

# AI HACKATHON DOCUMENTATION

This guide provides essential information to help you get started with resources and tools you might need or use throughout the hackathon.

**Puhti Project invitation:** <https://my.csc.fi/projects/invitation/a154ee53-03e8-413c-9d8a-9ff8e4aee243>

We have two themes from which you can choose and build your project

## 1. On boarding Assistant

An AI onboarding assistant that can provide personalized, accessible, and efficient support to new employees during their initial weeks at the company

Data and starter code: <https://github.com/ngnmai02/onboarding-template.git>

## 2. Smart meeting assistant

A chatbot-like AI agent that can automatically listen to meetings, transcribe audio to text, and generate concise, accurate summaries

Data and starter code: <https://github.com/CSCfi/smart-meeting-agent/tree/master>

## Platforms and Resources

### 1. MyCSC Portal

#### Access and Registration:

- URL: <https://my.csc.fi>
- Sign in with your (HAKA login) or CSC credentials.
- After logging in, connect to the Hackathon project (**Project\_2014873**) and log in to Puhti

### 2. Puhti

#### Access Puhti:

- SSH login or
- Login through OoD: <https://www.puhti.csc.fi/>

#### File System:

- **Home directory:** /users/username
- **Project directory:** /projappl/project\_2014873  
*Shared space for project data and results. Good for collaboration and medium-to-large datasets.*
- **Scratch directory:** /scratch/project\_2014873  
*High-performance, temporary storage. Not backed up. Best used for large intermediate data or temporary files created during computations.*

#### Important Guidelines:

- Do **not** use any internal, sensitive, or non-public CSC data during the hackathon. This includes confidential documents, internal policies, and personal information
- Do **not** use the home directory for heavy I/O operations. Avoid using it for intensive computation or large file storage.
- Do **not** store critical data long-term in the scratch directory, as it may be purged regularly.
- Run heavy computational jobs from the scratch directory to optimize I/O performance.
- Backup essential results to the project directory. (Will be deleted on the expiry of the Project)

# Slurm Job Scheduler

It efficiently allocates and manages computational resources, schedules tasks, and monitors resource utilization. With Slurm, you will submit, monitor, and manage batch jobs across a distributed computing environment.

## Capabilities of Slurm:

- Allocate resources like CPUs, GPUs, memory, and nodes.
- Schedule jobs with detailed parameters (e.g., time limits, priority, resource specifications).
- Automate execution of complex workflows.
- Enable parallel job execution for optimal performance.

## Sample Job Script:

```
#!/bin/bash
#SBATCH --account=project_2014873
#SBATCH --partition=gputest
#SBATCH --nodes=1
#SBATCH --ntasks=1
#SBATCH --cpus-per-task=10
#SBATCH --mem=32G
#SBATCH --time=15
#SBATCH --gres=gpu:v100:1,nvme:10
```

```
module purge
module load pytorch
```

```
tar xf /scratch/project_2001234/cifar-10-python.tar.gz -C $LOCAL_SCRATCH
srn python3 cifar10_cnn.py --data_path=$LOCAL_SCRATCH/cifar-10-batches-py
```

## Submitting a Job:

```
sbatch myjob.sh
```

## Monitoring Jobs:

- List your jobs:  
`squeue -u username`
- Cancel a job:  
`scancel JOB_ID`

## Important links

Slurm job documentation: Puhti: [Puhti example scripts - Docs CSC](#)

Getting started with Machine learning: <https://docs.csc.fi/support/tutorials/ml-starting/>

Machine learning guide: <https://docs.csc.fi/support/tutorials/ml-guide/>

Python guide <https://docs.csc.fi/support/tutorials/python-usage-guide/>

# Virtual Environments and Wrappers

## Some reasons to use Virtual Environments:

- Avoid conflicts between project dependencies.
- Ensure reproducibility of your software environment.
- Easily manage and switch between different setups.

## Python Virtual Environments (venv)

- **Create environment:**

```
python -m venv myenv
```

- **Activate environment:**

```
source myenv/bin/activate
```

- **Install dependencies:**

```
pip install -r requirements.txt
```

## Wrappers (Singularity)

Wrappers (containers) like Singularity - Apptainer, encapsulate software and its dependencies into a self-contained environment. They provide a standardized way to package software, making it portable and ensuring consistent execution across various platforms (Puhti ).

## Reasons to use Wrappers:

- Facilitate software portability and reproducibility.
- Easily deploy software across different computing environments.
- Manage complex dependencies within a single container.

Documentation on Puhti: [Containers - Docs CSC](#)

## Further Support

For support and troubleshooting:

- Visit: <https://docs.csc.fi>

[https://github.com/jomugumya/Ai\\_hakathon\\_Documentation/blob/main/AI%20hackathon%20Documentation.pdf](https://github.com/jomugumya/Ai_hakathon_Documentation/blob/main/AI%20hackathon%20Documentation.pdf)