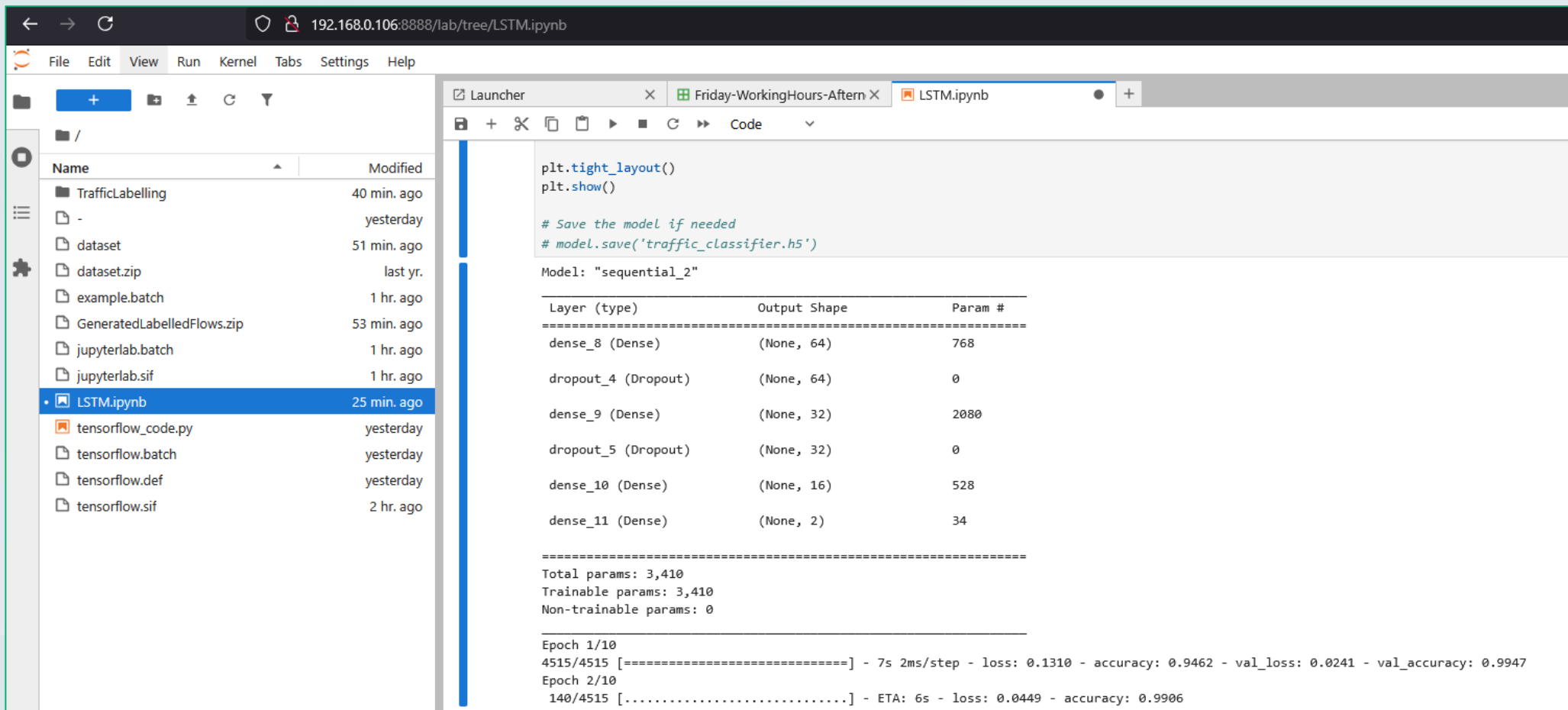
- Webes fejlesztői környezet elérése a lekérdezett IP cím alapján
 - port → 8888
 - token alapú autentikáció → a batch fájlban került definiálásra

```
Slurm_PaaS User
konrad@slurm-master:~$ scontrol getaddrs $(scontrol show job 3 | grep "NodeList=slurm" | cut -d '=' -f 2) | col2 | cut -d ':' -f 1
192.168.0.106
konrad@slurm-master:~$ squeue -u $USER
      JOBID PARTITION    NAME    USER  ST       TIME  NODES NODELIST(REASON)
        3 interacti jupyterl  konrad  R       1:24      1 slurm-worker-gpu-g2-large-1
konrad@slurm-master:~$
```



Interaktív job

- Webes fejlesztői környezet használata
 - `/home` mappa elérhető a konténerizált környezetben is



The screenshot shows a JupyterLab web interface. The left sidebar contains a file browser with a list of files and folders. The main area on the right shows a code editor with the following content:

```
plt.tight_layout()
plt.show()

# Save the model if needed
# model.save('traffic_classifier.h5')
```

Model: "sequential_2"

Layer (type)	Output Shape	Param #
dense_8 (Dense)	(None, 64)	768
dropout_4 (Dropout)	(None, 64)	0
dense_9 (Dense)	(None, 32)	2080
dropout_5 (Dropout)	(None, 32)	0
dense_10 (Dense)	(None, 16)	528
dense_11 (Dense)	(None, 2)	34

=====
Total params: 3,410
Trainable params: 3,410
Non-trainable params: 0

Epoch 1/10
4515/4515 [=====] - 7s 2ms/step - loss: 0.1310 - accuracy: 0.9462 - val_loss: 0.0241 - val_accuracy: 0.9947
Epoch 2/10
140/4515 [.....] - ETA: 6s - loss: 0.0449 - accuracy: 0.9906

Interaktív job

- GPU erőforrás ellenőrzése
 - elérhető a webes fejlesztői környezetben
 - **singularity exec --nv** → parapméter, GPU hozzáférés a konténer által
 - globális beállítás, a felhasználónak nem szükséges használnia

The screenshot displays a web-based interactive environment. On the left is a file explorer for the directory `/demo_interactive/`. It lists several files and folders: `TrafficLabelling` (modified 12 hr. ago), `Check_GPU.ipynb` (modified now), `dataset.zip` (modified last yr.), `LSTM.ipynb` (modified 7 min. ago), and `Preprocess.ipynb` (modified 12 hr. ago). The `Check_GPU.ipynb` file is selected.

On the right is a terminal window showing the output of a command. The output includes the NVIDIA-SMI command, the driver version (550.144.03), the CUDA version (12.4), and a table of GPU information. Below this, it shows the processes running on the GPU, including the `/usr/bin/python` process.

```
flow) (3.2.2)
Mon Feb 17 08:02:10 2025

+-----+
| NVIDIA-SMI 550.144.03                  Driver Version: 550.144.03          CUDA Version: 12.4                  |
+-----+-----+
| GPU   Name                               Persistence-M | Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf              Pwr:Usage/Cap |      Memory-Usage | GPU-Util  Compute M. |
|                                           MIG M.         |
+-----+-----+
|  0    GRID V100DX-8C                     On          | 00000000:06:00.0 Off |             0         |
| N/A   N/A   P0               N/A /  N/A   | 2370MiB / 8192MiB |    0%      Default   |
|                                           N/A           |
+-----+-----+

+-----+
| Processes:                                |
| GPU   GI    CI        PID   Type   Process name                      GPU Memory |
|      ID    ID                             |              Usage |
+-----+-----+
|  0    N/A  N/A       87019    C   /usr/bin/python                   2369MiB   |
+-----+-----+
```

MP

- Közös memóriájú párhuzamos programozás
- A fő szál párhuzamos szálakat indít, amelyek ugyanazt a memóriaterületet osztják meg
- Egyetlen gépen történő kód párhuzamosításra használják, több mag/processzor esetén



MP

```
GNU nano 7.2
#!/bin/bash
#SBATCH --job-name=omp_job      # Job name
#SBATCH --nodes=1               # Run on a single node
#SBATCH --ntasks=1             # Run a single task
#SBATCH --cpus-per-task=4       # Use 4 CPU cores
#SBATCH --mem=4G                # Request 4 GB of memory
#SBATCH --time=00:30:00         # Time limit: 30 minutes
#SBATCH --output=%j_output.log  # Standard output log
#SBATCH --error=%j_error.log    # Standard error log

# Set the number of OpenMP threads to match requested CPUs
export OMP_NUM_THREADS=$SLURM_CPUS_PER_TASK

# Compile the OpenMP program (if needed)
gcc -fopenmp mp.c -o mp

# Run the program
srun ./mp
```

```
Slurm_PaaS
konrad@slurm-master:~$ nano mp.batch
konrad@slurm-master:~$ sbatch mp.batch
Submitted batch job 110
konrad@slurm-master:~$ cat 110_
110_error.log  110_output.log
konrad@slurm-master:~$ cat 110_output.log
Sum: 2499999975000000.000000
Time taken: 0.114638 seconds
Number of threads used: 4
konrad@slurm-master:~$ cat 110_error.log
konrad@slurm-master:~$ █
```



MPI

- OpenMPI
 - Elosztott memóriájú párhuzamos programozás
 - Üzenetküldés olyan folyamatok között, amelyek saját, elkülönített memóriaterülettel rendelkeznek
 - Több gépen átívelő párhuzamos számításokhoz tervezve egy számítási klaszterben
 - Potenciális erőforrás limitáció



MPI – 2 node

```
Slurm_PaaS
GNU nano 7.2
#!/bin/bash
#SBATCH --job-name=mpi_2n_job
#SBATCH --output=mpi_2n.out
#SBATCH --error=mpi_2n.err
#SBATCH --ntasks-per-node=2
#SBATCH --nodes=2
#SBATCH --gres=gpu:nvidia:1

# Compile the OpenMPI program (if needed)
mpicc -o mpi_code mpi_code.c

# Run MPI script
srun -n 2 ./mpi_code
```

```
Slurm_PaaS
konrad@slurm-master:~/demo_mpi$ sbatch mpi_2n.batch
Submitted batch job 260
konrad@slurm-master:~/demo_mpi$ cat mpi_2n.out
Process rank 1 running on node slurm-worker-gpu-g2-large-1
Rank 1: I received: Message from master
Process rank 3 running on node slurm-worker-gpu-g2-large-2
Rank 3: I am a worker process
Process rank 2 running on node slurm-worker-gpu-g2-large-2
Rank 2: I am a worker process
Process rank 0 running on node slurm-worker-gpu-g2-large-1
Rank 0: I am the master process
konrad@slurm-master:~/demo_mpi$
```

- OpenMPI
 - CPU és RAM erőforrások egyidejű használata
 - Slurm node-ok közötti kommunikáció



MPI – 4 node

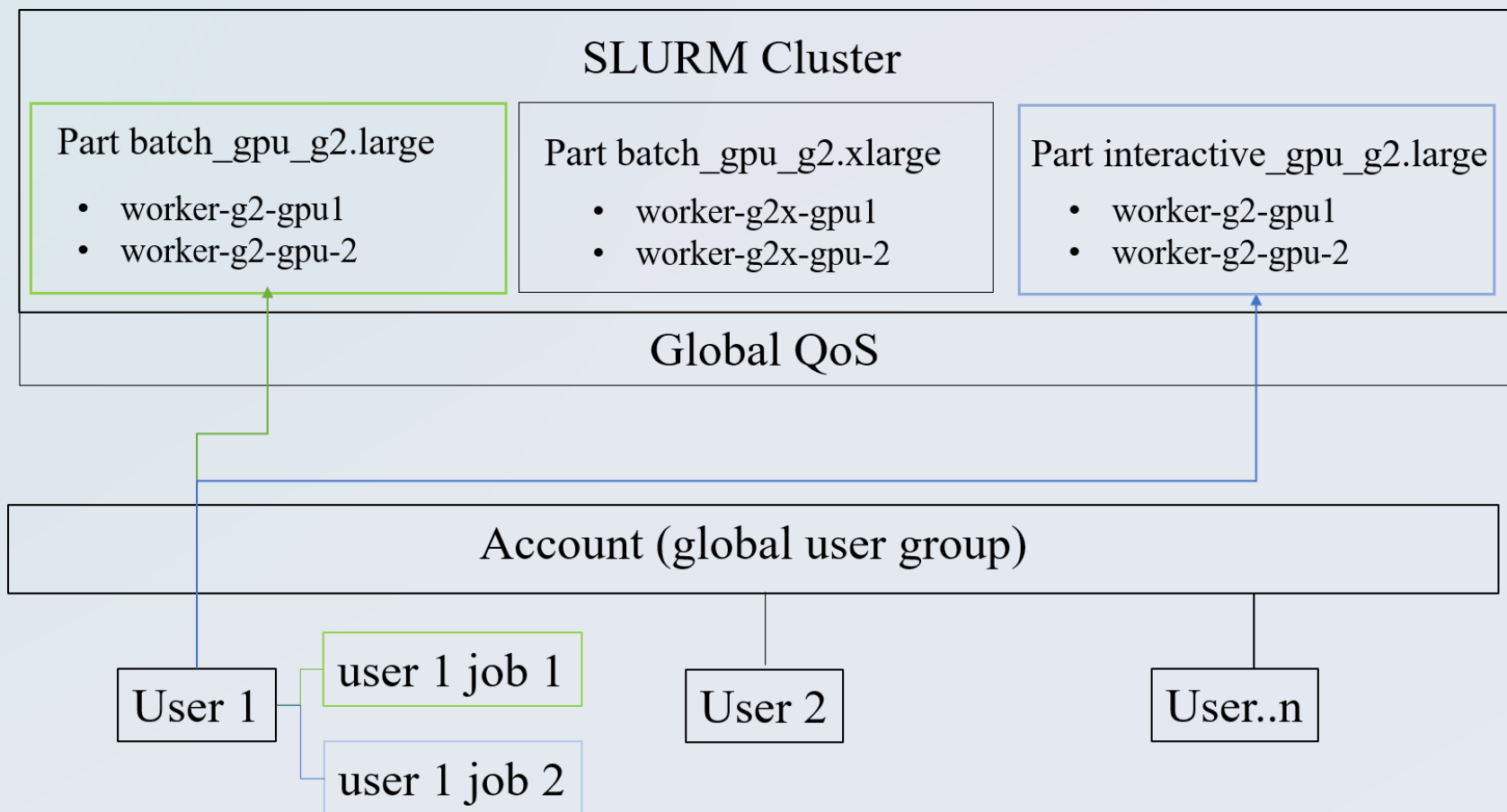
```
Slurm_PaaS
konrad@slurm-master:~/demo_mpi$ sbatch mpi_4n.batch
Submitted batch job 261
konrad@slurm-master:~/demo_mpi$ cat mpi_4n.out
Process rank 1 running on node slurm-worker-gpu-g2-large-1
Rank 1: I received: Message from master
Process rank 5 running on node slurm-worker-gpu-g2-large-3
Rank 5: I am a worker process
Process rank 0 running on node slurm-worker-gpu-g2-large-1
Rank 0: I am the master process
Process rank 3 running on node slurm-worker-gpu-g2-large-2
Rank 3: I am a worker process
Process rank 7 running on node slurm-worker-gpu-g2-large-4
Rank 7: I am a worker process
Process rank 4 running on node slurm-worker-gpu-g2-large-3
Rank 4: I am a worker process
Process rank 6 running on node slurm-worker-gpu-g2-large-4
Rank 6: I am a worker process
Process rank 2 running on node slurm-worker-gpu-g2-large-2
Rank 2: I am a worker process
konrad@slurm-master:~/demo_mpi$ █
```

```
Slurm_PaaS
konrad@slurm-master:~/demo_mpi$ sbatch mpi_4n.batch
Submitted batch job 261
konrad@slurm-master:~/demo_mpi$ cat mpi_4n.out
Process rank 1 running on node slurm-worker-gpu-g2-large-1
Rank 1: I received: Message from master
Process rank 5 running on node slurm-worker-gpu-g2-large-3
Rank 5: I am a worker process
Process rank 0 running on node slurm-worker-gpu-g2-large-1
Rank 0: I am the master process
Process rank 3 running on node slurm-worker-gpu-g2-large-2
Rank 3: I am a worker process
Process rank 7 running on node slurm-worker-gpu-g2-large-4
Rank 7: I am a worker process
Process rank 4 running on node slurm-worker-gpu-g2-large-3
Rank 4: I am a worker process
Process rank 6 running on node slurm-worker-gpu-g2-large-4
Rank 6: I am a worker process
Process rank 2 running on node slurm-worker-gpu-g2-large-2
Rank 2: I am a worker process
konrad@slurm-master:~/demo_mpi$ █
```

- OpenMPI
 - CPU és RAM erőforrások egyidejű használata
 - Slurm node-ok közötti kommunikáció



Slurm – kvóta limitációk

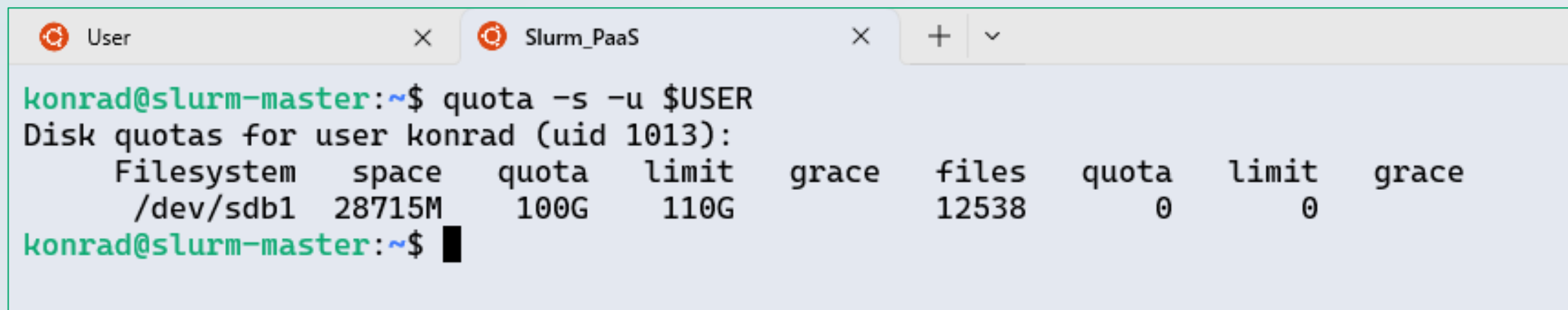


Slurm – kvóta limitációk

- Felhasználói szintű kvóták
 - felhasználónként 100GB tárhely áll rendelkezésre
 - */storage* hálózati meghajtó
 - felhasználónként 1 feladat futhat adott időben
 - lehet parameter sweep feladat (**n** db job steps)
 - felhasználónként 3 feladat lehet beütemezve egyszerre
 - *queue*

Slurm – kvóta limitációk

- Felhasználónként 100GB tárhely áll rendelkezésre
 - `quota -s -u $USER`



A terminal window with two tabs: 'User' and 'Slurm_PaaS'. The command `quota -s -u $USER` has been executed, showing disk quotas for user konrad (uid 1013). The output is a table with 10 columns: Filesystem, space, quota, limit, grace, files, quota, limit, grace. The data row shows /dev/sdb1 with 28715M space, 100G quota, 110G limit, 0 grace, 12538 files, 0 quota, 0 limit, and 0 grace.

```
konrad@slurm-master:~$ quota -s -u $USER
Disk quotas for user konrad (uid 1013):
   Filesystem    space   quota   limit   grace   files   quota   limit   grace
    /dev/sdb1  28715M   100G   110G         12538         0         0
konrad@slurm-master:~$
```



Slurm – kvóta limitációk

- Felhasználónként 1 feladat futhat adott időben
- Felhasználónként 3 feladat lehet beütemezve egyszerre

```
User x Slurm_PaaS x + v
konrad@slurm-master:~$ sbatch -p interactive_gpu_g2.large_8 jupyterlab.batch
Submitted batch job 158
konrad@slurm-master:~$ sbatch -p batch_gpu_g2.xlarge_16 jupyterlab.batch
Submitted batch job 159
konrad@slurm-master:~$ sbatch -p batch_gpu_g2.2xlarge_32 jupyterlab.batch
Submitted batch job 160
konrad@slurm-master:~$ sbatch -p interactive_gpu_g2.large_8 jupyterlab.batch
sbatch: error: QOSMaxSubmitJobPerUserLimit
sbatch: error: Batch job submission failed: Job violates accounting/QOS policy (job submit limit, user's size and/or time limits)
konrad@slurm-master:~$ squeue -o "%.18i %.20P 15j %.8u %.8T %.10M %.20R"
      JOBID      PARTITION 15j      USER      STATE      TIME      NODELIST(REASON)
      160 batch_gpu_g2.2xlarge 15j      konrad      PENDING      0:00      (QOSMaxJobsPerUserLi
      159 batch_gpu_g2.xlarge_ 15j      konrad      PENDING      0:00      (QOSMaxJobsPerUserLi
      158 interactive_gpu_g2.l 15j      konrad      RUNNING      0:24      slurm-worker-gpu-g2-large-1
konrad@slurm-master:~$ █
```



Slurm – kvóta limitációk

- Feladat szintű kvóták
 - Futási idő limitáció → *CANCELED* feladat státusz
 - Addig számított eredmények elérhetőek
 - **Batch** partíciók → 24 óra
 - **Interactive** partíciók → 7 nap



Slurm – kvóta limitációk

- Cél → maximális erőforrás kihasználtság fair módon
- Igény és rendelkezésre álló erőforrás esetén
 - megengedőbb kvóta limitációk
 - más *Batch* és *Interactive* partíció felosztás
 - több GPU erőforrás → skálázható klaszter
 - további példa kódok és előre build-elt image-ek



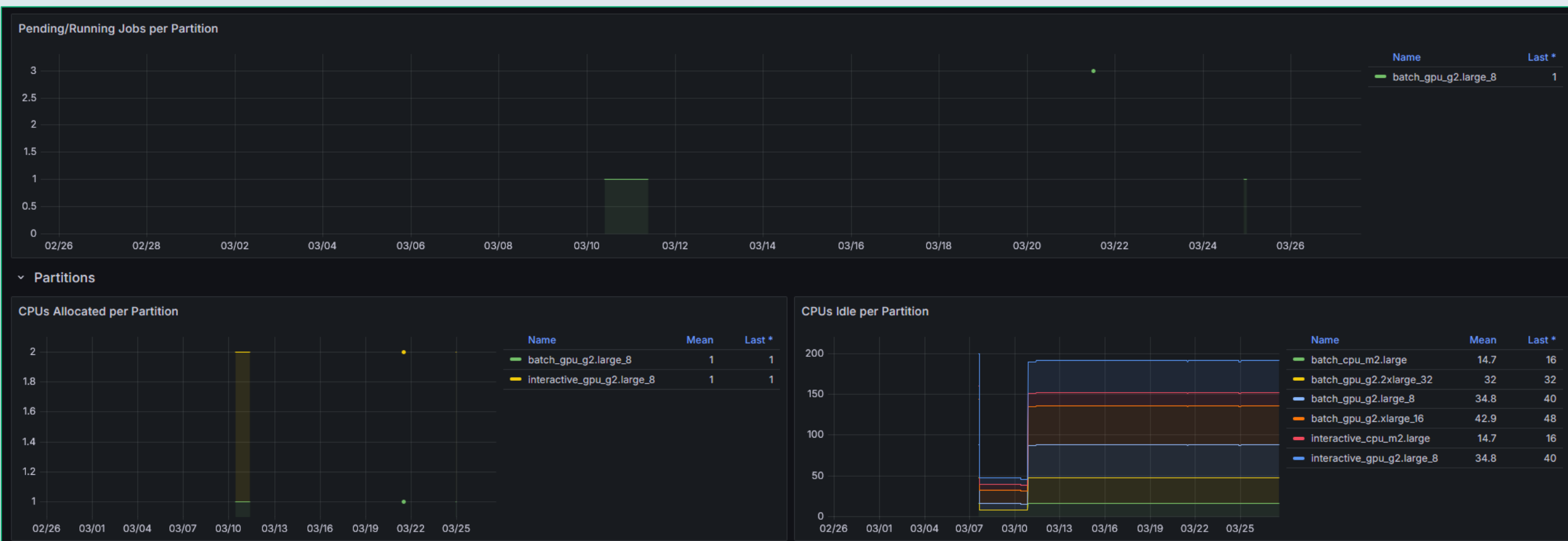
Monitoring

- Erőforrás-kihasználtság nyomon követése
 - GPU
 - CPU
 - Partíciók
- Publikus dashboard
- <https://jobs.slurm.science-cloud.hu/public-dashboards/a54b9e235e194854b9f452622bb8fff6>



Monitoring

- Csak a job státusza elérhető



Összefoglalás

- Egy elosztott, többfelhasználós rendszerben:
 - hogyan tudnak a felhasználók **egyéni függőségeket** definiálni?
 - *Singularity*
 - hogyan tudunk **dinamikusan** és **fair módon** különböző GPU erőforrásokat allokálni?
 - *GPU partíciók*
 - hogyan tudunk **párhuzamos számítást** támogató környezetet biztosítani?
 - *MP & MPI*
 - *Parameter Sweep*



Köszönöm a figyelmet!

