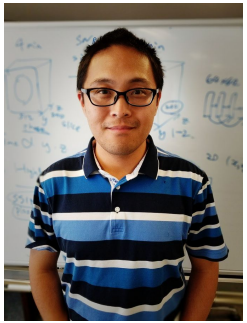
# Creating AI Work Groups Within the Enterprise: Developers Share Their Best Practices

GTC Silicon Valley 2019 - Breakout Session S9483

**Michael Balint**
Senior Product Manager
NVIDIA
@michaelbalint

**Markus Weber**
Senior Product Manager
NVIDIA
@MarkusAtNVIDIA



SUBTLE MEDICAL

**Liren Zhu**
Head of Engineering
@SubtleMedical



NVIDIA.

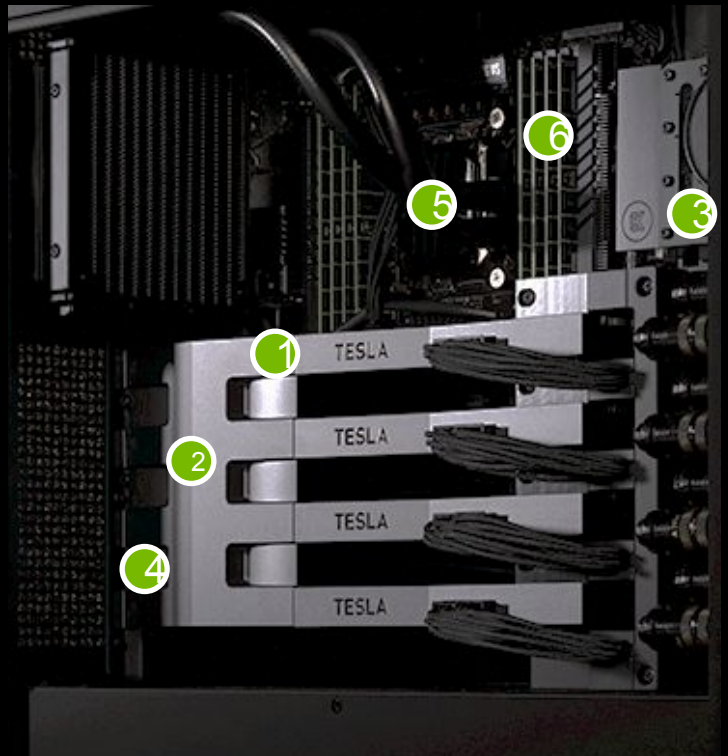
# Topics

- ➔ Intro to DGX Station
- ➔ Sharing Your GPU Compute Resource
  - Basic
  - Intermediate
  - Advanced
  - Futures
- ➔ Takeaways


NVIDIA.

# NVIDIA DGX STATION

Groundbreaking AI in Your Office




## The AI Workstation
for Data Science Teams

## Key Features

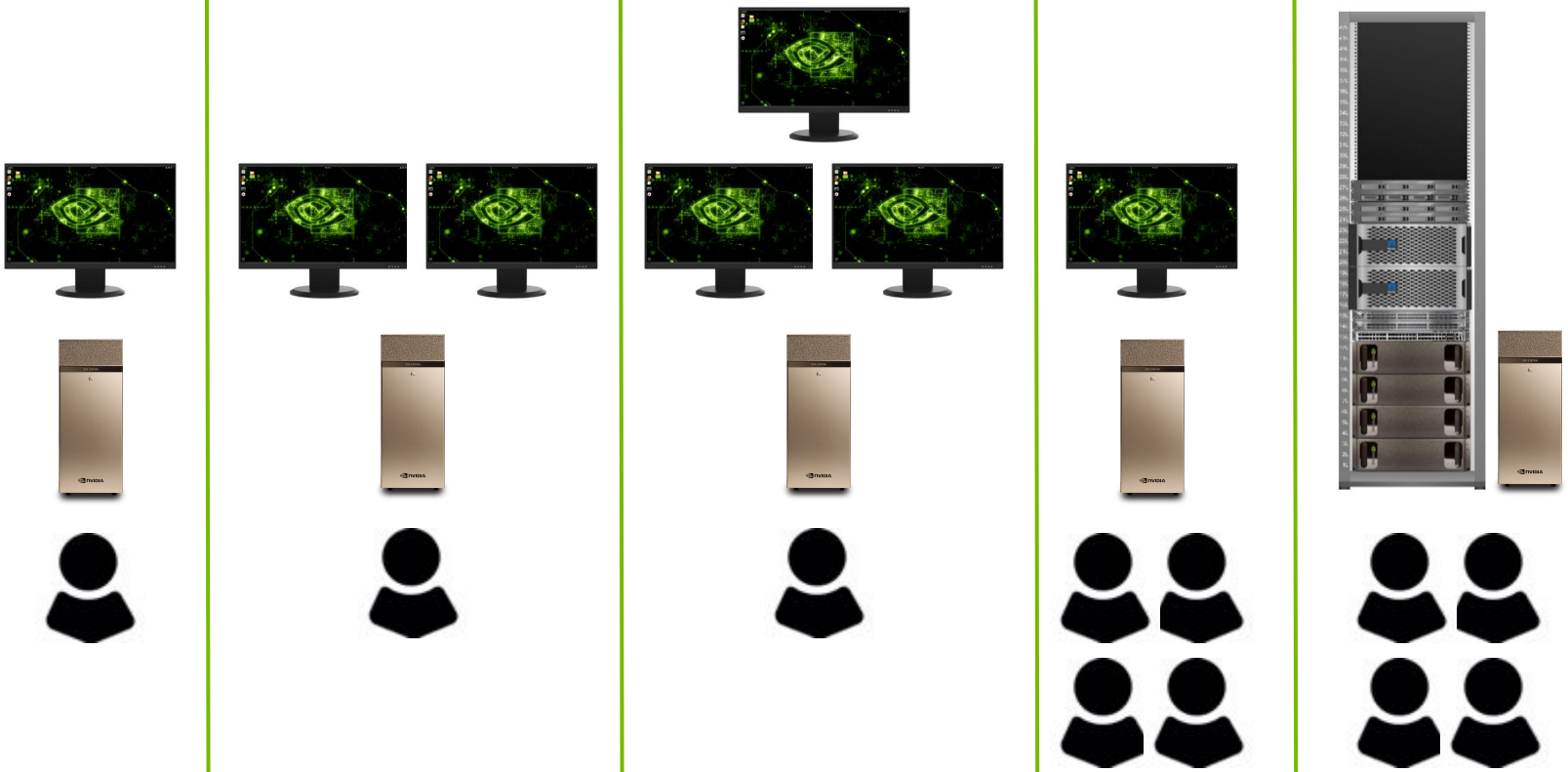
1. 4 x NVIDIA Tesla V100 GPU (32GB)
2. 2nd-gen NVLink (4-way)
3. Water-cooled design
4. 3 x DisplayPort (4K resolution)
5. Intel Xeon E5-2698 20-core
6. 256GB DDR4 RAM


# DGX STATION SPECIFICATIONS

## At a Glance

| | |
|---|---|
| **GPUs** | 4x NVIDIA® Tesla® V100 |
| **TFLOPS (GPU FP16)** | 500 |
| **GPU Memory** | 32 GB per GPU |
| **NVIDIA Tensor Cores** | 2,560 (total) |
| **NVIDIA CUDA Cores** | 20,480 (total) |
| **CPU** | Intel Xeon E5-2698 v4 2.2 GHz (20-core) |
| **System Memory** | 256 GB RDIMM DDR4 |
| **Storage** | Data: 3 x 1.92 TB SSD RAID 0<br>OS: 1 x 1.92 TB SSD |
| **Network** | Dual 10GBASE-T LAN (RJ45) |
| **Display** | 3x DisplayPort, 4K Resolution |
| **Additional Ports** | 2x eSATA, 2x USB 3.1, 4x USB 3.0 |
| **Acoustics** | < 35 dB |
| **Maximum Power Requirements** | 1500 W |
| **Operating Temperature Range** | 10 - 30 ºC |
| **Software** | Ubuntu Desktop Linux OS<br>DGX Recommended GPU Driver<br>CUDA Toolkit |

# Deployment Scenarios



# Today's Focus



NVIDIA.

# Today’s Focus



NVIDIA.

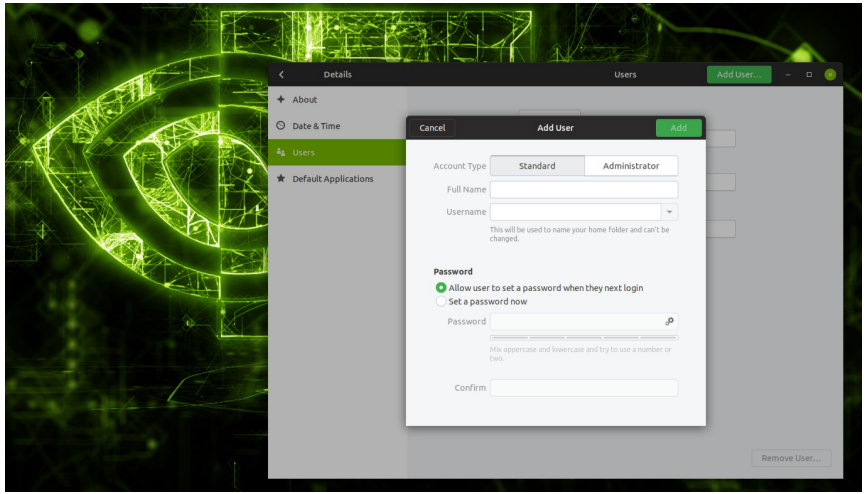
# Basic Sharing

# Adding OS Users

## CLI

```
adduser[--home DIR][--shell SHELL][--no-create-home][--uid ID][--firstuid ID][--lastuid ID][--gecos GECOS][--ingroup
GROUP | --gid ID][--disabled-password][--disabled-login][--add_extra_groups][--encrypt-home]USER
```

## GUI




## IT

- LDAP
- Active Directory
- NIS / NIS+
- …

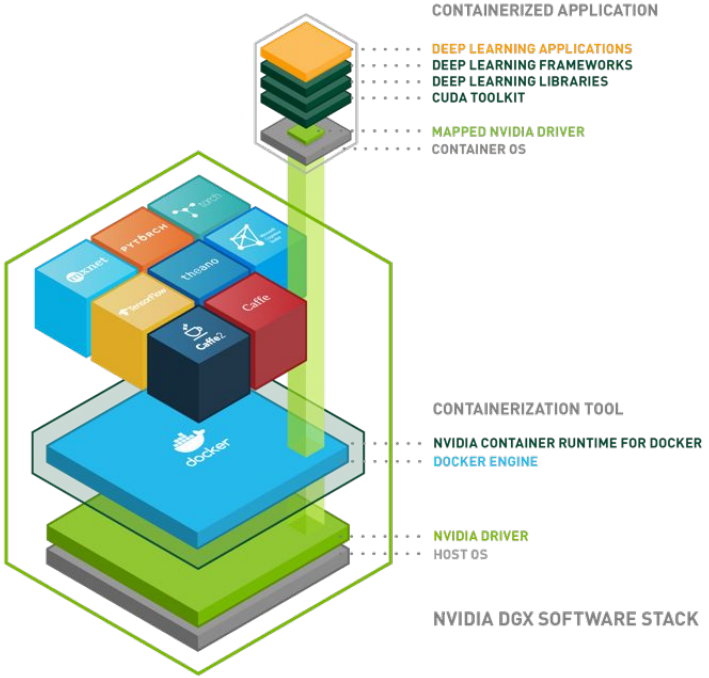


NVIDIA.

# Secure Shell (SSH)

```
$ systemctl status sshd
● ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2019-03-01 15:10:29 PST; 6 days ago
 Main PID: 1623 (sshd)
    Tasks: 1 (limit: 6143)
   CGroup: /system.slice/ssh.service
           └─1623 /usr/sbin/sshd -D
```

```
$ ssh demouser@10.110.42.125
The authenticity of host '10.110.42.125 (10.110.42.125)' can't be established.
ECDSA key fingerprint is SHA256:Zn2ucVJ4lYXXw2RBwRPZ2oMcf9pLS3XGWX0+rq6YFDQ.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.110.42.125' (ECDSA) to the list of known hosts.
demouser@10.110.42.125's password:
Welcome to NVIDIA DGX Station Version 4.0.5 (GNU/Linux 4.15.0-45-generic x86_64)
Last login: Fri Mar  8 13:03:13 2019 from 10.110.65.223
demouser@markusstation:~$
```




# DGX STATION SOFTWARE STACK

## Fully Integrated Software for Instant Productivity

Advantages:

Instant productivity with NVIDIA optimized deep learning frameworks

- Caffe, Caffe2, PyTorch, TensorFlow, MXNet, and others

Performance optimized across the entire stack

Faster Time-to-Insight with pre-built, tested, and ready to run framework containers

Flexibility to use different versions of libraries like libc, cuDNN in each framework container

# Using Individual GPUs

```
$ docker run -e NVIDIA_VISIBLE_DEVICES=0,1 --rm nvidia/cuda nvidia-smi
Thu Mar  7 23:34:24 2019
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 410.104      Driver Version: 410.104      CUDA Version: 10.0     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  Tesla V100-DGXS...  On   | 00000000:07:00.0  On |                    0 |
| N/A   36C    P0    37W / 300W |    432MiB / 16125MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   1  Tesla V100-DGXS...  On   | 00000000:08:00.0 Off |                    0 |
| N/A   36C    P0    38W / 300W |      0MiB / 16128MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID   Type   Process name                             Usage      |
|=============================================================================|
+-----------------------------------------------------------------------------+
```

```
$ docker run -e NVIDIA_VISIBLE_DEVICES=2,3 --rm nvidia/cuda nvidia-smi
Thu Mar  7 23:35:13 2019
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 410.104      Driver Version: 410.104      CUDA Version: 10.0     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  Tesla V100-DGXS...  On   | 00000000:0E:00.0 Off |                    0 |
| N/A   36C    P0    38W / 300W |      0MiB / 16128MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   1  Tesla V100-DGXS...  On   | 00000000:0F:00.0 Off |                    0 |
| N/A   36C    P0    40W / 300W |      0MiB / 16128MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID   Type   Process name                             Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

# Using Individual GPUs

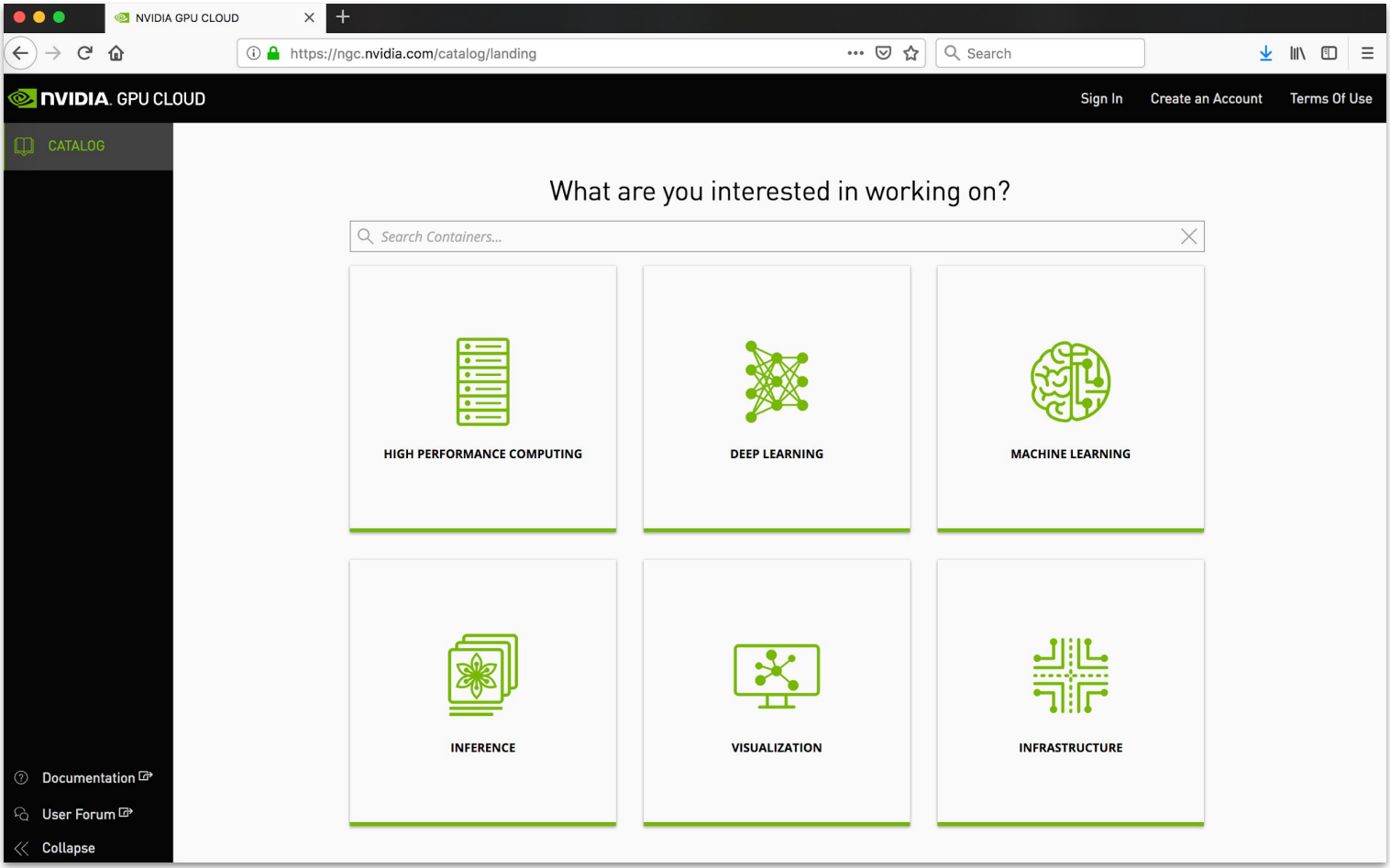
```
$ docker run -e NVIDIA_VISIBLE_DEVICES=0,1 --rm nvidia/cuda nvidia-smi
Thu Mar  7 23:34:24 2019
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 410.104      Driver Version: 410.104      CUDA Version: 10.0     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  Tesla V100-DGXS...  On   | 00000000:07:00.0  On |                    0 |
| N/A   36C    P0    37W / 300W |    432MiB / 16125MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   1  Tesla V100-DGXS...  On   | 00000000:08:00.0 Off |                    0 |
| N/A   36C    P0    38W / 300W |      0MiB / 16128MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID   Type   Process name                             Usage      |
|=============================================================================|
+-----------------------------------------------------------------------------+
```

```
$ docker run -e NVIDIA_VISIBLE_DEVICES=2,3 --rm nvidia/cuda nvidia-smi
Thu Mar  7 23:35:13 2019
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 410.104      Driver Version: 410.104      CUDA Version: 10.0     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  Tesla V100-DGXS...  On   | 00000000:0E:00.0 Off |                    0 |
| N/A   36C    P0    38W / 300W |      0MiB / 16128MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   1  Tesla V100-DGXS...  On   | 00000000:0F:00.0 Off |                    0 |
| N/A   36C    P0    40W / 300W |      0MiB / 16128MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID   Type   Process name                             Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

# NVIDIA GPU Cloud (NGC)



# Using Individual GPUs

## Real-World Execution Examples

**Joe:**

```
docker run -e NVIDIA_VISIBLE_DEVICES=0 --rm nvcr.io/nvidia/pytorch:19.02-py3 python \
/workspace/examples/upstream/mnist/main.py
```

```
docker run -e NVIDIA_VISIBLE_DEVICES=1 --rm nvcr.io/nvidia/pytorch:18.11-py2 python \
/workspace/examples/upstream/mnist/main.py
```

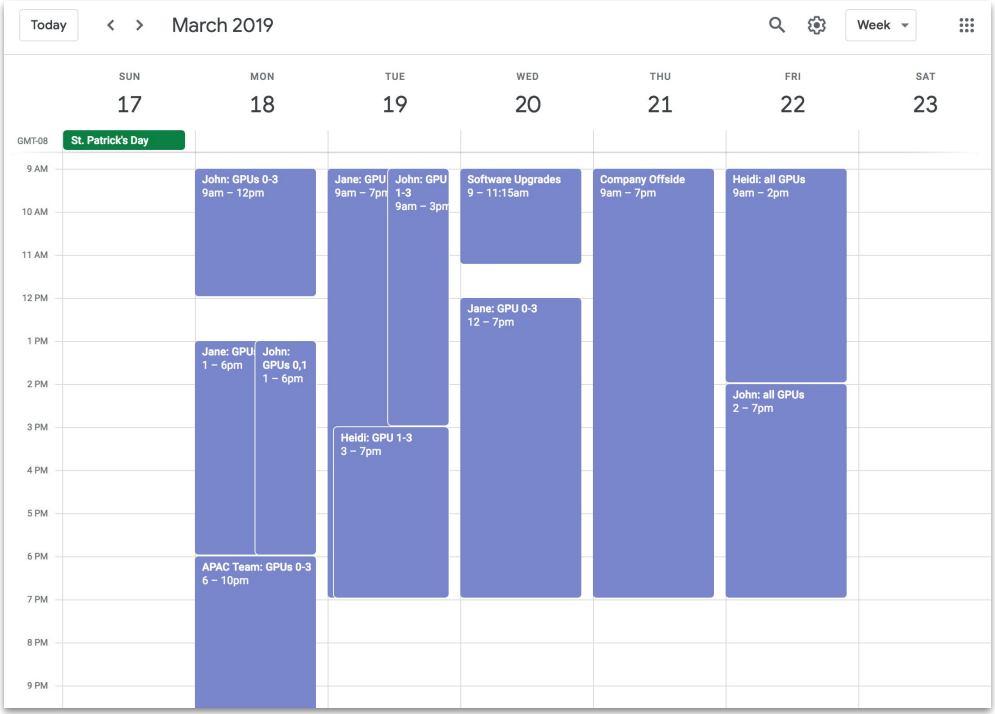
**Jane:**

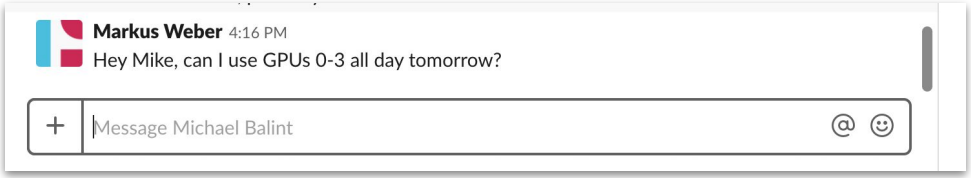
```
docker run --it -e NVIDIA_VISIBLE_DEVICES=2,3 --rm -v /home/jane/data/mnist:/data/mnist
nvcr.io/nvidia/tensorflow:19.02-py3
```



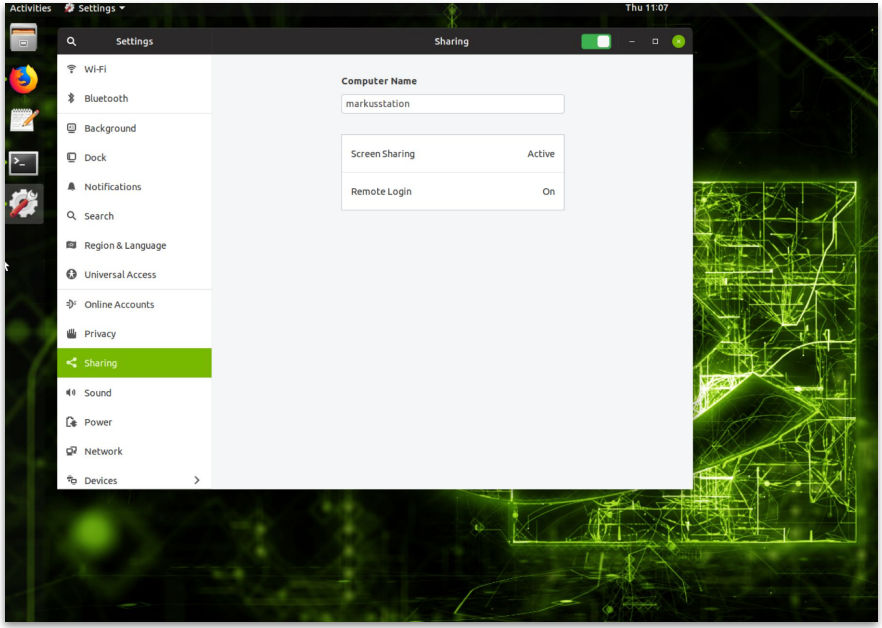
NVIDIA.

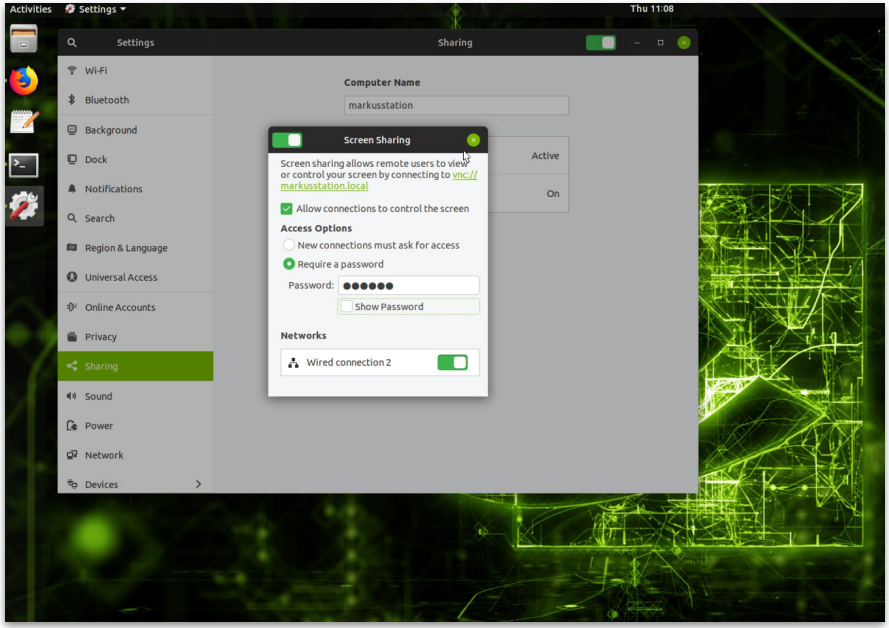
# “Manual” Sharing



# Using VNC


Activities
Settings
Thu 11:07
Settings
Sharing
Wi-Fi
Bluetooth
Background
Dock
Notifications
Search
Region & Language
Universal Access
Online Accounts
Privacy
Sharing
Sound
Power
Network
Devices
Computer Name
markusstation
Screen Sharing
Active
Remote Login
On


Activities
Settings
Thu 11:08
Settings
Sharing
Wi-Fi
Bluetooth
Background
Dock
Notifications
Search
Region & Language
Universal Access
Online Accounts
Privacy
Sharing
Sound
Power
Network
Devices
Computer Name
markusstation
Active
On
Screen Sharing
Screen sharing allows remote users to view or control your screen by connecting to vnc://markusstation.local
Allow connections to control the screen
Access Options
New connections must ask for access
Require a password
Password:
Show Password
Networks
Wired connection 2

# Intermediate Sharing



NVIDIA.

# Data Storage

## Internal RAID 0 | Internal RAID 5 | External DAS

```
$ lsblk
NAME   MAJ:MIN RM   SIZE RO TYPE
MOUNTPOINT
sda      8:0    0   1.8T  0 disk
├─sda1   8:1    0   487M  0 part  /boot/efi
└─sda2   8:2    0   1.8T  0 part  /
sdb      8:16   0   1.8T  0 disk
└─md0    9:0    0   5.2T  0 raid0 /raid
sdc      8:32   0   1.8T  0 disk
└─md0    9:0    0   5.2T  0 raid0 /raid
sdd      8:48   0   1.8T  0 disk
└─md0    9:0    0   5.2T  0 raid0 /raid
```

```
$ sudo configure_raid_array.py -m raid5
```

```
$ sudo configure_raid_array.py -m raid0
```

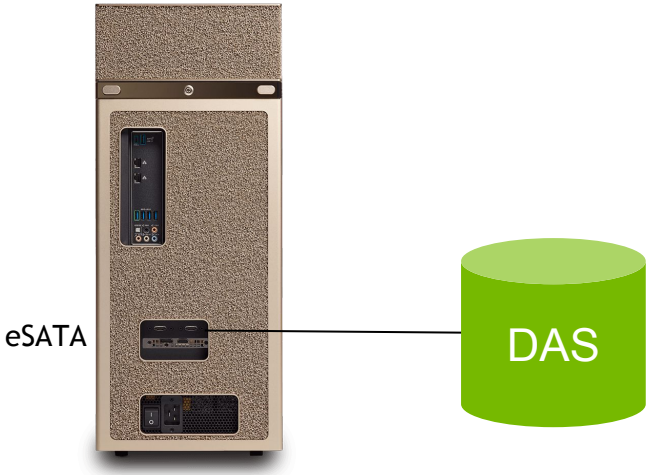
# Data Storage

## Internal RAID 0 | Internal RAID 5 | External DAS

```
$ lsblk
NAME   MAJ:MIN RM   SIZE RO TYPE
MOUNTPOINT
sda      8:0    0   1.8T  0 disk
├─sda1   8:1    0   487M  0 part  /boot/efi
└─sda2   8:2    0   1.8T  0 part  /
sdb      8:16   0   1.8T  0 disk
└─md0    9:0    0   5.2T  0 raid0 /raid
sdc      8:32   0   1.8T  0 disk
└─md0    9:0    0   5.2T  0 raid0 /raid
sdd      8:48   0   1.8T  0 disk
└─md0    9:0    0   5.2T  0 raid0 /raid
```

```
$ sudo configure_raid_array.py -m raid5
```

```
$ sudo configure_raid_array.py -m raid0
```



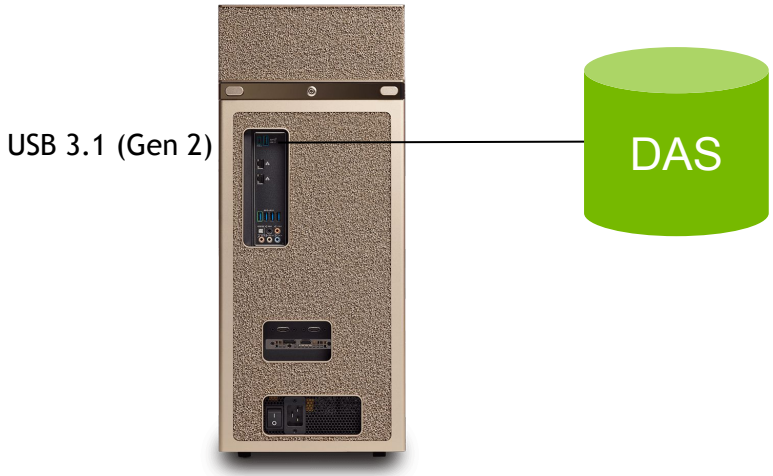
# Data Storage

## Internal RAID 0 | Internal RAID 5 | External DAS

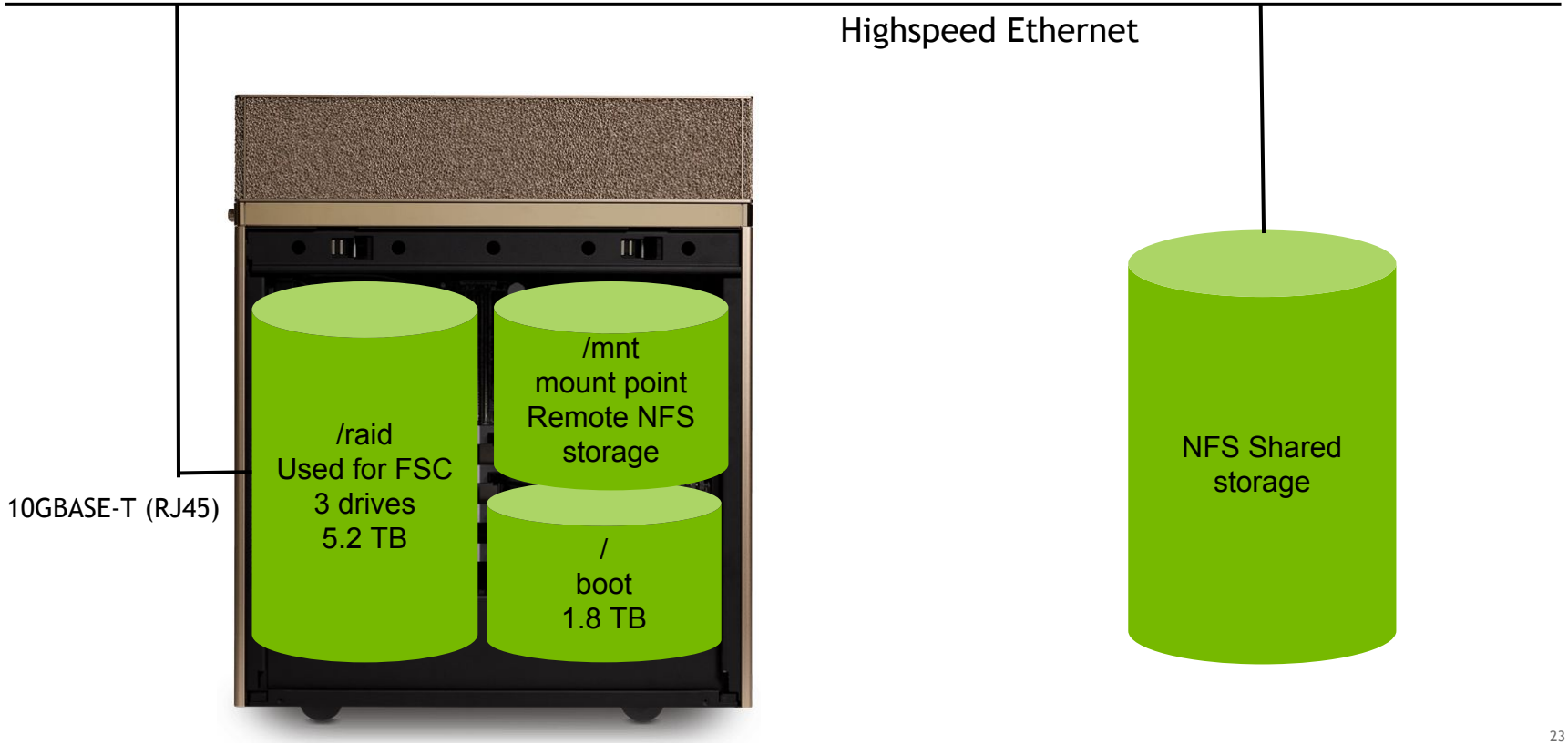
```
$ lsblk
NAME   MAJ:MIN RM   SIZE RO TYPE
MOUNTPOINT
sda      8:0    0   1.8T  0 disk
├─sda1   8:1    0   487M  0 part  /boot/efi
└─sda2   8:2    0   1.8T  0 part  /
sdb      8:16   0   1.8T  0 disk
└─md0    9:0    0   5.2T  0 raid0 /raid
sdc      8:32   0   1.8T  0 disk
└─md0    9:0    0   5.2T  0 raid0 /raid
sdd      8:48   0   1.8T  0 disk
└─md0    9:0    0   5.2T  0 raid0 /raid
```

```
$ sudo configure_raid_array.py -m raid5
```

```
$ sudo configure_raid_array.py -m raid0
```



# Configuring a NFS Cache



# Configure NFS mount and cache

- What is NFS fsc?
- FS-Cache (fsc) caches NFS client requests onto local storage
  - Improve NFS read I/O only
- Implemented using the `cachefilesd` daemon
  - Persistent cache is mounted at `/raid (`lsblk` to check)`
  - Install `cachefilesd` if not installed already (`systemctl status cachefilesd`)
- Add `fsc` to the `mount` command or in */etc/fstab*
- Why are we configuring it?
  - DGX Station includes ~5.2TB SSDs in RAID0
  - These SSDs can definitely be used for application caching
- Use NFS drives for long term data storage

NVIDIA

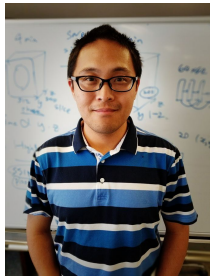
USA - United States

PLATFORMS › DEVELOPERS › COMMUNITY › SHOP DRIVERS › SUPPORT ABOUT NVIDIA ›

HOME DEEP LEARNING DRIVING GAMING PRO GRAPHICS AUTONOMOUS MACHINES HEALTHCARE AI PODCAST

## How Subtle Medical Is Using AI to Slash Costs, Risks of Medical Scans

July 23, 2018 by KAREN XIA

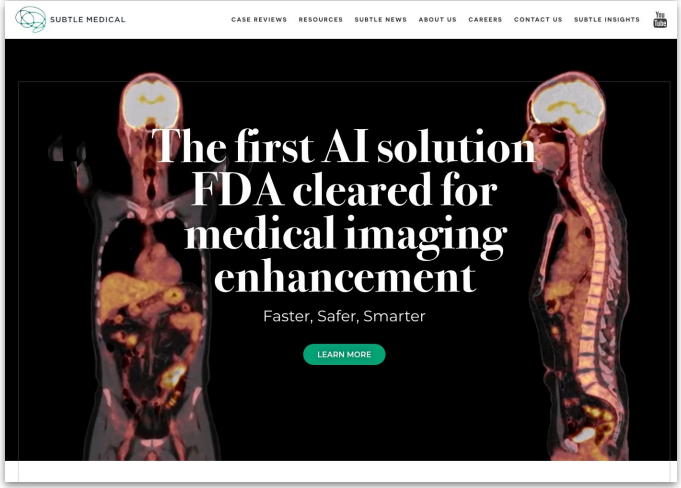



SUBTLE MEDICAL

## Liren Zhu

Head of Engineering

@SubtleMedical



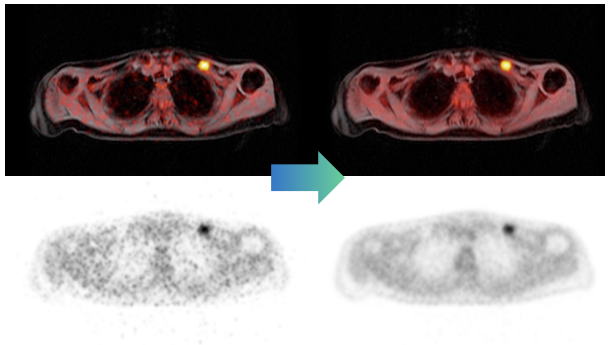



NVIDIA

# Subtle Medical AI Technologies

*Apply Deep Learning to enhance lower quality images to high quality diagnostic standards*

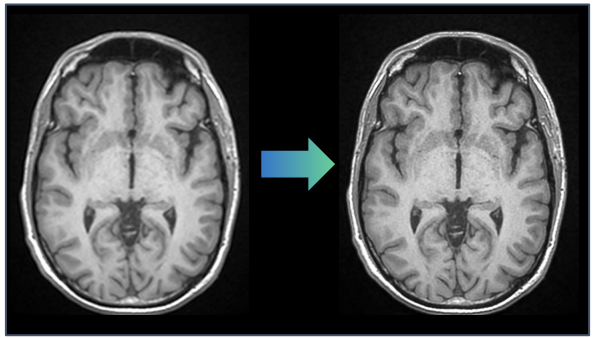
## SubtlePET™



4x faster acquisition enhanced

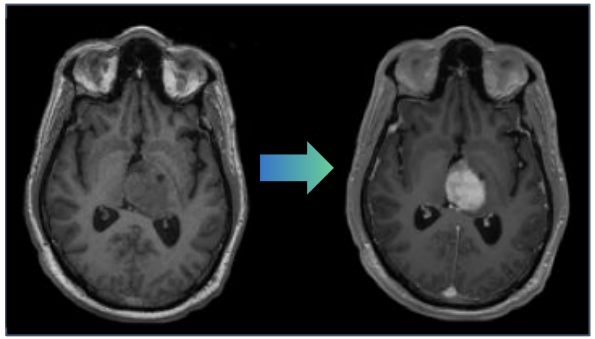
FDA 510(k) Cleared
CE Mark Approved

## SubtleMR™*



4x faster acquisition enhanced

## SubtleGAD™*



90% contrast reduction

*Disclaimer: Investigational use only. Not available for commercial distribution or sale.

# Advanced Sharing



NVIDIA.

# DeepOps

## What is it?

- Official NVIDIA solution for cluster management
- Highly modular!
- Deploy a Kubernetes cluster
- GPU-enabled (of course)
- Deployment via Ansible
- Optional services, demo examples included
- Code is open source
  - GitHub: https://github.com/NVIDIA/deepops/
  - Releases
    - tagged like NGC, year.month (ex: release-19.03)
    - documentation is particular to release


**Elegant way to share a single DGX Station or connect it to a cluster**

# DeepOps

## Demo: Setup cluster on a single DGX Station

```
# clone the release you'd like to use
git clone https://github.com/NVIDIA/deepops/tree/release-19.03
# update the submodules (in this case, kubespray)
git submodule update --init
# install software prerequisites and copy default configuration
./scripts/setup.sh
# create server inventory (using a single node, in this case)
./scripts/k8s_inventory.sh <station-node-ip>
# deploy the kubernetes cluster
ansible-playbook -i k8s-config/hosts.ini -b playbooks/k8s-cluster.yml
```


NVIDIA.

# Kubernetes

## Demo: Test GPUs

```
# quick test on 4 gpus

kubectl run gpu-test --rm -t -i --restart=Never --image=nvidia/cuda
--limits=nvidia.com/gpu=4 -- nvidia-smi
```



NVIDIA.

# Kubernetes

## Demo: Define a job

```
# deepops/test/pytorch-job.yml

apiVersion: batch/v1
kind: Job
metadata:
  name: pytorch-job
spec:
  backoffLimit: 5
  template:
    spec:
      imagePullSecrets:
        - name: nvcr.dgxkey
      containers:
        - name: pytorch-container
          image: nvcr.io/nvidia/pytorch:19.02-py3
          command: ["/bin/sh"]
          args: ["-c", "python /workspace/examples/upstream/mnist/main.py"]
          resources:
            limits:
              nvidia.com/gpu: 2
      restartPolicy: Never
```

# Kubernetes

## Demo: Launch a job using an NGC registry container

```
# launch first job with first user
kubectl create -f pytorch-job.yml
# launch another job with second user
kubectl create -f pytorch-job2.yml
# observe that both are using the cluster w/ different GPUs
kubectl get jobs
kubectl get pods
# kick off a third job, watch it queue (pending)
kubectl create -f pytorch-job3.yml
kubectl get pods
```


NVIDIA.

# Kubernetes

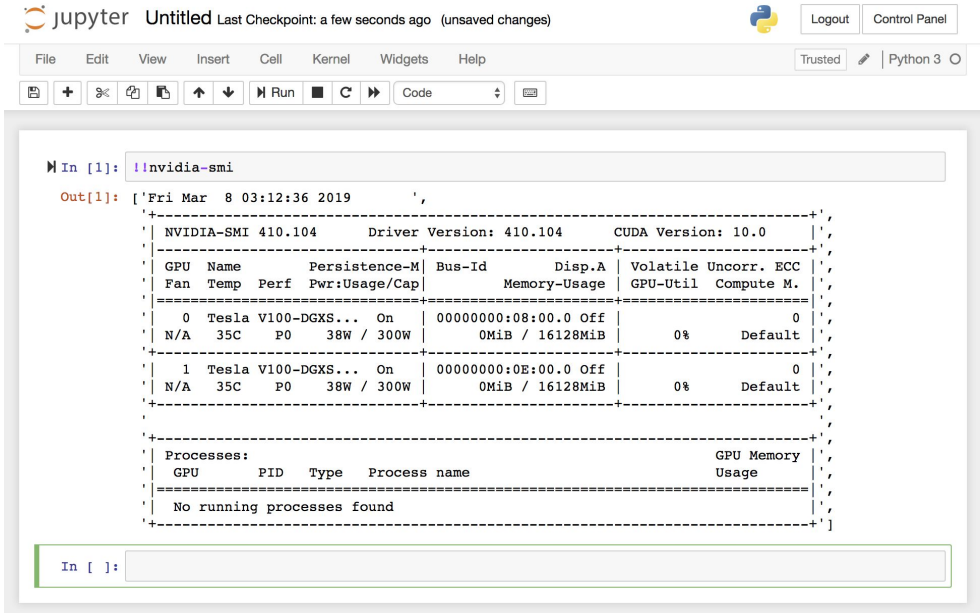
## Demo: Monitoring

```
# deploy monitoring
./scripts/k8s_deploy_monitoring.sh
# show grafana interface
http://<station-node-ip>:30200
```

# Kubernetes

## Demo: Kubeflow

```
# deploy rook w/ ceph
./scripts/k8s_deploy_rook.sh
# deploy kubeflow
./scripts/k8s_deploy_kubeflow.sh
# go to kubeflow interface
```



# Future



NVIDIA.

# DeepOps & Kubernetes

## DeepOps

- **Current:** official solution
  - Ubuntu & RHEL support
  - Kubernetes & Slurm
  - optional services and demo examples
- **Future:**
  - high-performance networking (IB/RoCE)
  - simplified usage
  - air-gapped deployment
  - continuous updates per customer feedback

## Kubernetes

- **Current:** contribute to mainstream K8s
  - Device Plugin
  - Driver Container
  - Monitoring Agent
- **Future:** NVIDIA GPU Operator
  - V1: manage lifecycle of installing GPUs (detecting GPUs, installing driver, etc)
  - V2: access metrics in your custom monitoring stack


NVIDIA.

# To Summarize

| Basic | Intermediate | Advanced |
|---|---|---|
| OS Users | Internal Storage | DeepOps |
| SSH | External Storage | Kubernetes |
| Docker / Containers | NFS Cache | Scripts |
| NGC | | Scheduling |
| Manual Scheduling | | Orchestration |
| VNC | | Monitoring |


NVIDIA.