Tutorial: Expanding AI/ML Coursework on Your Campus Using Jupyter Notebooks powered by NRP

Tuesday, January 28 10am-12pm

Why we're here

Artificial Intelligence (AI) and Machine Learning (ML) have become everyday terms, transforming industries, research, and education. The purpose of today's focus on JupyterHub and the National Research Platform (NRP) is to explore how these tools can be leveraged in teaching and learning, particularly for AI and ML.

Why JupyterHub and NRP?

1. Accessible Computational Environment
2. Hands-On Learning for AI/ML
3. Consistent, Reproducible Environments
4. Scalability (Many faculty start with instruction and move to research)

Takeaways: Nuts and Bolts for Leveraging JupyterHub on NRP for AI/ML Instruction

71%

Students stating that AI will become an essential part of most professions

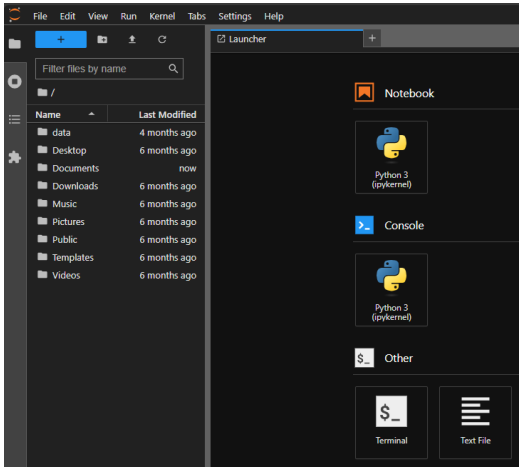
Results from the 2024 SDSU Student AI Survey (*n* = 10,162)
https://aaai.sdsu.edu/

Agenda

- What is JupyterHub
- What SDSU has done
- JupyterHub options on NRP
  - NRP hosted JupyterHub
  - Custom JupyterHub Deployment
    - Deployment guide
    - CI/CD for automatic deployments
- Distributing Notebooks
  - GitHub
  - GitHub Classroom
- Data sharing
  - Drag-and-Drop through UI
  - Shared storage
  - Linux utilities and scripts
  - S3-compatible object storage
- Building your own containers
- Supporting your JupyterHub
  - Ticketing
  - Documentation

What is JupyterHub

- JupyterHub is a multi-user server for Jupyter Notebooks
- Designed to support many users by spawning, managing, and proxying many singular Jupyter Notebook servers
- Enables running JupyterLab at scale
    - Web-based interactive development environment for notebooks, code, and data
    - Flexible interface allows users to configure and arrange workflows in data science, scientific computing, computational journalism, and machine learning
    - Supports languages such as Python, R, and Julia along with popular libraries like Pandas, PyTorch, and TensorFlow



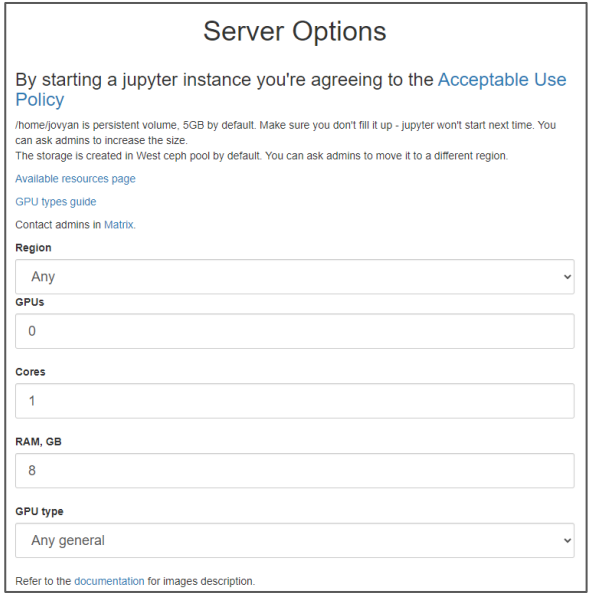
What SDSU has done

SDSU | San Diego State University

- Followed the Bring-Your-Own-Resource (BYOR) model with the VERNE instructional cluster
- Set up a single, central JupyterHub instance for instruction
- Started with same containers as NRP
  - Eventually started creating custom containers for courses
- Piloted with 2 courses in the Spring of 2023, up to 19 courses for Fall 2024
- Courses from several colleges and academic disciplines spanning graduate and undergraduate level
  - ASTR 201 Astronomy for Science Majors
  - LING 572 Python Scripting
  - MIS 429 Artificial Intelligence
  - PH 700A Principle Program Data Structures in Public Health

JupyterHub options on NRP

- Option 1: Use the NRP hosted JupyterHub
  - See the JupyterHub Service docs
  - Easy way to get started
  - 5 GB initial storage per user, must be increased per user
- Option 2: Custom deployment
  - Management models
    - One JupyterHub for all courses
    - Customized JupyterHub per course
  - Resources
    - Use the NRP
    - BYOR

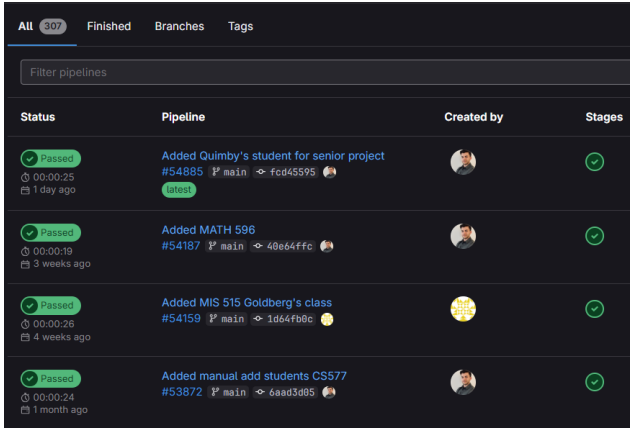


Custom JupyterHub Deployment

- Don’t need to BYOR to start, just use the resources available on NRP
- NRP deploying JupyterHub guide
  - Ask to become a namespace admin in the NRP Matrix Nautilus Support channel
  - Create a namespace for your JupyterHub instance
  - Register a new CILogon application
    - Enables users to authenticate with their institutional SSO credentials
  - Install Helm command line tool
    - “Helm is the best way to find, share, and use software built for Kubernetes”
  - Deploy the official JupyterHub Helm chart
    - Follow the instructions in the official JupyterHub deployment guide
    - Customize NRP provided template configuration file

CI/CD for automatic deployments

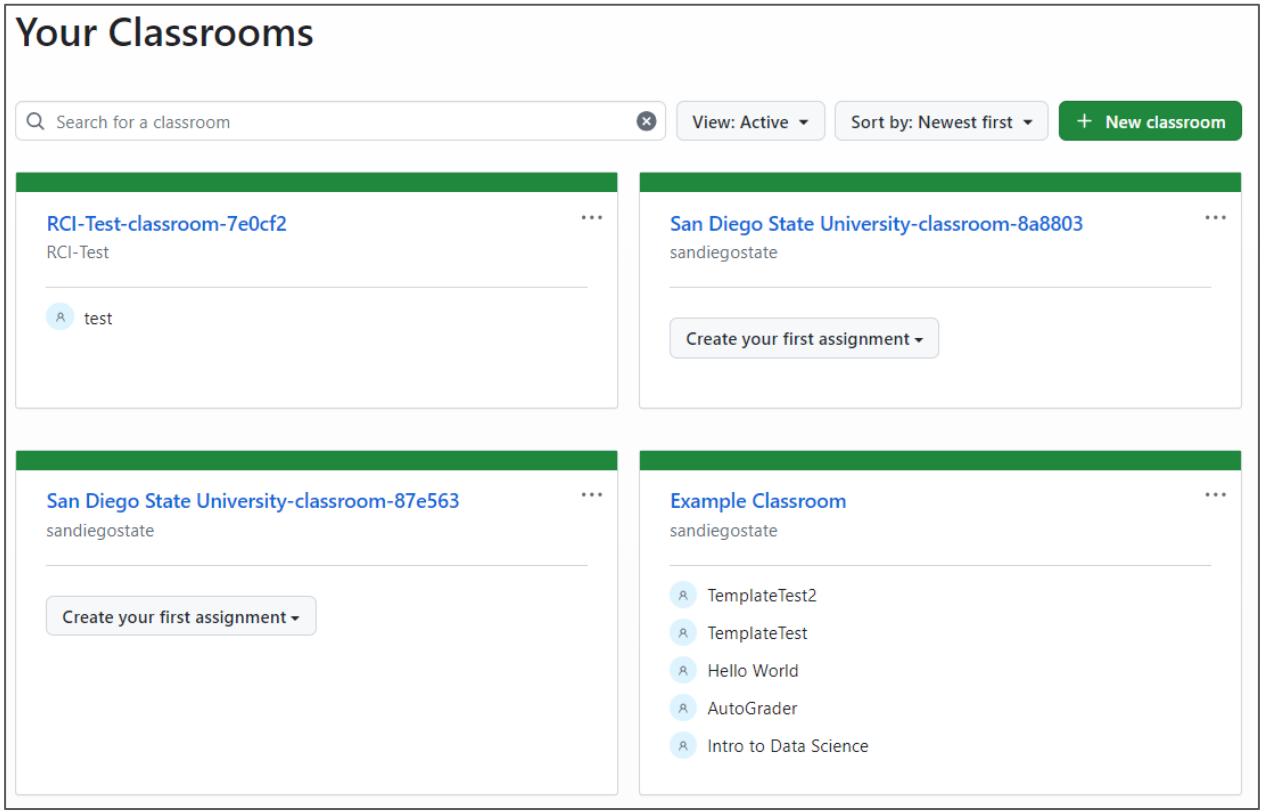
- Continuous Integration and Continuous Delivery (CI/CD)
  - Fancy term for automating the deployment process
- Why set up automatic deployments?
  - Helps reduce manual error when making changes
  - May need to update container images each term or academic year
  - May need to add users throughout the term
  - May need to upgrade JupyterHub
  - May need to revert changes
- NRP has an integrating GitLab and Kubernetes guide
  - Requires making an NRP GitLab account
    - Not to be confused with a gitlab.com account
  - Create a project
  - Upload code
  - Define CI/CD Pipeline in .gitlab-ci.yml file
  - Reference GitLab project



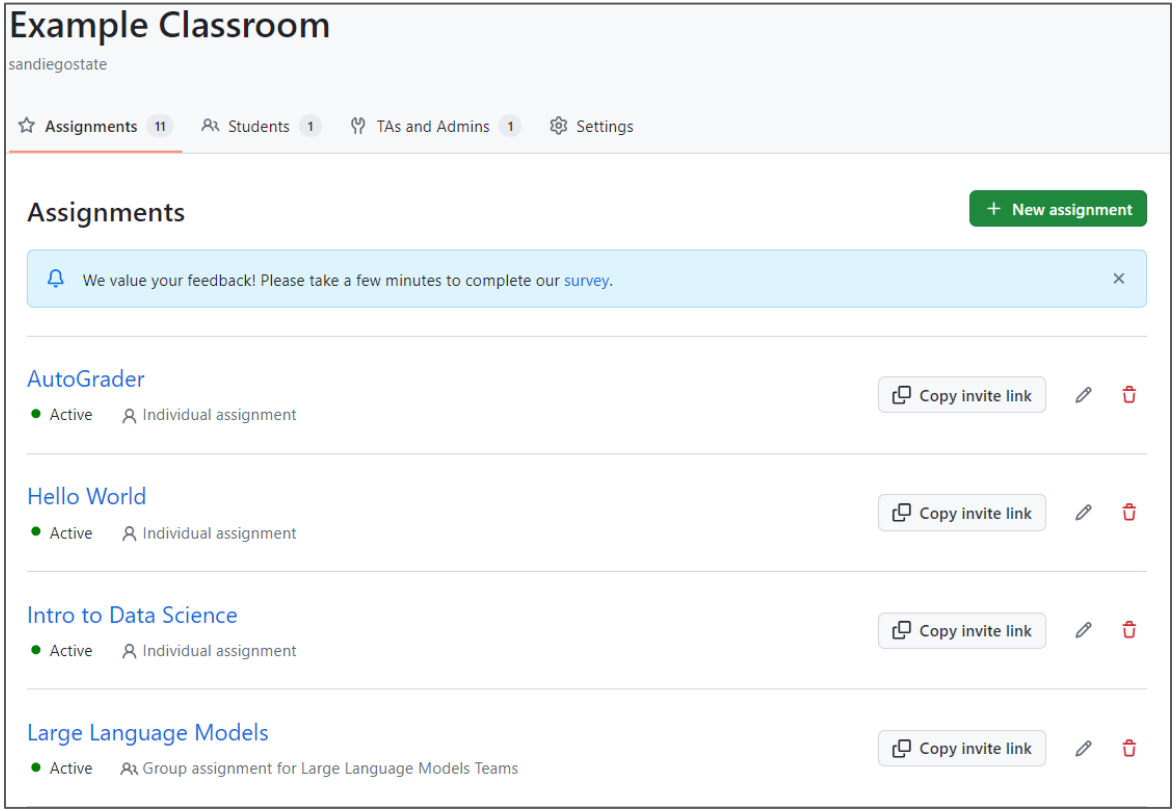
Distributing Notebooks

- All of the official Jupyter notebook containers come with git pre-installed
- GitHub
  - Straightforward and does not require students to have an account for cloning/downloading notebooks
  - Does require students to make an account for submitting work
- GitHub Classroom
  - Free service from GitHub tailored to teaching
    - Individual and group assignments, submission deadlines, and automatic grading
    - Optional roster integration with several LMS (Canvas, Google Classroom, LTI-compatible)
    - Assignments can be reused across classrooms
  - Requires faculty to create an organization
  - Requires students to create an account
  - More info: GitHub Classroom

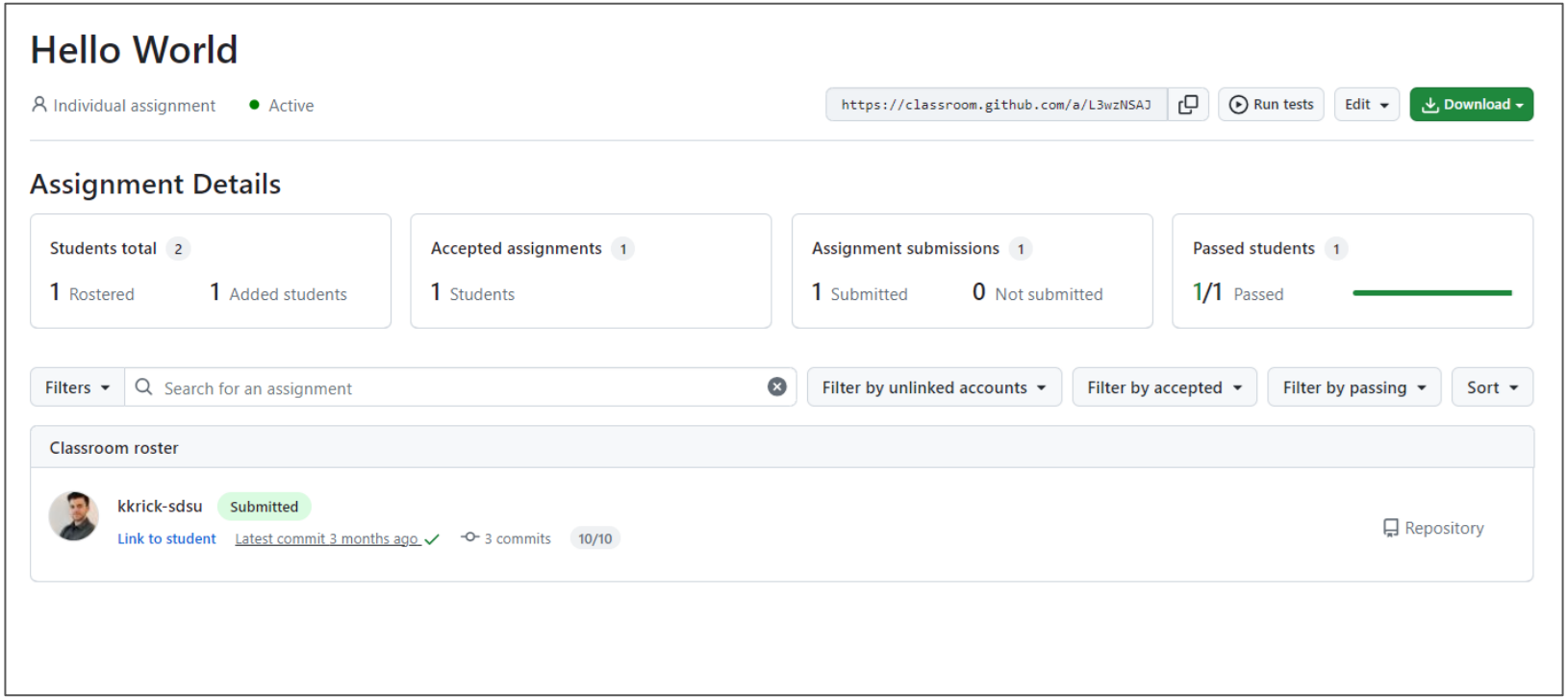
Distributing Notebooks: GitHub Classroom



Distributing Notebooks: GitHub Classroom



Distributing Notebooks: GitHub Classroom



Distributing Notebooks: JupyterLab Git Extension

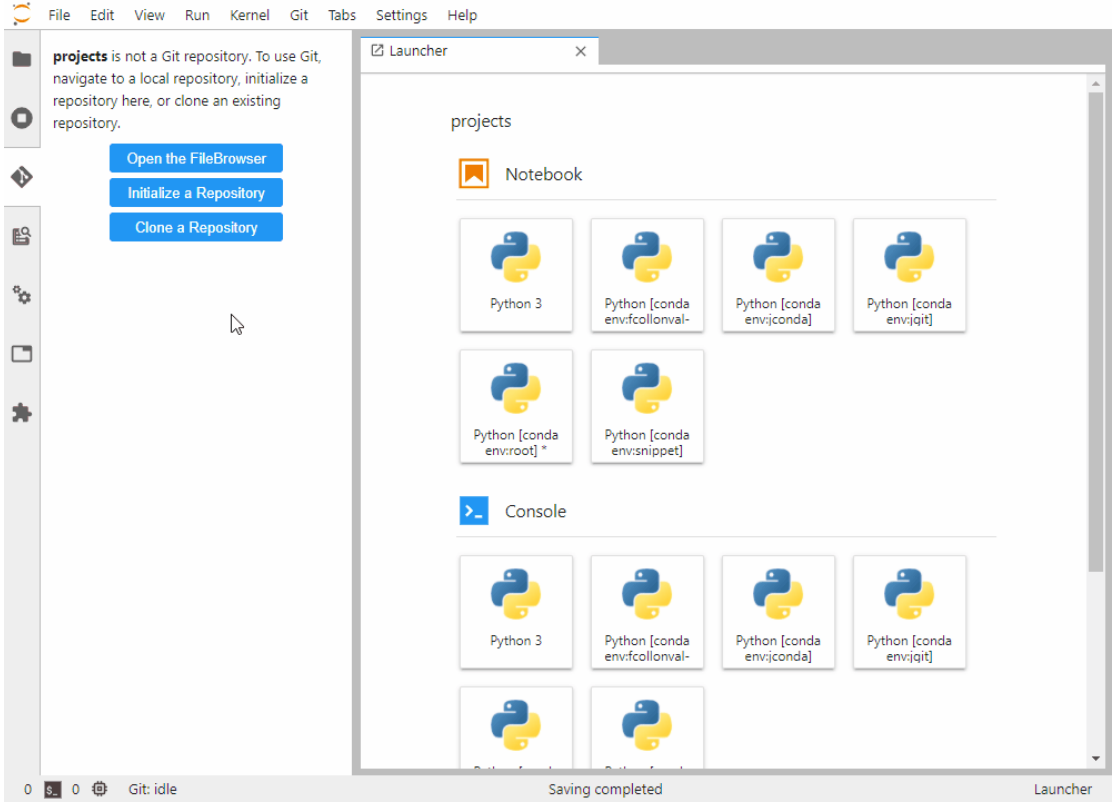
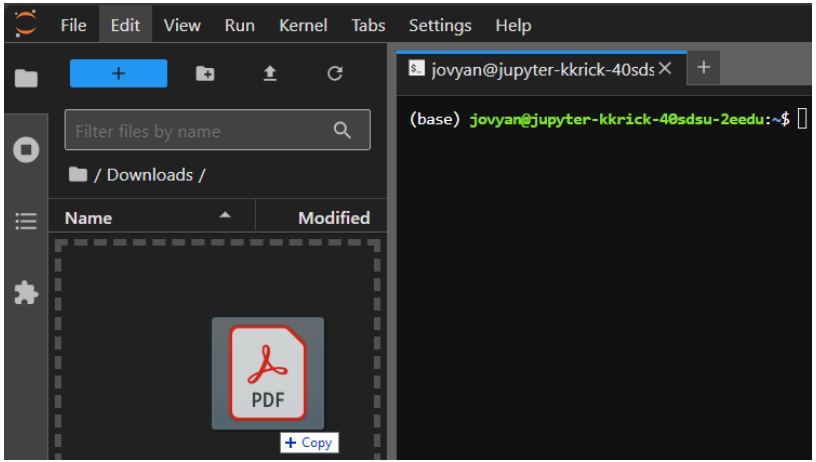
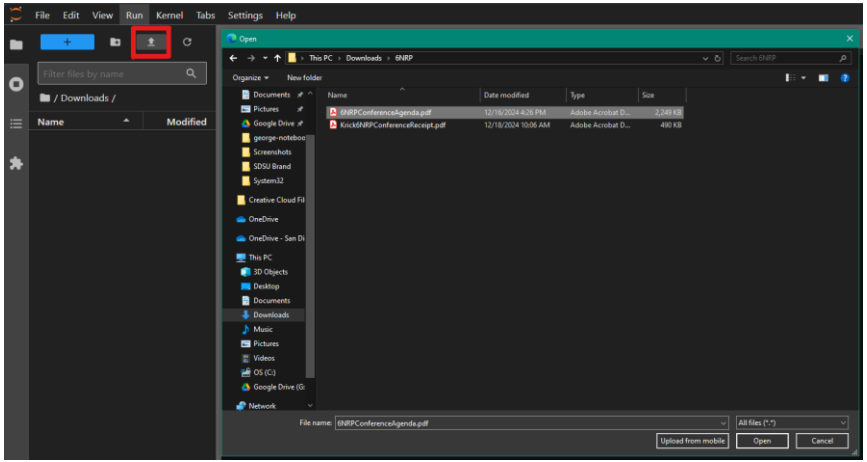


Image credit: [jupyterlab/jupyterlab-git: A Git extension for JupyterLab]

Data Sharing: JupyterLab UI

- JupyterLab has drag-and-drop and file selection upload options
  - Good for small datasets of < 50MB
  - Can provide files in your institution’s learning management system (LMS)
  - Students can download and then upload files





Data Sharing: Mounting Shared Storage

- File Storage
  - Allows multiple users access to the same data at the same time
- Create a PersistentVolumeClaim (PVC) with support for ReadWriteMany (RWX) in your namespace
  - Choose a CephFS storage pool close to the physical location/region of the nodes you want to run on
  - Modify "`storageClassName`" value in the next slide as appropriate
  - Modify "`storage`" value in next slide as appropriate for data size
- You may schedule a pod in the same namespace to transfer files
  - After files are transferred, recommend running "chown -R 1000:100 /path/to/pvc"
- Add the PVC to the singleuser spec in your helm-values.yaml
  - Recommend mounting it as read-only so that students do not overwrite the data
  - Recommend mountPath of "/home/jovyan/shared"

Data Sharing: Mounting Shared Storage - Make RWX PVC

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: shared
spec:
  accessModes:
  - ReadWriteMany
  resources:
    requests:
      storage: 5Gi
  storageClassName: rook-cephfs
```

- Save contents to the left in file “shared-pvc.yaml”
- Run the commands below:

```
kubectl apply -f shared-pvc.yaml
```

```
kubectl get pvc | grep shared
```

- Verify your new PVC was created and is the desired size

Data Sharing: Mounting Shared Storage - Data Transfer

```
apiVersion: v1
kind: Pod
metadata:
  name: shared-data-pod
spec:
  restartPolicy: Never
  containers:
  - name: shared-data
    image: ubuntu:latest
    resources:
      limits:
        memory: 2Gi
        cpu: 1
      requests:
        memory: 2Gi
        cpu: 1
    command: ["sleep", "infinity"]
    volumeMounts:
    - name: shared-data-volume
      mountPath: /data
  volumes:
  - name: shared-data-volume
    persistentVolumeClaim:
      claimName: shared
```

- Save contents to the left in file “shared-data-pod.yaml”
- Run the commands below:

```
kubectl apply -f shared-data-pod.yaml

kubectl exec -it shared-data-pod -- bash
```

- Transfer files via preferred method
  - wget, curl, script, rclone etc.
- When done, run the commands below

```
chown -R 1000:100 /data

exit

kubectl delete -f shared-data-pod.yaml
```

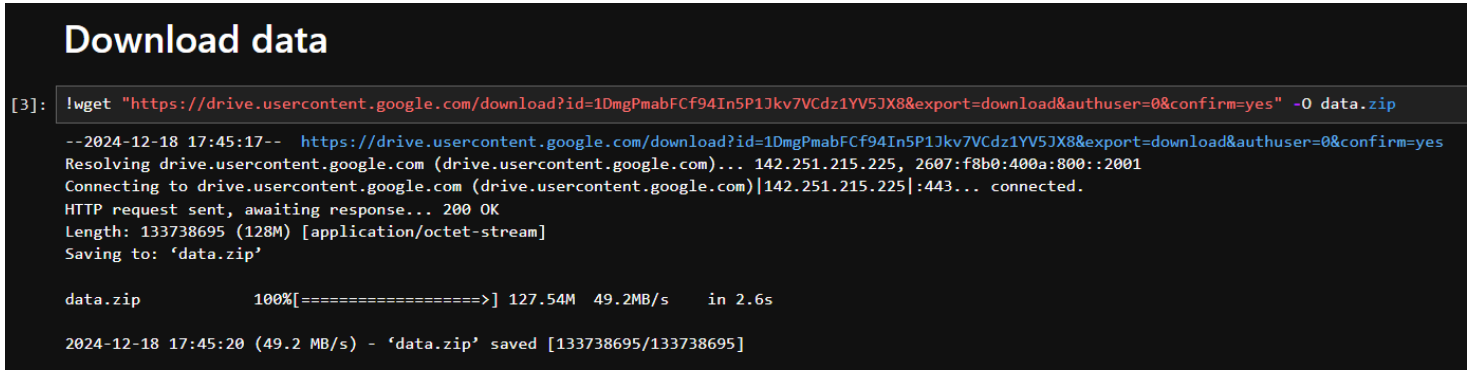
Data Sharing: Mounting Shared Storage in Notebooks

```
singleuser:
  storage:
    extraVolumes:
    - name: shared-data-volume
      persistentVolumeClaim:
        claimName: shared
    extraVolumeMounts:
    - name: shared-data-volume
      mountPath: /home/jovyan/shared
      readOnly: true
```

- Modify your helm-values.yaml file to include the extraVolumes and extraVolumeMounts logic under your singleuser > storage section
- Deploy these changes to your hub
  - This may be manual or via CI/CD depending on how you configured it
- Start a notebook via your hub and verify that you see the files you expect at the path /home/jovyan/shared

Data Sharing: Linux scripting

- Official Jupyter notebook images come with curl and wget
  - Can add commands to notebooks to download publicly available datasets
  - Can distribute shell scripts to execute in JupyterLab’s linux terminal



Data Sharing: Example data transfer script

```
#!/bin/env bash

cd /data

mkdir /data/processed

mkdir /data/processed/1of2

# Get data.zip
wget "https://drive.usercontent.google.com/download?id=1DmgPmabFCf94In5P1Jkv7VCdz1YV5JX8&export=download&authuser=0&confirm=yes" -O data.zip

# Get requirements.txt
wget "https://raw.githubusercontent.com/csu-tide/k8s-recipes/master/indexed-job/requirements.txt" -O requirements.txt

# Get pre_process_wiki_text.py
wget "https://raw.githubusercontent.com/csu-tide/k8s-recipes/master/indexed-job/pre_process_wiki_text.py" -O pre_process_wiki_text.py

# Get wiki-text.sh
wget "https://raw.githubusercontent.com/csu-tide/k8s-recipes/master/indexed-job/wiki-text.sh" -O wiki-text.sh

unzip data.zip

rm data.zip
```

```
(base) jovyan@jupyter-kkrick-40sdsu-2eedu:~$ bash get-data.sh
--2024-12-18 17:49:52--  https://drive.usercontent.google.com/download?id=1DmgPmabFCf94In5P1Jkv7VCdz1YV5JX8&export=download&authuser=0&confirm=yes
Resolving drive.usercontent.google.com (drive.usercontent.google.com)... 142.251.116.132, 2607:f8b0:4004:c1b::84
Connecting to drive.usercontent.google.com (drive.usercontent.google.com)|142.251.116.132|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 133738695 (128M) [application/octet-stream]
Saving to: ‘data.zip’

data.zip                                            100%[=================================================================================
```

Data Sharing: NRP S3-compatible object storage

- Can use NRP S3-compatible object storage
  - Best for large datasets I.E. > 1GB
  - Recommend using rclone
    - Most likely will need to install this
    - `sudo -v ; curl https://rclone.org/install.sh | sudo bash`
    - If building your own containers, recommend baking this utility in
  - Can distribute shared credentials to bucket
    - Recommend read-only credentials
    - Can distribute credentials as a file “rclone.conf”
    - Students can then copy the file where it belongs
      - mkdir ~/.config/rclone
      - cp ~/rclone.conf ~/.config/rclone/rclone.conf

Data Sharing: Rclone for S3-compatible object storage

rclone.conf U

rclone.conf

```
[nrp-s3]
type = s3
provider = Ceph
access_key_id = XXXXXXXXXXXXXXXXXXXX
secret_access_key = XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
endpoint = https://s3-west.nrp-nautilus.io
acl = private
```

```
(base) jovyan@jupyter-kkrick-40sdsu-2eedu:~$ rclone config
Current remotes:

Name                 Type
====                 ====
nrp-s3               s3

e) Edit existing remote
n) New remote
d) Delete remote
r) Rename remote
c) Copy remote
s) Set configuration password
q) Quit config
e/n/d/r/c/s/q>
```

```
(base) jovyan@jupyter-kkrick-40sdsu-2eedu:~$ rclone lsd nrp-s3:quick-benchmarks/output/mpi-cuda/a100
          0 2000-01-01 00:00:00        -1 2023-12-15
          0 2000-01-01 00:00:00        -1 2024-01-04
```

```
(base) jovyan@jupyter-kkrick-40sdsu-2eedu:~$ rclone ls nrp-s3:quick-benchmarks/output/mpi-cuda/a100/2024-01-04
    23787 morphine.1.out
    24161 morphine.2.out
    24909 morphine.4.out
    24569 psb5.1.out
    24943 psb5.2.out
    25691 psb5.4.out
    35818 taxol.1.out
    36192 taxol.2.out
    36940 taxol.4.out
    46001 valinomycin.1.out
    46375 valinomycin.2.out
    47123 valinomycin.4.out
```

```
(base) jovyan@jupyter-kkrick-40sdsu-2eedu:~$ rclone copy -LP nrp-s3:quick-benchmarks/output/mpi-cuda/a100/2024-01-04 ./2024-01-04
Transferred:      387.216 KiB / 387.216 KiB, 100%, 0 B/s, ETA -
Transferred:           12 / 12, 100%
Elapsed time:         1.4s
```

Building your own containers

- Why build your own containers?
  - Need specialized software i.e. linux binaries
  - Need a lot of packages or an entire pip/conda environment
  - Don’t want to install a package every time after startup anymore
- Start with NRP docker tutorial
  - Install Docker
  - Find good base images
    - Jupyter official images are a great place to start
  - Upload to Docker Hub or NRP GitLab
  - Add to your JupyterHub Helm values and re-deploy your hub

Building your own containers: Example “LLM Notebook”

Dockerfile > ...

```
ARG BASE_IMAGE=quay.io/jupyter/pytorch-notebook:cuda12-2024-07-29

FROM ${BASE_IMAGE}

# Switch to root for software installs
USER root
WORKDIR /opt

# Install rclone, Ollama and VS Code Server
RUN curl https://rclone.org/install.sh | bash \
 && curl -fsSL https://ollama.com/install.sh | sh \
 && curl -fsSL https://code-server.dev/install.sh | sh

# Fix any permissions issues caused by installing software via root
RUN fix-permissions "${CONDA_DIR}" \
 && fix-permissions "/home/${NB_USER}"

# Switch back to notebook user
USER $NB_USER
WORKDIR /home/${NB_USER}

# Add LLM packages
RUN mamba install -y -n base \
    bitsandbytes \
    transformers \
    peft \
    accelerate \
    trl \
    ollama-python \
    openai \
    pyaudio \
    portaudio \
    cuda-nvcc \
    deepspeed \
    langchain \
    huggingface_hub \
    auto_gptq \
    autoawq \
    xformers \
    dask-kubernetes \
    chromadb

```

```
kkrick@ITD-DAD-E057347:~/rci/custom-images/llm-notebook$ docker build . -f Dockerfile -t kkricksdsu/llm-notebook:v2.1
[+] Building 4.1s (5/10)
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 1.07kB
 => [internal] load metadata for quay.io/jupyter/pytorch-notebook:cuda12-2024-07-29
 => [internal] load .dockerignore
 => => transferring context: 2B
 => CACHED [1/7] FROM quay.io/jupyter/pytorch-notebook:cuda12-2024-07-29@sha256:a4daeb15b458ee35f00b701d7d625534d356ba637fa6656e342af70fdaef6a53
 => [2/7] WORKDIR /opt
 => [3/7] RUN curl https://rclone.org/install.sh | bash  && curl -fsSL https://ollama.com/install.sh | sh  && curl -fsSL https://code-server.dev/install.sh | sh
 => => #   % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
 => => #                                  Dload  Upload   Total   Spent    Left  Speed
 => => # 100  4734  100  4734    0     0   2935      0  0:00:01  0:00:01 --:--:--  2936
```

There are no container images stored for this project

With the Container Registry, every project can have its own space to store its Docker images. More Information

CLI Commands

If you are not already logged in, you need to authenticate to the Container Registry by using your GitLab username and password. If you have Two-Factor Authentication enabled, use a personal access token instead of a password.

```
docker login gitlab-registry.nrp-nautilus.io
```

You can add an image to this registry with the following commands:

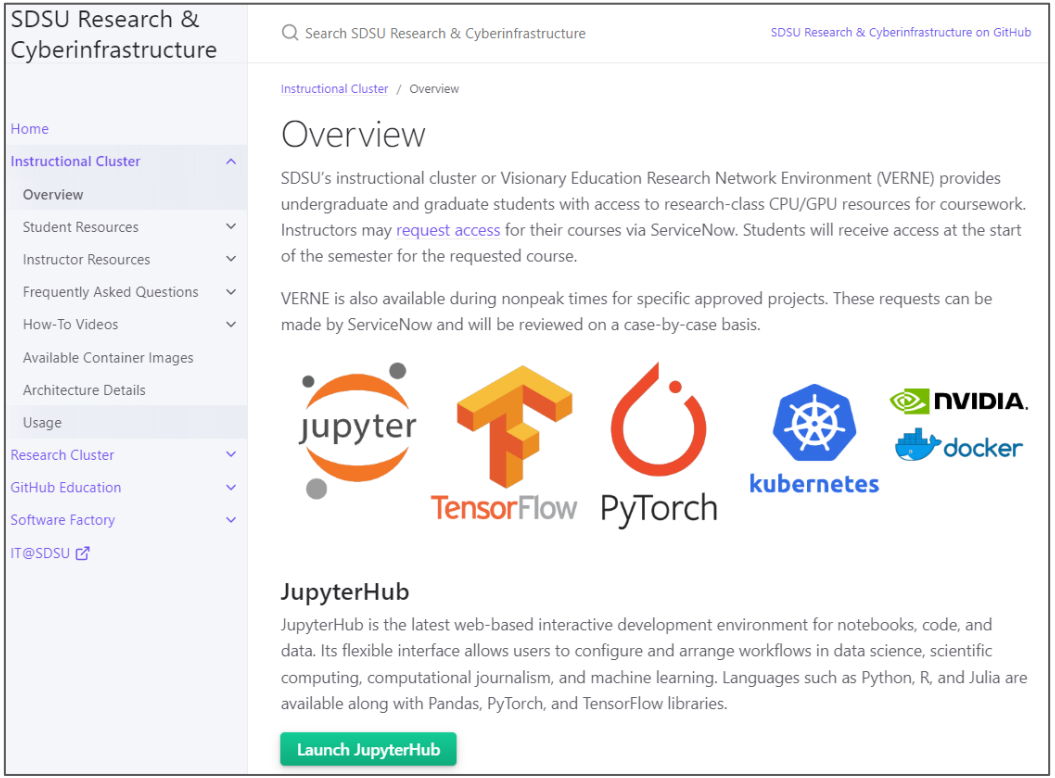
```
docker build -t gitlab-registry.nrp-nautilus.io/sdsu/llm-n
```

```
docker push gitlab-registry.nrp-nautilus.io/sdsu/llm-noteb
```

Supporting your JupyterHub: Documentation

- Questions to answer in your documentation
  - How to request access?
  - What resources are available?
  - How much storage capacity is available?
  - What is the data retention policy?
  - What software is available?
- GitHub pages hosts static sites for free

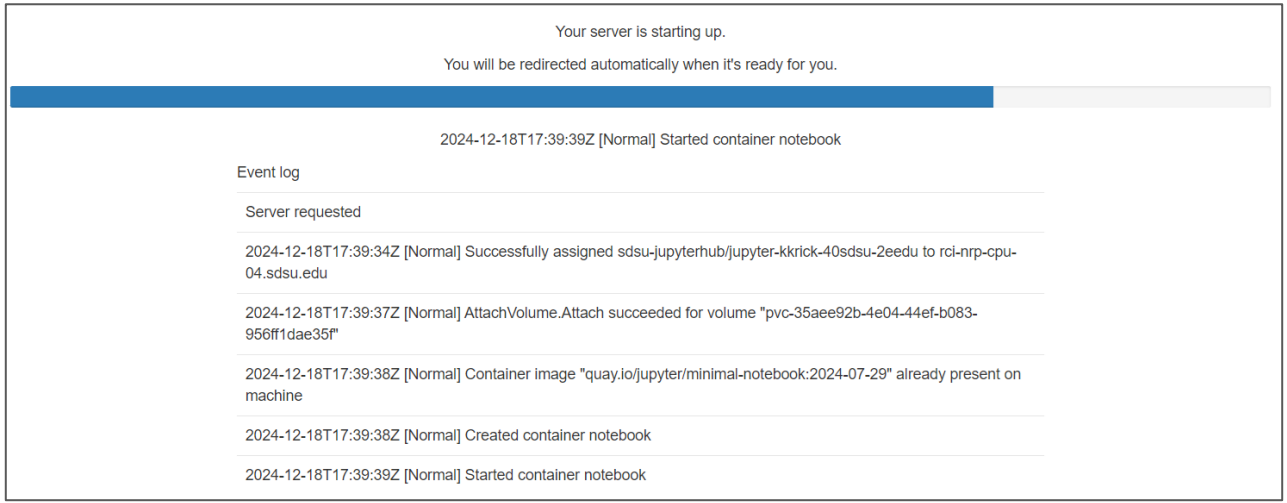
Supporting your JupyterHub: Documentation example



https://sdsu-research-ci.github.io/instructionalcluster

Supporting your JupyterHub: Reporting Issues

- Provide instructions for how faculty and/or students can get support
  - How to request access
  - Instructions for reporting issues
    - what information they should report
    - I.E. Recommend including screenshot of the start up logs



JupyterHub Cookbook

1. Try out NRP JupyterHub	https://docs.nationalresearchplatform.org/userdocs/jupyter/jupyterhub-service/
2. Deploy your own JupyterHub ● Create namespace ● Register CILogon ● Configure custom helm chart ● Deploy JupyterHub helm chart	https://docs.nationalresearchplatform.org/userdocs/jupyter/jupyterhub/
3. Automate your JupyterHub Deployment	https://docs.nationalresearchplatform.org/userdocs/development/k8s-integration/
4. Build and publish your own containers	https://docs.nationalresearchplatform.org/userdocs/tutorial/docker/