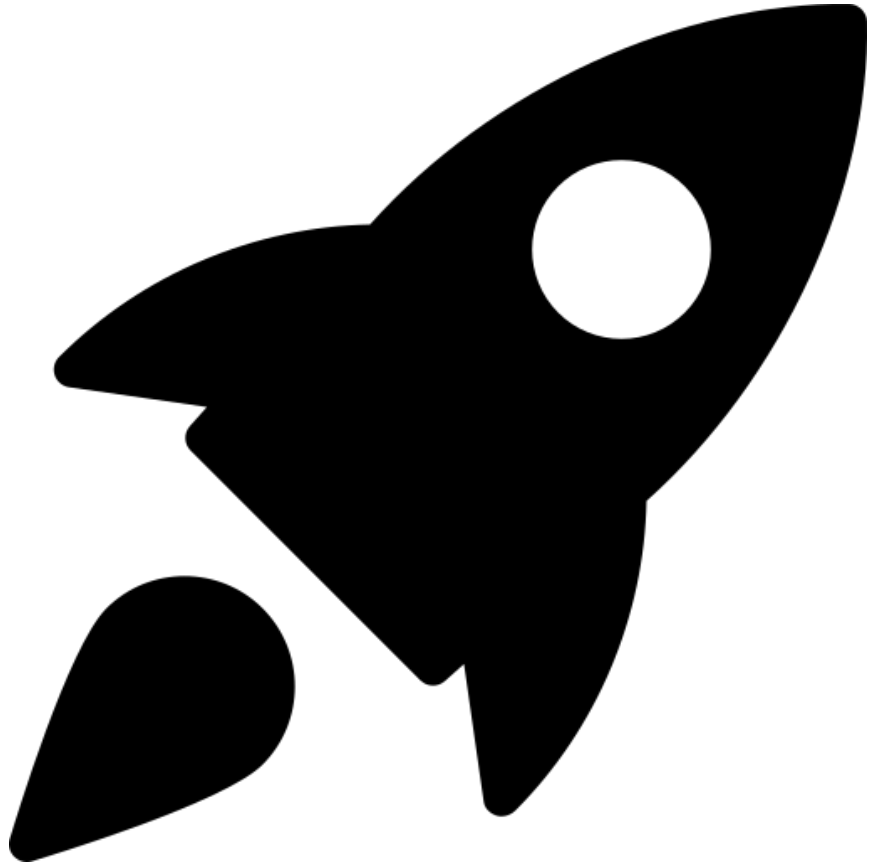
Getting up to speed @AIRLab



Things are moving fast, so this presentation might be outdated and/or incomplete.
Contribution and improvements are welcome!

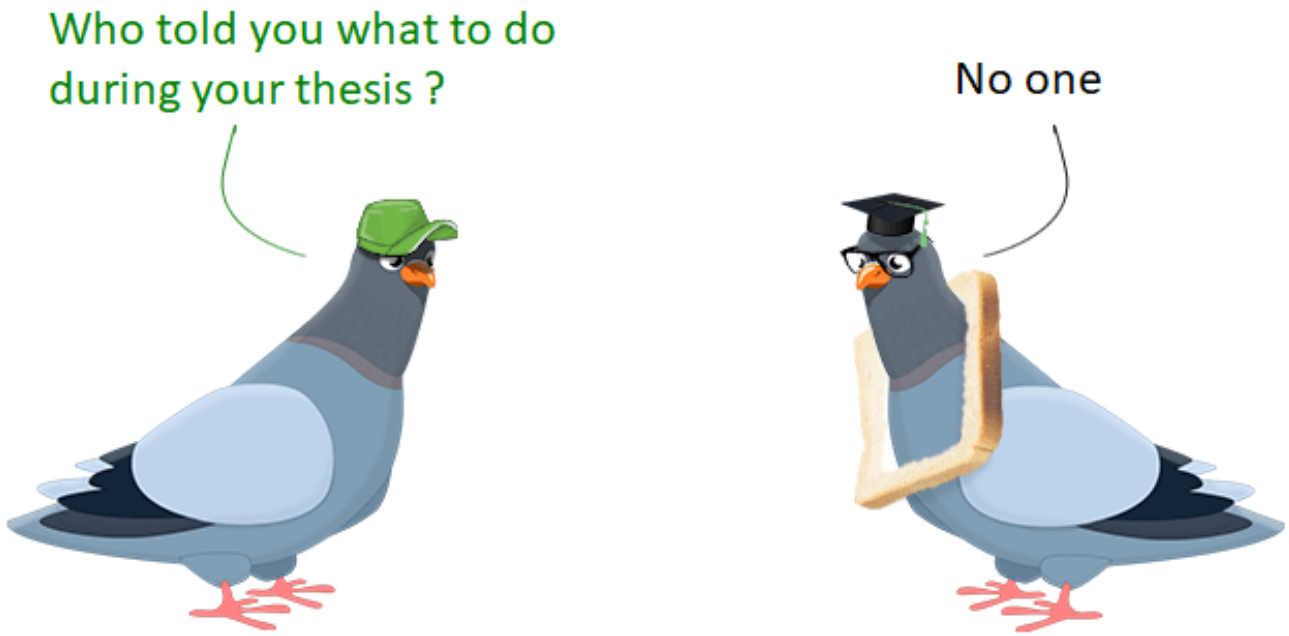
Contacts: nico.catalano@polimi.it

Disclaimer

Most of the concepts in this presentation are just <u>suggestions for a great start</u>

-> It can´t contain everything you will ever need

Every thesis is different, **you will learn to learn by yourself!**



Tesista´s check list

Make sure to be able to access the **building 7** https://airlab.deib.polimi.it/life-in-airlab/service-pages/becoming-an-airlab-user/

Ask your co-advisor for the **VPN** credentials if not already provided

Ask your co-advisor for the servers' **credentials** if not already provided

Ask your co-advisor the **AIRLab Discord** server joining link if not already provided

How to approach the thesis' challenges

- The thesis is **YOUR PROJECT**, so it should reflect **YOUR INTEREST** and ambition
- The co-advisor is there to help you getting there
- You should enjoy the ride
- There is not a fixed time frame

- **Do not hesitate to talk to your co-advisor!**

Things to know before start writing

- Notes on how to write the thesis:
  https://airlab.deib.polimi.it/wp-content/uploads/2021/03/How-to-write-a-thesis.pdf

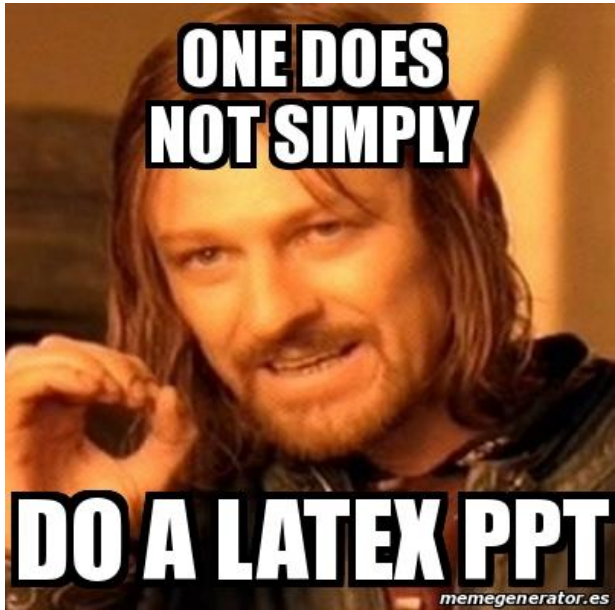
- Rules:
  https://www.ingindinf.polimi.it/en/1/teaching/lectures-and-exams/degree-examinations

- POLIMI's LATEX (Overleaf) template:
  https://www.overleaf.com/gallery/tagged/polimi

- POLIMI's MS PowerPoint template:
  https://polimi365-my.sharepoint.com/:f:/g/personal/10306579_polimi_it/Eu7UoRKAz-pHmHKuXXwdFd8Bn4JVyBfeYbMOm3uIGg4pkg?e=zLNdnj

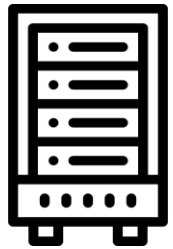
General Suggestions

- Take notes during meetings
- Schedule regular update meeting with your co-advisor
- Always prepare what to show and talk about for scheduled meetings
- Keep track of your readings (eg using Zotero)
- **Do not use Latex for presentations, only for your thesis!**



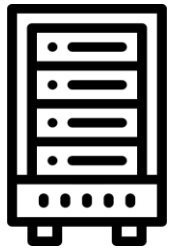
AIRLab Network

AIRLab´s servers



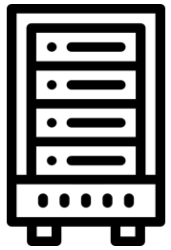
westworld – GPU Server

8 x 11GB Nvidea 1080 TI
40 core
252 GB RAM



elysium – GPU Server

4 x 24GB Nvidea RTX6000
1 x 12GB Nvidea Titan X
1 x 11GB Nvidea 1080TI
48 core
252 GB RAM



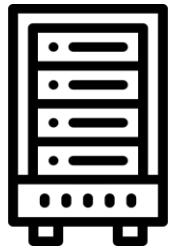
magrathea – GPU Server

8 x 24GB Nvidea RTX6000
112 core
252 GB RAM

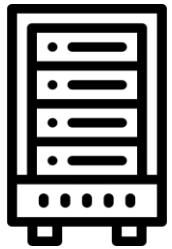
multiverse – NAS

Notice! To obtain the servers IPs, please ask your co-supervisor.

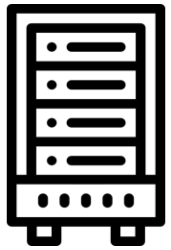
AIRLab´s servers



westworld – GPU Server



elysium – GPU Server



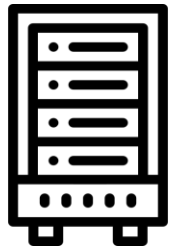
magrathea – GPU Server

multiverse – NAS

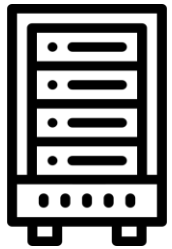
Computation only

Storage

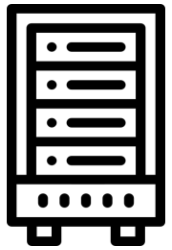
AIRLab´s servers



westworld – GPU Server



elysium – GPU Server



magrathea – GPU Server

multiverse – NAS

Students

PhDs and Postdocs

Computation only

Storage

Your account on the servers

- Same user/password on westworld, elysium (and magrathea)
- Home directory in each server:

/surname

/dataset

/shared -> link to multiverse

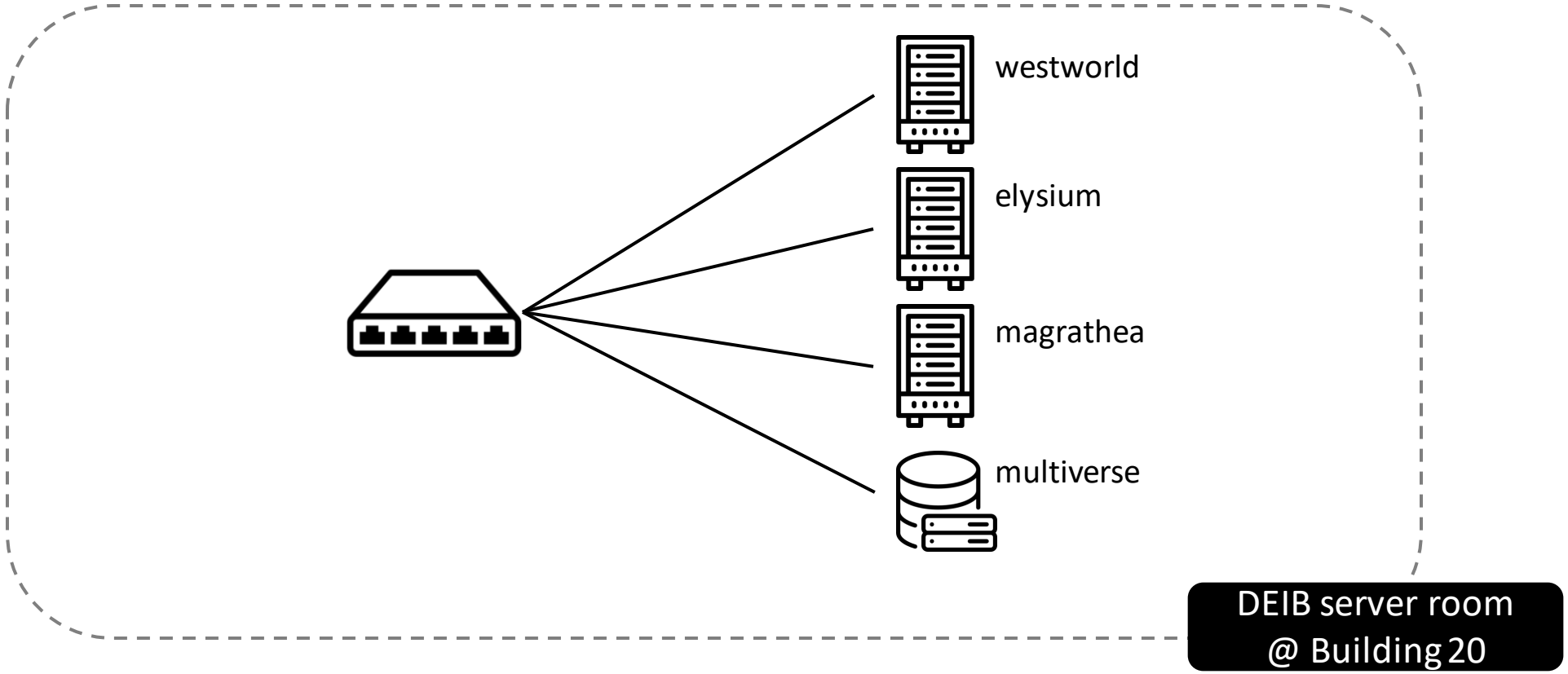
/private -> link to multiverse

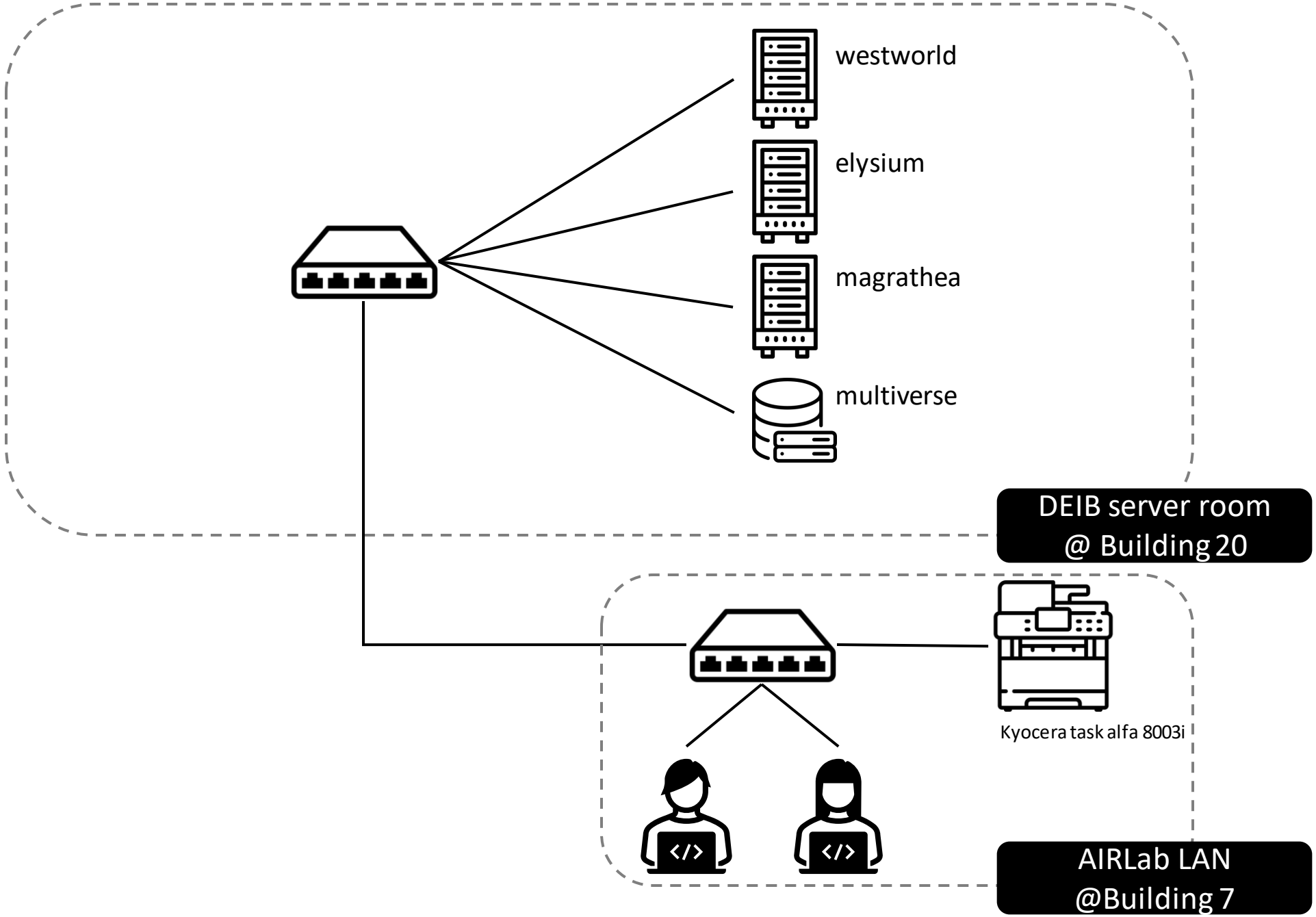
/storage -> link to multiverse

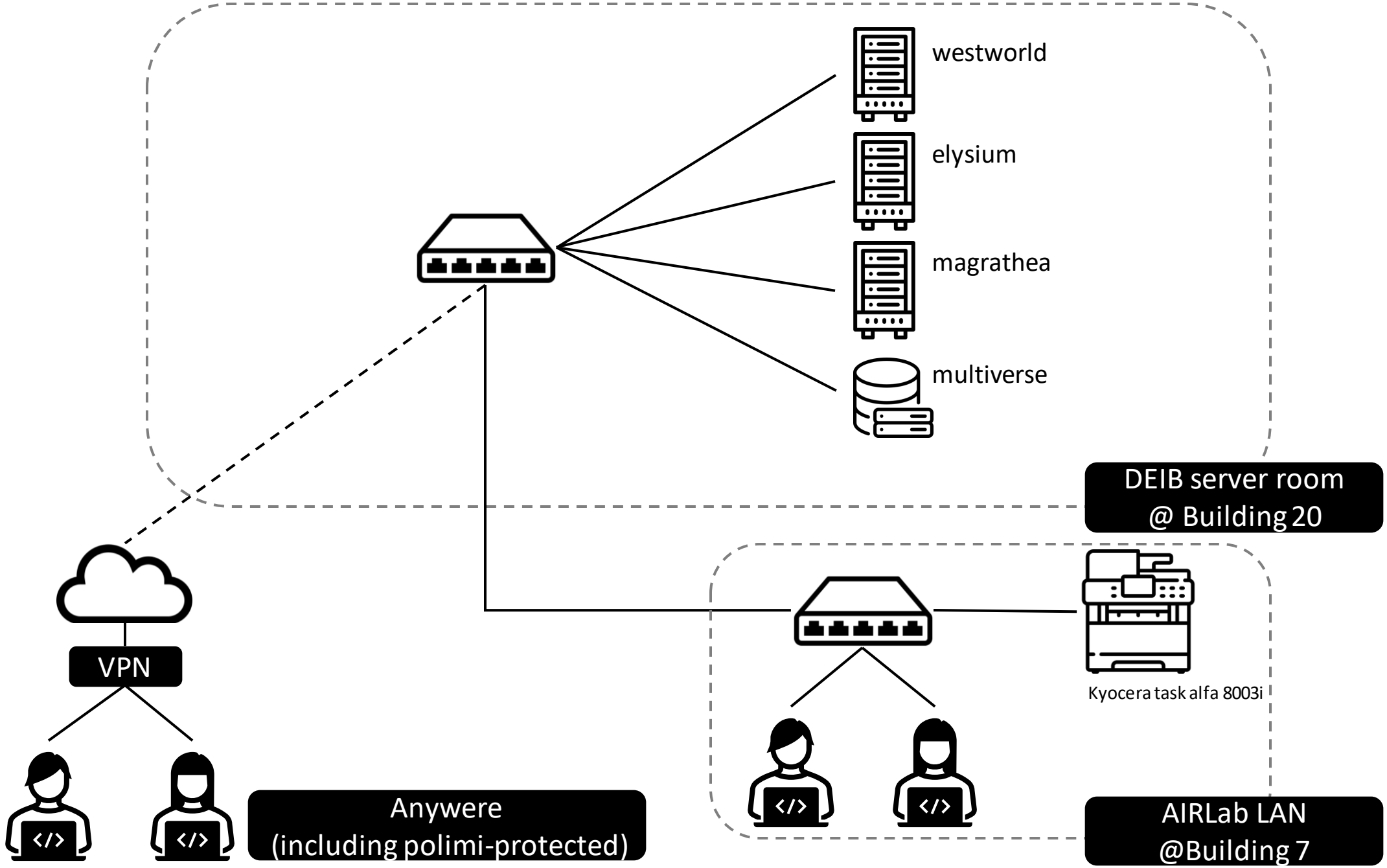
Your home folder is supposed to be empty besides the link you already find. Please put your code under /surname/storage and the datasets in /surname/dataset/private

What is on multiverse is visible from all servers

AIRLab´s network




westworld
elysium
magrathea
multiverse
DEIB server room
@ Building 20
Kyocera task alfa 8003i
AIRLab LAN
@Building 7


westworld
elysium
magrathea
multiverse
DEIB server room
@ Building 20
VPN
Kyocera task alfa 8003i
Anywere
(including polimi-protected)
AIRLab LAN
@Building 7

Use and share resources

Run experiments

- Share resources (GPU, CPU, ..)
- Virtualization
- Avoid dependencies conflicts
- Easier reproducibility

If you are not familiar with docker is advisable to check out some introductory tutorial on line

Book resources

Before using any GPU or CPU check that is not already used and book it for you

- Westworld resources: https://docs.google.com/spreadsheets/d/1n6HDbSX0Pe0zcRA0iBYDStrud7d_yRhBSrpFld4WNxs/edit#gid=1311829678

- Elysium resources: https://docs.google.com/spreadsheets/d/1wvzz0ZgPypepkZJtZahAmIJxV-iOIhsR7_s5Mht4I1A/edit#gid=0

- Magrathea resources: https://docs.google.com/spreadsheets/d/1zpmh4hSZp8u45HZ4p2aKrMqlBsSQHbYEthdIzRkKj2o/edit#gid=0

Tutorial

- Get ssh certificate for servers
- Configure ssh autocomplete
- Configure network drive on ubuntu
- Configure ssh certificate for github on server
- Tmux
- Nvidia-smi
- Introduction to docker
- “Tutorial” pytorch + jupyter + tensorboard

Tutorial

- The following tutorial assumes you are using Ubuntu as operative system and VS Code as editor
- It mainly cover how I have configurated my machine -> there might be better ways, please tell me about it

- You can use whatever OS/Editor you prefer

Configure VPN

If you can't connect to the AIRLab ethernet network, you can always connect to any of the servers using the VPN.

Instructions for OSX and Windows on POLIMI web site at
https://www.ict.help.polimi.it/network-vpn-global/

VPN installation on Ubuntu

```
sudo apt install network-manager-openconnect network-manager-openconnect-gnome -y
```

Extract certificates (use the unique password you created the certificate with):
(with newer openssl versions, from 3.0, append the "-legacy" option to the following commands)

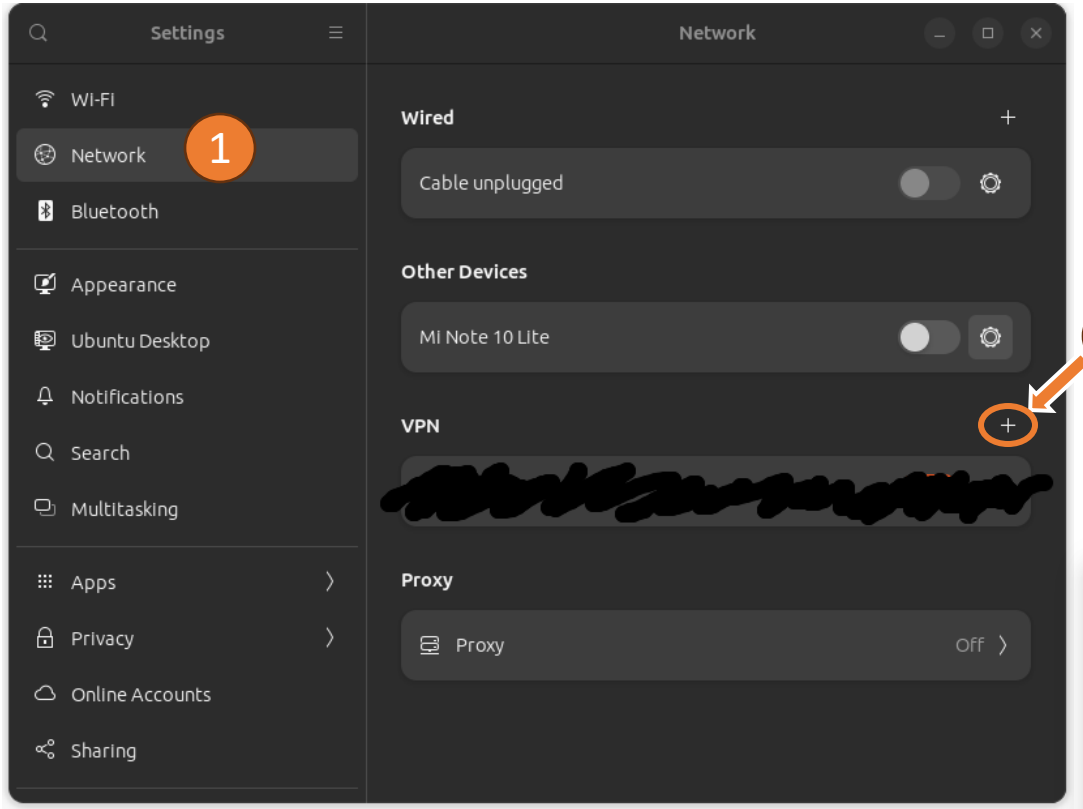
```
openssl pkcs12 -in YOUR_CERTIFICATE_FILE.p12 -out usercert.pem -clcerts -nokeys
openssl pkcs12 -in YOUR_CERTIFICATE_FILE.p12 -out userkey.pem -nocerts -nodes
openssl pkcs12 -in YOUR_CERTIFICATE_FILE.p12 -cacerts -nokeys -chain -out userca.pem
```

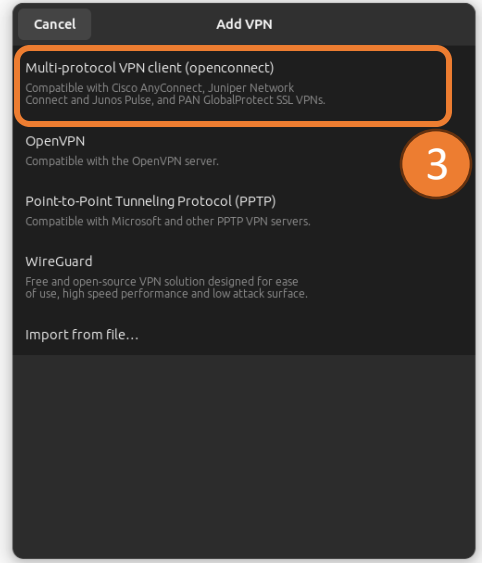
Configure in GNOME

Under settings:
1. Network
2. Add VPN
3. Multi-protocol VPN client (openconnect)

Fill with:
- VPN Protocol: Palo Alto Networks GlobalProtect
- Gateway: gp-deib.vpn.polimi.it
- CA Certificate: userca.pem
- User Certificate: usercert.pem
- Private Key: userkey.pem

Launch the vpn from the menu filling in your polimi login credentials (name.surname@(mail.)polimi.it and password)





Connect to the servers

... Or how I configured my ssh client

Names instead of IP

- Instead of typing each time the IP of the server you want to connect to -> **use DNS entry**
- Add the following lines to your `/etc/hosts` file:

```
# AirLab servers
<magrathea_ip>   magrathea    magra
<elysium_ip>        elysium        ely
<westworld_ip>    westworld    ww
<multiverse_ip>    multiverse    multi
```

Now you can also use "nicknames" for the server (eg, `ww` instead of `westworld`)

SSH certificate

- Each time you connect to a server you need to put your credentials
- Tedious, error prone, need to remember a password

-> use ssh certificate

https://linuxhint.com/generate-ssh-key-ubuntu/

Do it for all the server you can connect to

SSH config

- Typing `ssh surname@server` each time is tedious as the username is always the same!
- We can configure ssh to do it for us!
- Add to your `.ssh/config` the following lines:

```
Host ely
  HostName ely
  User YOUR_USERNAME_HERE
  Port 22

Host magra
  HostName magra
  User YOUR_USERNAME_HERE
  Port 22

Host ww
  HostName ww
  User YOUR_USERNAME_HERE
  Port 22
```

ssh config

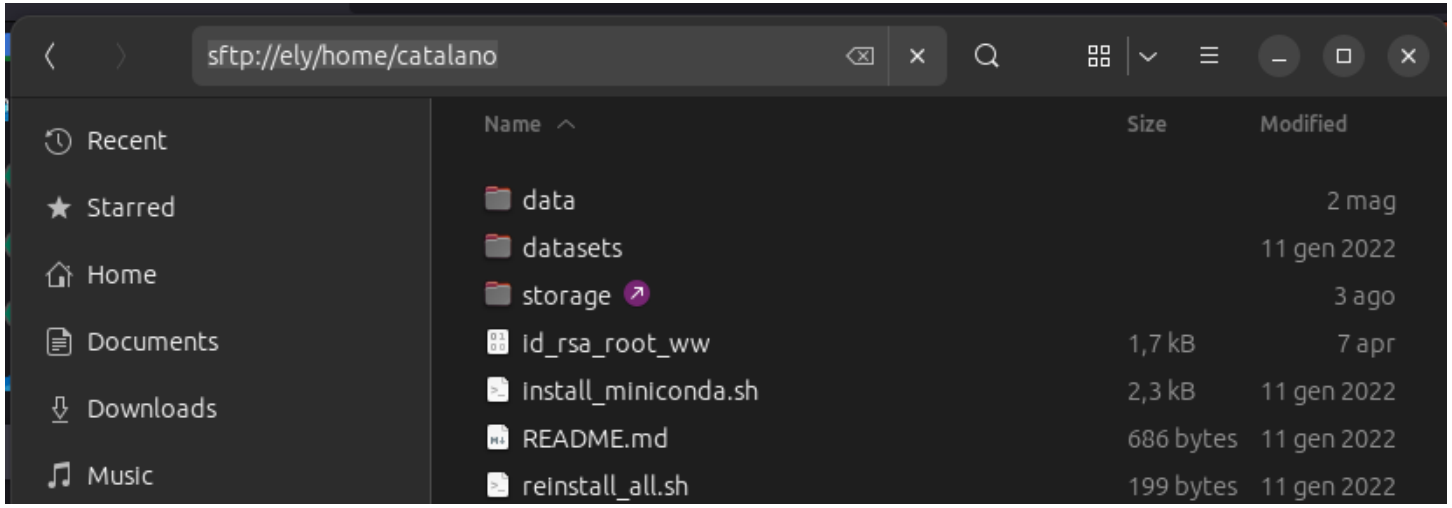
Finally now you can connect to any server with just:

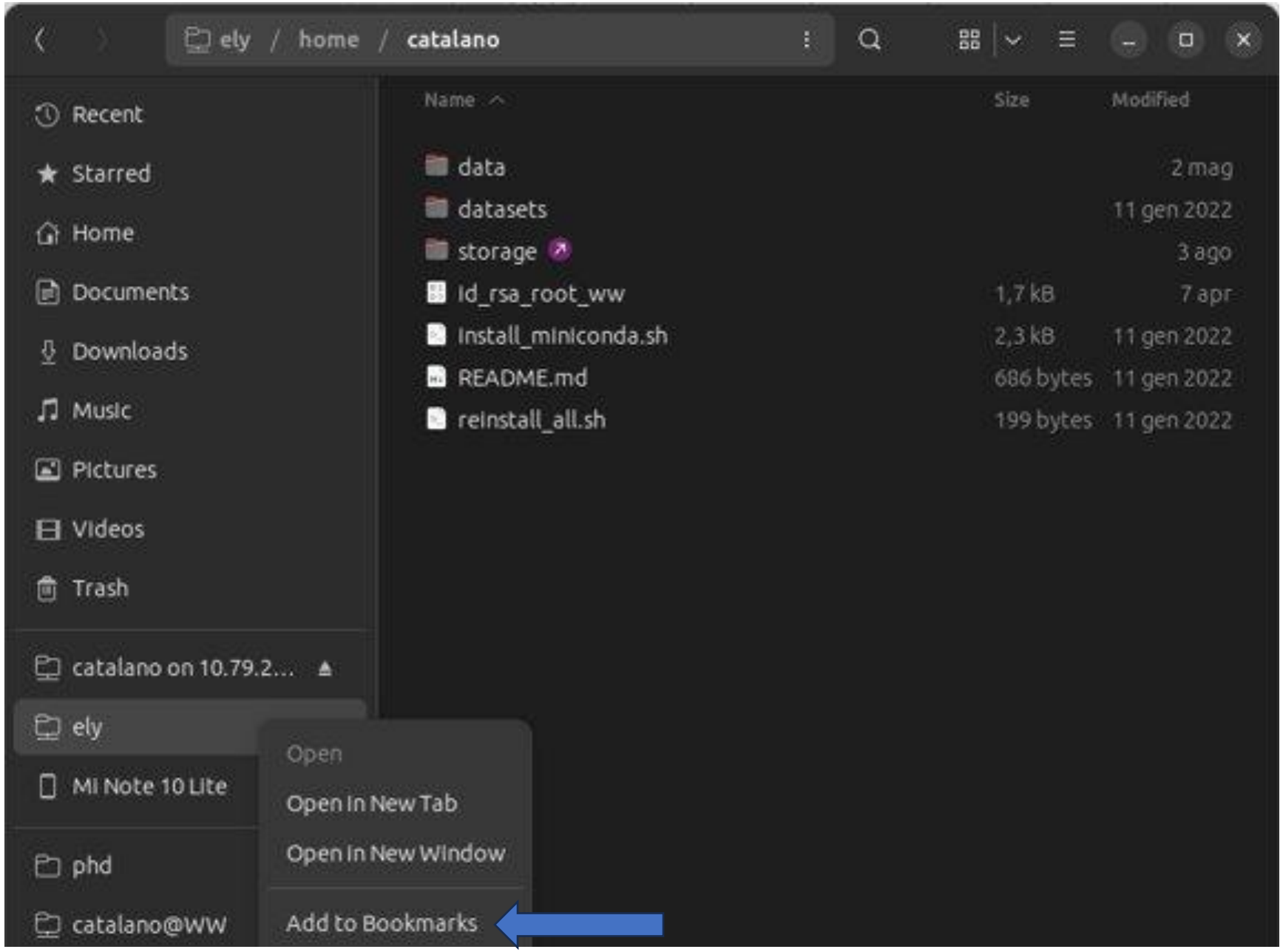
`ssh server_nickname`

Bonus: you should now also have tab-autocomplete for server_nickname

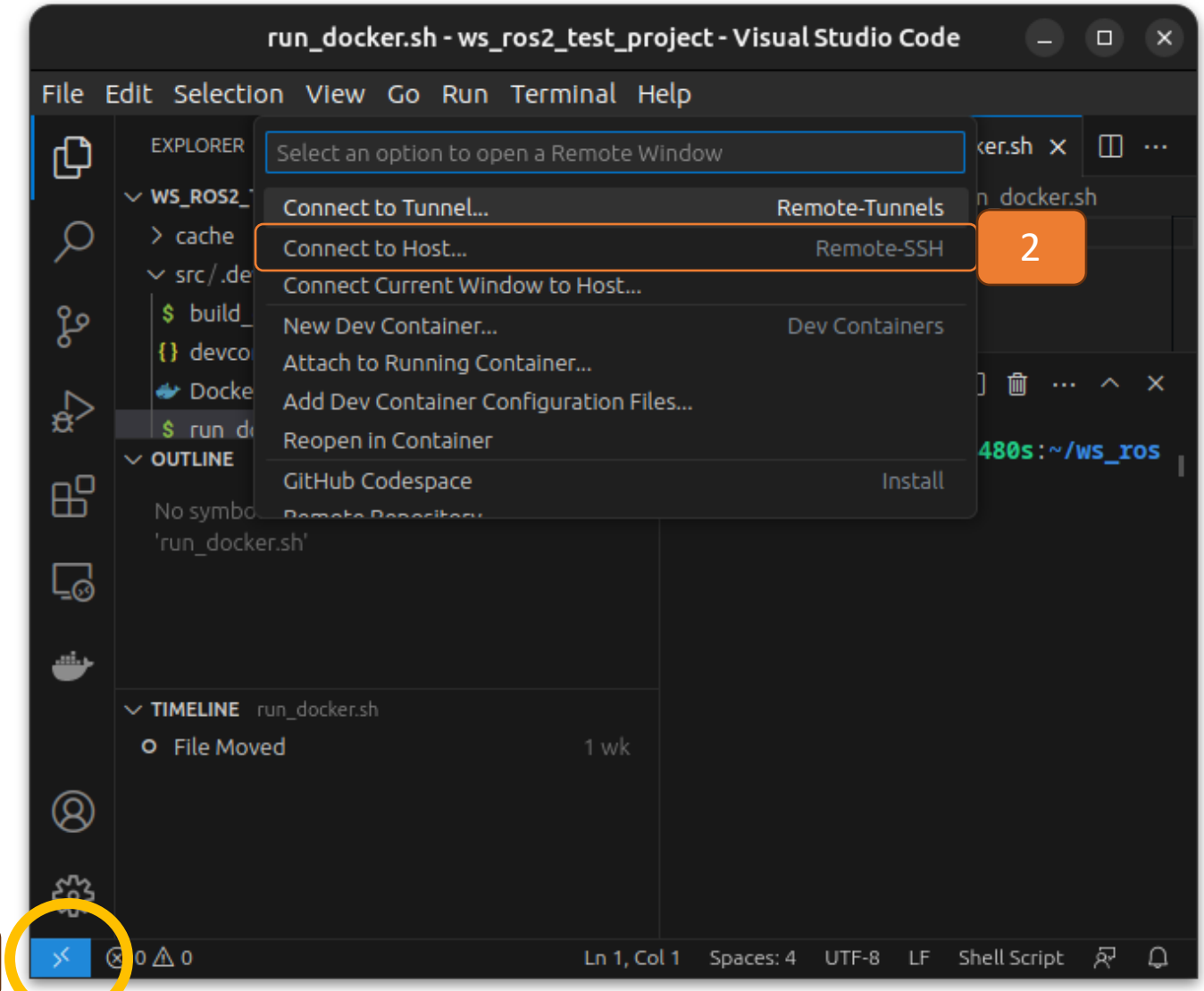
Move files from/to servers

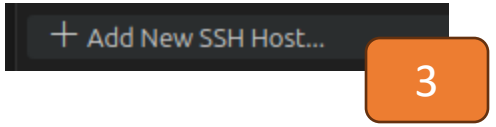
- You can use FTP, scp, and rsync
- But you can also map your home on the server in nautilus on your local machine
- Use sftp://server_name/home/username as address
- Save this location to bookmarks




ely / home / catalano
Recent
Starred
Home
Documents
Downloads
Music
Pictures
Videos
Trash
catalano on 10.79.2...
ely
Mi Note 10 Lite
phd
catalano@WW
Name
Size
Modified
data
2 mag
datasets
11 gen 2022
storage
3 ago
id_rsa_root_ww
1,7 kB
7 apr
install_miniconda.sh
2,3 kB
11 gen 2022
README.md
686 bytes
11 gen 2022
reinstall_all.sh
199 bytes
11 gen 2022
Open
Open In New Tab
Open In New Window
Add to Bookmarks

Configure VS Code Remote SSH





Have a github repo for your project!

- Sync your work on github repo
- Following these instruction on
  the sever you are developing will let you use git hub

https://docs.github.com/en/authentication/connecting-to-github-with-ssh/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent

https://docs.github.com/en/authentication/connecting-to-github-with-ssh/adding-a-new-ssh-key-to-your-github-account

tmux: keeping ssh session alive when disconnect from server

- Running a time intensive script and want to turn off your pc?
- Want to login on the same shell from different location?

use tmux!

- Cheat sheet here: https://tmuxcheatsheet.com/

Check how to create, attach, detach and delete a tmux session

tmux: things to know

- Scroll a tmux pane:

“Ctrl + b” followed by “[“ (then you can use the arrows)

- tmux copy and paste:

Enter ‘copy mode’ by pressing “CTRL + b”, [.

Use the arrow keys to go to the position from where you want to start copying. Press “CTRL + SPACE” to start copying.

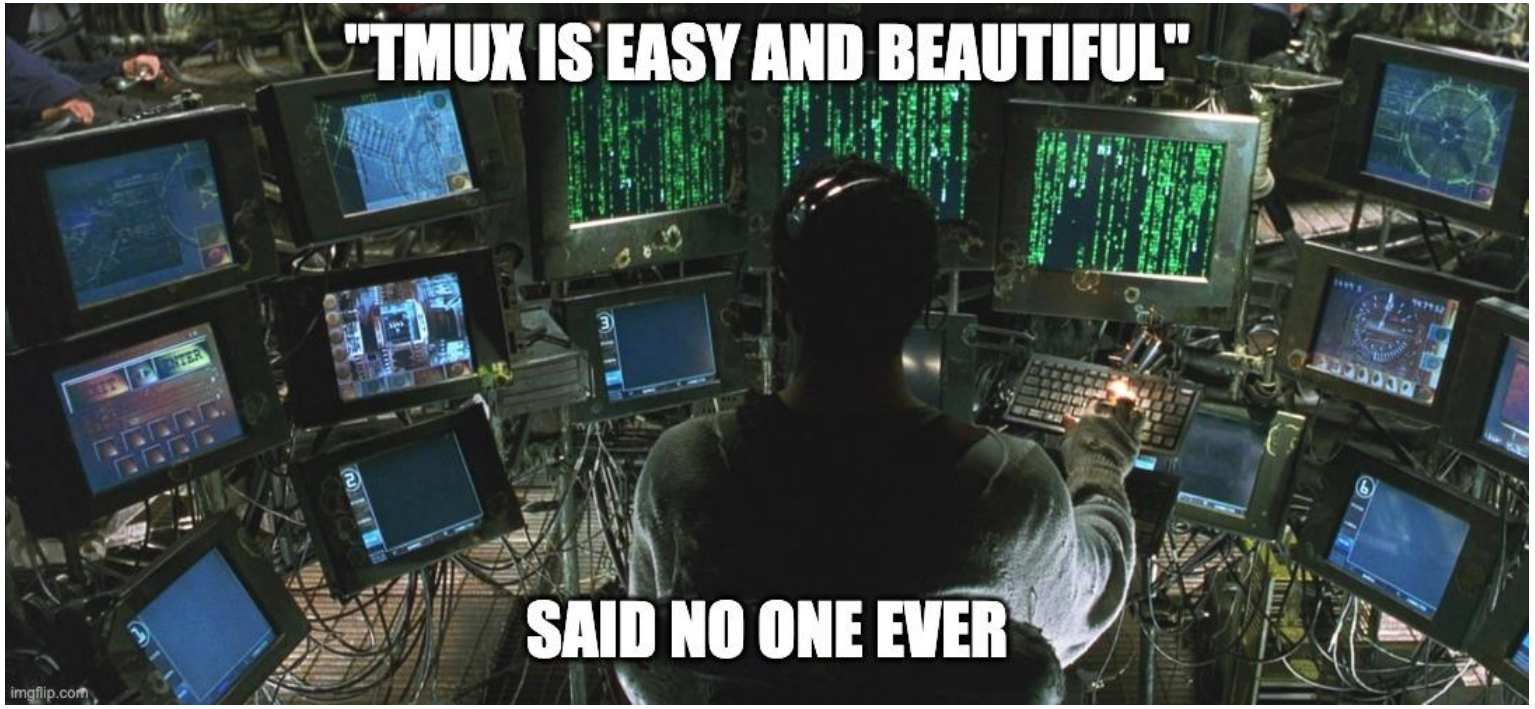
Use arrow keys to go to the end of text you want to copy. Press “ALT + w” or “CTRL+ w” to copy into the tmux buffer.

Press “CTRL +b”,] to paste in a possibly different Tmux pane/window.

tmux alternatives??

Do you know tmux alternatives?

Please let me know!!



Check GPU status

- Use command `nvidia-smi`

```
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 520.56.06    Driver Version: 520.56.06    CUDA Version: 11.8     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Quadro RTX 6000     On   | 00000000:04:00.0 Off |                  Off |
| 74%   56C    P2   197W / 180W |   9777MiB / 24576MiB |     40%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   1  Quadro RTX 6000     On   | 00000000:05:00.0  On |                  Off |
| 87%   60C    P2    99W / 180W |  17301MiB / 24576MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   2  NVIDIA TITAN X ...  On   | 00000000:83:00.0 Off |                  N/A |
| 91%   66C    P2   179W / 180W |  10051MiB / 12288MiB |    100%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   3  NVIDIA GeForce ...  On   | 00000000:84:00.0 Off |                  N/A |
| 32%   37C    P8     9W / 180W |     13MiB / 11264MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   4  Quadro RTX 6000     On   | 00000000:87:00.0 Off |                  Off |
| 78%   57C    P2   151W / 180W |   6655MiB / 24576MiB |     74%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   5  Quadro RTX 6000     On   | 00000000:88:00.0 Off |                  Off |
| 36%   40C    P8    14W / 180W |  17301MiB / 24576MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|    0   N/A  N/A      2558      G   /usr/lib/xorg/Xorg                  8MiB |
|    0   N/A  N/A     14390      C   /usr/bin/python3                 3122MiB |
|    0   N/A  N/A   2533867      C   python                           6642MiB |
|    1   N/A  N/A      2559      G   /usr/lib/xorg/Xorg                  8MiB |
|    1   N/A  N/A    388610      C   /usr/bin/python3                17288MiB |
|    2   N/A  N/A      2560      G   /usr/lib/xorg/Xorg                  8MiB |
|    2   N/A  N/A   3829190      C   python                           4314MiB |
|    2   N/A  N/A   3829728      C   python                           4080MiB |
|    2   N/A  N/A   3840660      C   python                           1644MiB |
|    3   N/A  N/A      2561      G   /usr/lib/xorg/Xorg                  8MiB |
|    4   N/A  N/A      2562      G   /usr/lib/xorg/Xorg                  8MiB |
|    4   N/A  N/A   2523223      C   python                           6642MiB |
```

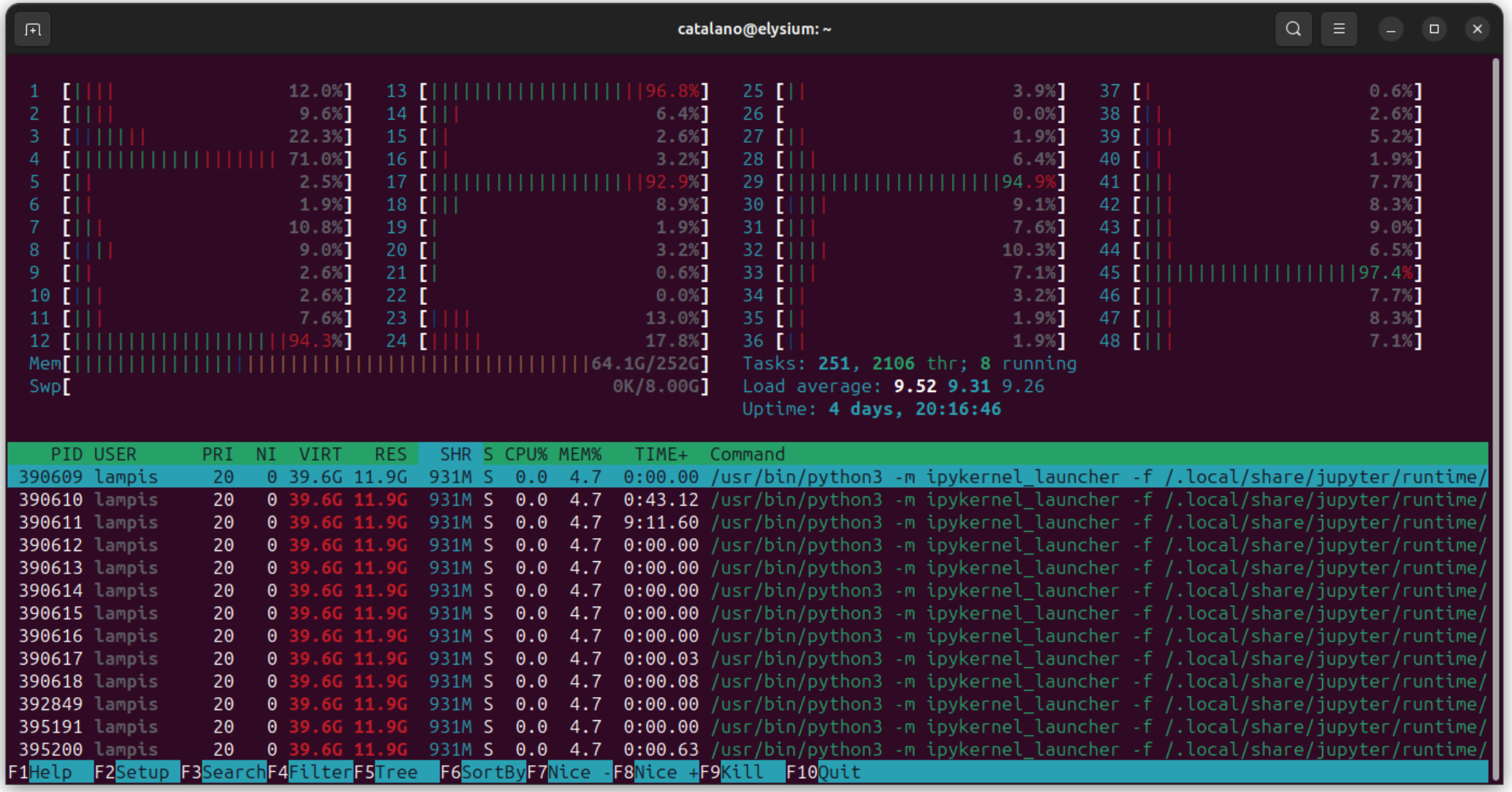
Get information about a process

- Use `ps aux | grep PROCESS_PID`

- Useful to get the process' user

"task manager" for the command line

- Use `htop`

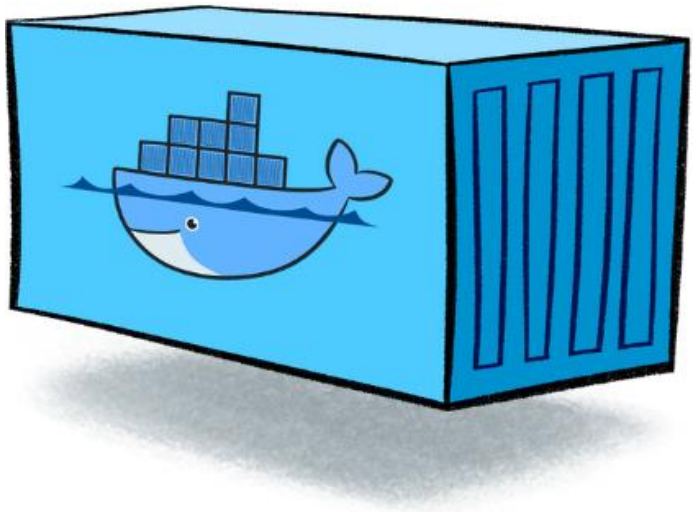


Developing pipeline - hig level picture

Docker Image

The "virtual environment"
- what library to make available to your script
- what dependences to install
- ..

RUN



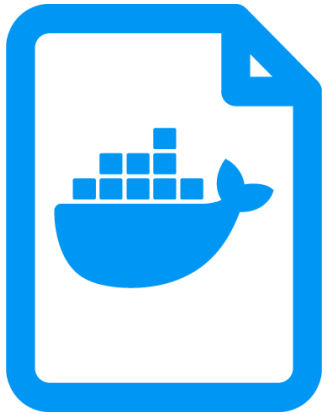
Docker Container

The image actually running. **It is immutable**

Assign resources (GPUs, CPUs)

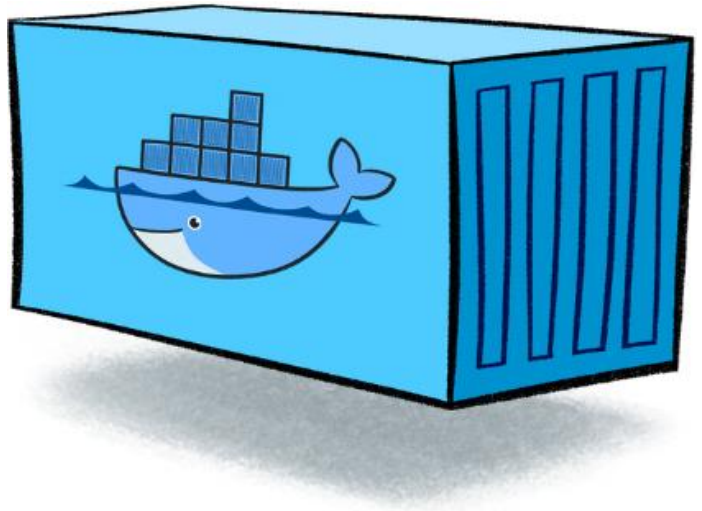
Think of it as a "*virtual machine*"

Developing pipeline - hig level picture



BUILD

RUN



Docker File

Docker Image description
text file

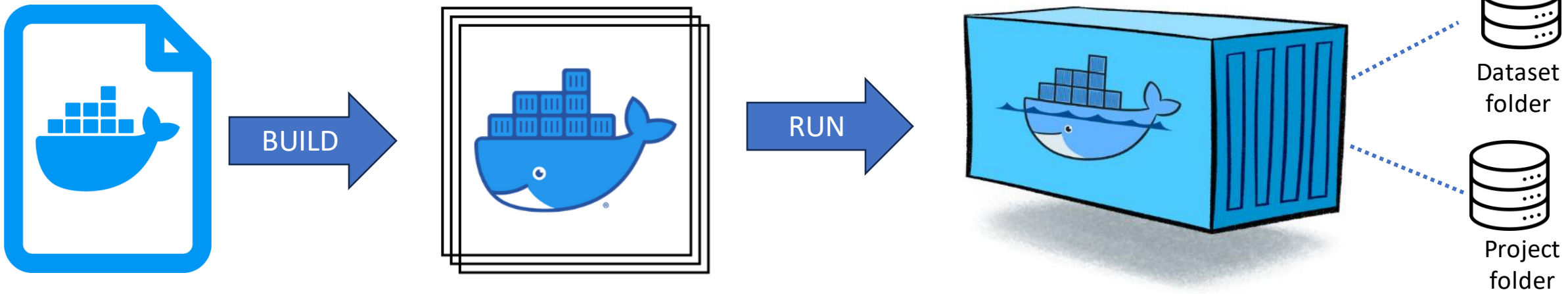
Docker Image

Binary file

Docker Container

Running **process**
(or processes)

Developing pipeline - hig level picture



Docker File

Docker Image

The "virtual enviorment"
- what library to make available to your script
- what dependeces to install
- ..

Docker Container

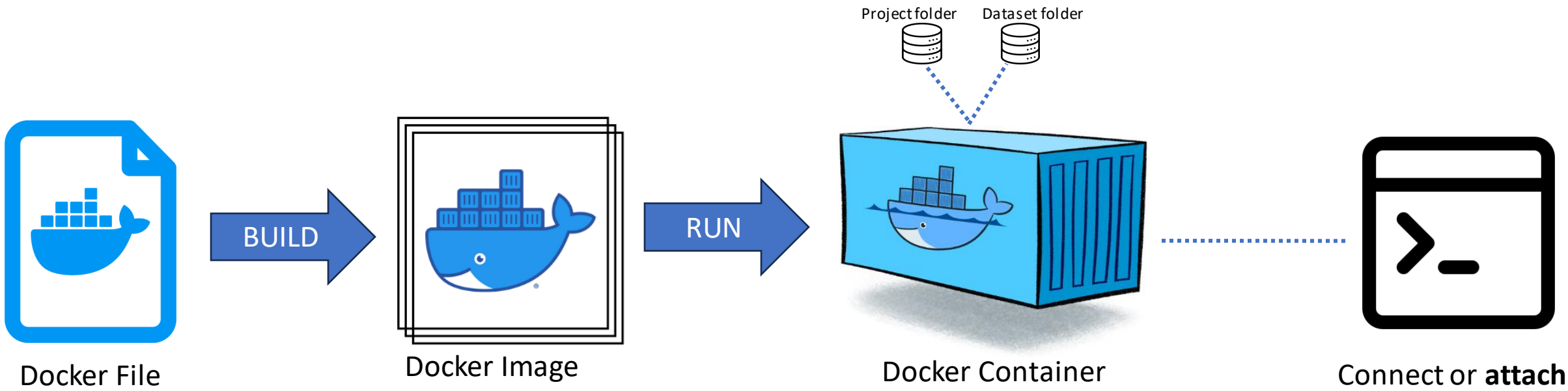
The image actually running, its filesystem is immutable

Assign resources (GPUs, CPUs)

Think of it as a "*virtual machine*"

Attach external drive (read and **write**)

Developing pipeline - hig level picture



Docker File

Docker Image

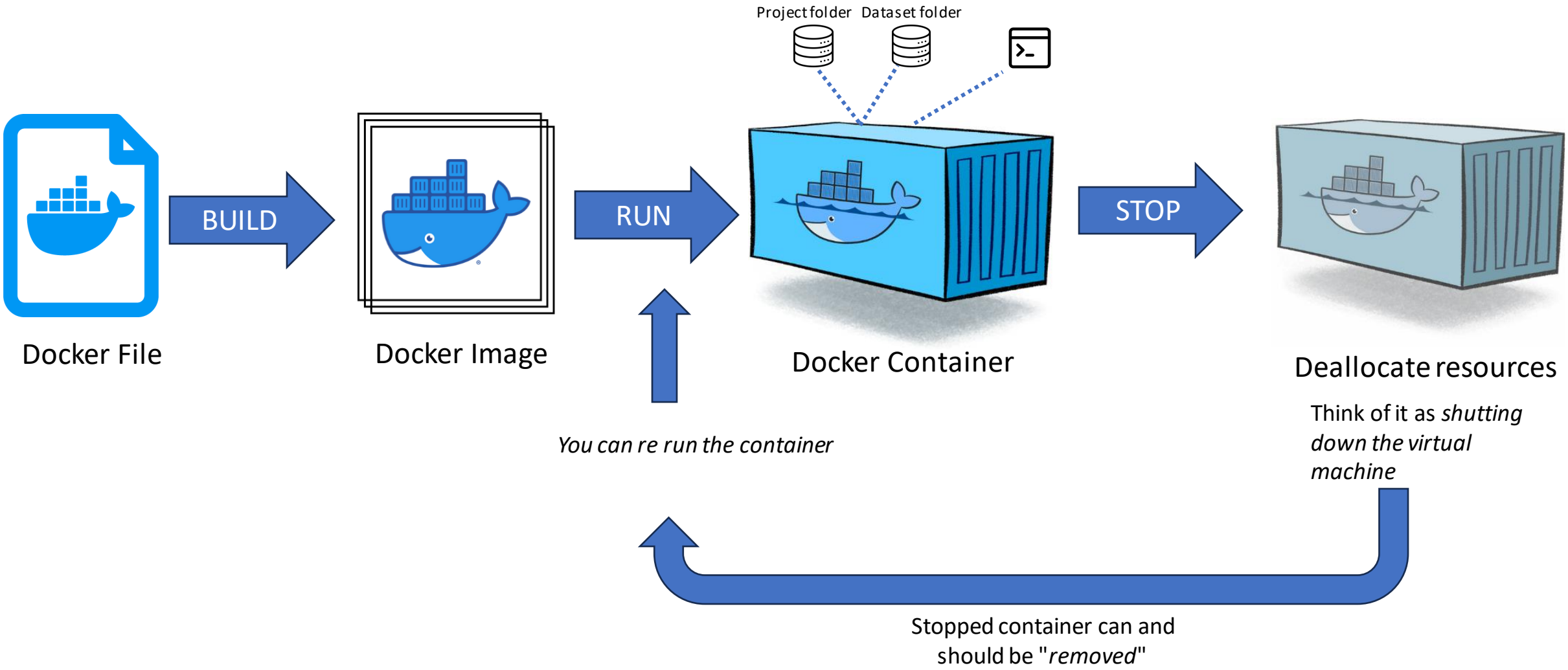
Docker Container

Connect or **attach** to a running container

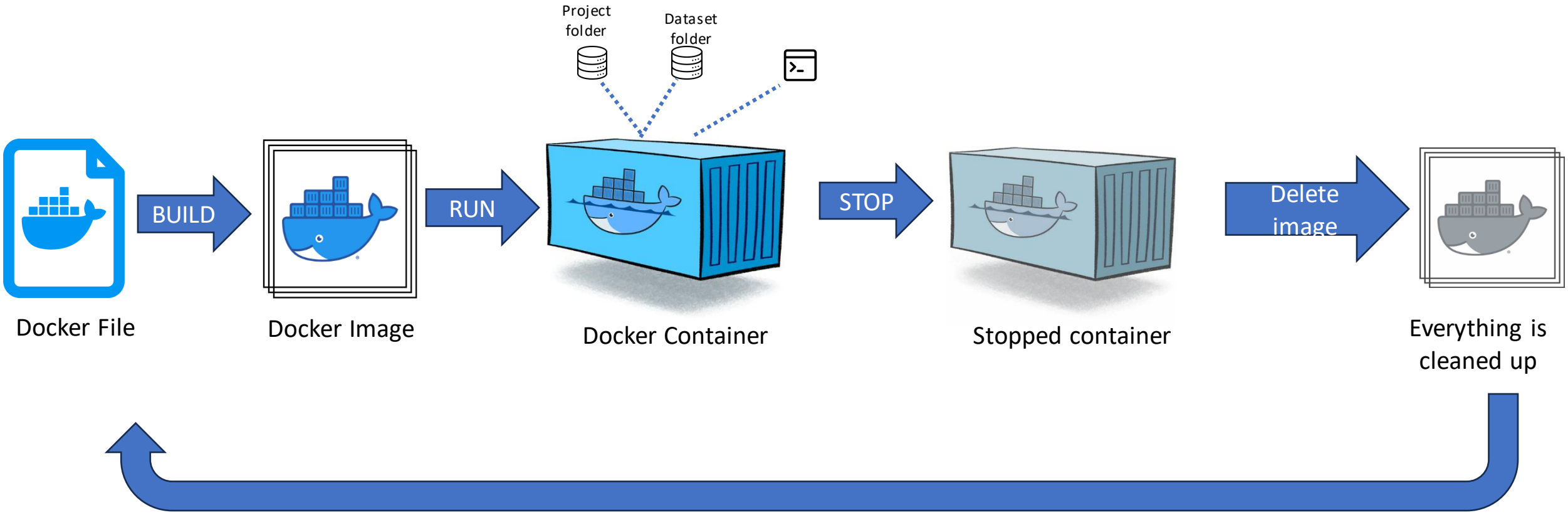
From here you can execute command as you are in a virtual machine separate from the hosting machine.

Developing/running/debugging ecc..

Developing pipeline - hig level picture



Developing pipeline - hig level picture



Docker useful commands

Build a new docker image:
docker build --rm -t your_surname/image_name

Visualize docker images:
docker images
docker images | grep your_surname

Run contanier
docker run
with some sugar
run-docker

Visulize running containers:
docker ps
docker ps | grep your_surname

Atach to a running contatiner:
docker exec -it CONTAINER_ID bash

Stop container:
docker stop CONTAINER_ID

Remove contanier:
docker rm CONTAINER_ID

Remove a docker image:
docker rmi your_surname/image_name

Example of docker run

- You need to specify your UID and GID
  - Use command `id surname` to get it
- You need to specify which image to use
- You need to specify the resulting docker container name

Example of docker run

- Set UID and GID as 1085
- Set container name surname_docker_example
- Image to use surname/example_image:v0

```
docker run -e HOST_UID=1085 -e HOST_GID=1085 --name surname_docker_example surname/example_image:v0
```

Example of docker run

- Use GPU 7
- Use CPU from 104 to 111

```
docker run --gpus device=7 --cpuset-cpus 104-111 -e HOST_UID=1085 -e HOST_GID=1085 -u 1085:20002 --name surname_docker_example surname/example_image:v0
```

Example of docker run

- Set the working directory as /home/surname/exp

```
docker run --gpus device=7 --cpuset-cpus 104-111 -w /home/catalano/exp -e HOST_UID=1085 -e HOST_GID=1085 -u 1085:20002 --name surname_docker_example surname/example_image:v0
```

Example of docker run

- Mount the dataset folder
- Mount the project folder

```
docker run --gpus device=7 --cpuset-cpus 104-111 --mount type=bind,source=/home/surname/storage/project,target=/home/surname/exp --mount type=bind,source=/multiverse/datasets/surname/,target=/home/surname/exp/private_datasets -w /home/catalano/exp -e HOST_UID=1085 -e HOST_GID=1085 -u 1085:20002 --name surname_docker_example surname/example_image:v0
```

Example of docker run

- Use parameter -d to let the container run in the background
- Use parameter -- rm to automatically remove the container when it exits
- Use parameter -it to have a console

```
docker run -d --rm -it --gpus device=7 --cpuset-cpus 104-111 --mount
type=bind,source=/home/surname/storage/project,target=/home/surname/exp --mount
type=bind,source=/multiverse/datasets/surname/,target=/home/surname/exp/private_datasets -w /home/catalano/exp -e
HOST_UID=1085 -e HOST_GID=1085 -u 1085:20002 --name surname_docker_example surname/example_image:v0
```

Other parameters

- You might also want to :
  - set a memory limit
  - Forword some TCP port outside the container
  - ..
- Google/ChatGPT it

The lazy way

- Write a docker file starting from an existing one
- Have a build script
- Run the container via a bash script
- Attach your IDE to the running container
  - In VS Code you can:
    - Forward ports from a running container
    - Run Jupiter notebooks in the attached container w/o using the browser
- When you finish, stop the container, remove it, remove the image.

- Check your IDE plugins, there might be something useful for handling docker

Toy example

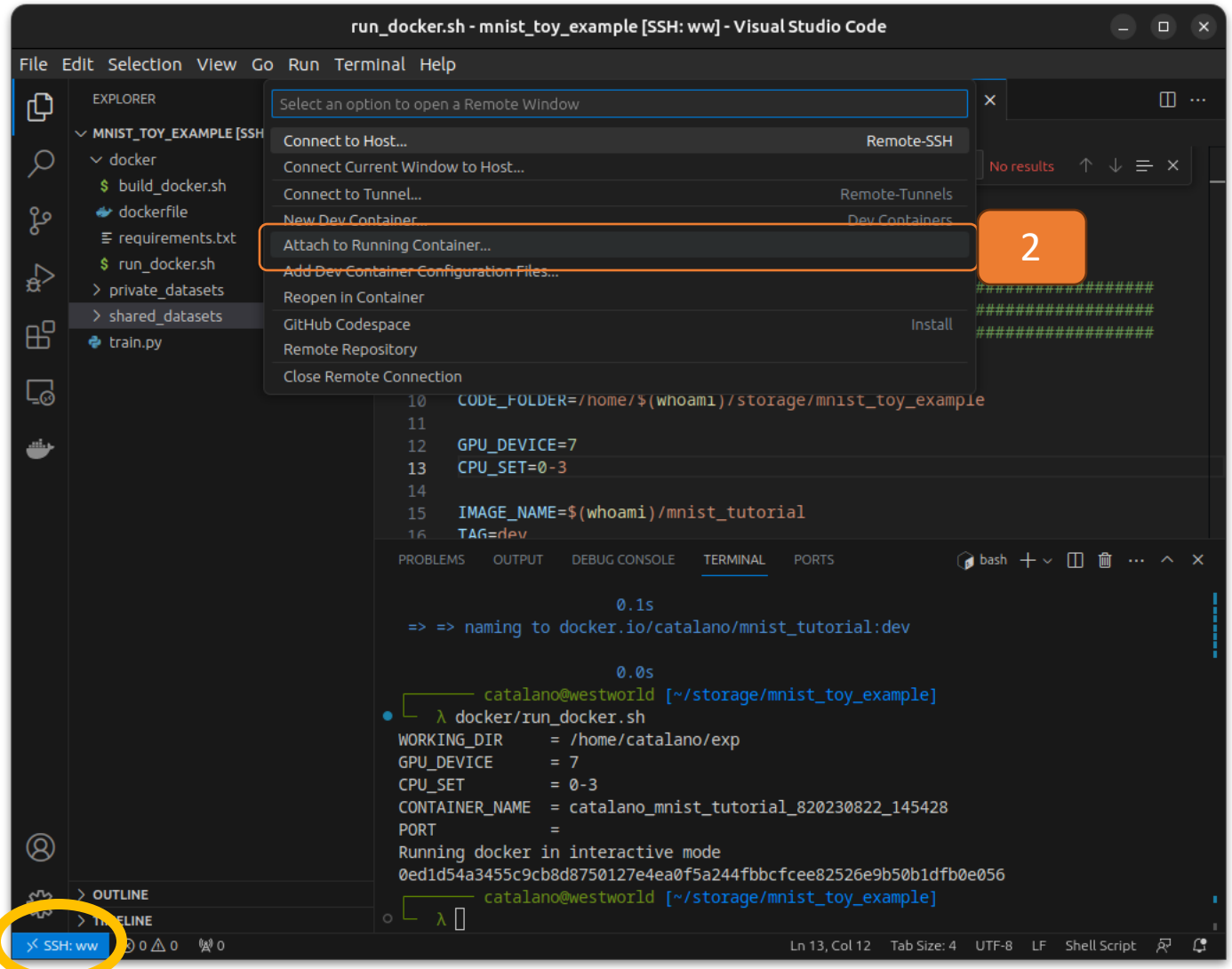
- MNIST classification example:

Repo on github

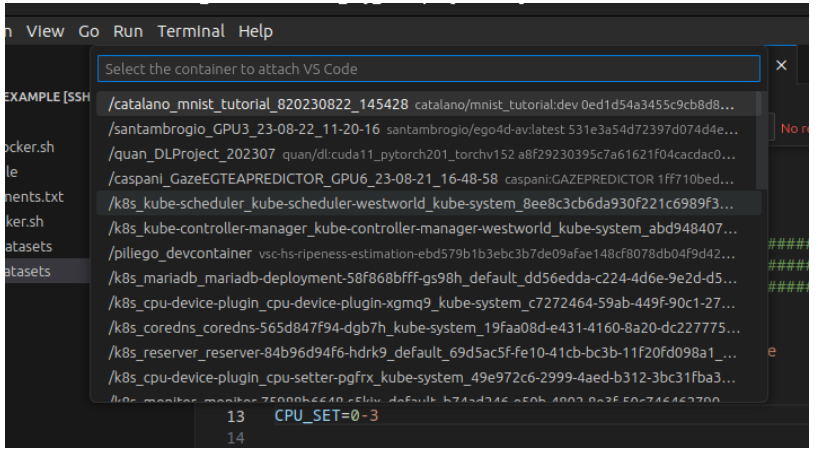
https://github.com/AIRLab-POLIMI/MNIST_Toy_Example

- We create the environment with the docker file, and install the python libs in requirements.txt
- Build the image with docker/build_docker.sh (it takes some time)
- Book the resources needed
- Run the container with
- Attach VS Code to the running container

Attach VS Code to an already running container



3 Select your container from the list



- Your project will be in /home/surname/exp
- CTRL + j to open a terminal
- **Now you can play**

Tensorboard

- Store and compare logs from many experiments
- Can visulize many kind of data:
  - Scalars:
    - Loss value, accuracy ..
  - Images
  - Model architecture
  - Embeddings

Tensorboard

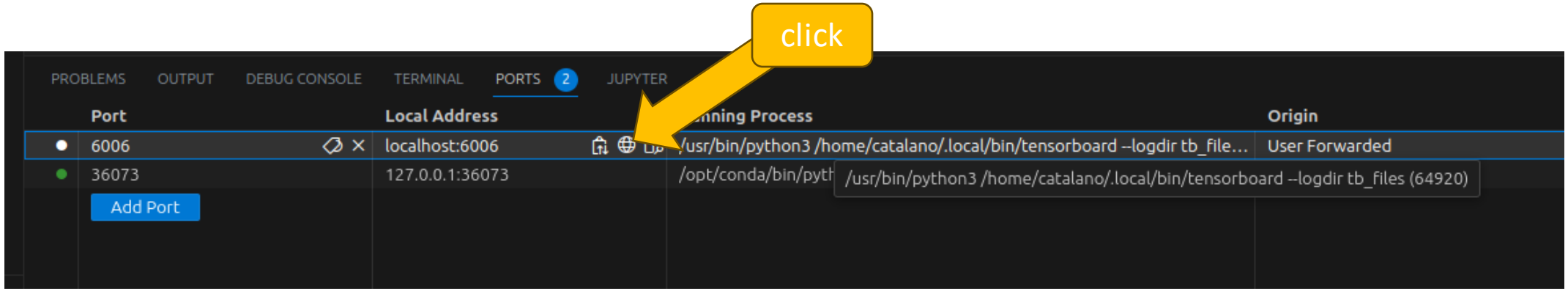
1. Create a folder for all your experiments
2. For each experiment create a "writer" with a meaningful name
3. Add values to the writer
4. To visualize the data run tensorbord with command:

tensorboard --logdir TB_DIRECTORY

By default it will listen on port 6006

Port forwarding

- When you run **tensorboard** VSCode should see that and auto forward its port outside.
- If not, or if you want to forward something else you can force the port forwarding
- ALT + j to open the bottom panel, go to the PORTS tab
- "Add Port"

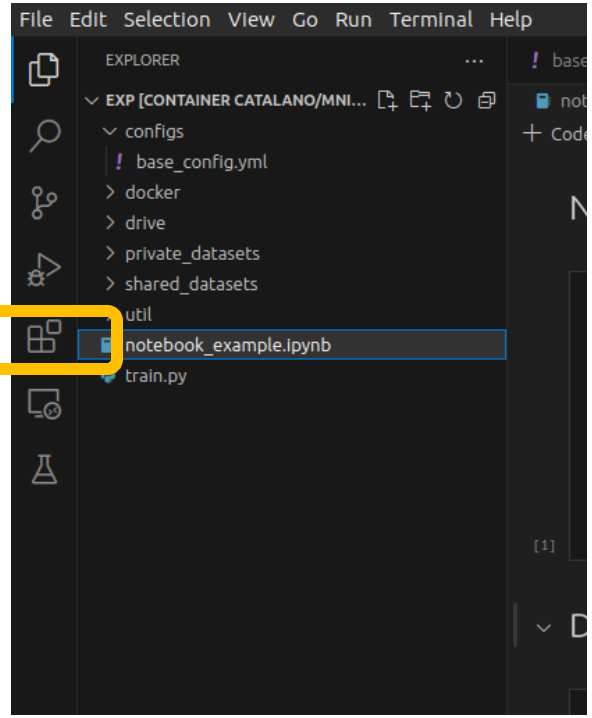


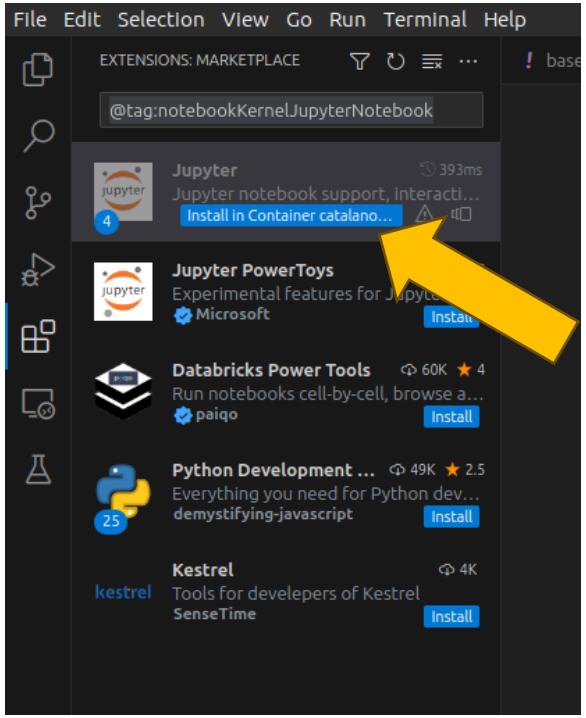
Tensorboard - Projector

- Let you project (PCA, UMAP,t-SNE) high dimensional object to a 2D or 3D space
- -> allow to have an idea on how "far apart" objects are in the embedding space
- For some scenarion can be extremly insigthfull (eg let you see outliers)
- Not well documented, but for each embedding you can attach some metadata (images, text, numbers, boolean) to do filters.
- Check example

Jupyter notebook

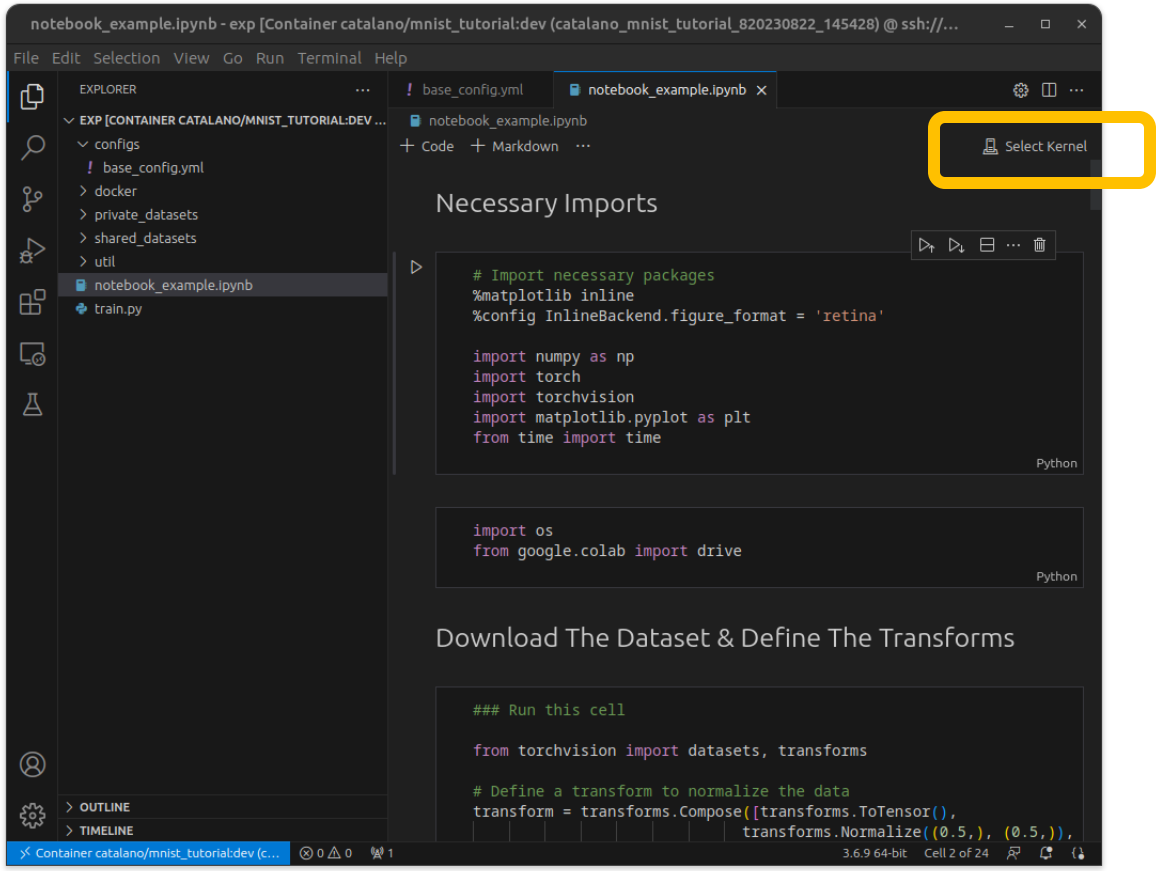
- Install the extension @tag:notebookKernelJupyterNotebook





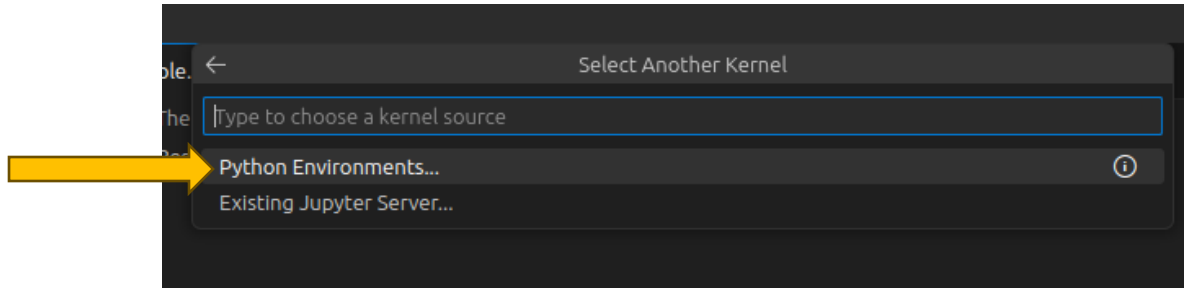
Jupyter notebook

- Install the extension @tag:notebookKernelJupyterNotebook
- Create/Open a ipynb file
- Select a kernel



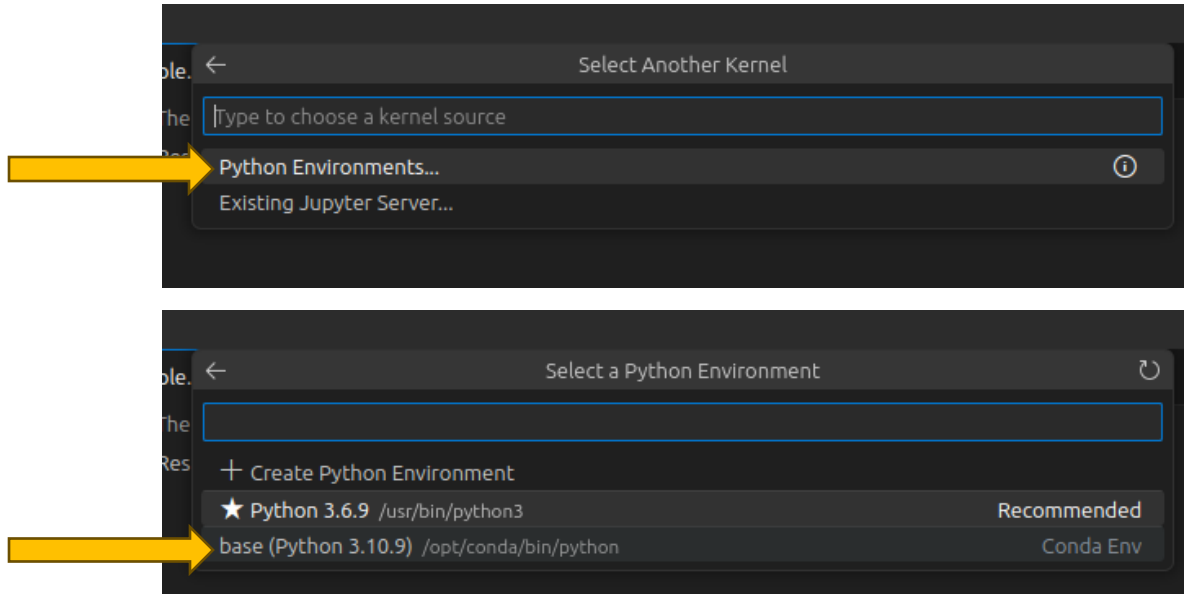
Jupyter notebook

- Install the extension @tag:notebookKernelJupyterNotebook
- Create/Open a ipynb file
- Select a kernel
- "python enviorment"



Jupyter notebook

- Install the extension @tag:notebookKernelJupyterNotebook
- Create/Open a ipynb file
- Select a kernel
- "python enviorment"
- "base"



Endorsed practice

- Use configuration files and keep the same source code to implement small variations in your experiments (use flags and values from a config file)
- Use tensorboard to log values
  - Losses
  - Accuracy
  - Ecc ..
- Use a log file instead of printing to the console

- Most importantly

Always code as if the guy who ends up maintaining your code will be a violent psychopath who knows where you live.