Group mini-project

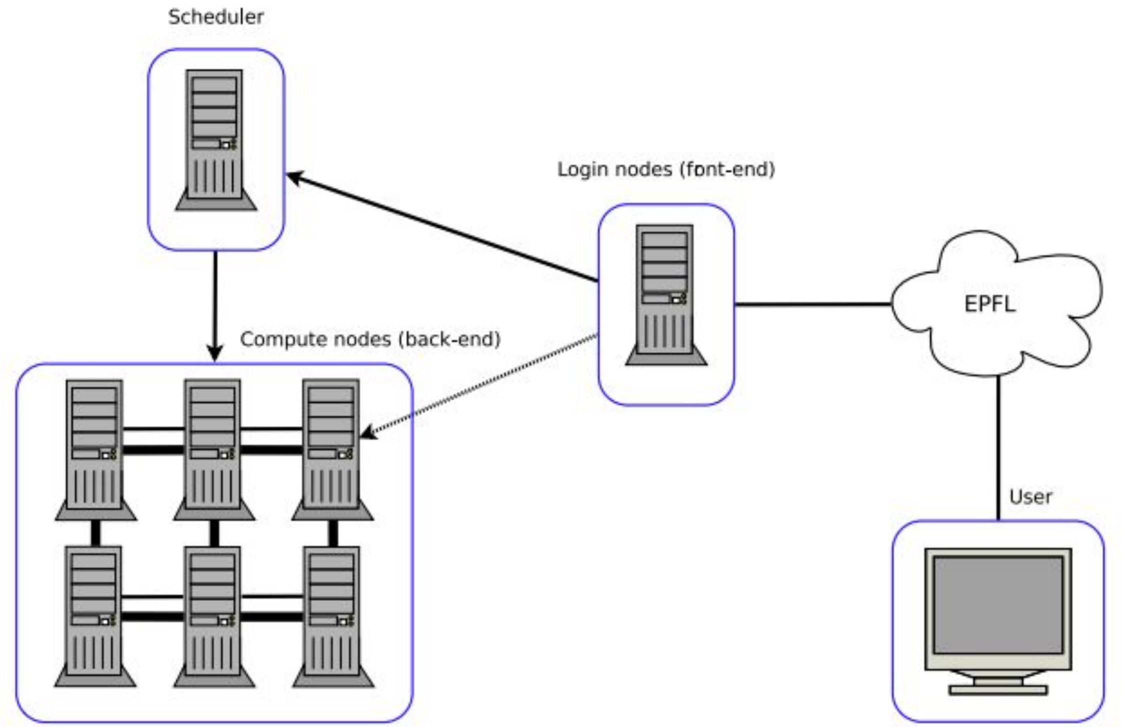
Introduction to the computational resources

The computational demands of Deep Learning models

- Training modern deep learning models on CPUs can take days or even weeks.
- Large datasets and complex architectures demand high memory capacity and fast computation.
- GPUs accelerate parallelized matrix operations, significantly reducing training time.
- However, your own computer, even with a GPU, might not be sufficient, we need more computational power → We need to use a **cluster**

What is a cluster ?

A **cluster** is a network of multiple interconnected computers (nodes) that work together, combining their processing power to handle large-scale (deep learning) tasks efficiently



Antolin, P. (2025). *Illustration of a cluster in MATH-454 Parallel and High Performance Computing, Lecture 1*. EPFL.

The Research Computing Platform (RCP)

- For your mini-project you will use the Research Computing Platform. It provides an access to a GPU cluster of over 400+ GPUs. The cluster is based on the following concepts:

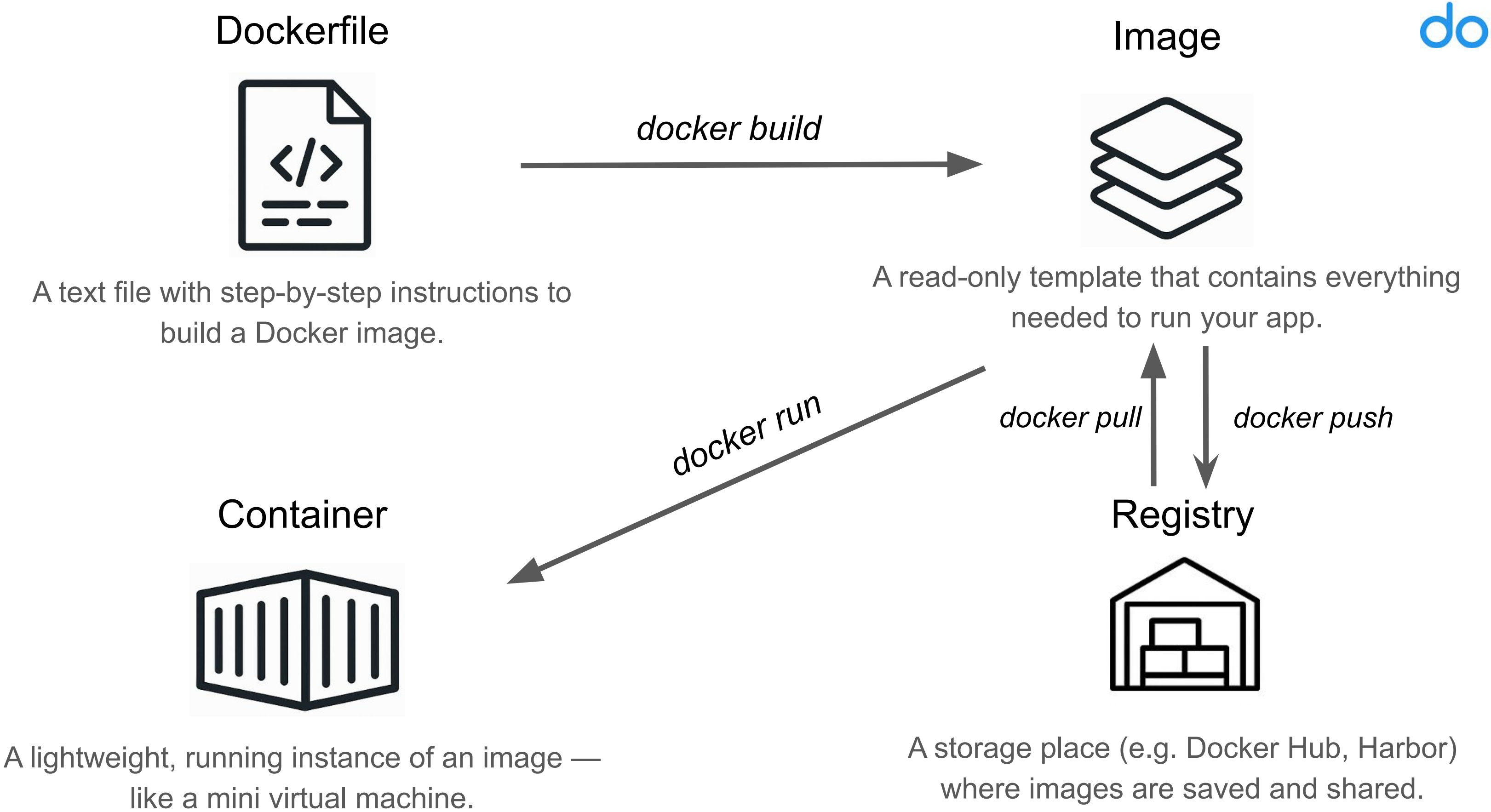
Flexibility: The cluster works with *images* that define a runtime environment, like "*Ubuntu 22.04 with Python and certain packages installed*", allowing everyone to tailor their environment to their unique needs.

You will build them with **Docker**.

Optimization: The RCP Cluster optimizes resources behind the scenes, but we keep it simple and show only the essential steps to run your mini-project scripts.

Fairness and Robustness: A custom scheduler, **run:ai**, makes sure resources are shared fairly between labs and ensures jobs continue running despite GPU failures.

How Docker works? (4 concepts)



Dockerfile

A text file with step-by-step instructions to build a Docker image.

docker build

Image

A read-only template that contains everything needed to run your app.

docker run

docker pull

docker push

Container

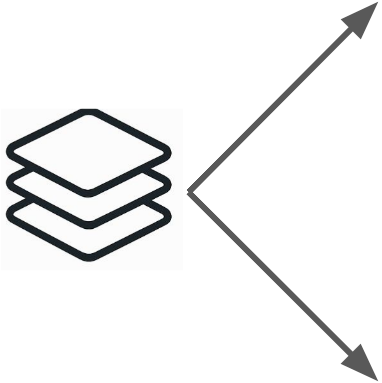
A lightweight, running instance of an image — like a mini virtual machine.

Registry

A storage place (e.g. Docker Hub, Harbor) where images are saved and shared.

How Docker works?

❏ How to get a Docker image?



- **Use existing images** from a **registry**

  e.g., Docker Hub—many official or community-contributed images are already available. Use docker pull to download and run base image.

- **Build your own image using a Dockerfile (custom env)**

  When your project has specific dependencies or tools, you write a Dockerfile and build your own image.

  Note: In this course, images will be built locally on your laptop, and then uploaded to the RCP Registry (Harbor).

What do you need to do to use the RCP ?

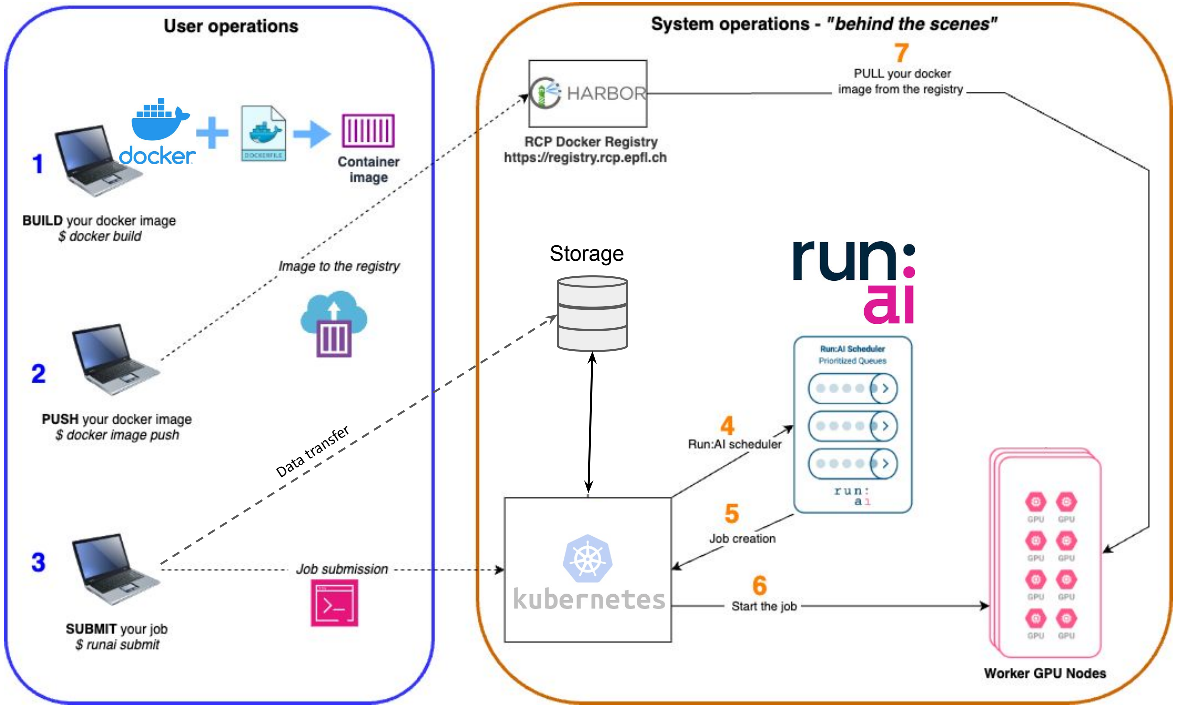
You will need to learn to:

1. Build a **Docker** image (or use a pre-existing one)

2. Push images to the a registry so that is accessible from the cluster

3. Transfer data and Python files from your computer to the cluster, and submit jobs using **run:ai**

Everything else is handled by the cluster itself (4./5./6./7.)



We've put together some documentation to help you get started quickly !

What is the structure within the RCP ?

- `/home/{username}`

This is the active working directory, from which you can run jobs. As this is your personal space on the cluster, feel free to create any folders and organize them as you wish. For example, you can do `mkdir mini_project`, and then `cd mini_project`.

You'll be in `/home/{username}/mini_project`and can create other folder such as `data` or import files such as `main.py`, `train.py`, ...

- `/mnt/course-ee-559/collaborative/group-<group-number>/`

This shared folder allows all team members to store and retrieve files. One member can upload the datasets, while others can access them from their home folders. Use this space for any files that need to be shared, such as model weights. You can NOT run jobs from there.

Before starting the mini-project

- Ensure your code runs without errors first by testing it on small number of iterations and a subset of your dataset before executing it fully on the cluster.
  You can also use interactive jobs to debug your code with the parameter `--interactive` in the `runai submit` command.

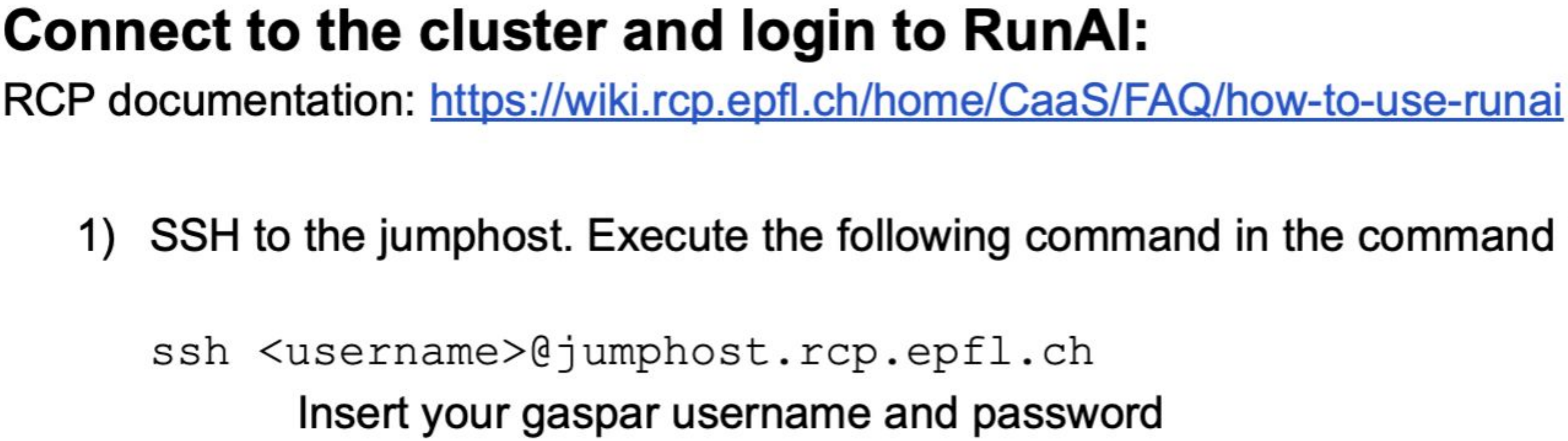
- Familiarize yourself with basic Bash commands to feel more comfortable when working with the cluster (see also this MIT crash course)
- Check the official documentations
  - RPC Wiki : https://wiki.rcp.epfl.ch/en/home/CaaS
  - Run:ai : https://docs.run.ai/v2.18/Researcher/cli-reference/Introduction/

Some useful tools

- For version control and collaborative work use GitHub (This also allows you to directly push/pull modifications of your files from the cluster to your GitHub).
- For VS Code users, install the Remote-SSH extension which allow you to easily modify any file on the cluster from VS Code directly.
- WandB (Weights and Biases) is a great tool for logging, visualizing experiments, and tracking performance over time.
- For Windows users: It is recommended to use Git Bash to connect to the cluster and transfer files, as it provides an environment similar to the Linux-based cluster. (You can also use PowerShell.)
  For macOS and Linux: Everything works the same by default.

Cluster tutorial

Links in the tutorial are provided for you to be able to refer to original sources if you want more details about the steps or something is not working. However, you should be able to set up the job submission by simply following the instructions in the tutorial.



Connect to the cluster and login to RunAI:

RCP documentation: https://wiki.rcp.epfl.ch/home/CaaS/FAQ/how-to-use-runai

1) SSH to the jumphost. Execute the following command in the command line

```
ssh <username>@jumphost.rcp.epfl.ch
```

Insert your gaspar username and password

RCP two types of jobs

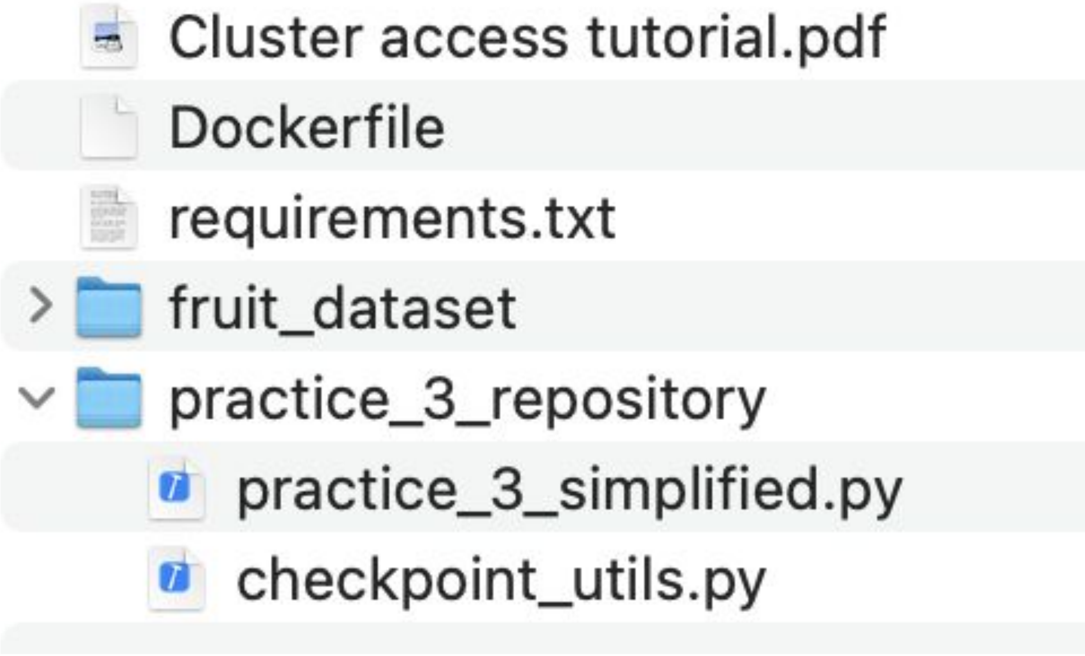
	Interactive jobs	Train jobs
Purpose	Testing & Development.	Training & Compute.
Maximum GPUs	● User: 1x A100 or 1x V100. ● Lab allowed max quota (8x A100 AND 8x V100).	Available GPUs.
Maximum duration	12 hours.	Until job finishes.
Considerations	● Easy interaction with the pod through ‘interactive shell’ (‘ssh’ like), jupyter or vscode.	● Use job **checkpoints** to resume after job killed. ● Avoid using sleep infinity.

Cluster tutorial

- Everywhere where you see `username` you have to replace it with your Gaspar username, unless specifically said otherwise. Everywhere you have to put in password (password, not verification code or anything else) - that is your gaspar password.
- Your local PC - is the computer on which you are currently working. Server or jumphost is an remote epfl server, which can be assessed by ssh through

```
ssh <username>@jumphost.rcp.epfl.ch
```

Cluster tutorial



Cluster access tutorial.pdf

Dockerfile

requirements.txt

fruit_dataset

practice_3_repository

practice_3_simplified.py

checkpoint_utils.py

By the next week lab try to:

- Connect to the cluster and login to runai
- Copy necessary files to the cluster
- Run a simple job with the default nvidia cluster image
- Creating your own docker image and pushing it to registry
- Running a job with your own docker image
- Prepare your questions for the next week