Beginning Steps:

1. Create an XSEDE Account
   - [https://portal.xsede.org/#/guest](https://portal.xsede.org/#/guest)
2. Set up DUO for multi-factor authentication (Note: Yes this is necessary)
3. Let the XSEDE allocation manager (your professor) what your username is to add you to the project
4. Create a PSC account to be able to connect to Bridges
   - [https://apr.psc.edu/autopwdreset/autopwdreset.html](https://apr.psc.edu/autopwdreset/autopwdreset.html)

Connecting to Bridges

1. Connect through SSH to a namenode using XSEDE Single Sign-on to Bridges
   * Host Name = bridges.psc.xsede.org
   * Port = 2222
   
   or
   
   You can connect to XSEDE instead then use the gsissh command to connect to Bridges
   * HostName = login.xsede.org
   * Port = 22

   Note: anecdotally from seeing various students try to connect, if you are having trouble connecting straight to Bridges, try connecting to XSEDE first

2. Enter Login and Password

3. Select 2-factor authentication method (1 for duo mobile)
   * open app and check yes
   
   Note: necessary to set up 2 factor authentication

What the Login Menu should look like

What a successful login looks like
Once Connected:
For transferring files:
1. Connect through SFTP to Bridges
   *Host Name = data.bridges.psc.edu
   *Port = 22

- You can tell you are on namenode by username@br###

```bash
[dmhughes@br018 ~]$ [dmhughes@br018 ~]$]
```

- $HOME directory is where you are at login. $HOME has a limited amount of storage and it is advised that it does not reliably store anything

```
/home/username = $HOME
```

```bash
[dmhughes@br018 ~]$ $HOME
-bash: /home/dmhughes: Is a directory
```

- $SCRATCH is the pylon5 storage directory (this is where hdfs files are also created)
  o important to note while this storage is much larger, if the Spark application fails to delete files each time you run your program, it will eventually fill up

```
/pylon5/groupname/username = $SCRATCH
```

```bash
[dmhughes@br018 ~]$ $SCRATCH
-bash: /pylon5/c: /dmhughes: Is a directory
```

Some simple commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>projects</td>
<td>Display allocations that you belong to</td>
</tr>
<tr>
<td>id -Gn</td>
<td>check all groupnames</td>
</tr>
<tr>
<td>id –gn</td>
<td>check current groupname</td>
</tr>
<tr>
<td>change_primary_group “name”</td>
<td>change groupname that you run jobs under</td>
</tr>
</tbody>
</table>

Note: Changing your primary group name is important because this is the default group for charging the cost of a job to
Interactive Usage for Spark:

Note: Ensure that you are on the right allocation with the right type of nodes (GPU vs RM)
- If you get an invalid qos specification error then that means that you either entered the command wrong or your allocation does not have access to the GPU/RM nodes

Example Command Order
$ interact -N 4 -t 01:00:00  // 1 hour is default if not -t option
   //this gets RM-shared nodes, for GPU there are additional arguments
   //You can tell you are in a compute node by @r###
$ module load hadoop
$ start-hadoop.sh    //at this point the HDFS is created, ready to run Spark jobs

Loading data onto HDFS
$ hdfs dfs –put data.txt
My example:
$ hdfs dfs –put $SCRATCH/matrix/matrix_5m.mtx

For example, this is one of my commands:
$ spark-submit --class Main --master yarn --deploy-mode cluster --driver-memory 115g --executor-memory 115g --conf spark.driver.maxResultSize="0" rsvdtest.jar mtxr matrix_10k.mtx 10 $HOME//rsvdtest/output_10k.txt 1

There are certain default options to be aware of such as how the driver memory and executory memory default to 1g.

Output from job:
$ yarn logs -applicationId yarnapplicationId  //to view full output of yarn

Note: Runtime errors will throw a useless Spark error on the console, so it is often necessary to use yarn logs, although they are difficult to read.
Batch Usage
Bridges uses SLURM for batch jobs.

<table>
<thead>
<tr>
<th>Example Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>$ sbatch &quot;name of bash script&quot;  //to submit a batch job</td>
</tr>
<tr>
<td>$ squeue –u &quot;username&quot;    // check job status</td>
</tr>
<tr>
<td>// PD = pending, R = running, CG = completing</td>
</tr>
<tr>
<td>$ scancel “job id”        //to cancel job</td>
</tr>
</tbody>
</table>

Example of a batch script I have used to run a Spark job on a RM partition

```bash
#!/bin/bash
#SBATCH -N 4
#SBATCH -t 03:00:00

#measure start time
START_TIME=$SECONDS

#variables
MAX_ITERATIONS=10
BLOCK_SIZE=10
NUM_RUNS=1

#echo commands to stdout
set -x

#load spark and hdfs
module load hadoop
start-hadoop.sh

echo "delaying" #this is here because the jobs have canceled due to nodes not being prepared
sleep 15s

#load matrices
hdfs dfs -put /pylon5/cc5fpep/dmhughes/matrix/matrix_1m.mtx

#make the output directory
mkdir $HOME//multiplytest/fulltest/

#run spark jobs
spark-submit --class Main --master yarn --deploy-mode cluster --driver-memory 100g --executor-memory 100g multiplytest.jar mtxr matrix_1m.mtx $BLOCK_SIZE $MAX_ITERATIONS $HOME//multiplytest/fulltest/output_1m.txt $NUM_RUNS

#output end time
echo "Finished in (seconds): 
ELAPSED_TIME=$(($SECONDS - $START_TIME))
echo "$(($ELAPSED_TIME/60)) min $(($ELAPSED_TIME%60)) sec"
```
Interactive Usage for GPU:

*Example Command Order for using Keras for a CNN*

```
$ interact --gpu       or       $interact --p GPU --gres=gpu:type:N //ex: type=p100, N=2
```

Note: additional arguments such as “-t 02:00:00” to get more time for a session

```
[dmhughes@gpu048 ~]$ //You can tell you are in a GPU node by username@gpu###
```

```
$ module load keras     //also loads CUDA
$ module load anaconda3  //to get PIL
$ python assignment_2.py //run python file
```

Note: Similar to when using the RM nodes for Spark, make sure you that you are charging to the correct project with the correct allocation, otherwise: “Invalid qos specification”

*Another Side Note about a specific error I encountered:*

- If you are trying to use transfer learning from Keras on your model, you will get an error when Bridges tries to connect to the github link in the Keras code.
- Instead, download the weights and then transfer them through SFTP to Bridges
Batch Usage:

Example of a batch script I have used to run a Keras' CNN on a gpu partition

```bash
#!/bin/bash
#SBATCH -p GPU
#SBATCH -t 03:00:00
#SBATCH --gres=gpu:p100:2

#measure start time
START_TIME=$SECONDS

#variables

#echo commands to stdout
set -x

#load modules
module load keras
module load anaconda3

echo "delaying"
sleep 5s

#age
#python assignment_2.py --bn resnet50 --ttn $SCRATCH/datasets/hw2/train_target_age.csv --pon $HOME/cs4990/test_pred_age_newEpoch40.csv --traindd $SCRATCH/datasets/hw2/train --testdd $SCRATCH/datasets/hw2/test --bs 32 --e 40 --o age --fli 25

#gender

#rm -f final_model_weights.h5

#output end time
echo "Finished in (seconds): "
ELAPSED_TIME=$(($SECONDS - $START_TIME))
echo "$((ELAPSED_TIME/60)) min $((ELAPSED_TIME%60)) sec"
```
Useful Links:

XSEDE portal:
https://portal.xsede.org/#/guest

Bridges Links:
https://apr.psc.edu/autopwdreset/autopwdreset.html //change password and set up account
https://www.psc.edu/bridges/user-guide/connecting-to-bridges
https://www.psc.edu/bridges/user-guide/running-jobs
https://www.psc.edu/bridges/user-guide/hadoop-and-spark