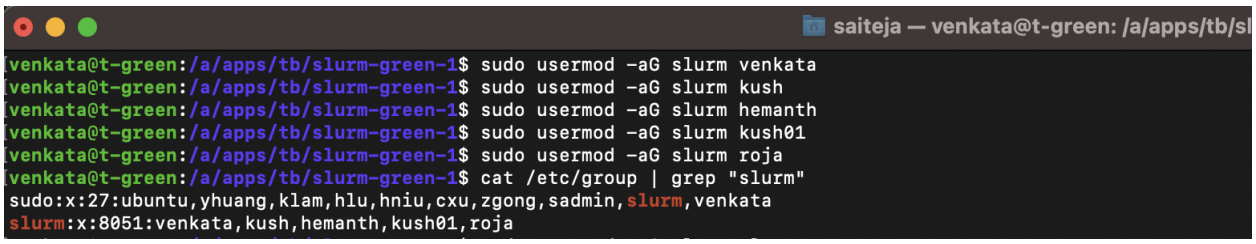
MILESTONE 4: Account Provisioning

· Add each team member's user account to the proper groups to grant them the necessary access. I added users to the "slurm" group, allowing them to submit jobs to Slurm.

sudo usermod -aG slurm venkata
sudo usermod -aG slurm kush
sudo usermod -aG slurm roja
sudo usermod -aG slurm hemanth



· This command adds a new user account named "c74" to the SLURM accounting system, with a description of "class", associated with the organization "neu", and allowed to use the cluster "green".

sacctmgr -i add account c74 Description=class Organization=neu cluster=green

Now we see the accounts and clusters using the following commands below:

sacctmgr show account
sacctmgr show clusters


```
class=(kush,kush01,roja,venkata,hemanth)
for u in "${class[@]}"; do
   #echo "user=$u"
   #sacctmgr -i delete account $u
   sacctmgr -i delete user $u
   #echo sacctmgr -i delete user $u
   sacctmgr -i add user $u Accounts+=c74 DefaultAccount=c74 cluster=green defaultqos=normal qos=normal,green
   #sacctmgr -i update user $u set AdminLevel=Admin
   #echo sacctmgr -i update user $u set defaultqos=normal qos=normal
   #echo sacctmgr -i update user $u set defaultqos=p1q1 cluster=p01
done
```

The above script creates or modifies SLURM user accounts for users named "kush", "kush01", "roja", "venkata", and "hemanth". It first deletes any existing accounts for those users, then creates new accounts with the specified account name, default account, cluster, and quality of service settings. It also includes some commented-out lines that can be used to update or modify account settings for these users.

```
slurm@t-green:/home/venkata$ for u in "${class[@]}"; do
>     #echo "user=$u"
>     #sacctmgr -i delete account $u
>     sacctmgr -i delete user $u
>     #echo sacctmgr -i delete user $u
>     sacctmgr -i add user $u Accounts+=c74 DefaultAccount=c74 cluster=green defaultqos=normal qos=normal,green
>     #sacctmgr -i update user $u set AdminLevel=Admin
>     #echo sacctmgr -i update user $u set defaultqos=normal qos=normal
>     #echo sacctmgr -i update user $u set defaultqos=p1q1 cluster=p01
> done
 Nothing deleted
 Adding User(s)
  kush
  kush01
  roja
  venkata
  hemanth
 Settings =
  Default Account = c74
 Associations =
  U = kush      A = c74        C = green
  U = kush01    A = c74        C = green
  U = roja      A = c74        C = green
  U = venkata   A = c74        C = green
  U = hemanth   A = c74        C = green
 Non Default Settings
  QOS           = +green,+normal
  DefQOS        = normal
slurm@t-green:/home/venkata$
```

We see the account users using the command below:

```
slurm@t-green:/home/venkata$ sacctmgr show user
      User   Def Acct     Admin
---------- ---------- ---------
     dhruv            Administ+
   hemanth        c74 Administ+
      hniu                None
      kush        c74 Administ+
    kush01        c74 Administ+
     manoj                None
     pawan            Administ+
      prem            Administ+
      roja        c74 Administ+
      root       root Administ+
     varun            Administ+
   venkata        c74 Administ+
    yhuang            Administ+
slurm@t-green:/home/venkata$
```

MILESTONE 5

Integrate the Slurm controller and worker nodes

sinfo -N -o "%.20N %.15C %.8t %.12m %.12P %10f %.15G %.40E"

```
[slurm@t-green:/home/venkata$ sinfo -N -o "%.20N %.15C %.8t %.12m %.12P %10f %.15G %.40E"
             NODELIST   CPUS(A/I/O/T)    STATE       MEMORY    PARTITION AVAIL_FEAT            GRES                                    REASON
            i-hemanth         0/1/0/1     idle            1       batch* (null)             (null)                                      none
            i-hemanth         0/1/0/1     idle            1          ig0 (null)             (null)                                      none
               i-kush         0/1/0/1     idle            1       batch* (null)             (null)                                      none
               i-kush         0/1/0/1     idle            1          ig0 (null)             (null)                                      none
               i-roja         0/1/0/1     unk*            1       batch* (null)             (null)                                      none
               i-roja         0/1/0/1     unk*            1          ig0 (null)             (null)                                      none
            i-venkata         0/1/0/1     unk*            1       batch* (null)             (null)                                      none
            i-venkata         0/1/0/1     unk*            1          ig0 (null)             (null)                                      none
slurm@t-green:/home/venkata$
```

Test the integrated system and ensure it is functioning properly

Testing from team servers to all individual nodes

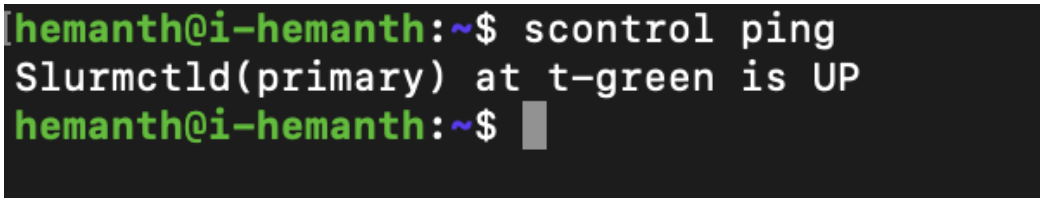
srun -p ig0 --nodes=4 hostname

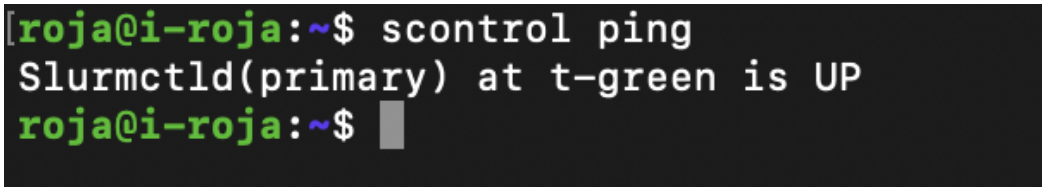
```
venkata@t-green:~$ srun -p ig0 --nodes=4 hostname
i-kush
i-venkata
i-roja
i-hemanth
venkata@t-green:~$
```

Individual Ping screenshots to team server

1.
```
venkata@i-venkata:~$ scontrol ping
Slurmctld(primary) at t-green is UP
venkata@i-venkata:~$
```

2.
```
kush01@i-kush:~$ scontrol ping
Slurmctld(primary) at t-green is UP
kush01@i-kush:~$
```

3. 

4. 

MILESTONE 6: Optimization, scaling and Qos

This command adds a new Quality of Service (QoS) named "green" to the SLURM workload manager. The QoS specifies a maximum of 1 CPU per user, a priority of 10, and a "DenyOnLimit" flag that prevents jobs from running once the specified limit is reached. Additionally, this QoS has a maximum walltime limit of 5 minutes.

sacctmgr -i add qos green set MaxTRESPerUser=cpu=1 priority=10 Flags=DenyOnLimit MaxWall=0-0:5:0

```
[slurm@t-green:/home/venkata$ sacctmgr -i add qos green set MaxTRESPerUser=cpu=1 priority=10 Flags=DenyOnLimit MaxWall=0-0:5:0
 Adding QOS(s)
  green
 Settings
  Description    = green
  Flags          = DenyOnLimit
  MaxTRESPerUser = cpu=1
  MaxWall        = 00:05:00
  Priority       = 10
```

This command displays a list of all Quality of Service (QoS) configurations that have been defined in the SLURM workload manager. It provides information such as the QoS name, description, and resource allocation settings, including limits on CPU usage, memory usage, and run time.

sacctmgr show qos

```
venkata@t-green:~$ sacctmgr show qos
      Name   Priority  GraceTime    Preempt   PreemptExemptTime PreemptMode                                Flags UsageThres UsageFactor       GrpTRES   GrpTRESMins GrpTRESRunMin GrpJobs GrpSubmi
t     GrpWall       MaxTRES MaxTRESPerNode   MaxTRESMins     MaxWall     MaxTRESPU MaxJobsPU MaxSubmitPU     MaxTRESPA MaxJobsPA MaxSubmitPA        MinTRES
---------- ---------- ---------- ---------- ----------- ------------- ---------- ---------- ----------- ---------- ---------- ---------- ------------- ------------- ------------- ------- --------
- ---------- ---------- ---------- ---------- ----------- ------------- ---------- ---------- ----------- ---------- ---------- ---------- -------------
    normal          0   00:00:00                                cluster                                                    1.000000
      p1q1         10   00:00:00                                cluster                          DenyOnLimit               1.000000
                                                        00:05:00         cpu=1
     green         10   00:00:00                                cluster                          DenyOnLimit               1.000000
                                                        00:05:00         cpu=1
     black         10   00:00:00                                cluster                          DenyOnLimit               1.000000
                                                        00:05:00         cpu=1
venkata@t-green:~$
```

QOS Limitations

squeue is a command-line utility in the SLURM workload manager that displays information about jobs running or waiting to run in a SLURM-managed cluster. It provides a real-time view of job status, including job IDs, user IDs, partition names, start and end times, and resource allocations. The output of the squeue command can be sorted and filtered in various ways to display only the desired information, such as jobs from a specific user, or jobs running on a particular node or partition. The squeue command is a useful tool for monitoring and managing jobs in a SLURM-managed HPC environment.

```
venkata@t-green:/home/venkata/test/slurm_arr$ sbatch sub_arr.sh
sbatch: error: QOSMaxCpuPerUserLimit
sbatch: error: Batch job submission failed: Job violates accounting/QOS policy (job submit limit, user's size and/or time limits)
venkata@t-green:/home/venkata/test/slurm_arr$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
venkata@t-green:/home/venkata/test/slurm_arr$ cat sub_arr.sh
#!/bin/bash
#SBATCH --job-name=arrayTest
#SBATCH --nodes=2
#SBATCH --ntasks=4
#SBATCH --time=5:00
#SBATCH --mem=2mb                    # Job memory request on each node
#SBATCH --output=j_%A_%a.out
#SBATCH --error=j_%A_%a.err
#SBATCH --partition batch
#SBATCH --array=1-4

y=$(($SLURM_ARRAY_TASK_ID+2));
echo "$SLURM_ARRAY_TASK_ID" $y
sleep 10

# sbatch sub_arr.sh
```

The first command updates the configuration of the "green" Quality of Service (QoS) in the SLURM workload manager to change the maximum number of CPUs that can be used per user from 1 to 4.
The second command displays a list of all QoS configurations in the SLURM workload manager, including the updated "green" QoS with the new CPU limit.
In summary, these commands modify the resource allocation settings of the "green" QoS and then display the updated QoS configuration.

sacctmgr -i update qos green set MaxTRESPerUser=cpu=4
sacctmgr show qos

```
venkata@t-green:/home/venkata/test/slurm_arr$ sacctmgr show qos
      Name   Priority  GraceTime    Preempt   PreemptExemptTime PreemptMode                                 Flags UsageThres UsageFactor       GrpTRES   GrpTRESMins GrpTRESRunMin GrpJobs GrpSubmi
t     GrpWall       MaxTRES MaxTRESPerNode   MaxTRESMins     MaxWall     MaxTRESPU MaxJobsPU MaxSubmitPU     MaxTRESPA MaxJobsPA MaxSubmitPA       MinTRES
---------- ---------- ---------- ---------- ------------------- ----------- ---------- ---------- ------------------------------------- ---------- ----------- ------------- ------------- ------------- ------- --------
- ----------- ------------- -------------- ------------- ----------- ------------- --------- ----------- ------------- --------- ----------- -------------
    normal          0   00:00:00                                cluster                                                      1.000000

      p1q1         10   00:00:00                                cluster                           DenyOnLimit                1.000000
                                                        00:05:00          cpu=1
     green         10   00:00:00                                cluster                           DenyOnLimit                1.000000
                                                        00:05:00          cpu=4
     black         10   00:00:00                                cluster                           DenyOnLimit                1.000000
                                                        00:05:00          cpu=1
```

MILESTONE 7: Advanced Parallel Jobs

Task 1: Job array

We run these commands for the sub_arr.sh files

[-d ~/test/slurm_arr] | mkdir -p ~/test/slurm_arr
cd ~/test/slurm_arr
cp /a/util/demo/slurm/run/slurm_arr/sub_arr.sh ~/test/slurm_arr/

```
#!/bin/bash
#SBATCH --job-name=arrayTest
#SBATCH --nodes=4
#SBATCH --ntasks=4
#SBATCH --time=5:00
#SBATCH --mem=2mb                    # Job memory request on each node
#SBATCH --output=j_%A_%a.out
#SBATCH --error=j_%A_%a.err
#SBATCH --partition batch
#SBATCH --array=1-4

y=$(($SLURM_ARRAY_TASK_ID+2));
echo "$SLURM_ARRAY_TASK_ID" $y
sleep 10

# sbatch sub_arr.sh
```

This command submits a job to the SLURM workload manager using the shell script sub_arr.sh. The sbatch command reads the script and submits it to the SLURM controller for scheduling and execution. The script sub_arr.sh will contain instructions for the SLURM controller to allocate resources for the job, such as the number of CPU cores, memory requirements, and expected run time.
Once the job is submitted, the sbatch command will return a job ID number that can be used to track the status of the job using other SLURM commands, such as squeue or sacct.

sbatch sub_arr.sh

```
venkata@t-green:/home/venkata/test/slurm_arr$ [ -d ~/test/slurm_arr ] | mkdir -p ~/test/slurm_arr
venkata@t-green:/home/venkata/test/slurm_arr$ cd ~/test/slurm_arr
venkata@t-green:/home/venkata/test/slurm_arr$ cp /a/util/demo/slurm/run/slurm_arr/sub_arr.sh ~/test/slurm_arr/
venkata@t-green:/home/venkata/test/slurm_arr$ vi sub_arr.sh
venkata@t-green:/home/venkata/test/slurm_arr$ sbatch sub_arr.sh
Submitted batch job 295
venkata@t-green:/home/venkata/test/slurm_arr$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
         295_[2-4]     batch arrayTes  venkata PD       0:00      4 (Resources)
             295_1     batch arrayTes  venkata  R       0:03      4 i-hemanth,i-kush,i-roja,i-venkata
venkata@t-green:/home/venkata/test/slurm_arr$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
ig0          up   infinite      4  alloc i-hemanth,i-kush,i-roja,i-venkata
batch*       up   infinite      4  alloc i-hemanth,i-kush,i-roja,i-venkata
venkata@t-green:/home/venkata/test/slurm_arr$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
             295_4     batch arrayTes  venkata PD       0:00      4 (Resources)
             295_3     batch arrayTes  venkata  R       0:07      4 i-hemanth,i-kush,i-roja,i-venkata
venkata@t-green:/home/venkata/test/slurm_arr$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
ig0          up   infinite      4  alloc i-hemanth,i-kush,i-roja,i-venkata
batch*       up   infinite      4  alloc i-hemanth,i-kush,i-roja,i-venkata
venkata@t-green:/home/venkata/test/slurm_arr$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
```

sacct is a command-line tool used to display accounting information for SLURM jobs. The sacct -j option followed by a job ID number can be used to display detailed information about a specific job.

sacct -j 295
--format=user,jobid,jobname,Partition,NodeList,ntasks,reqtres%30,submit,start,end,elapsed,Timelimit,ReqMem,State

```
venkata@t-green:/home/venkata/test/slurm_arr$ sacct -j 295 --format=user,jobid,jobname,Partition,NodeList,ntasks,reqtres%30,submit,start,end,elapsed,Timelimit,ReqMem,State
     User JobID           JobName  Partition        NodeList   NTasks                       ReqTRES              Submit               Start                 End    Elapsed  Timelimit     ReqMem      State
--------- ------------ ---------- ---------- --------------- -------- ------------------------------ ------------------- ------------------- ------------------- ---------- ---------- ---------- ----------
  venkata 295_1        arrayTest      batch i-hemanth,i-ku+          billing=4,cpu=4,mem=8M,node=4 2023-04-20T20:10:41 2023-04-20T20:10:41 2023-04-20T20:10:51   00:00:10   00:05:00         8M  COMPLETED
          295_1.batch       batch                i-hemanth        1                                2023-04-20T20:10:41 2023-04-20T20:10:41 2023-04-20T20:10:51   00:00:10                     COMPLETED
  venkata 295_2        arrayTest      batch i-hemanth,i-ku+          billing=4,cpu=4,mem=8M,node=4 2023-04-20T20:10:41 2023-04-20T20:10:52 2023-04-20T20:11:02   00:00:10   00:05:00         8M  COMPLETED
          295_2.batch       batch                i-hemanth        1                                2023-04-20T20:10:52 2023-04-20T20:10:52 2023-04-20T20:11:02   00:00:10                     COMPLETED
  venkata 295_3        arrayTest      batch i-hemanth,i-ku+          billing=4,cpu=4,mem=8M,node=4 2023-04-20T20:10:41 2023-04-20T20:11:03 2023-04-20T20:11:13   00:00:10   00:05:00         8M  COMPLETED
          295_3.batch       batch                i-hemanth        1                                2023-04-20T20:11:03 2023-04-20T20:11:03 2023-04-20T20:11:13   00:00:10                     COMPLETED
  venkata 295_4        arrayTest      batch i-hemanth,i-ku+          billing=4,cpu=4,mem=8M,node=4 2023-04-20T20:10:41 2023-04-20T20:11:14 2023-04-20T20:11:25   00:00:11   00:05:00         8M  COMPLETED
          295_4.batch       batch                i-hemanth        1                                2023-04-20T20:11:14 2023-04-20T20:11:14 2023-04-20T20:11:25   00:00:11                     COMPLETED
```

ls -al j_295*

```
venkata@t-green:/home/venkata/test/slurm_arr$ ls -al j_295*
-rw-rw-r-- 1 venkata venkata 0 Apr 20 20:10 j_295_1.err
-rw-rw-r-- 1 venkata venkata 4 Apr 20 20:10 j_295_1.out
-rw-rw-r-- 1 venkata venkata 0 Apr 20 20:10 j_295_2.err
-rw-rw-r-- 1 venkata venkata 4 Apr 20 20:10 j_295_2.out
-rw-rw-r-- 1 venkata venkata 0 Apr 20 20:11 j_295_3.err
-rw-rw-r-- 1 venkata venkata 4 Apr 20 20:11 j_295_3.out
-rw-rw-r-- 1 venkata venkata 0 Apr 20 20:11 j_295_4.err
-rw-rw-r-- 1 venkata venkata 4 Apr 20 20:11 j_295_4.out
venkata@t-green:/home/venkata/test/slurm_arr$
```

cat j_295_*.out

```
venkata@t-green:/home/venkata/test/slurm_arr$ cat j_295_*.err
venkata@t-green:/home/venkata/test/slurm_arr$ cat j_295_*.out
1 3
2 4
3 5
4 6
venkata@t-green:/home/venkata/test/slurm_arr$
```

Task 2: Chain submission

vi chain.sh

```
F=$(sbatch --parsable sub1.sh)&& sbatch --dependency=afterok:${F} sub2.sh
```

vi sub1.sh

```
#!/bin/bash
#SBATCH --job-name=first        # Job name
#SBATCH --partition=batch          # send job to c0 partition
#SBATCH --ntasks=1              # Ask for 1 CPU
#SBATCH --time=00:02:00         # Time limit hrs:min:sec
#SBATCH --mem=2mb               # Job memory request on each node
#SBATCH --output=j_f_%j.out     # Standard output and error log
#SBATCH --error=j_f_%j.err      # Standard output and error log

cd ~/test/slurm_chain

ml Python/3.9.6-GCCcore-11.2.0
rm first.txt

sleep 1m

python -c "with open('first.txt', 'w') as fw: s=f\"first job, job ID is {$SLURM_JOB_ID}\n\"; fw.write( s); "

# sbatch sub1.sh
# squeue
```

vi sub2.sh

```
#!/bin/bash
#SBATCH --job-name=second       # Job name
#SBATCH --partition=batch          # send job to c0 partition
#SBATCH --ntasks=1              # Ask for 1 CPU
#SBATCH --time=00:05:00         # Time limit hrs:min:sec
#SBATCH --mem=2mb               # Job memory request on each node
#SBATCH --output=j_s_%j.out     # Standard output and error log
#SBATCH --error=j_s_%j.err      # Standard output and error log

cd ~/test/slurm_chain

ml Python/3.9.6-GCCcore-11.2.0
python -c "fo=open('first.txt', 'r+');line=fo.read();print (\"Read Line: %s\" % (line));fo.close()"
```

[-d ~/test/slurm_chain] | mkdir -p ~/test/slurm_chain
cd ~/test/slurm_chain
cp -r /a/util/demo/slurm/run/slurm_chain/ ~/test/slurm_chain/
bash chain.sh

```
venkata@t-green:/home/venkata/test/slurm_arr$ [ -d ~/test/slurm_chain ] | mkdir -p ~/test/slurm_chain
venkata@t-green:/home/venkata/test/slurm_arr$ cd ~/test/slurm_chain
venkata@t-green:/home/venkata/test/slurm_chain$ cp /a/util/demo/slurm/run/slurm_chain/chain.sh ~/test/slurm_chain/
venkata@t-green:/home/venkata/test/slurm_chain$ cp -r /a/util/demo/slurm/run/slurm_chain/ ~/test/slurm_chain/
venkata@t-green:/home/venkata/test/slurm_chain$ ls -al
total 52
drwxr-xr-x 3 venkata venkata 6144 Apr 20 20:15 .
drwxrwxr-x 7 venkata venkata 6144 Apr 20 14:40 ..
-rw-r--r-- 1 venkata venkata   74 Apr 20 20:15 chain.sh
-rw-rw-r-- 1 venkata venkata   25 Apr 20 14:49 first.txt
-rw-rw-r-- 1 venkata venkata    0 Apr 20 14:48 j_f_280.err
-rw-rw-r-- 1 venkata venkata    0 Apr 20 14:48 j_f_280.out
-rw-r--r-- 1 venkata venkata   57 Apr 20 14:40 j_f_37.err
-rw-r--r-- 1 venkata venkata    0 Apr 20 14:40 j_f_37.out
-rw-r--r-- 1 venkata venkata    0 Apr 20 14:40 j_s_38.err
-rw-r--r-- 1 venkata venkata   36 Apr 20 14:40 j_s_38.out
drwxr-xr-x 2 venkata venkata 6144 Apr 20 20:15 slurm_chain
-rw-r--r-- 1 venkata venkata  632 Apr 20 14:45 sub1.sh
-rw-r--r-- 1 venkata venkata  574 Apr 20 14:45 sub2.sh
venkata@t-green:/home/venkata/test/slurm_chain$ vi chain.sh
venkata@t-green:/home/venkata/test/slurm_chain$ vi sub1.sh
venkata@t-green:/home/venkata/test/slurm_chain$ vi sub2.sh
venkata@t-green:/home/venkata/test/slurm_chain$ bash chain.sh
Submitted batch job 300
venkata@t-green:/home/venkata/test/slurm_chain$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
               300     batch   second  venkata PD       0:00      1 (Dependency)
               299     batch    first  venkata  R       0:03      1 i-hemanth
venkata@t-green:/home/venkata/test/slurm_chain$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
ig0          up   infinite      1  alloc i-hemanth
ig0          up   infinite      3   idle i-kush,i-roja,i-venkata
batch*       up   infinite      1  alloc i-hemanth
batch*       up   infinite      3   idle i-kush,i-roja,i-venkata
venkata@t-green:/home/venkata/test/slurm_chain$
```

sacct -j 299,300
--format=user,jobid,jobname,Partition,NodeList,ntasks,reqtres%30,submit,start,end,elapsed,Timelimit,ReqMem,State

```
venkata@t-green:/home/venkata/test/slurm_chain$ sacct -j 299,300 --format=user,jobid,jobname,Partition,NodeList,ntasks,reqtres%30,submit,start,end,elapsed,Timelimit,ReqMem,State
     User JobID           JobName  Partition        NodeList   NTasks                       ReqTRES              Submit               Start                 End    Elapsed  Timelimit     ReqMem      State
--------- ------------ ---------- ---------- --------------- -------- ------------------------------ ------------------- ------------------- ------------------- ---------- ---------- ---------- ----------
  venkata 299              first      batch       i-hemanth          billing=1,cpu=1,mem=2M,node=1 2023-04-20T20:16:00 2023-04-20T20:16:00 2023-04-20T20:17:03   00:01:03   00:02:00          2M  COMPLETED
          299.batch        batch                  i-hemanth       1                               2023-04-20T20:16:00 2023-04-20T20:16:00 2023-04-20T20:17:03   00:01:03                       COMPLETED
  venkata 300             second      batch       i-hemanth          billing=1,cpu=1,mem=2M,node=1 2023-04-20T20:16:00 2023-04-20T20:17:03 2023-04-20T20:17:04   00:00:01   00:05:00          2M  COMPLETED
          300.batch        batch                  i-hemanth       1                               2023-04-20T20:17:03 2023-04-20T20:17:03 2023-04-20T20:17:04   00:00:01                       COMPLETED
venkata@t-green:/home/venkata/test/slurm_chain$
```

Output after execution

```
venkata@t-green:/home/venkata/test/slurm_chain$ cat first.txt
first job, job ID is 299
venkata@t-green:/home/venkata/test/slurm_chain$ cat j_s_300.out
Read Line: first job, job ID is 299

venkata@t-green:/home/venkata/test/slurm_chain$
```

Individually submit an array job to cluster

User: venkata

Create new batch file

vi ind_job_arr.sh

Add the following content

```
#!/bin/bash
#SBATCH --job-name=t_green_ind_venkata
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=1
#SBATCH --cpus-per-task=1
#SBATCH --time=00:01:00
#SBATCH --mem=1mb
#SBATCH --array=1-2
#SBATCH --output=j_%A_%a.out
#SBATCH --error=j_s_%j.err
cat /a/util/demo/slurm/run/slurm_arr/file_${SLURM_ARRAY_TASK_ID}.txt
```

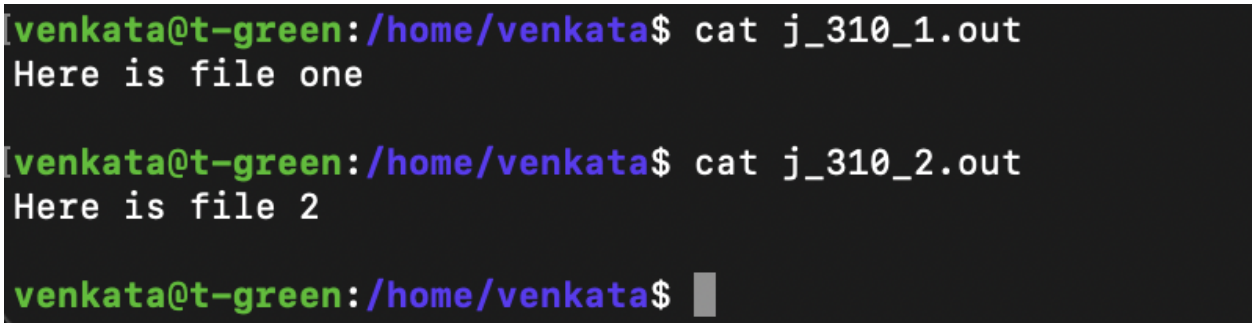
Execute the sbatch command

sbatch ind_job_arr.sh

```
venkata@t-green:/home/venkata$ vi ind_job_arr.sh
venkata@t-green:/home/venkata$ sbatch ind_job_arr.sh
Submitted batch job 310
venkata@t-green:/home/venkata$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
ig0          up   infinite      4   idle i-hemanth,i-kush,i-roja,i-venkata
batch*       up   infinite      4   idle i-hemanth,i-kush,i-roja,i-venkata
```

```
venkata@t-green:/home/venkata$ cat j_310_
j_310_1.out  j_310_2.out
venkata@t-green:/home/venkata$ cat j_310_
j_310_1.out  j_310_2.out
```

Outputs are logged into the files: j_%job%_1.out, j_%job%_2.out



Individually submit an array job to cluster

User: kush

Create new batch file

vi ind_job_arr.sh

Add the following content

```
#!/bin/bash
#SBATCH --job-name=t_green_ind_kush
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=1
#SBATCH --cpus-per-task=1
#SBATCH --time=00:01:00
#SBATCH --mem=1mb
#SBATCH --array=1-2
#SBATCH --output=j_%A_%a.out
#SBATCH --error=j_s_%j.err
cat /a/util/demo/slurm/run/slurm_arr/file_${SLURM_ARRAY_TASK_ID}.txt
```

Execute the sbatch command

sbatch ind_job_arr.sh

Outputs are logged into the files: j_%job%_1.out, j_%job%_2.out

```
kush01@t-green:/home/kush$ vi ind_job_arr.sh
kush01@t-green:/home/kush$ ls
start_inst.sh  stop_inst.sh
kush01@t-green:/home/kush$ cd
kush01@t-green:/home/kush01$  vi ind_job_arr.sh
kush01@t-green:/home/kush01$ sbatch ind_job_arr.sh
Submitted batch job 312
kush01@t-green:/home/kush01$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
ig0          up   infinite      4   idle i-hemanth,i-kush,i-roja,i-venkata
batch*       up   infinite      4   idle i-hemanth,i-kush,i-roja,i-venkata
kush01@t-green:/home/kush01$ cat j_312_
j_312_1.out  j_312_2.out
kush01@t-green:/home/kush01$ cat j_312_1.out
Here is file one

kush01@t-green:/home/kush01$ cat j_312_2.out
Here is file 2

kush01@t-green:/home/kush01$
```

Individually submit an array job to cluster

User: hemanth

Create new batch file

vi ind_job_arr.sh

Add the following content

```
#!/bin/bash
#SBATCH --job-name=t_green_ind_hemanth
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=1
#SBATCH --cpus-per-task=1
#SBATCH --time=00:01:00
#SBATCH --mem=1mb
#SBATCH --array=1-2
#SBATCH --output=j_%A_%a.out
#SBATCH --error=j_s_%j.err
cat /a/util/demo/slurm/run/slurm_arr/file_${SLURM_ARRAY_TASK_ID}.txt
```

Execute the sbatch command

sbatch ind_job_arr.sh

Outputs are logged into the files: j_%job%_1.out, j_%job%_2.out

```
[hemanth@t-green:/home/hemanth$ vi ind_job_arr.sh
[hemanth@t-green:/home/hemanth$ sbatch ind_job_arr.sh
Submitted batch job 314
[hemanth@t-green:/home/hemanth$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
ig0          up   infinite      4   idle i-hemanth,i-kush,i-roja,i-venkata
batch*       up   infinite      4   idle i-hemanth,i-kush,i-roja,i-venkata
[hemanth@t-green:/home/hemanth$ cat j_314_
j_314_1.out  j_314_2.out
[hemanth@t-green:/home/hemanth$ cat j_314_1.out
Here is file one

[hemanth@t-green:/home/hemanth$ cat j_314_2.out
Here is file 2

hemanth@t-green:/home/hemanth$
```

Individually submit an array job to cluster

User: roja

Create new batch file

vi ind_job_arr.sh

Add the following content

```
#!/bin/bash
#SBATCH --job-name=t_green_ind_roja
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=1
#SBATCH --cpus-per-task=1
#SBATCH --time=00:01:00
#SBATCH --mem=1mb
#SBATCH --array=1-2
#SBATCH --output=j_%A_%a.out
#SBATCH --error=j_s_%j.err
cat /a/util/demo/slurm/run/slurm_arr/file_${SLURM_ARRAY_TASK_ID}.txt
```

Execute the sbatch command

sbatch ind_job_arr.sh

Outputs are logged into the files: j_%job%_1.out, j_%job%_2.out

```
roja@t-green:/home/roja$ vi ind_job_arr.sh
roja@t-green:/home/roja$ sbatch ind_job_arr.sh
Submitted batch job 316
roja@t-green:/home/roja$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
ig0          up   infinite      4   idle i-hemanth,i-kush,i-roja,i-venkata
batch*       up   infinite      4   idle i-hemanth,i-kush,i-roja,i-venkata
roja@t-green:/home/roja$ cat j_316_
j_316_1.out  j_316_2.out
roja@t-green:/home/roja$ cat j_316_1.out
Here is file one

roja@t-green:/home/roja$ cat j_316_2.out
Here is file 2

roja@t-green:/home/roja$
```

Sub assignment 3: All the team members have registered for Lucidcharts.

And have collaborated for the following diagram

https://lucid.app/lucidchart/7f2494da-b82c-4474-87aa-53ef9354b0af/edit?viewport_loc=-102%2C-103%2C2581%2C1300%2CNSTQuOvS557b&invitationId=inv_2afb396e-faa1-4383-983f-855f4f08869b