Introduction to PSC

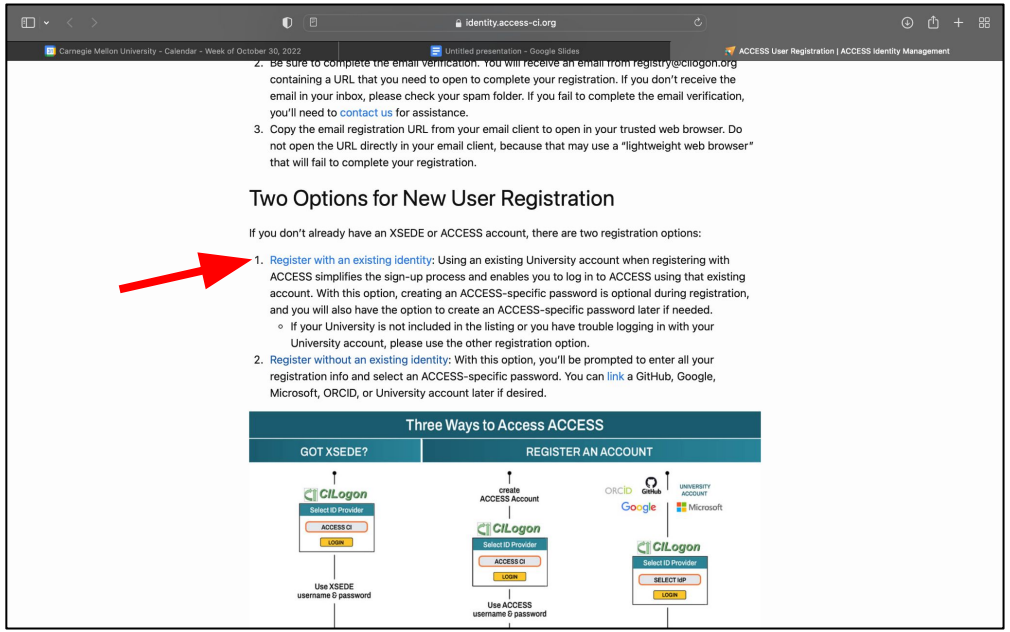
Pittsburgh Supercomputing Center

Aparajith Srinivasan
11785 Fall 2022, Recitation 10

Thanks to an uncredited student of Prof. Bhiksha for helping with PSC and the slides

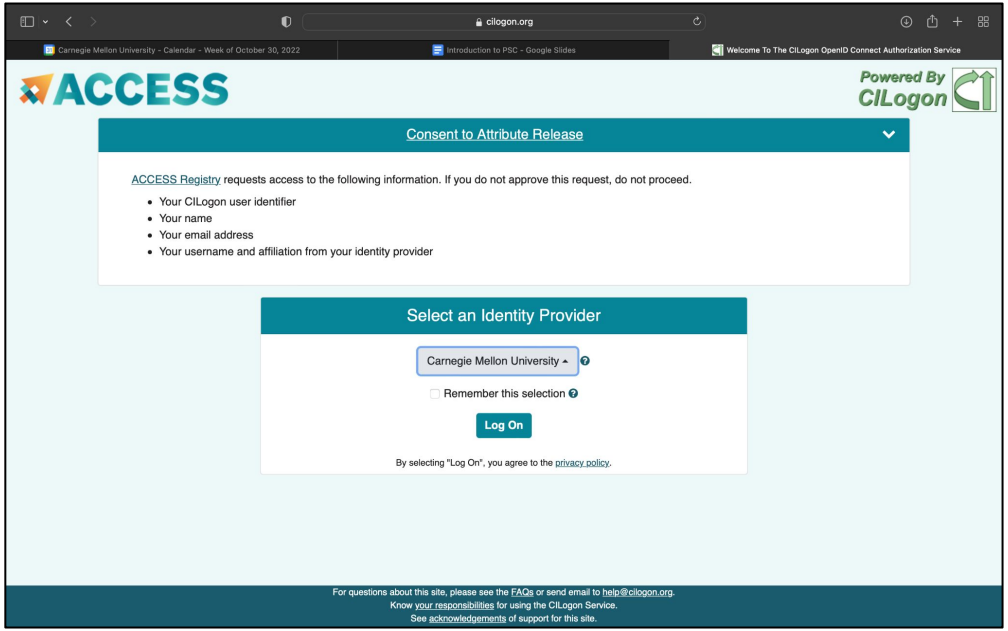
ACCESS Signup

- Make sure to use your Andrew ID Email. If you have an XSEDE account, select Register with an existing identity.
- Link to sign up: https://identity.access-ci.org/new-user



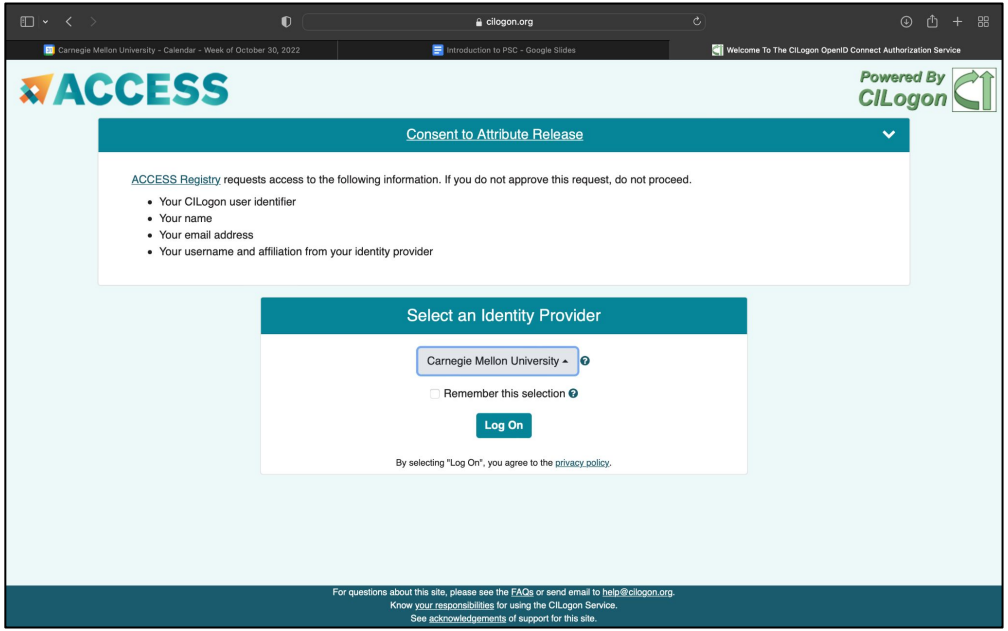
ACCESS Signup

- Make sure to use your Andrew ID Email. If you have an XSEDE account, select Register with an existing identity.
- Link to sign up: https://identity.access-ci.org/new-user
- Use your andrew account details
- Log on and finish signing up



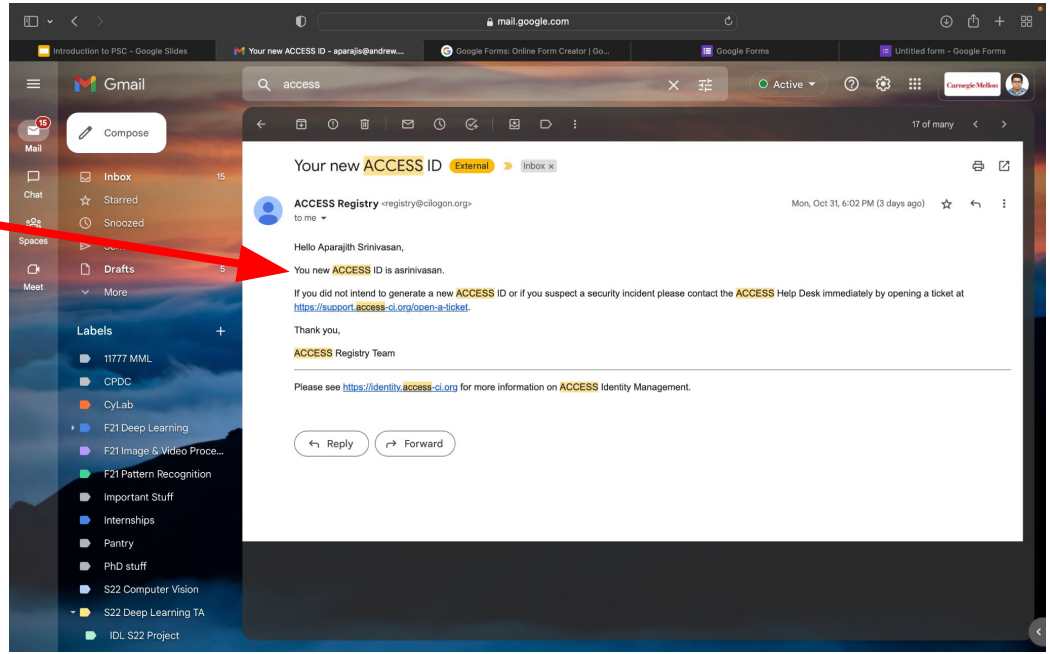
ACCESS Signup

- Make sure to use your Andrew ID Email. If you have an XSEDE account, select Register with an existing identity.
- Link to sign up: https://identity.access-ci.org/new-user
- Use your andrew account details
- Log on and finish signing up



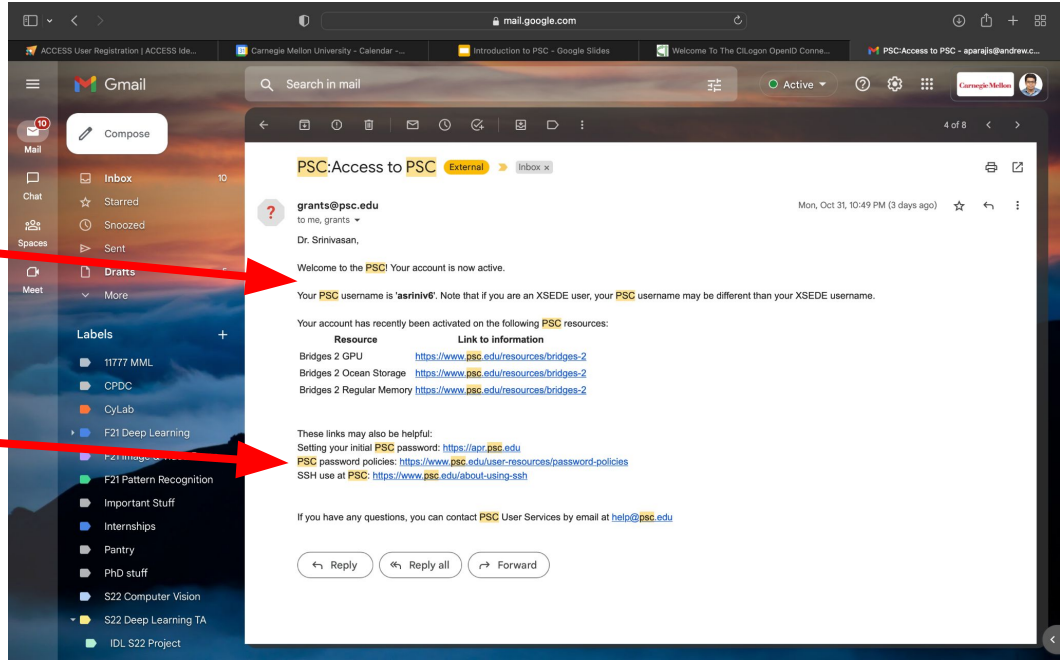
ACCESS Signup

- After Log on, you will receive an email with your ACCESS ID
- Make a note of this ACCESS ID and fill this form with the same
- https://forms.gle/7qnewdPyVSdybmHj7
- You get **1 BONUS** point for doing this



ACCESS Signup

- After a few hours, you will get a mail from grants@psc.edu with your PSC username
- Make a note of this username
- Set your initial password following this link
- Both these info will be required to connect to a GPU in PSC



PSC and Bridges 2

Bridges-2 is a High Performance Computing platform provided under the NSF ACCESS program by the Pittsburgh Supercomputing Center (PSC). Bridges-2 has:

1. Login nodes that you can launch jobs on, check status of existing jobs and job queues. These nodes are login011,login012,login013 or login014

2. GPU nodes are where you will launch your training or inference or feature extraction jobs. These nodes are called gpu0xx

DO NOT RUN ANY CODE ON THE LOGIN NODE

Credits: Prof. Shinji Watanabe (11751/187781)

Login to PSC with ssh

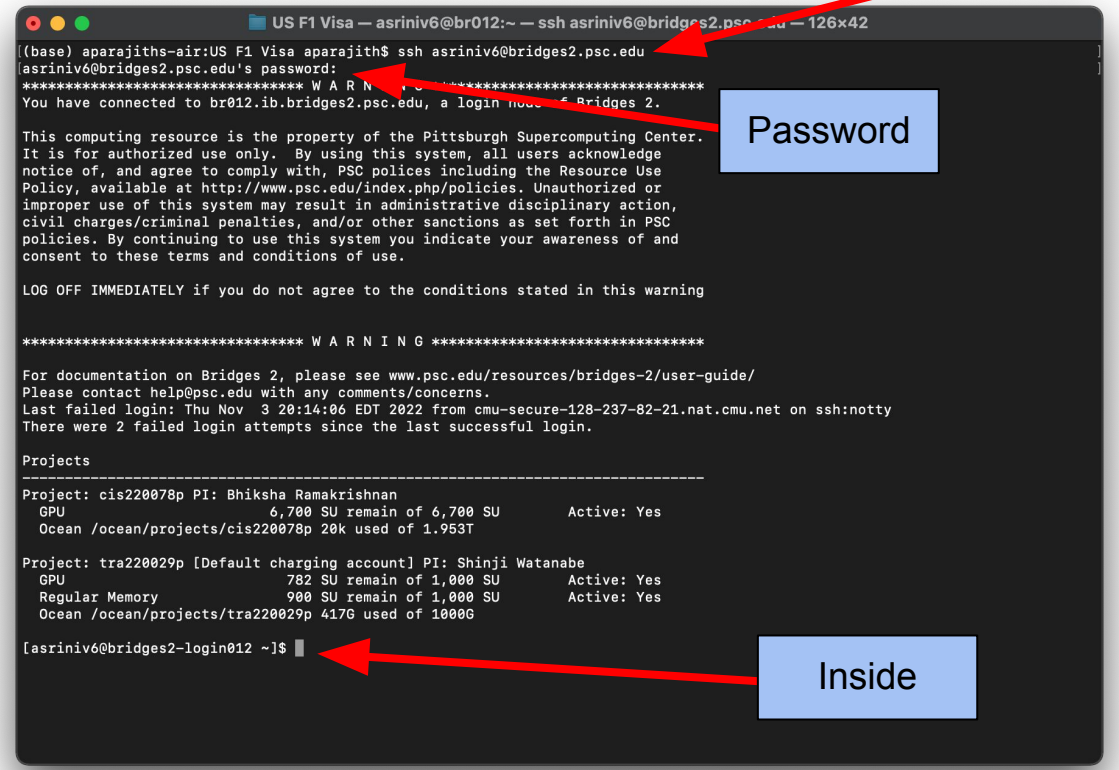
- Use this command from your terminal

ssh <username>@bridges2.psc.edu

- Enter your password when prompted
- To enable passwordless SSH login, you can try this after creating RSA keys using ssh-keygen (`ssh-keygen -t rsa -b 4096`)
- You can login to individual login nodes by doing:

ssh <username>@br012.ib.bridges2.psc.edu

Login to PSC with ssh



*** You are at the login node now**

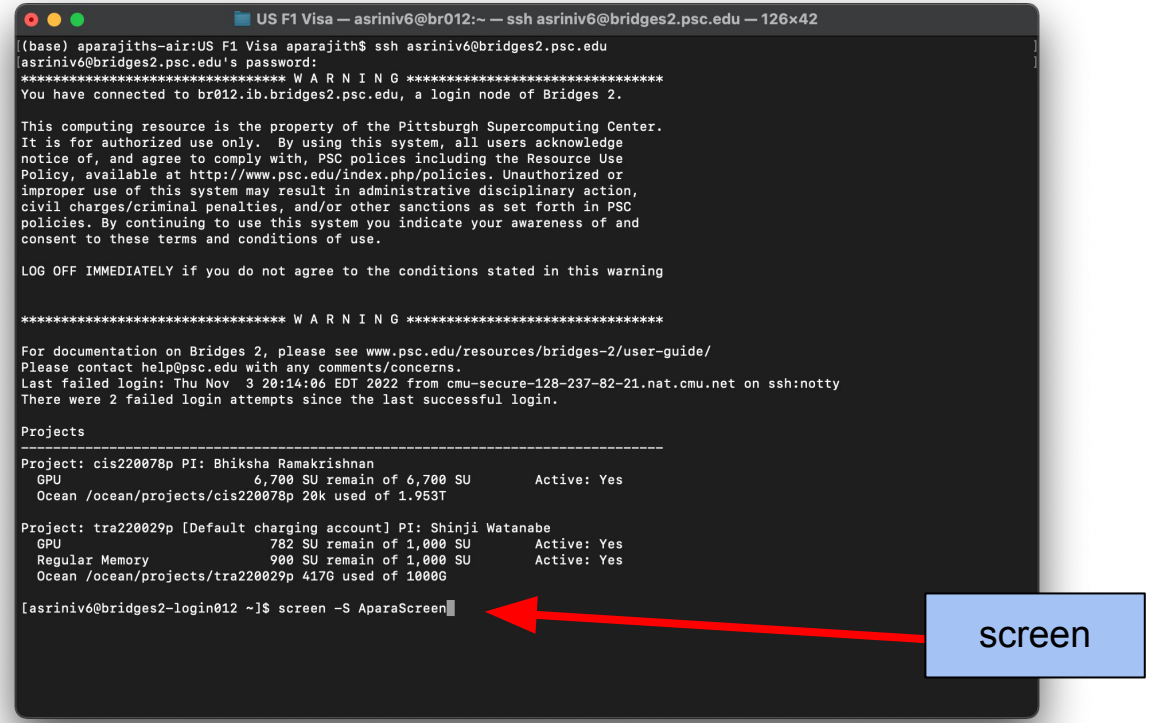
Login to PSC with ssh

- Use TMux or Screen to make sure that you can have background execution
- After logging into with your login node, you need to create a screen session and then request a compute node
- Do all your experiments inside a screen so that your progress does not get deleted when your laptop dies
- TMux is a similar thing which people use with Mac but screen is the most predominantly used.
- Just open a screen with

screen -S <name>

- Can exit a screen with ***exit***
- Take a look at all the other screen commands. You can run multiple screens too
- **Note down your login node**

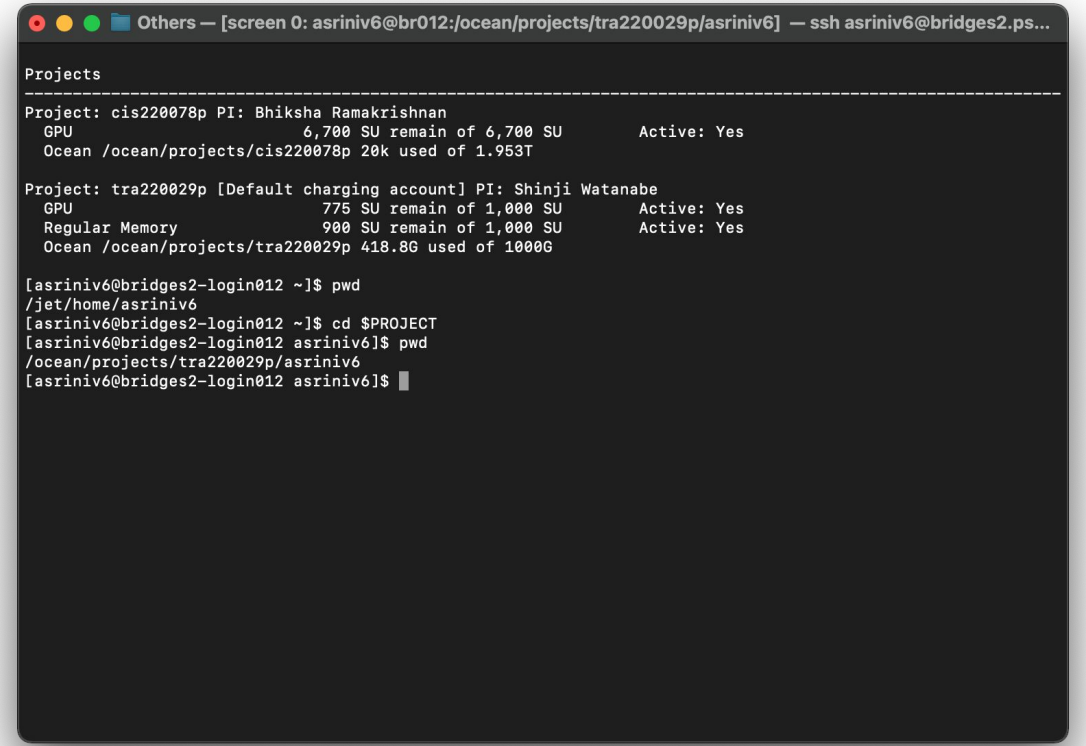
Login to PSC with ssh



Changing Directories

- By default you are logged in to the HOME directory. This is shared by everyone and don’t upload large files here
- If you do ***pwd***, you get this ***/jet/home/<username>***
- Change to the project directory with ***cd $PROJECT***
- Now, if you do pwd, you will get ***/ocean/projects/<project_id>/<username>***

Changing Directories



My output here is inside a screen

Requesting GPU Nodes

- You can request nodes in an interactive manner (where you are ble to access the bash shell directly and run commands), or using sbatch (where your code runs in the background)
- ***interact*** is used when you want a single GPU to test code for <= 8 hrs
- ***srun*** is used when you want an interactive shell on upto 8 GPUs for <= 48 hrs
- ***sbatch*** is used when you want to run code in the background on upto 8 GPUs for <=48 hrs

Requesting GPU Nodes: Interactive Allocations

interact

interact -gpu -t 08:00:00
Requests a single GPU node for 8 hrs

srun

srun -p GPU-small --gres=gpu:v100-32:2 -t 8:00:00 --pty bash (upto 2 GPUs for upto 8 hrs)

srun -p GPU-shared --gres=gpu:v100-32:4 -t 48:00:00 --pty bash (upto 4 GPUs for upto 48 hrs)

srun -p GPU --gres=gpu:v100-32:8 -t 48:00:00 --pty bash (upto 8 GPUs for upto 48 hrs)

Requesting GPU Nodes: sbatch Allocations

sbatch

sbatch -p GPU-small --gres=gpu:v100-32:2 -t 8:00:00 --job-name <name> ./run.sh (upto 2 GPUs for upto 8 hrs)

sbatch -p GPU-shared --gres=gpu:v100-32:4 -t 48:00:00 --job-name <name> ./run.sh (upto 4 GPUs for upto 48 hrs)

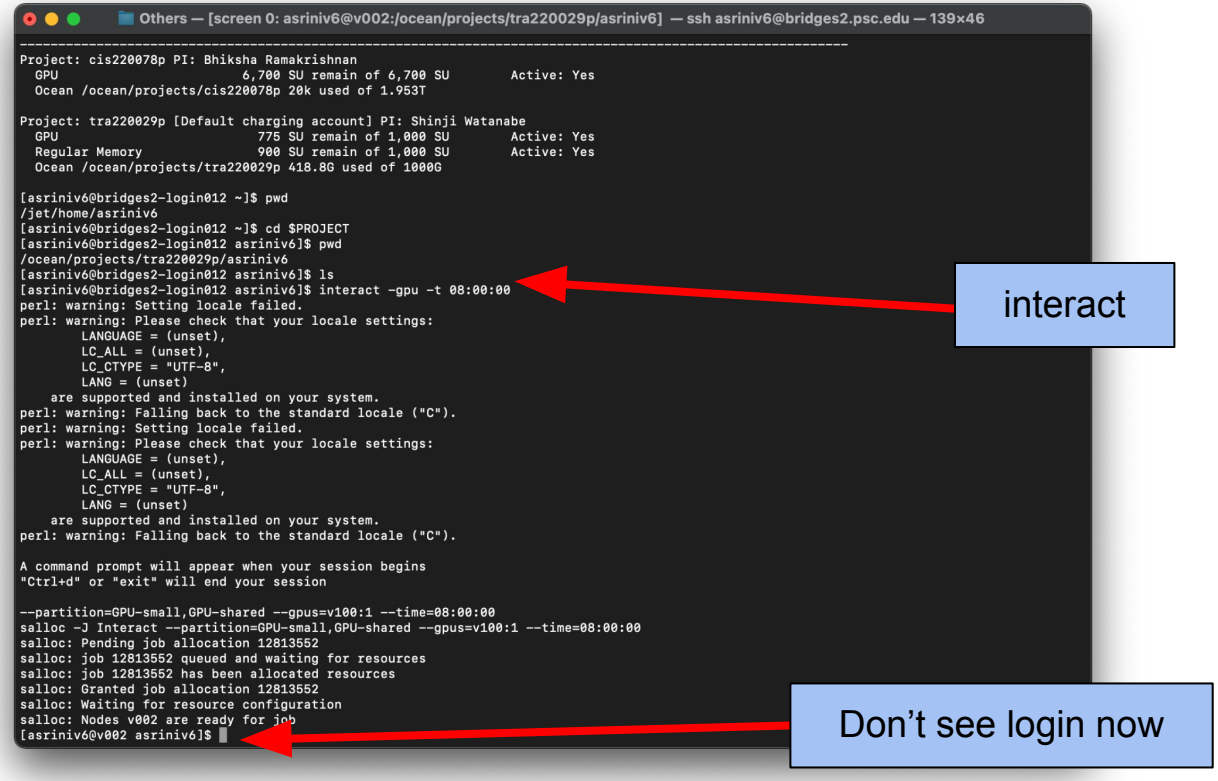
sbatch -p GPU --gres=gpu:v100-32:8 -t 48:00:00 --job-name <name> ./run.sh (upto 8 GPUs for upto 48 hrs)

Script run.sh must be written by you as a bash script

Other commands

- ***nvidia-smi*** - Tells you about the GPU usage.
  - Make sure that you use the complete GPU. If you have a 32GB GPU, you need to be using almost everything. Try increasing the batchsize until you get this.
- ***htop -*** Tells you about the processes, CPU Usage and RAM
  - If your job crashes, you need to request for more RAM

Requesting GPU Nodes



Uploading your files

- Use the command to upload your files

  scp <filename> <username>@bridges2.psc.edu:<project_dir>

- Enter password when prompted
- If you don’t specify ***<project_dir>***, the fill will be uploaded into your home directory

Creating venv

- Python3 is installed by default
- Create a virtual environment with ***venv*** and activate it

  https://docs.python.org/3/tutorial/venv.html

- Then do a ***pip3*** install to install all your libraries

The run.sh script

```
#! /bin/bash

python train.py
```