

High Performance Computing Facility

Indian Institute of Technology Goa

HPC user manual

Last updated: July 2024

Please read this manual carefully before using the HPC facility.

1. **IIT Goa faculty/ researchers/ students are authorized to access this facility.**
2. **These instructions are only for a user with some experience. You need proficiency in Linux and parallel programming.**
3. **For new user registrations visit https://hpc.iitgoa.ac.in/ and fill the "New user registration" form. You will receive an email upon completion of the user creation process.**
4. **User usage policy:**
    a) **Regular user (UG/ PG students) will get 100 GB storage limit.**
    b) **Ph.D. / Research users will get 1 TB storage limit.**
    c) **Faculty will get 20 TB storage limit.**

1. About the cluster IIT Goa HPC

This cluster having 16 CPU nodes & 1 GPU node and 1 DGX GPU node.

Configuration:

CPU Nodes:

- 2x Intel Xeon-Gold 6248 Processor, 20 Core, total 40 cores,
- 192 GB Memory.
- Mellanox 100Gbps Interconnect
- Total 16 nodes and 640 Cores for running MPI, OpenMP and
- Hybrid jobs.
- The Node names are node 1 to node 16.

GPU Node:

- 2x Intel Xeon-Gold 6248 Processor, 20 Core, total 40 cores,
- 192 GB Memory
- Mellanox 100Gbps Interconnect
- GPU Node having Nvidia Tesla V100 GPU Card and installed
- with necessary drivers and configured to work with slurm job
- scheduler.
- Total 40 Cores for running GPU jobs
- The node are named as gpu1.

Storage:

- This cluster also has 200 TB Lustre Storage in this storage is allotted to "/home".

1 a. How to Access HPC

To Access or login to the HPC can be done using SSH (Secure Shell). You can use software like Putty or directly access SSH from Terminal / command prompt.

HPC Host name: hpc.iitgoa.ac.in

1. Connecting from Terminal/ Command line.

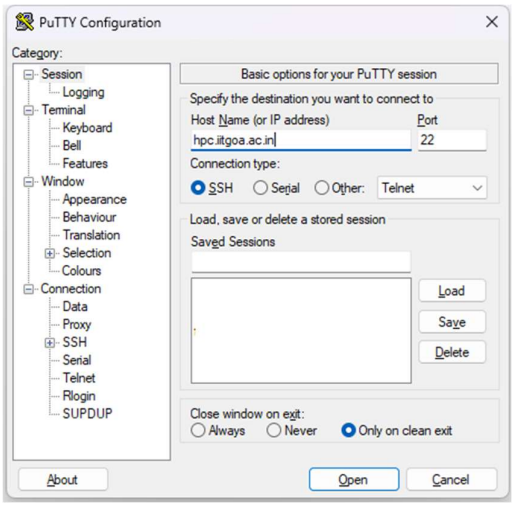
   Example:

   ssh user1@hpc.iitgoa.ac.in

   ssh demouser@hpc.iitgoa.ac.in

2. Connect using Putty.

1. Connect to the hpc.iitgoa.ac.in



2. Enter your username and then password



2.Slurm Job Scheduler - CLI

a. Submitting Jobs

To submit job, user need to create a job script as follows

For GPU Nodes:

```
#!/bin/bash
#SBATCH --job-name=newjob
#SBATCH --partition=gpu
#SBATCH --mail-user=user@iitgoa.ac.in
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=2
#SBATCH --gres=gpu:1
#SBATCH --mail-type=ALL
#SBATCH --workdir=/home/iitgoa/
#SBATCH --output=newjob%j.out

cd $SLURM SUBMIT DIR

echo $SLURM JOB NODELIST > hostfile $SLURM JOBID
```

For CPU Nodes:

To submit job user need to create a job script as follows

```
#!/bin/bash
#SBATCH -N 1
#SBATCH --ntasks-per-node=32
#SBATCH -J <testrun>
#SBATCH -p route
#SBATCH --time=24:00:00
#SBATCH -o slurmANAj.out
#STDOUT
#SBATCH -e slurmANAj.err
#STDERR
#SBATCH --export=all
```

```
#SBATCH--mail-user=user@iitgoa.ac.in
#SBATCH --mail-type=ALL

cd $SLURM SUBMIT DIR

echo $SLURM JOB NODELIST > hostfile $SLURM
JOBID


## OpenMP Case

## Make sure the ppn value and OMP NUM THREADS
value are same

 or

##leave with SLURM NTASKS env variable

#export OMP NUM THREADS=$SLURM NTASKS

#<your executable> >& out $SLURM JOBID


#MPI Case

### Only one executable allowed

mpirun -np <npvalue>
<your/executable/with/path> >& out $SLURM JOBID
```

In this above file change what are in <...> to its appropriate value and save as for example myscript.sh

Then submit the myscript.sh as follows :

```
#sbatch myscript.sh
```

This command submits jobs to scheduler and returns a job id. This jobid

will be used later for monitoring and managing jobs.

b. Monitor the schedule queue

```
#squeue
```

Above command will out put the information about the job is scheduler queue with its states and other details.

```
squeue

10131D PARTITION NAME USER ST TIME NODES NODE LIST (REASON)
```

In the above output ST means State of the Job in queue R means Running, P means Pending, C means Completing and etc.

c. Cancel a job

```
scancel <jobid>
```

This command will cancel or delete the job

```
scancel -u <username>
```

This command will cancel all the job of user <username>

```
scancel -t PENDING -u <username>
```

This command will cancel all Pending jobs of user <username>

d. Other scheduler commands

```
scontrol show jobid <jobid>
```

This command will show the full details of job which is in queue.

```
scontrol show jobid -dd <jobid>
```

This command will show the full details of job including its job script file which is in queue.

```
sacct                                                          --
format=JoblD,JobName,MaxRSS,UserCPU,SystemCPU,CpuTime  -j
<jobid>
```

This command will show the details of the job which

is completed and a day old.

For example:

```
sacct                                                          --
format=JoblD,JobName,MaxRSS,UserCPU,SystemCPU,CpuTime    -j
2588
JobID        JobName    MaxRSS     UserCPU    SystemCPU
 CPUTime

2588                    MA00_5LSt+ 09:30:13   02:15.525
 09:40:48
2588. batch batch       4588K      00:00.156  00:00.093
 09:40:48
2588.0       pmi_proxy  2020K      09:30:13   02:15.431
 00:18:09
```

```
scontrol hold <jobid>
```

This command hold the job which is queue but not running.

```
scontrol resume <jobid>
```

This command will release the hold job.

```
sstat --format=AvePages,AveRSS,AveVMSize,Jobl D
-j <jobid>
```

This command will show the statistics of a running job.

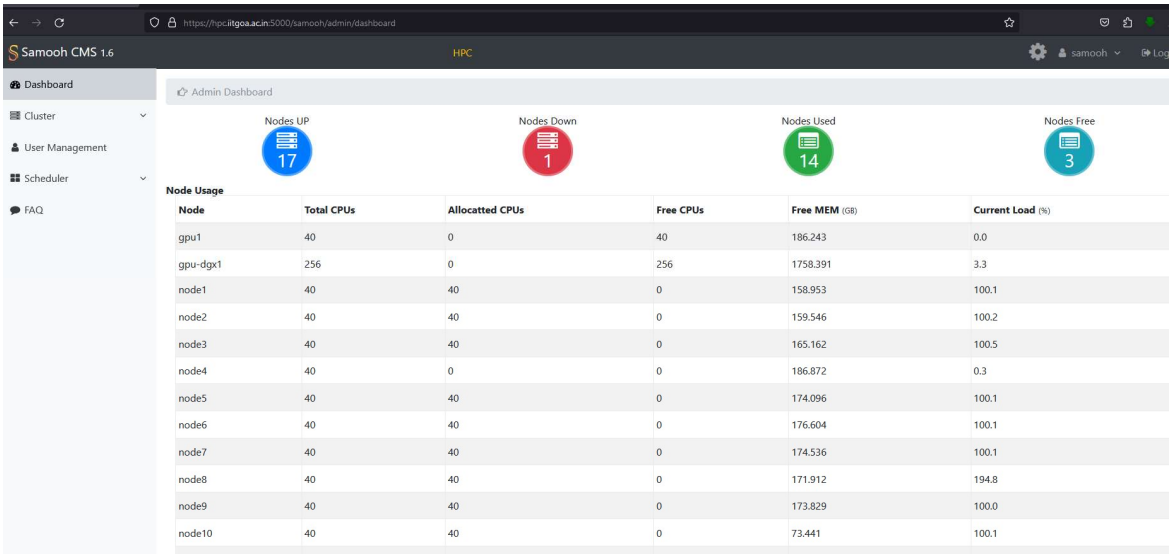
e. Slurm Partition details

This cluster is partitioned with respect to scheduler to use the resource fairly. The following is the details of the slurm partition information.

S.No	Partition	No of Nodes	No of Cores	Wall
1	cpu	16	640	INFINITE
2	gpu	1	40	INFINITE
3	All	17	680	INFINITE

3. Samooh CMS -Job submission portal

Samooh CMS version 1.6 is web based cluster management suite which provides the job submission and managing portal.

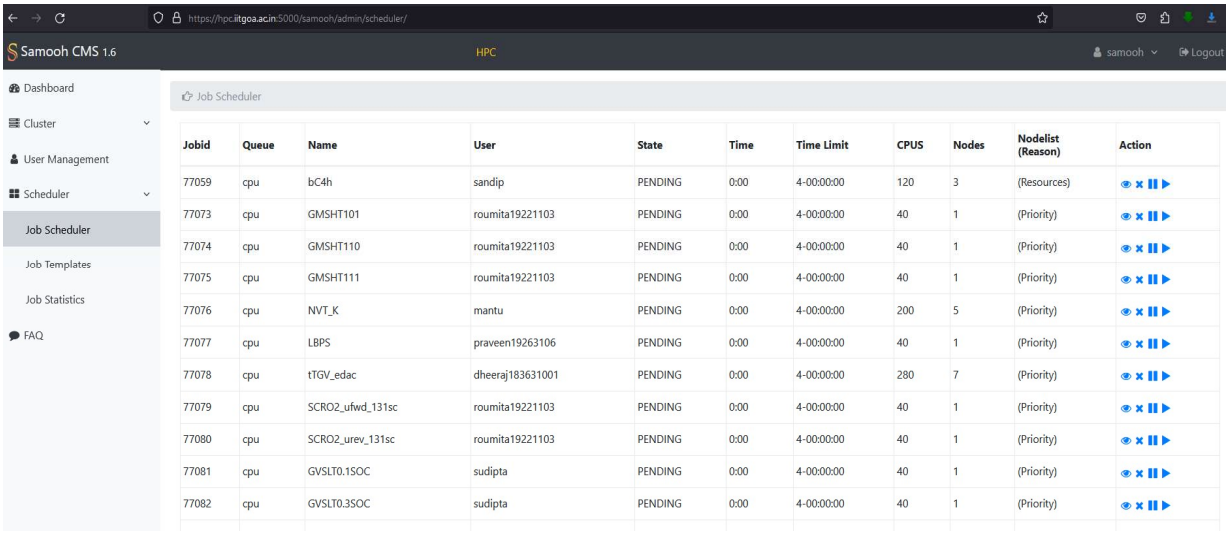


a. Dashbord

Users Dashboard will have Node Usage, Scheduler Partition usage details. Node Usage details include Node name, Total CPU, Allocated CPU, Free CPU, Free MEM and Current Load of each node. Scheduler Partition usage details includes Job ID, Partition Name, Username, Job Name, Job State, Time, Time Limit, CPUs, Nodes, Node list, Actions. Actions include Information of job, delete job, hold job and release holed job.

b. Job scheduler

Using this menu user can monitor the all their scheduled jobs details like Job ID, Partition Name, User name, Job Name, Job State, Time, Time Limit, CPUs, Nodes, Node list. Users also can do the actions include ® get information of job, "X" delete job, hold job and release old job.



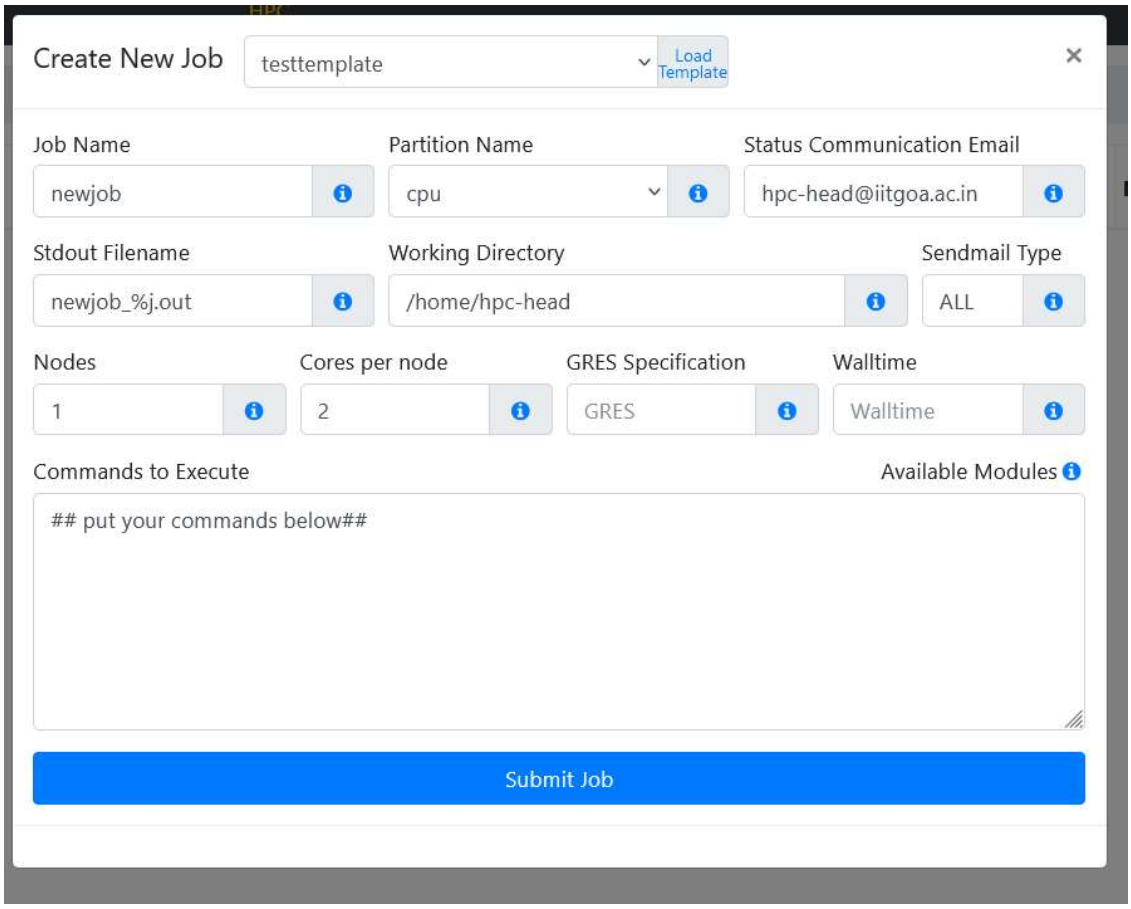
Jobid	Queue	Name	User	State	Time	Time Limit	CPUS	Nodes	Nodelist (Reason)	Action
77059	cpu	bC4h	sandip	PENDING	0:00	4-00:00:00	120	3	(Resources)	
77073	cpu	GMSHT101	roumita19221103	PENDING	0:00	4-00:00:00	40	1	(Priority)	
77074	cpu	GMSHT110	roumita19221103	PENDING	0:00	4-00:00:00	40	1	(Priority)	
77075	cpu	GMSHT111	roumita19221103	PENDING	0:00	4-00:00:00	40	1	(Priority)	
77076	cpu	NVT_K	mantu	PENDING	0:00	4-00:00:00	200	5	(Priority)	
77077	cpu	LBPS	praveen19263106	PENDING	0:00	4-00:00:00	40	1	(Priority)	
77078	cpu	tTGV_edac	dheeraj183631001	PENDING	0:00	4-00:00:00	280	7	(Priority)	
77079	cpu	SCRO2_ufwd_131sc	roumita19221103	PENDING	0:00	4-00:00:00	40	1	(Priority)	
77080	cpu	SCRO2_urev_131sc	roumita19221103	PENDING	0:00	4-00:00:00	40	1	(Priority)	
77081	cpu	GVSLT0.1SOC	sudipta	PENDING	0:00	4-00:00:00	40	1	(Priority)	
77082	cpu	GVSLT0.3SOC	sudipta	PENDING	0:00	4-00:00:00	40	1	(Priority)	

Using this menu user can submit new job to users. This will give window to provide necessary details like job name, communication email, stdout file, working

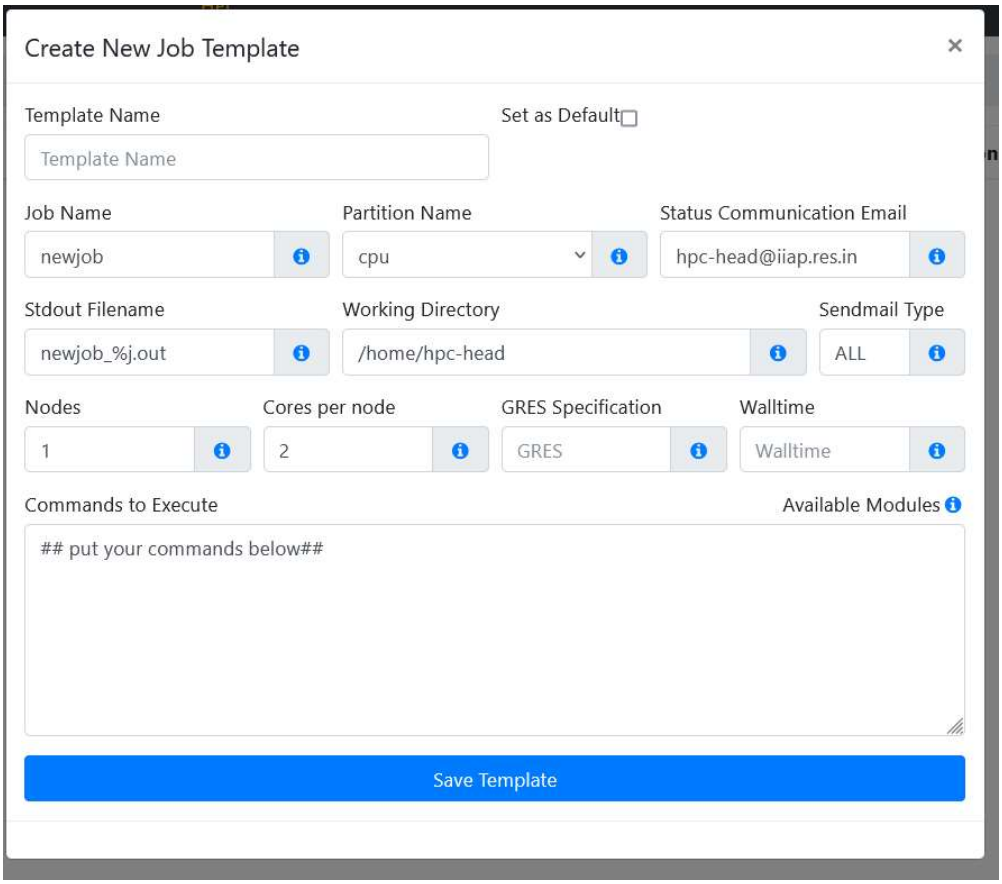
directory, mail type, number of nodes, and number of cores, wall time and very importantly job execution commands.

This add new job window has info button at each input, which will give required information about the input field.

c. Scheduler -> Job Scheduler -> Add Job



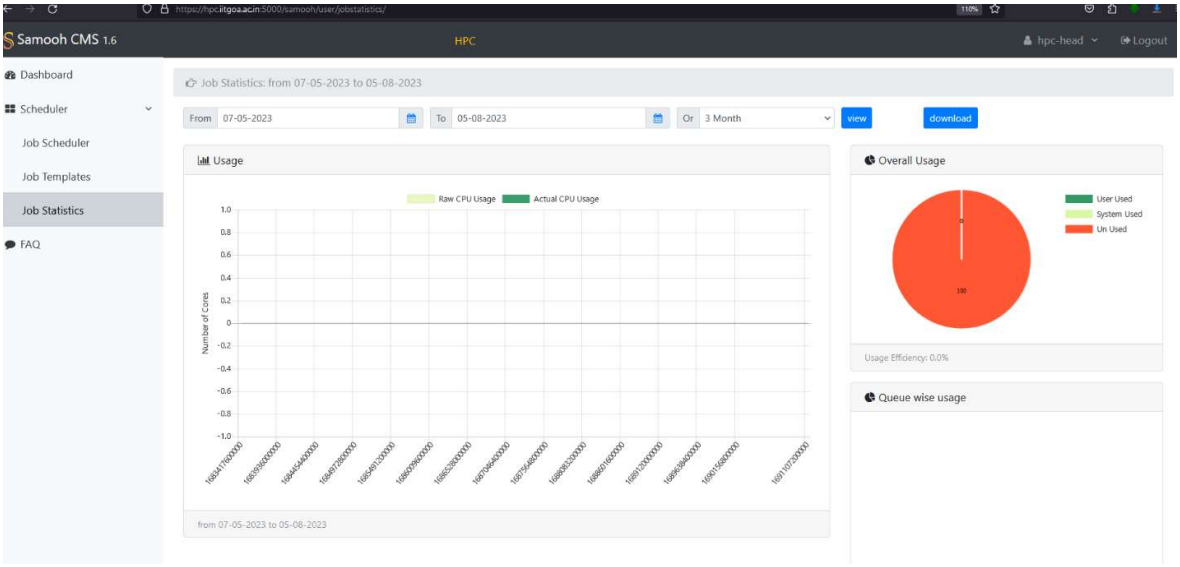
d. Job Template



Job Template is used for saving frequently submitted jobs details to the template show that it can be later used. Due to that its saving lot of re typing of job commands using this menu user can job template for various type of jobs. In this job template user can set important settings of job like job name, communication email, stdout file, working directory, mail type, number of nodes, and number of cores, wall time and very importantly job execution commands.

e. Job Statistics

Using this menu user can see their cluster usage statistics as graph from given date interval. This include Date wise number of core usage, Raw and Actual Usage, Overall Usage, and Queue/Partition wise usage.



4.Installing Packages

The master node is configured in such a way that all software installed in */home* will be available to all compute nodes. So we recommend to choose the installation path as follows.

```
/home/<usename>/softwares/<softwarename>/<version>
```

For Example: FFTW 3.3.7 installed in

```
/home/hpcuser/softwares/fftw/3.3.7
```

With this method we can have any number of version of same package.

So if the package need ./configure command to install then add

```
--prefix=
/home/<username>/softwares/<softwarename>/<version>
```

If the packed need cmake to install add -DINSTALL_PREFIX as follows

```
-DINSTA LL PREFIX
homekusemainc>/softwarcsi<softwarenamc>/<version>
```

Then do make and make install.

5.Module

Once the package is installed it is recommended to create module

for the same that package necessary environment variable loaded and

remove using module command

To load module

```
module load <modulename>/<version>
For ex: module load fftw/3.3.7
To unload module
module rm <mdoule>/<version>
For Ex: module rm fftw/3.3.7
```

Suppose if module version is not specified then the

higher version number is chosen automatically.

a. How to create module file

- Create folder for module

```
mkdir /home/username/modulefiles
```

- Create folder under */home/<username>/modulefiles/<packagename>* with package name

```
mkdir /home/<username>/modulefileg<packagename>
```

- then inside the */home/<username>/modulefiles/<packagename>*

create a file with version string with following main contents

```
cd /home/<username>/modulefiles<packagename>
```

vim 3.3.7 (version string)

1. First line should be #%Module1.0
2. Section module-what is, prepend-path as following example

```
#%Module1.0#####################################################################

proc ModulesHelp { } {
        puts stderr " "
        puts stderr "This module loads the example program"
        puts stderr "toolchain."
        puts stderr "\nVersion 1.0\n"
}
module-whatis "Name: example"
module-whatis "Version: 1.0"
module-whatis "Category: runtime library"
module-whatis "Description: example independant module"
module-whatis "URL http://www.google.com/"

set             version                                 1.0
prepend-path    PATH                            /opt/ohpc/pub/libs/example/1.0/bin
prepend-path     MANPATH                       /opt/ohpc/pub/libs/example/1.0/man
prepend-path    INCLUDE                        /opt/ohpc/pub/libs/example/1.0/include
prepend-path    LD_LIBRARY_PATH          /opt/ohpc/pub/libs/example/1.0/lib

setenv          EXAMPLE_DIR             /opt/ohpc/pub/libs/example/1.0
setenv          EXAMPLE_BIN              /opt/ohpc/pub/libs/example/1.0/bin
setenv           EXAMPLE_LIB             /opt/ohpc/pub/libs/example/1.0/lib
setenv          EXAMPLE_INC             /opt/ohpc/pub/libs/example/1.0/include
```

b. Available Modules

```
# module av

----------------- /opt/ohpc/pub/moduledeps/gnu9 ---------------
likwid/5.0.1         mpich/3.3.2-ofi       mpich/3.3.2-ucx  (D)
mvapich2/2.3.2       openblas/0.3.7              openmpi4/4.0.4
pdtoolkit/3.25.1

------------------ /opt/apps/modulefiles ---------------------
   cuda/11.1.1

----------------- /opt/ohpc/admin/modulefiles -----------------
   spack/0.15.0

----------------- /opt/ohpc/pub/modulefiles -------------------
 EasyBuild/4.3.0            cmake/3.16.2            hwloc/2.1.0
ohpc(L)                       papi/5.7.0        prun/2.0    (L)
valgrind/3.15.0             autotools   (L)    gnu9/9.3.0    (L)
libfabric/1.10.1    os       pmix/3.1.4          ucx/1.8.0

  Where:
   D:  Default Module
   L:  Module is loaded

Use "module spider" to find all possible modules and extensions.
Use "module keyword key1 key2 ..." to search for all possible
modules matching any of the "keys".
```

c. List loaded modules

```
module list
```

d. Load a module

```
module load <modulename>
```

e. Remove a loaded module

```
module remove <modulename>
```