Intro and access to HPC

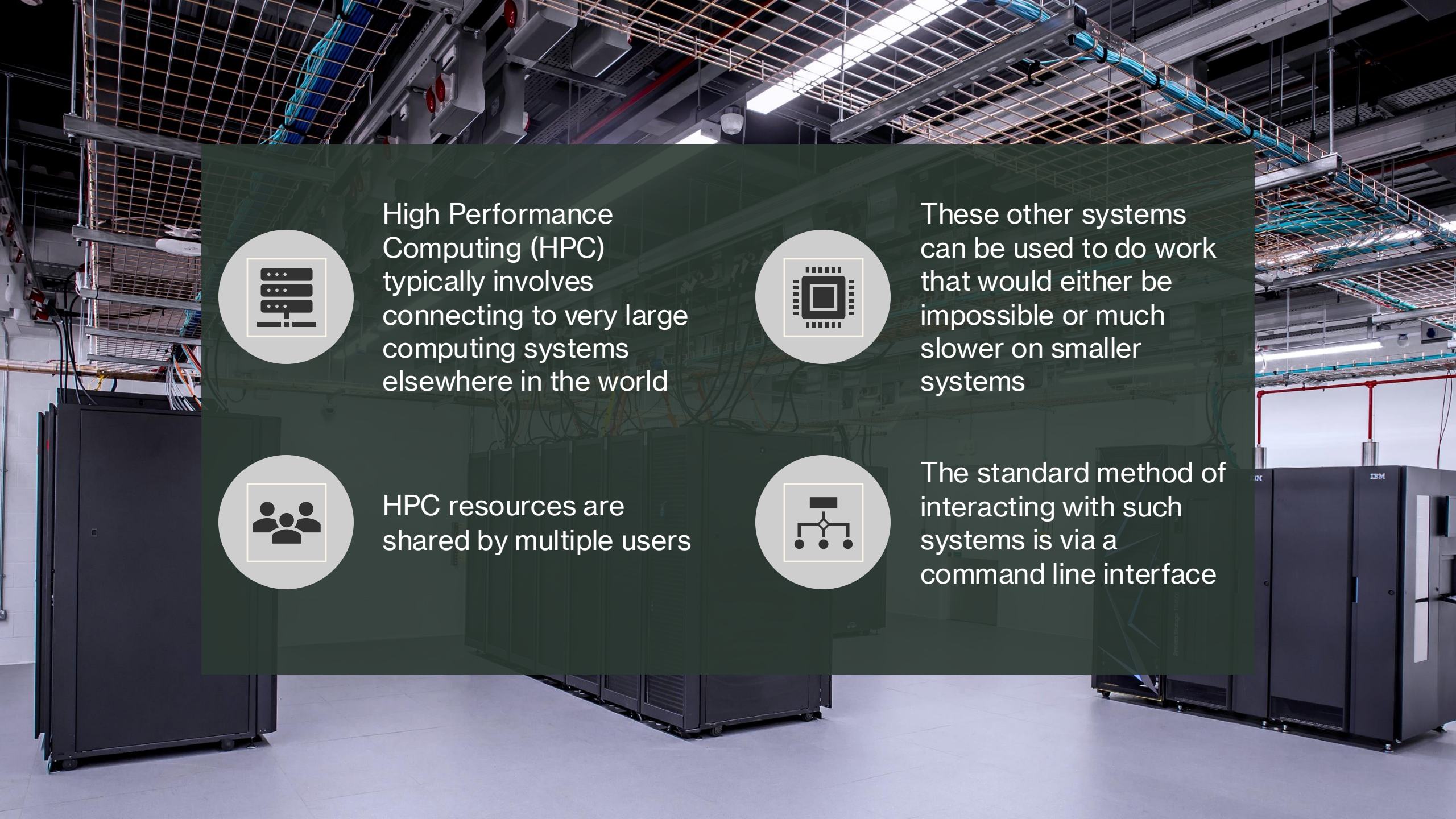
Turing HPC training 2025

25-26 November

Rosie Wood, Iain Stenson, David Llewellyn-Jones

Today's Schedule

Time	Session	Speaker
10:00 - 10:45	Intro and access to HPC	Iain Stenson
10:45 - 11:00	Coffee	
11:00 - 11:45	Intro and access to HPC	Rosie Wood
11:45 - 13:00	Multi-GPU inference	Rosie Wood
13:00 - 14:00	Lunch	
14:00 - 15:00	Multi-node training with Lightning	David Llewellyn-Jones
15:00 - 16:30	Python profiling	James Bishop



High Performance Computing (HPC) typically involves connecting to very large computing systems elsewhere in the world

These other systems can be used to do work that would either be impossible or much slower on smaller systems

HPC resources are shared by multiple users

The standard method of interacting with such systems is via a command line interface

UK HPC landscape

- Tier 0 – International (e.g. LUMI)
- Tier 1 – National (e.g. Isambard-AI, Dawn ARCHER2)
- Tier 2 – Regional or Consortium (e.g. Baskerville, JADE 2)
- Tier 3 – Local, University based (Apocrita at QMUL)

Image from https://www.gov.uk/government/publications/future-of-compute-review/the-future-of-compute-report-of-the-review-of-independent-panel-of-experts

Baskerville HPC

- 228 A100 GPUs
- 57 Compute Nodes
  - 4 GPUs per node
  - 11 A100-80 GPU nodes
  - 46 A100-40 GPU nodes
- 5 400 TB Storage

Dawn

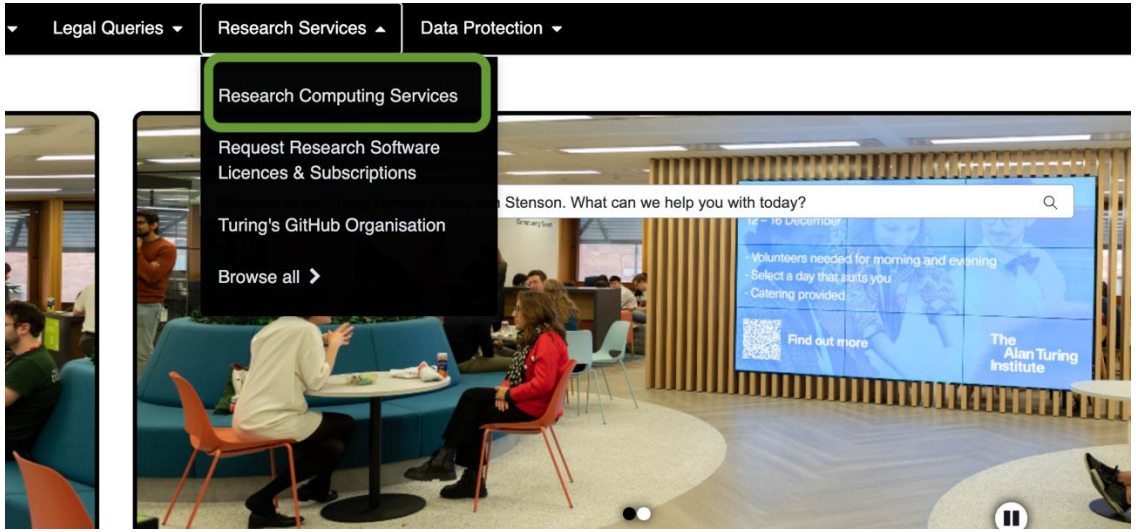
- 1 024 Intel® Data Center GPU Max 1550 GPUs
- 256 nodes,
  - 4 GPUs per node
  - 8 GPU tiles/stacks per node
  - 2 048 GPU *stacks/tiles*
- Each stack offers 64 GiB memory
- *Flat* or *composite* device hierarchy

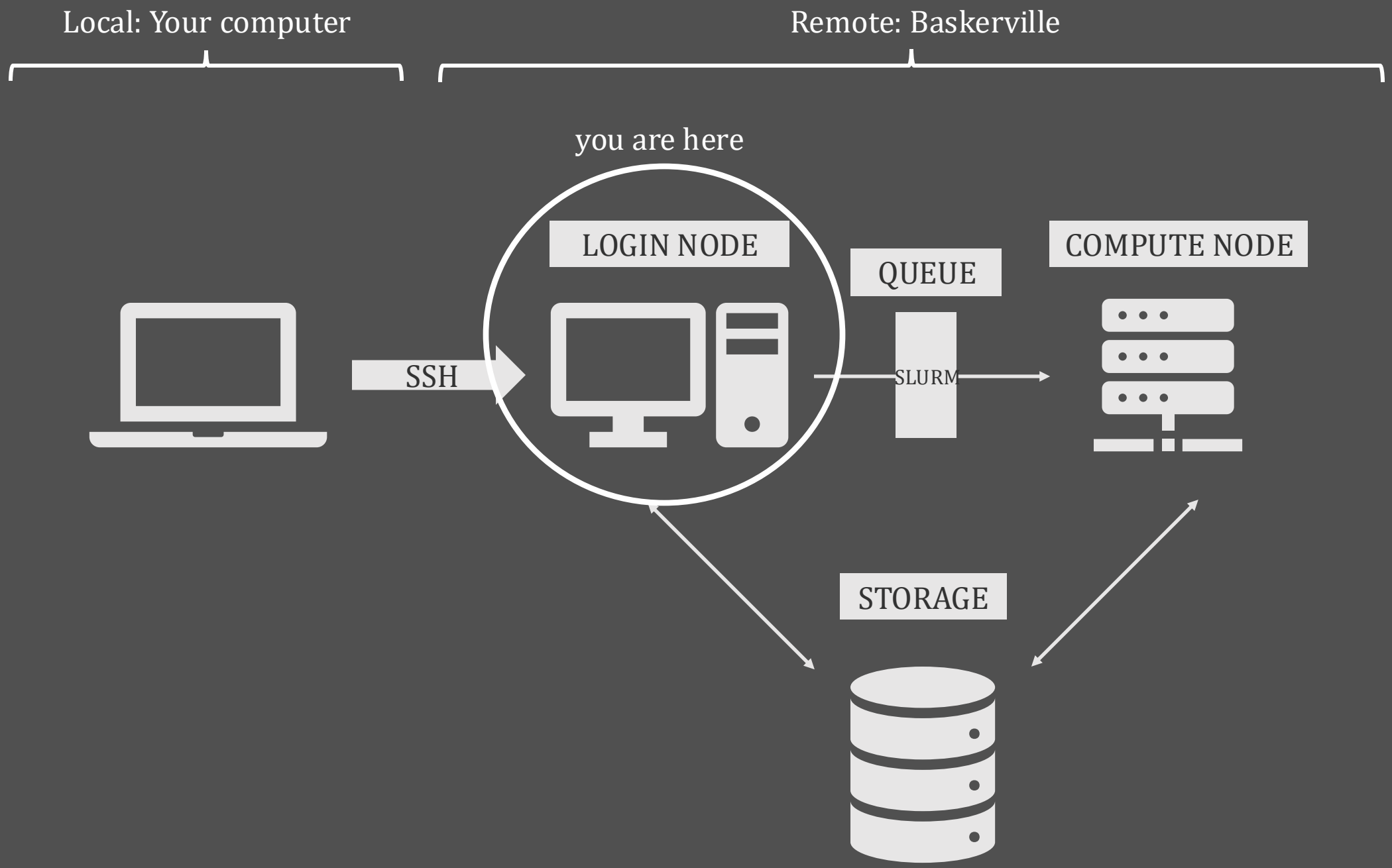
Isambard-AI

- 5 280 GH200 GPUs
- 1 320 compute nodes with GH200 chips.
  - 4 GH200 chips per node
  - Each GH200 chip has 1 Grace CPU and 1 H100 GPU
  - 72 cores per GPU
- Storage 27 000 TB

Applying for access

- Tier 0 / Tier 1
  - Apply directly via various methods.
- Isambard AI and DAWN
  - Apply via https://portal-airr.isambard.ac.uk/ and CC the Research Computing team.
- Baskerville
  - Alan Turing Institute's "Service Now"




Local: Your computer
Remote: Baskerville
you are here
LOGIN NODE
QUEUE
COMPUTE NODE
SSH
SLURM
STORAGE

Logging in to Baskerville

1. Login to Baskerville

   ```
   ssh uname@login.baskerville.ac.uk
   ```

2. Change directory to `/bask/projects/v/vjgo8416-training2511`

   ```
   cd /bask/projects/v/vjgo8416-hpc2511
   ```

3. Make directory for your work

   ```
   mkdir $USER
   ```

   ```
   cd $USER
   ```

4. Clone the training repo

   ```
   git clone --recursive https://github.com/alan-turing-institute/hpc-training-nov-2025.git
   ```

Logging in to Isambard-AI

1. Download `clifton` as per https://docs.isambard.ac.uk/user-documentation/guides/login/

   ```
   curl -L https://github.com/isambard-sc/clifton/releases/latest/download/clifton-macos-aarch64 -o clifton
   chmod u+x clifton
   mv clifton /usr/local/bin/
   ```

2. Run `clifton` and write your ssh config

   ```
   clifton auth --write-config true
   ```

3. Then, login to Isambard-AI

   ```
   ssh proj_name.aip2.isambard
   ```

Break 

10:45-11:00

Set up on a new HPC

- Adding to your .bashrc
  - aliases (e.g. `alias rm=‘rm -i’`)
  - cache directories (`PIP_CACHE`, `HF_HOME`)

- Linking your project directory
  - `ln -s /path/to/project/dir`
  - i.e. for today’s work: `ln -s /bask/projects/v/vjgo8416-hpc2511`

Useful tools/commands

- Terminal multiplexers – particularly useful on dodgy wifi
  - Tmux
  - Screen

- Baskerville specific useful commands:
- `my_baskerville` - lists information about your projects
- `bask-show-quota` - shows how much space you are using in your home directory and active projects
- `my_quota` - shows how much space you are using in your home directory

Storing data and working directories

Generally, work from your project directory not your home directory.

Baskerville

- Home space ($HOME):
  - 20 GB
  - For settings, ssh keys etc
  - Not for storing data
  - This size does not change
- Project space:
  - At *least* 1 TB (can request more)
  - Should be for data, job scripts, output etc

Isambard-AI

- Home space ($HOME):
  - 50 GB
  - For settings, ssh keys etc
  - Not for storing data
- Project space ($PROJECTDIR):
  - 50 TB
  - Should be for data, job scripts, output etc
- Scratch ($SCRATCH):
  - No hard limit
  - Deletes after end of project

Moving data

- For directories, use `-r` (recursive)
- Use `–p` to preserve the file permissions and modification times

scp

`scp -pr source destination`

- e.g. to move a file to Baskerville : `scp test.txt uname@login.baskerville.ac.uk:~/`
- e.g. to get a file from Baskerville: `scp uname@login.baskerville.ac.uk:~/test.txt .`

rsync

`rsync -vrutz source destination`

- e.g. to move a file to Baskerville: `rsync -vrutz test.txt uname@login.baskerville.ac.uk:~/`
- e.g. to get a file from Baskerville: `rsync -vrutz uname@login.baskerville.ac.uk:~/test.txt .`
  - Note: Trailing slash means files in this directory: `./my_dir/` = `./my_dir/*`

Moving data

1. Open a new terminal window/tab
2. Create a new file:

```
touch test.txt
```

3. Move your file to Baskerville:

```
scp test.txt uname@login.baskerville.ac.uk:~/
```

4. Login to Baskerville (or return to window where you are logged in)
5. List your files using `ls`

```
ls
```

Modules

- Many HPC systems manage software using modules
- “A *module* is a self-contained description of a software package – it contains the settings required to run a software package and, usually, encodes required dependencies on other software packages.” - https://carpentries-incubator.github.io/hpc-intro/15-modules/index.html
- View available modules with `module avail`
- Search for module with `module spider pytorch`
- Load modules with `module load PyTorch`, `module load Python/3.10.4`
- List loaded modules with `module list`
- Unload modules with `module unload PyTorch` or `module purge` to remove all modules

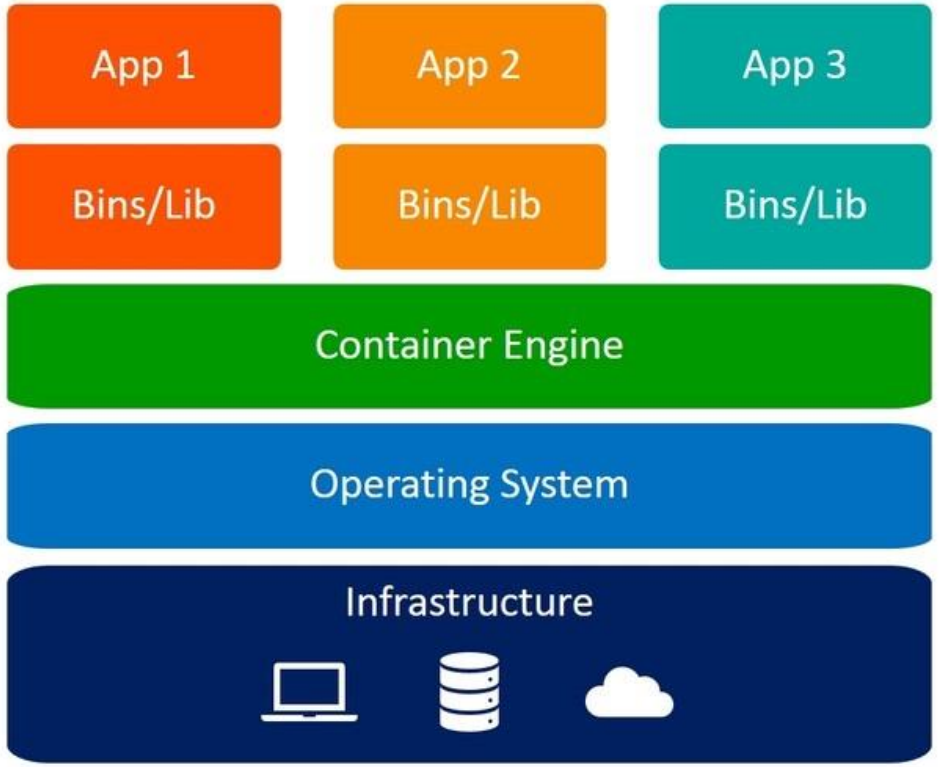
Self-installed software

- Not all software will be available as a module
- So, options for self installing software are:

1. Set up your own python environment using **venv**:
   - https://docs.baskerville.ac.uk/self-install/?h=venv#self-installing-python-software
   - `module load Python/3.11.3 (or whichever version you want to use)`
   - `python -m venv`
   - `source venv/bin/activate`
   - `pip install xxx`
2. Set up **conda**:
   - https://docs.isambard.ac.uk/user-documentation/guides/python/#conda-installing-and-using-miniforge
   - `conda create –n env_name python=3.11.3`
   - `conda activate env_name`
   - `conda install xxx`

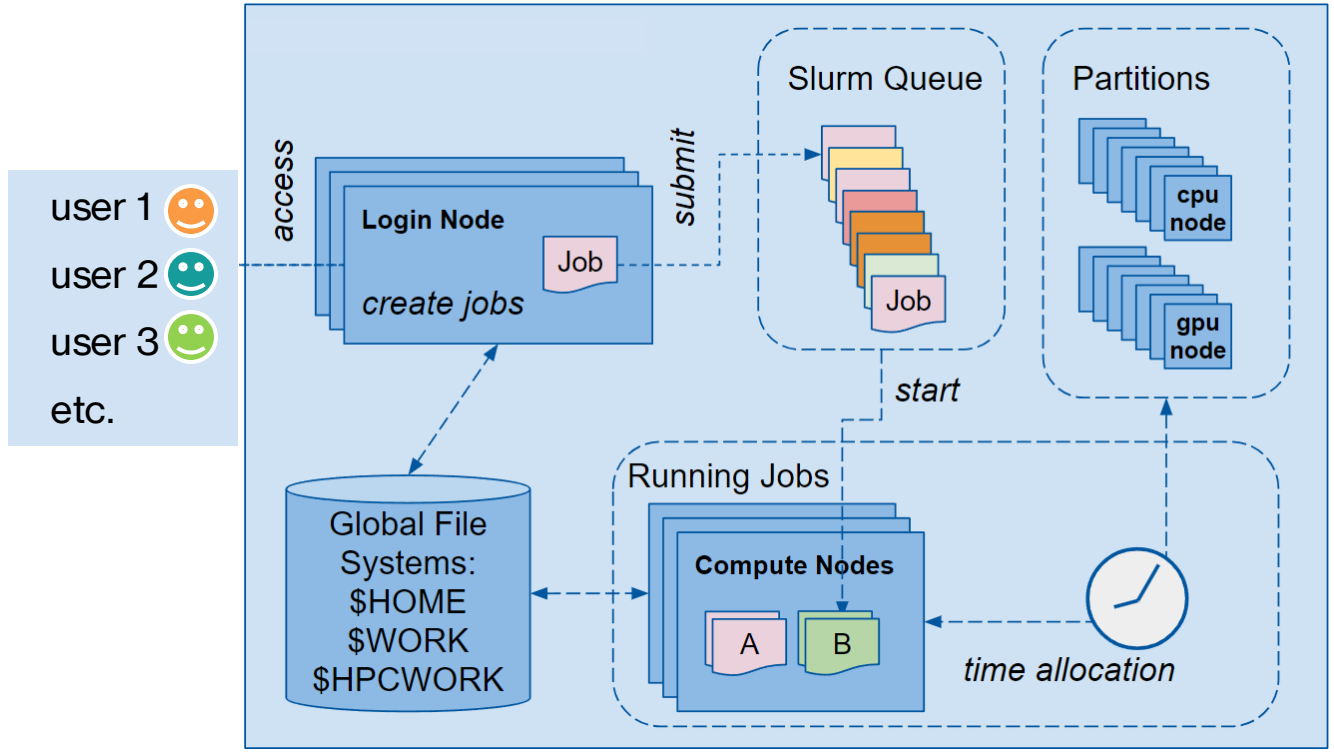
Containers

- Can completely package up your software + dependencies, including the operating system
- Can use pre-built images directly from Docker Hub (or other container registries)
- For HPC, you need to use:
  - Podman / Podman-HPC
  - Singularity/Apptainer



Slurm

- “Slurm is an open source, fault-tolerant, and highly scalable **cluster management and job scheduling system** for large and small Linux clusters.”
- Job scheduler
- Main need to know commands:
  - `sbatch`
  - `srun`(to get interactive job or can also use inside batch jobs)
  - `squeue`
  - https://slurm.schedmd.com/quickstart.html



https://help.itc.rwth-aachen.de/en/service/rhr4fjjutttf/article/6357a2a6944143a9867f71951e249737/

Interactive (srun) vs. batch (sbatch)

- Batch jobs are the most common way of running jobs on HPC
  - `sbatch`
  - non-interactive
  - you submit -> jobs enters queue -> job runs -> you get logs/output

- Interactive jobs are can be useful for running experimental workloads on compute nodes
  - `srun`
  - interactive

Lunch 

13:00-14:00