# JUPYTERLAB - SUPERCOMPUTING IN YOUR BROWSER

## Training course "Introduction to the usage and programming of supercomputer resources in Jülich"

2023-11-21 I JENS H. GÖBBERT (J.GOEBBERT@FZ-JUELICH.DE)
TIM KREUZER (T.KREUZER@FZ-JUELICH.DE)
ALICE GROSCH (A.GROSCH@FZ-JUELICH.DE

# MOTIVATION

## your thinking, your reasoning, your insides, your ideas

“It is all about using and building a machinery **interface** **between** computational researchers and data, supercomputers, laptops, cloud **and** your thinking, your reasoning, your insides, your ideas about a problem.”

Fernando Perez, Berkely Institute for Data Science
Founder of Project Jupyter

https://jupyter.org

JÜLICH
Forschungszentrum

# JUPYTER NOTEBOOK

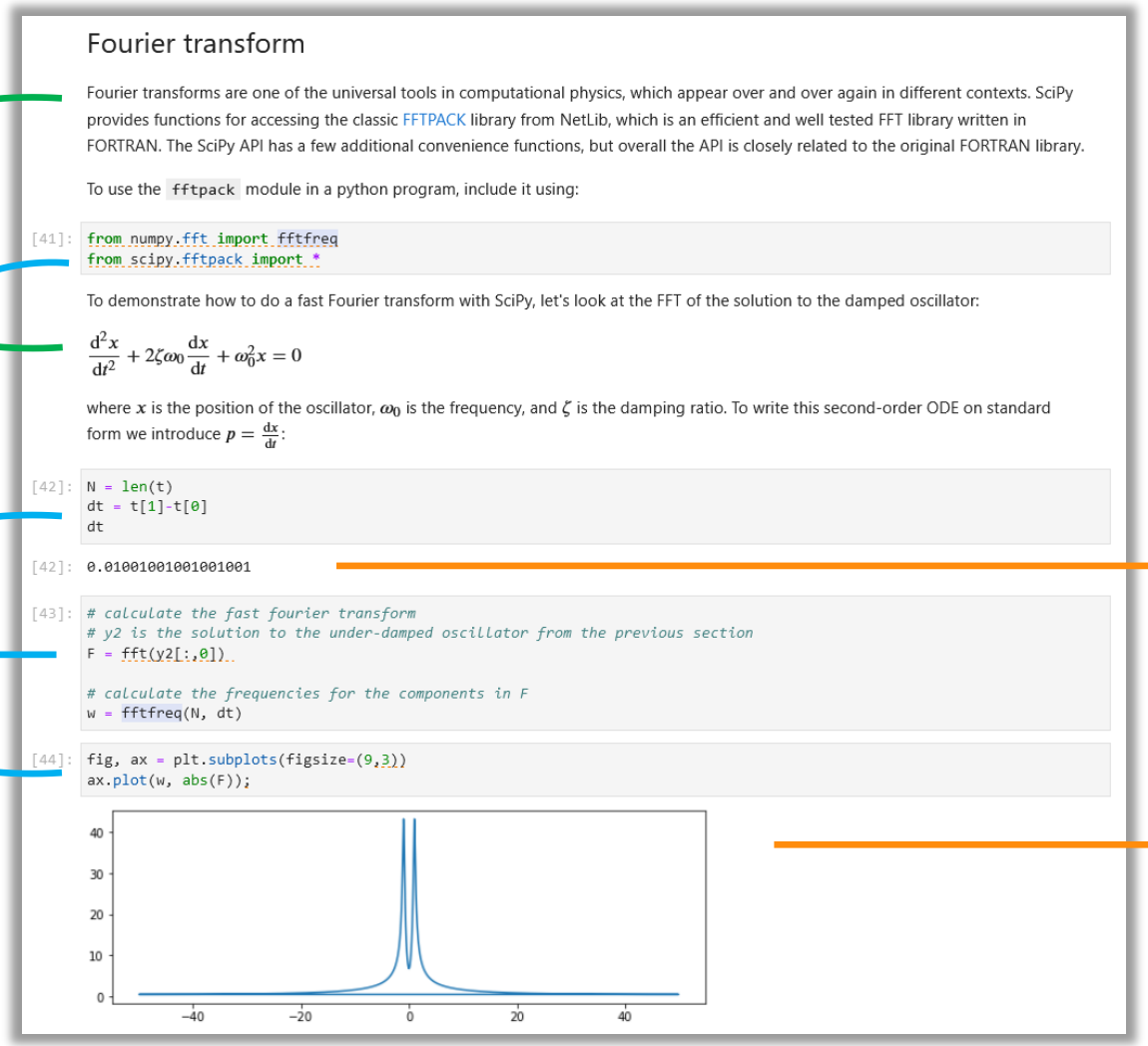
## creating reproducible computational narratives

Markdown Cells

Code Cells

Output

Output



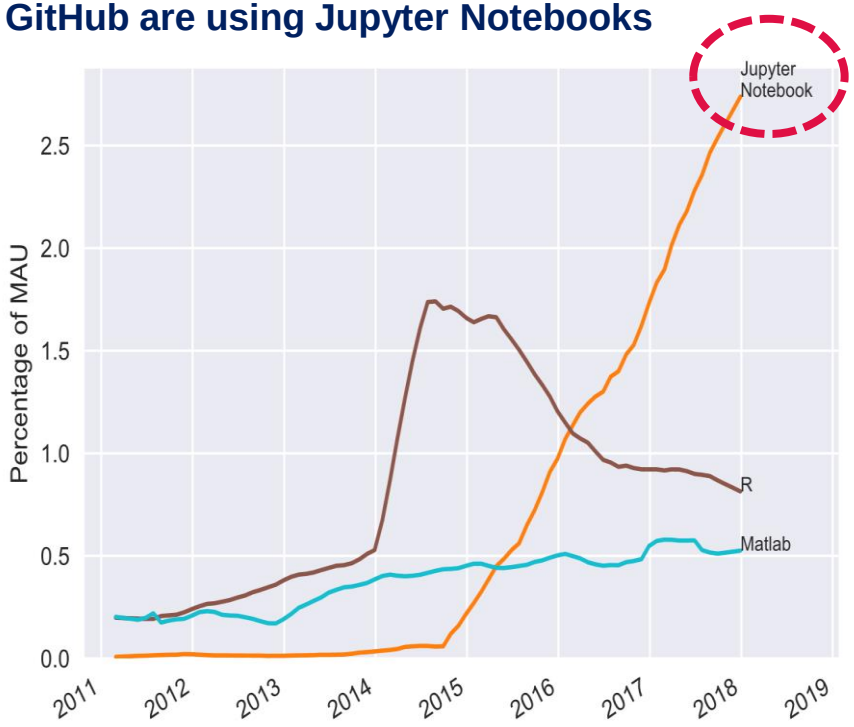

JÜLICH Forschungszentrum

# MOTIVATION

## Rise of Jupyter´s popularity

- In 2007, Fernando Pérez and Brian Granger announced
  „**Ipython**: a system for interactive scientific computing“ [1]
- In 2014, Fernando Pérez announced
  a spin-off project from IPython called **Project Jupyter**.
  - IPython continued to exist as a Python shell and a kernel for Jupyter, while the Jupyter notebook moved under the Jupyter name.
- In 2015, GitHub and the Jupyter Project announced
  native rendering of Jupyter notebooks file format (.ipynb files) on the **GitHub**
- In 2017, the **first JupyterCon** was organized by O'Reilly in New York City.
  Fernando Pérez opened the conference with an inspiring talk. [2]
- In 2018, **JupyterLab** was announced
  as the next-generation web-based interface for Project Jupyter.
- In 2019, JupyterLab 1.0 …
  In 2020, JupyterLab 2.0 …
  In 2021, JupyterLab 3.0 …
  In 2023, JupyterLab 4.0 expected in March 2023.

**Counting how many Monthly Active Users (MAU) on GitHub are using Jupyter Notebooks**




https://www.benfrederickson.com/ranking-programming-languages-by-github-users/
https://github.com/benfred/github-analysis


[1] Pérez F, Granger BE (2007) Ipython: a system for interactive scientific computing. Comput Sci Eng 9(3):21–29
[2] Pérez F, Project Jupyter: From interactive Python to open science -> https://www.youtube.com/watch?v=xuNj5paMuow




JÜLICH Forschungszentrum

# HISTORY OF JUPYTERLAB AT JSC


## Initial Basis

JupyterLab modules
**Authentication** via Unity/IdM
**Authorization** via UNICORE
Orchestration Docker Swarm
Synchronization of User-DBs
Basic Data Protection Regulation
Fulfill Safety Requirements

## Usage

Inplace Dokumentation
R, Julia, C++, Octave, Ruby
JupyterLabs on OpenStack
Dashboard Development
**JupyterLab Usability**
Kernel for Vis, DL
Testing & Benchmarking

## Features

**Remote Desktop** Integration
Optional 2-Factor Auth.
**Use for Workshops**
Specialized Functionalities
Enhanced Data Access
Extended Logging
Cross-Side Demonstration

## Redesign

Switch to **Kubernetes**
Redesign Management
Switch to **JupyterLab 3**
GPFS through UFTP
Support for User Extensions
Easybuild Modularization

## Customization

**Project/Community JHubs**
Upgrade JHub Entrance-UI
Comp. Resource Permissions
Maintenance Improvements
Upgrade of Load Balancer
**Modularization of Backend**
**External Clouds & HPC**



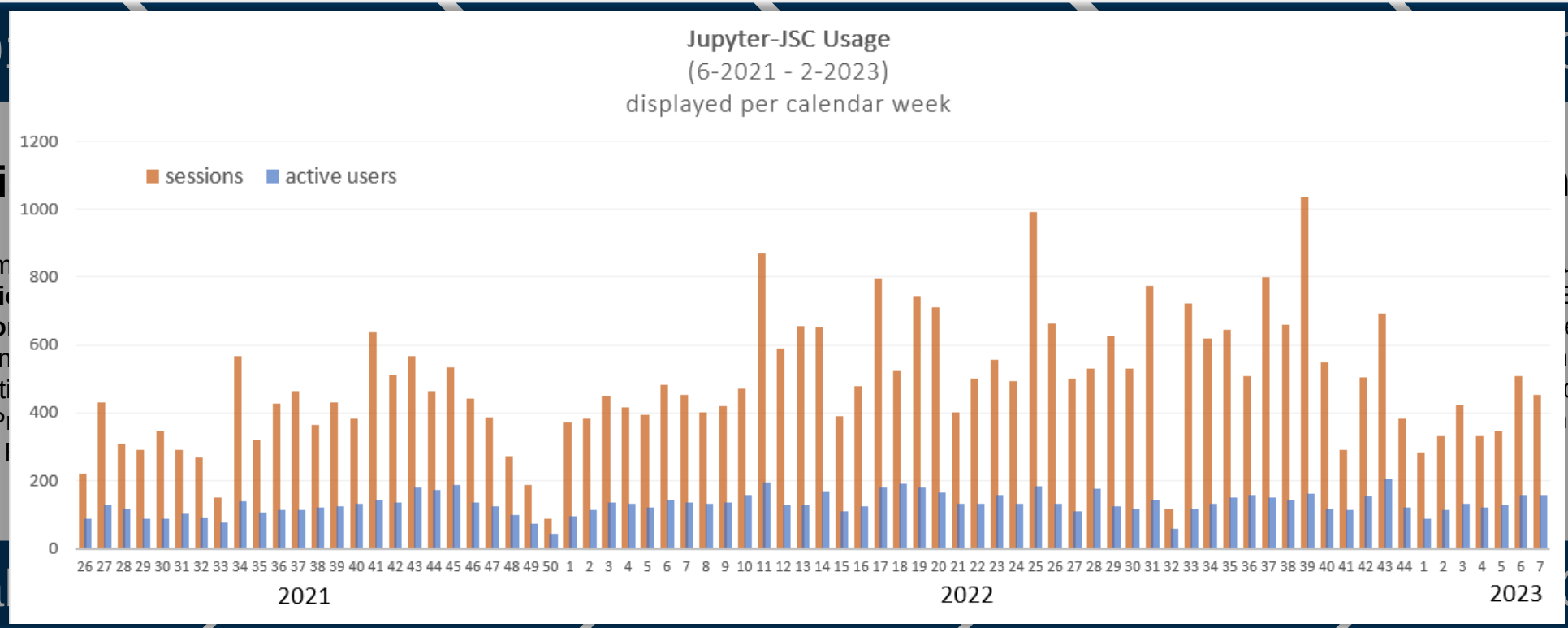

JÜLICH Forschungszentrum

# HISTORY OF JUPYTERLAB AT JSC

20

Initi

JupyterLab m
**Authenticati**
**Authorizatio**
Orchestration
Synchronizati
Basic Data P
Fulfill Safety




023

ization

**unity JHubs**
Entrance-UI
e Permissions
provements
d Balancer
**of Backend**

JLa

3+X



JÜLICH
Forschungszentrum

# TERMINOLOGY

# TERMINOLOGY

## What is JupyterLab

**JupyterLab**

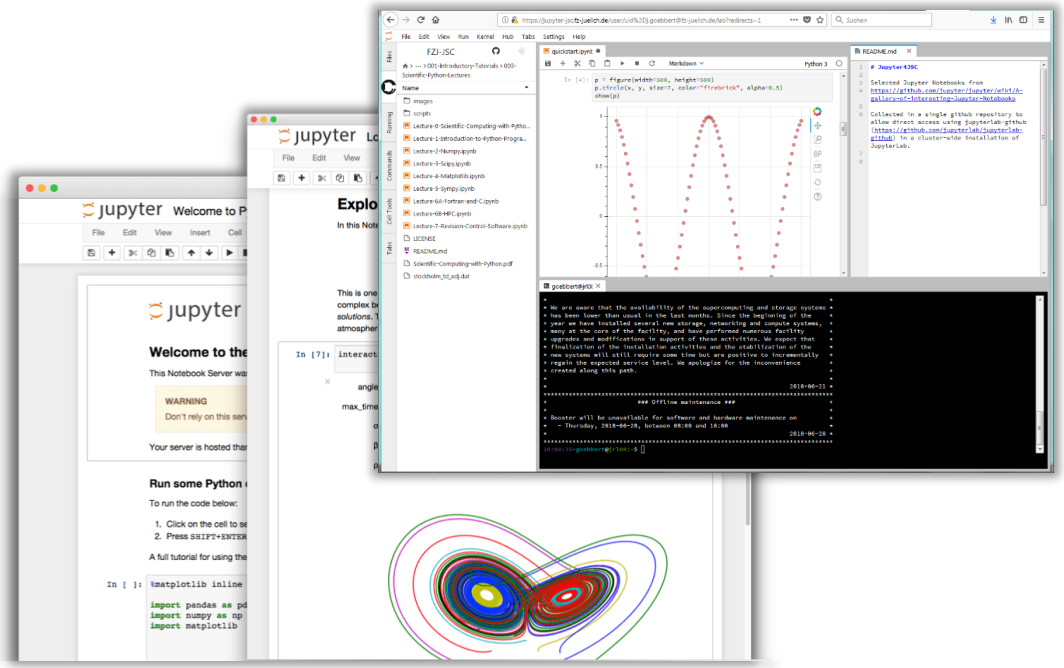
- **Interactive** working environment in the web browser
- For the creation of **reproducible** computer-aided narratives
- Very **popular** with researchers from all fields
- Jupyter = Julia + Python + R

Multi-purpose working environment

- Language agnostic
- Supports execution environments (“*kernels*”)
  - For dozens of languages: Python, R, Julia, C++, ...
- Extensible software design („*extensions*“)
  - many server/client plug-ins available
  - Eg. in-browser-terminal and file-browsing

Document-Centered Computing (“*notebooks*”)

- Combines code execution,
  rich text, math, plots and rich media.
- All-in-one document called Jupyter Notebook



https://jupyterlab.readthedocs.io

JÜLICH Forschungszentrum

# TERMINOLOGY

## What is a Jupyter Notebook?

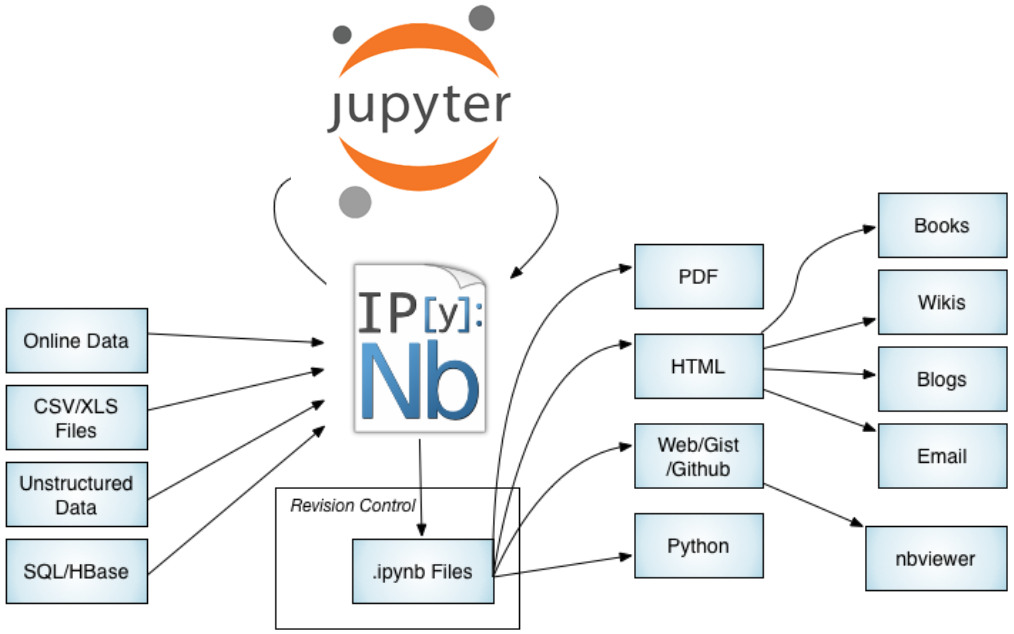
**Jupyter Notebook**

A notebook document (file extension .ipynb)
is a document that can be rendered in a web browser

- It is a file, which stores your work in JSON format
- Based on a set of open standards for interactive computing

- Allows development of custom applications with embedded interactive computing.
- Can be extended by third parties

- Directly convertible to PDF, HTML, LateX ...
- Supported by many applications
  such as GitHub, GitLab, etc..




https://jupyter-notebook.readthedocs.io/
https://github.com/jupyter/jupyter/wiki/A-gallery-of-interesting-Jupyter-Notebooks

# TERMINOLOGY

## What is a Jupyter Kernel?

**Jupyter Kernel**
A “kernel” refers to the separate process
which executes code cells within a Jupyter notebook.

Jupyter Kernel

- **run code** in different programming languages and environments.
- can be **connected to** a notebook (one at a time).
- **communicates** via ZeroMQ with the JupyterLab.

- Multiple **preinstalled** Jupyter Kernels can be found on our clusters
  - Python, R, Julia, Bash, C++, Ruby, JavaScript
  - Specialized kernels for visualization, quantum-computing

- You can easily **create your own kernel** which for example runs your specialized virtual Python environment.


https://jupyter-notebook.readthedocs.io/
https://github.com/jupyter/jupyter/wiki/Jupyter-kernels
https://zeromq.org

# TERMINOLOGY

## What is a JupyterLab Extension?

**JupyterLab Extension**

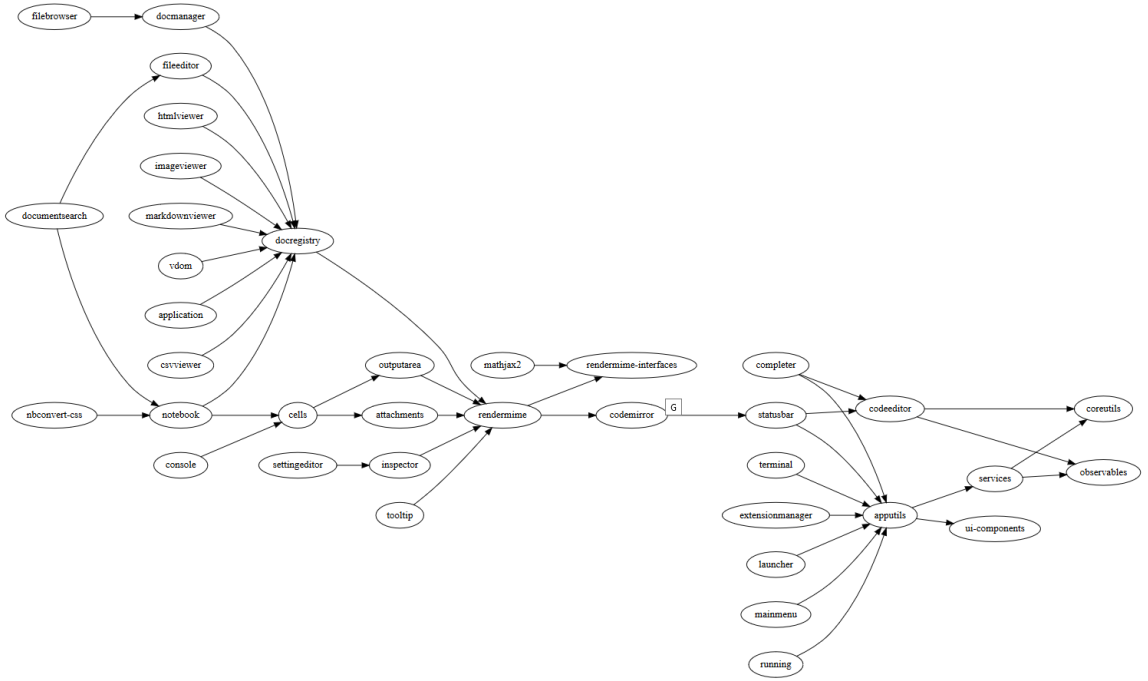
JupyterLab extensions can customize or enhance any part of JupyterLab.

JupyterLab Extensions

- provide new file viewers, editors, themes
- provide renderers for rich outputs in notebooks
- add items to the menu or command palette
- add keyboard shortcuts
- add settings in the settings system.

- Extensions can even provide an API for other extensions to use and can depend on other extensions.

The whole JupyterLab itself is simply a **collection of extensions** that are no more powerful or privileged than any custom extension.

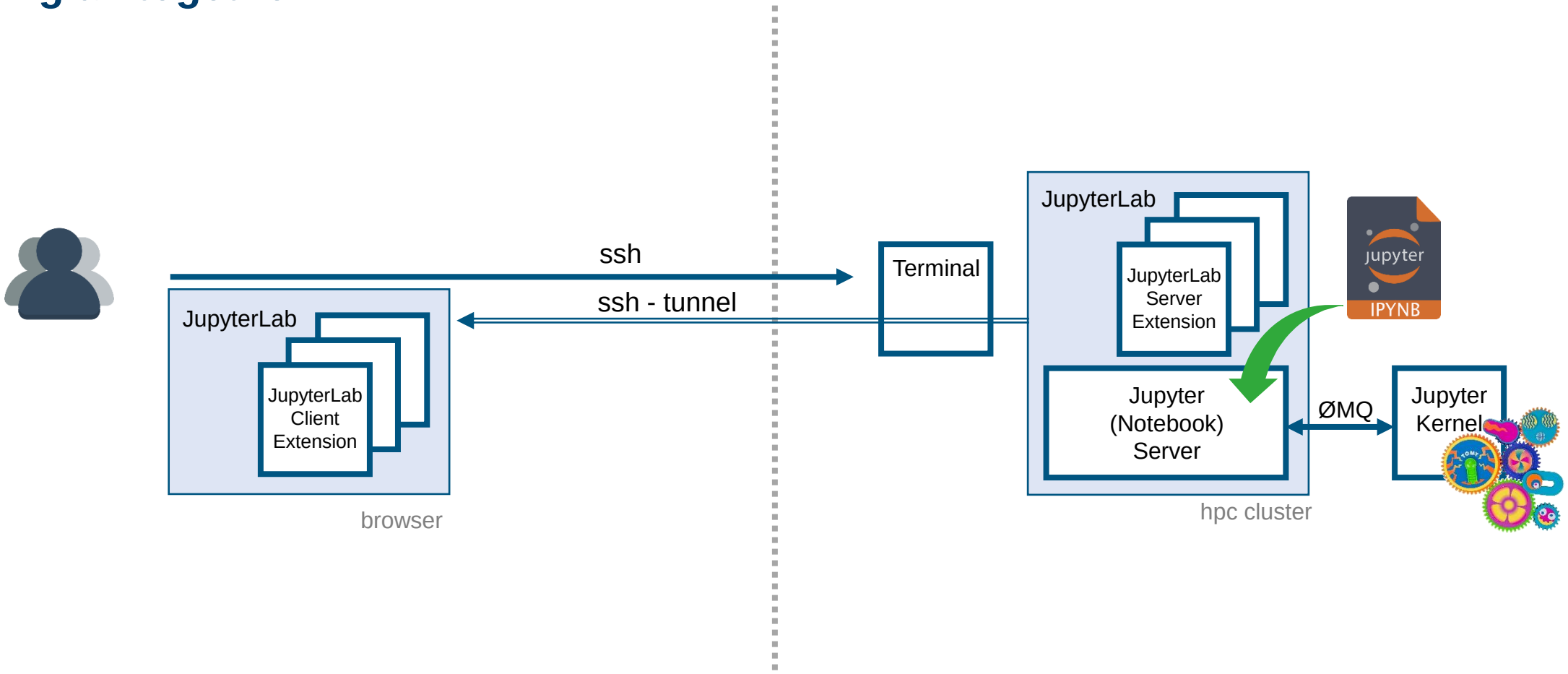



https://jupyterlab.readthedocs.io/en/stable/user/extensions.html
https://github.com/topics/jupyterlab-extension

JÜLICH Forschungszentrum

# TERMINOLOGY

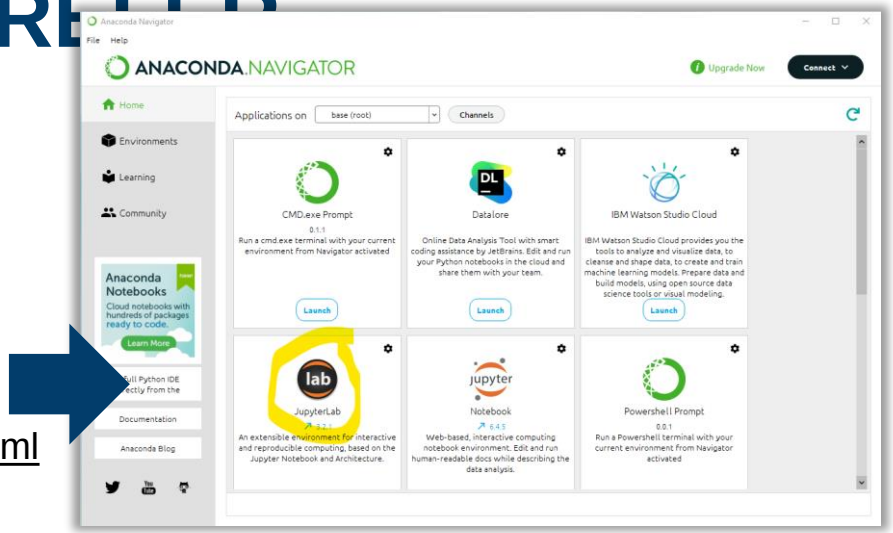
## Bringing all together



# INSTALLATION

# JUPYTERLAB - WHEREVER YOU PREFER

## Local, Remote, Browser-only




**Local installation:**

- **JupyterLab** installed using conda, mamba, pip, pipenv or docker.
  ➔ https://jupyterlab.readthedocs.io/en/stable/getting_started/installation.html

JÜLICH Forschungszentrum

# JUPYTERLAB - WHEREVER YOU PREFER

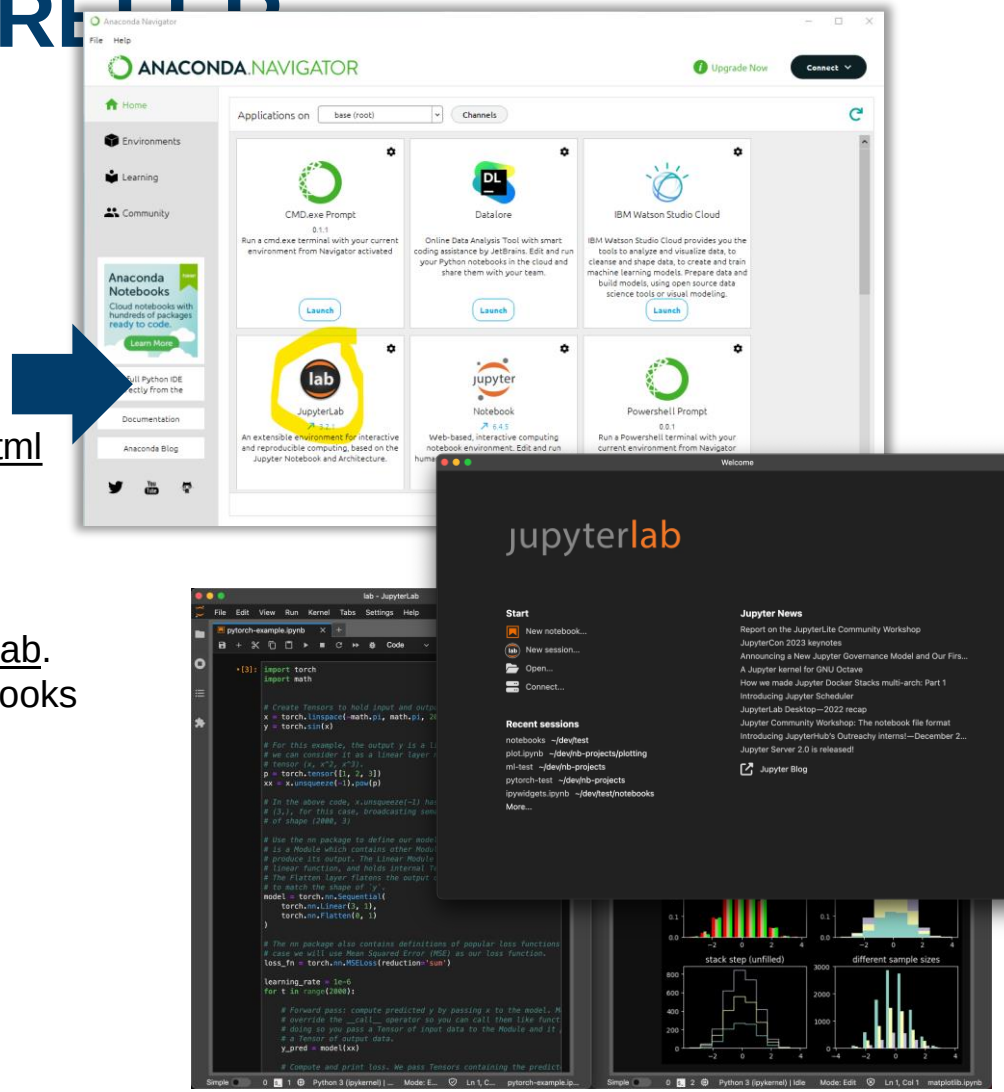
## Local, Remote, Browser-only

**Local installation:**

- **JupyterLab** installed using conda, mamba, pip, pipenv or docker.
  ➔ https://jupyterlab.readthedocs.io/en/stable/getting_started/installation.html
- **JupyterLab** installed as normal desktop application **= JupyterLab Desktop**
  ➔ https://github.com/jupyterlab/jupyterlab-desktop/releases

**JupyterLab Desktop** is the cross-platform desktop application for JupyterLab. It is probably the quickest and easiest way to get started with Jupyter notebooks on your personal computer, with the flexibility for advanced use cases.

(Windows, macOS, Debian/Ubuntu, RedHat/Fedora)




JÜLICH Forschungszentrum

# JUPYTERLAB - WHEREVER YOU PREFER

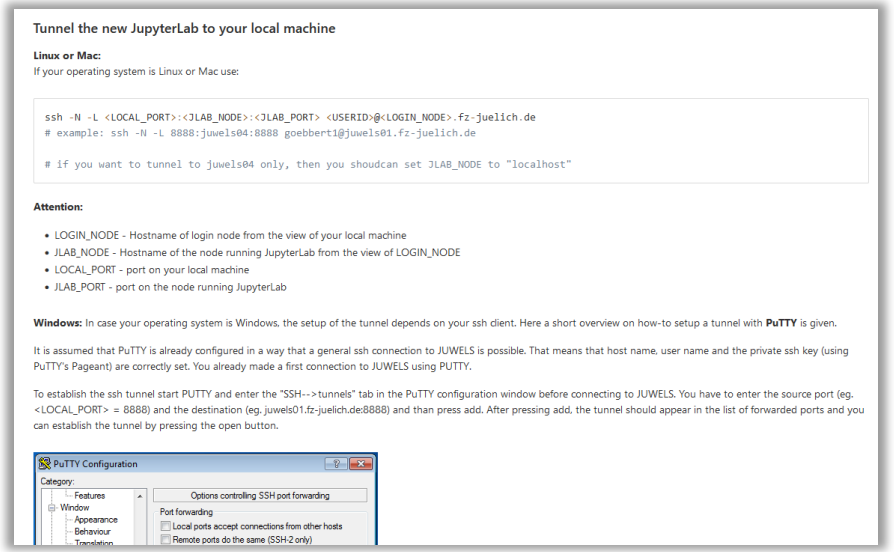
## Local, Remote, Browser-only

### Local installation:

- **JupyterLab** installed using conda, mamba, pip, pipenv or docker.
  ➔ https://jupyterlab.readthedocs.io/en/stable/getting_started/installation.html
- **JupyterLab** installed as normal desktop application **= JupyterLab Desktop**
  ➔ https://github.com/jupyterlab/jupyterlab-desktop/releases

### Remote (cluster) installation:

- **JupyterLab** installed on a remote server and accessed through the browser
  - in $HOME (e.g. using pip or miniconda)
  - system-wide (e.g. with Easybuild, Spark) by the admins.




JÜLICH Forschungszentrum

# JUPYTERLAB - WHEREVER YOU PREFER

## Local, Remote, Browser-only

### Local installation:

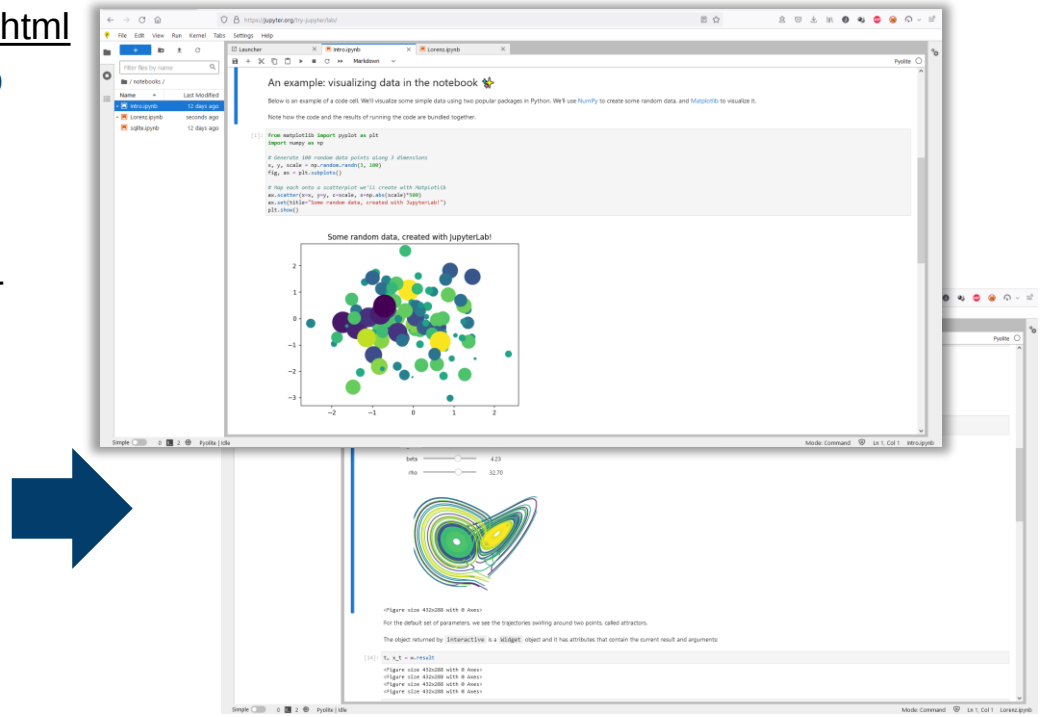
- **JupyterLab** installed using conda, mamba, pip, pipenv or docker.
  ➔ https://jupyterlab.readthedocs.io/en/stable/getting_started/installation.html
- **JupyterLab** installed as normal desktop application **= JupyterLab Desktop**
  ➔ https://github.com/jupyterlab/jupyterlab-desktop/releases

### Remote (cluster) installation:

- **JupyterLab** installed on a remote server and accessed through the browser
  - in $HOME (e.g. using pip or miniconda)
  - system-wide (e.g. with Easybuild, Spark) by the admins.

### Browser-only installation (limited feature set):

- **JupyterLab** local with server + client in your browser **= JupyterLite**
  Includes a browser-ready Python environment named Pyodide.
  ➔ https://jupyter.org/try-jupyter/lab

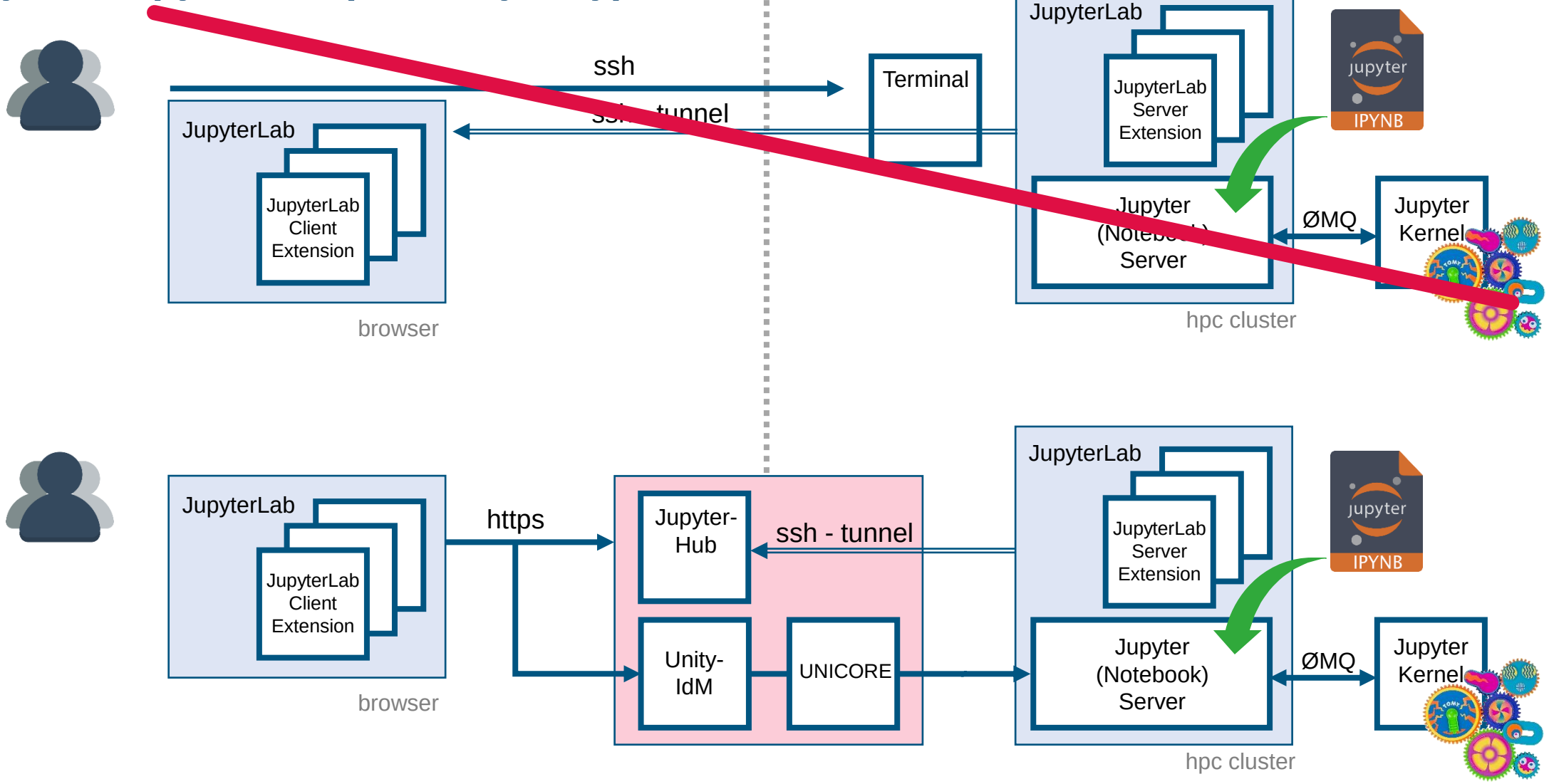



JÜLICH
Forschungszentrum

# START & LOGIN

# JUPYTER-JSC WEBSERVICE

## Start your JupyterLab (the easy way)



# PRE-ACCESS TODOS

## 1) Register & Login

- ✓ https://judoor.fz-juelich.de

## 2) Join a project

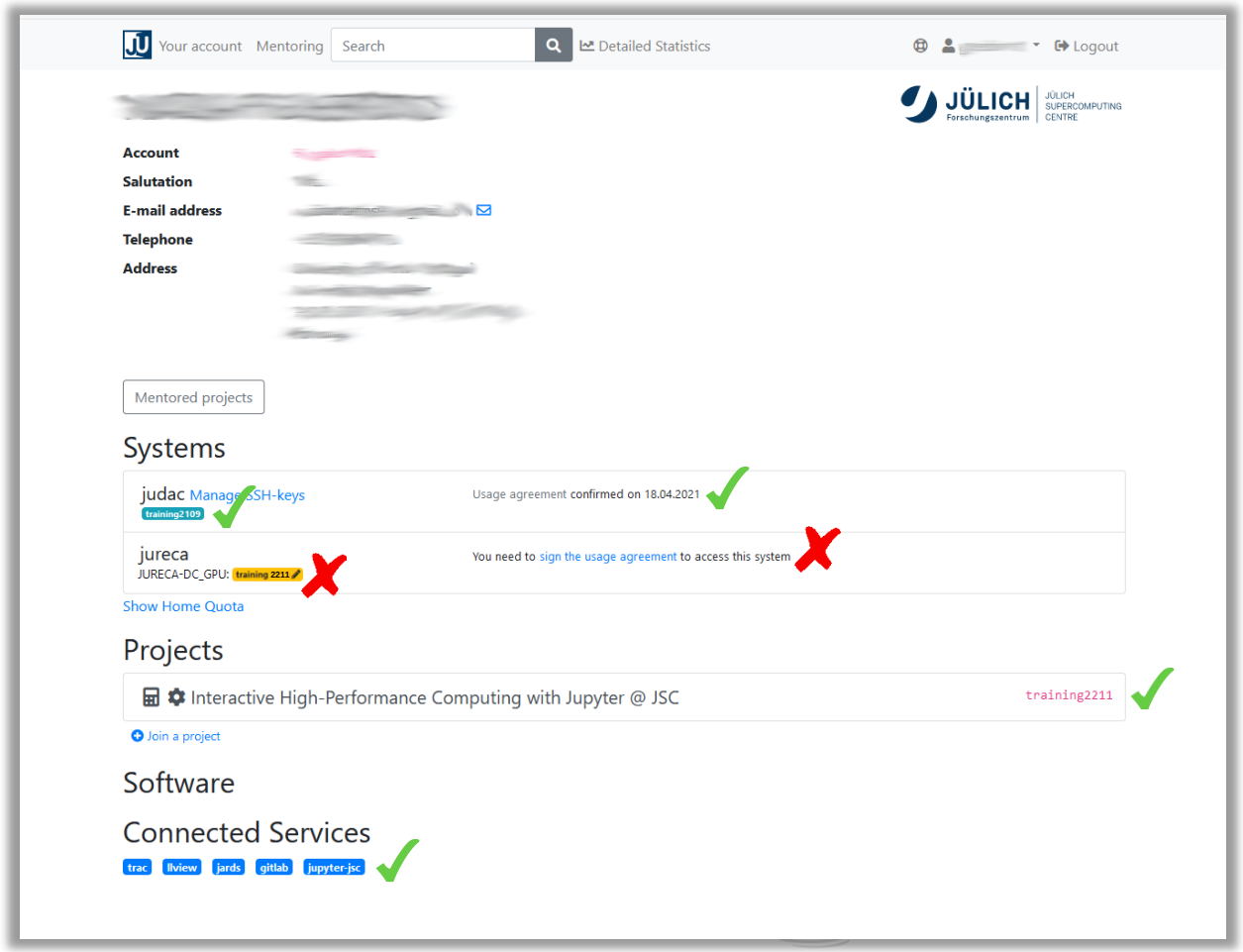
- ✓ Wait to get joined by the project PI

## 3) Sign usage agreement

- ✓ Wait for creation of HPC accounts

## 4) Check Connected Services:

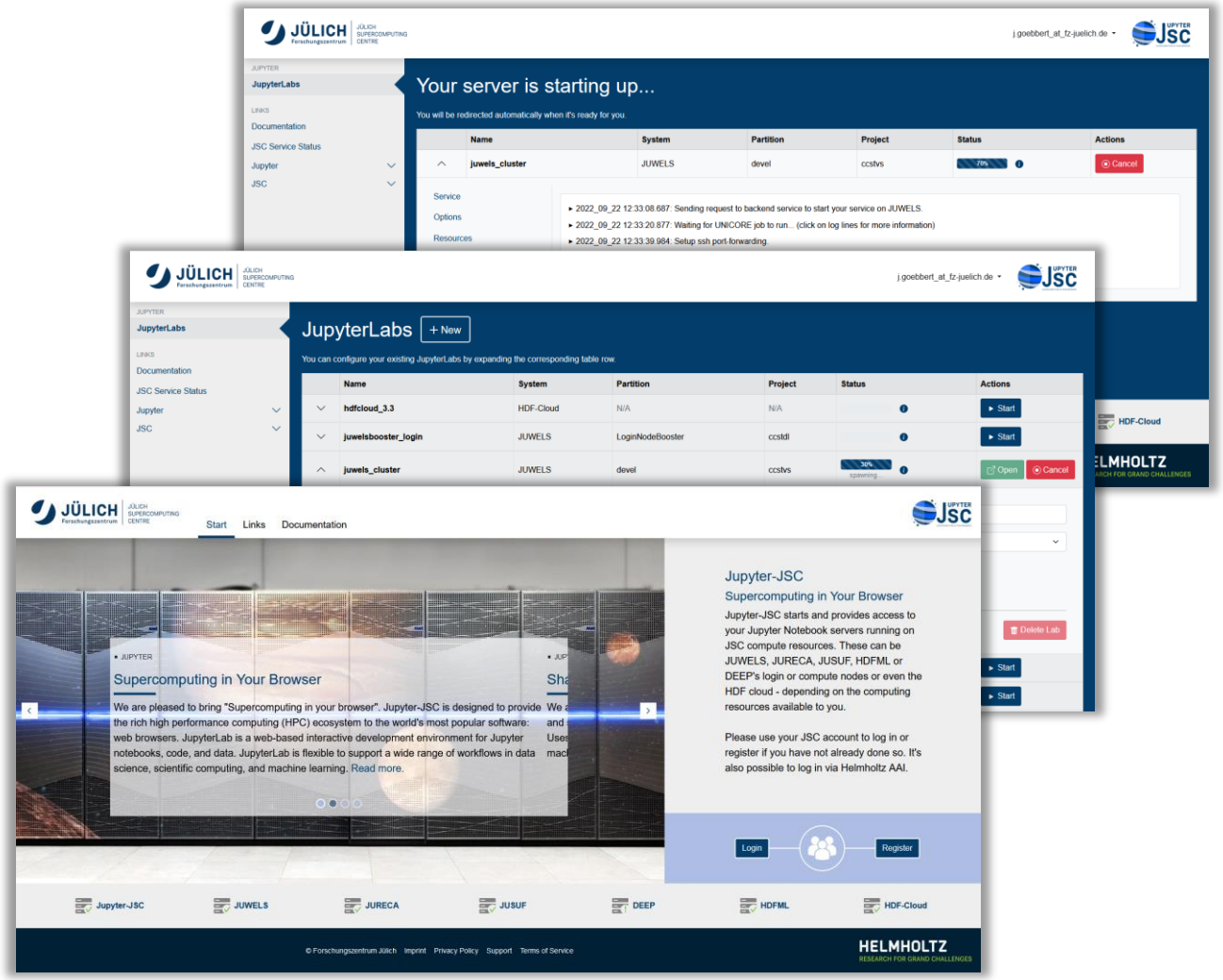
- ✓ jupyter-jsc

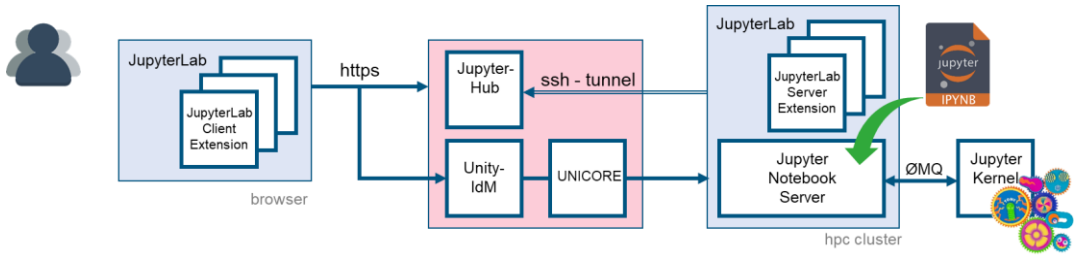


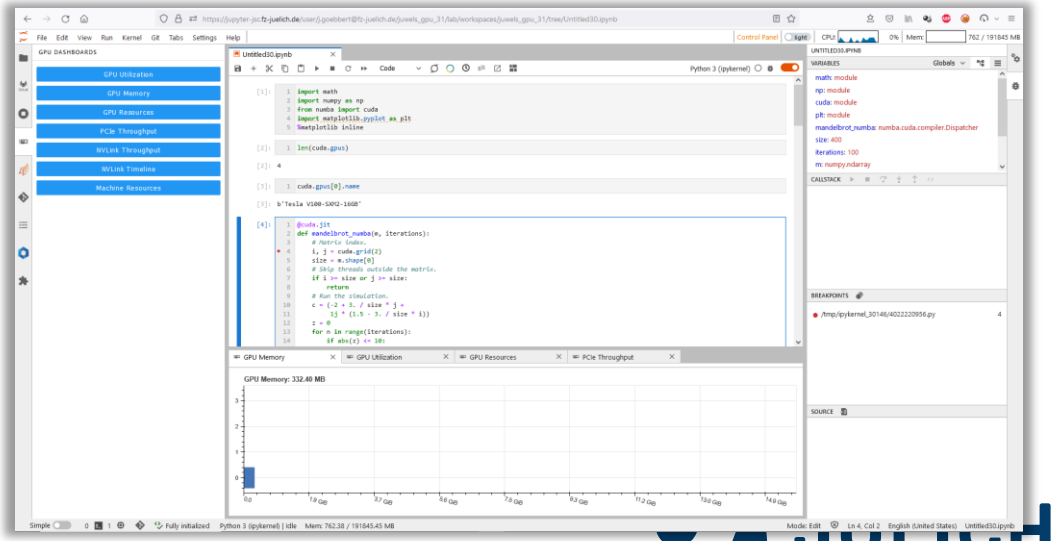

JÜLICH Forschungszentrum

# JUPYTER-JSC WEBSERVICE

## https://jupyter-jsc.fz-juelich.de



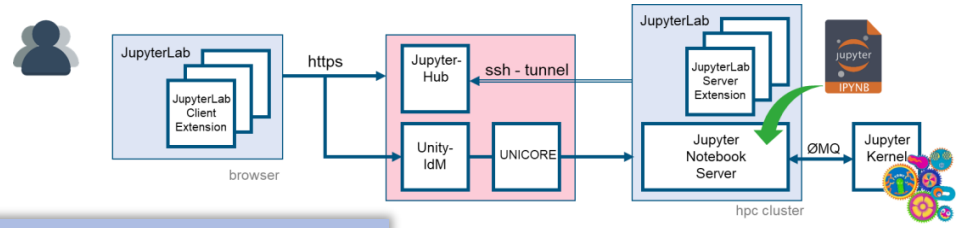




Forschungszentrum

# JUPYTER-JSC WEBSERVICE

## First time login

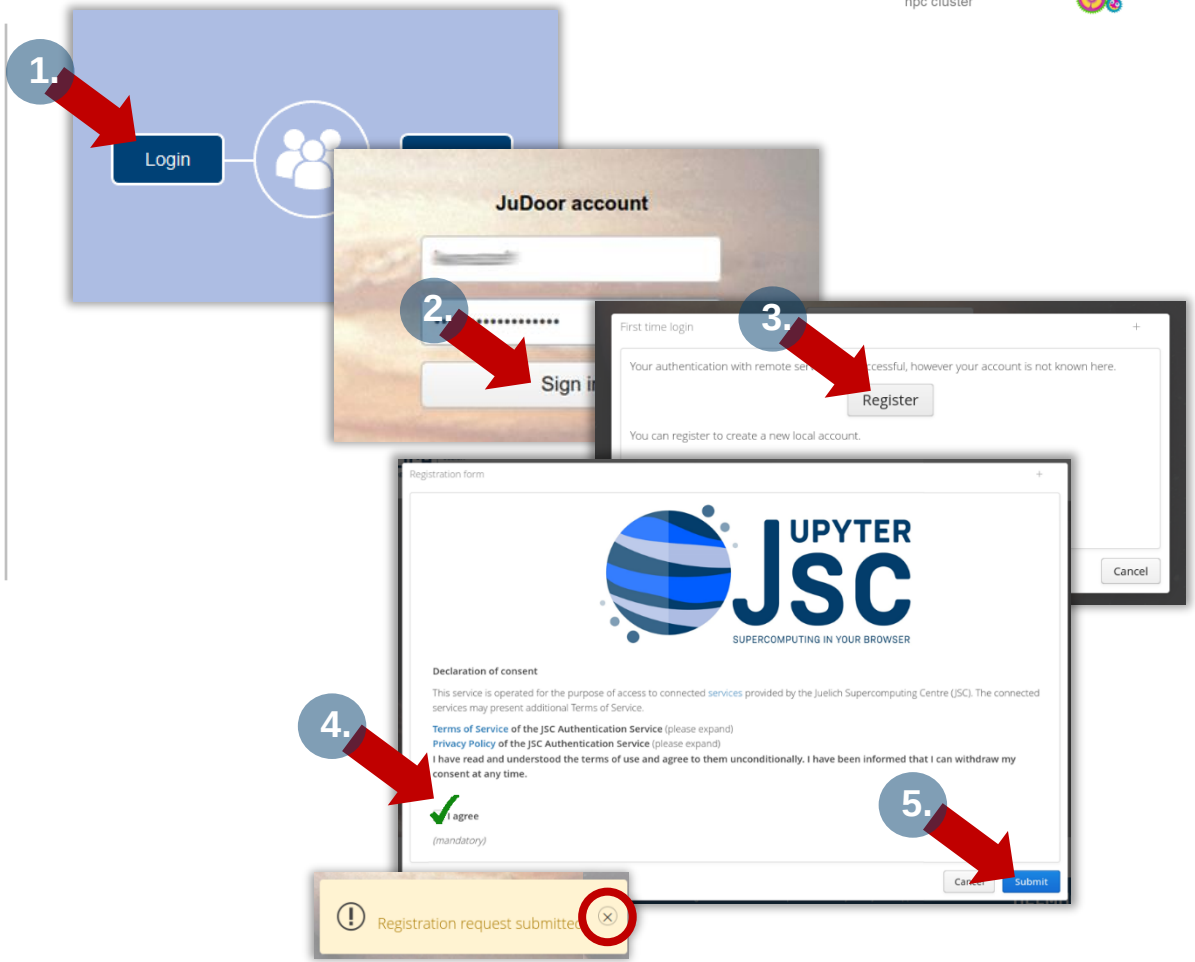


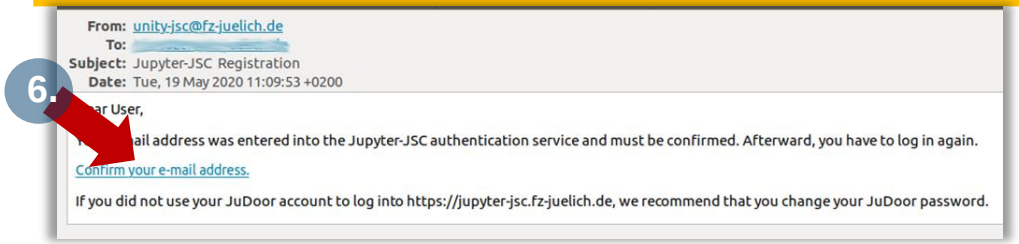

## => https://jupyter-jsc.fz-juelich.de

**Jupyter-JSC first time login**

- Requirements:
  - Registered at **judoor.fz-juelich.de**
    - **(check "Connected Services" = jupyter-jsc)**
  - Project membership + signed systems usage agreement
  - **Waited ~10 minutes**

1. Login at https://jupyter-jsc.fz-juelich.de
2. Sign in with your JSC account
3. Register to Jupyter-JSC
4. Accept usage agreement
5. Submit the registration
6. Wait for email and confirm your email address

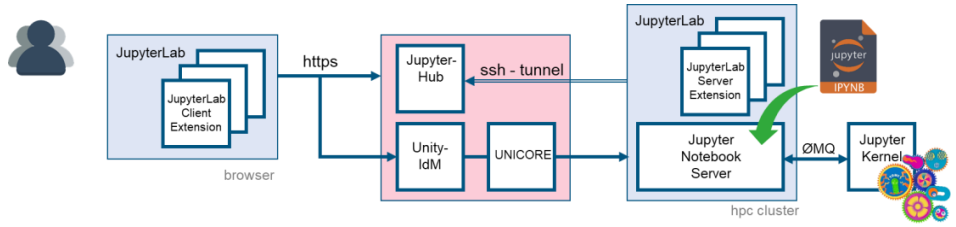


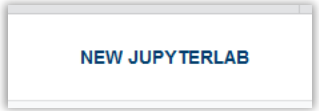

JÜLICH Forschungszentrum

# JUPYTER-JSC WEBSERVICE

## Control Panel




### A. New JupyterLab

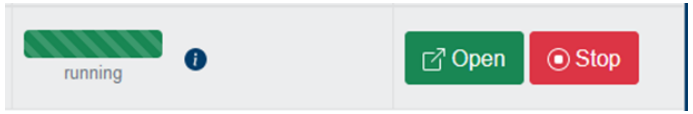


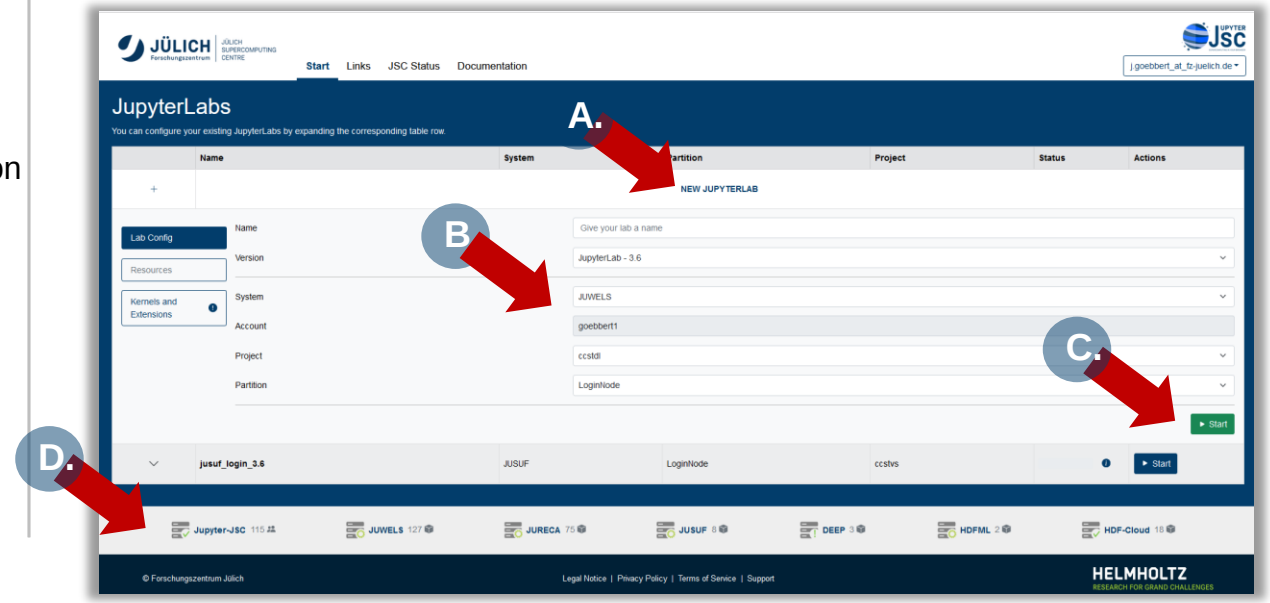

### B. Configuration Dialog

- Lab Config: set Name, Version, System, Account, Project, Partition
- Resources: if running on a compute node
- Kernels and Extensions: Optional addons

### C. Actions

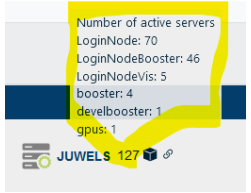
- Start/Open/Stop a JupyterLab
- Change/Delete **configuration**



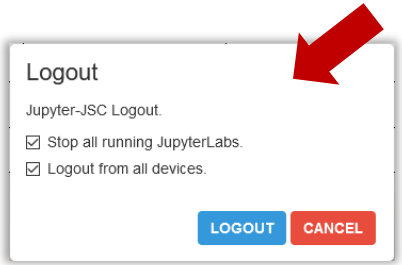

### D. Statusbar


- Shows, (hover to get more details)
  - Number of active users in the last 24h
  - Number of running JupyterLabs
- Click to see system status page




### E. Logout

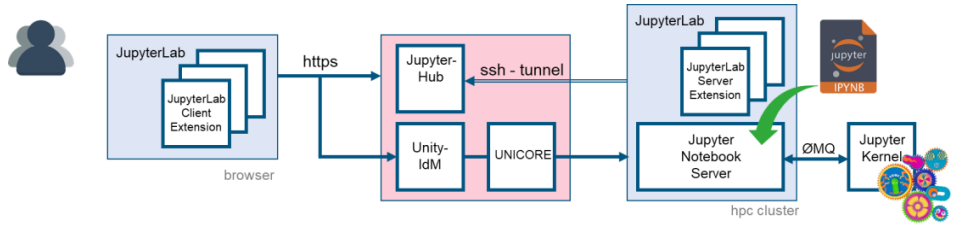
- **Logout will ask what you want to do with the running JupyterLabs – be careful what you answer!**




JÜLICH Forschungszentrum

# JUPYTER-JSC WEBSERVICE

## JupyterLab Configuration




**Jupyter-JSC – Configuration**

Available options **depend on**

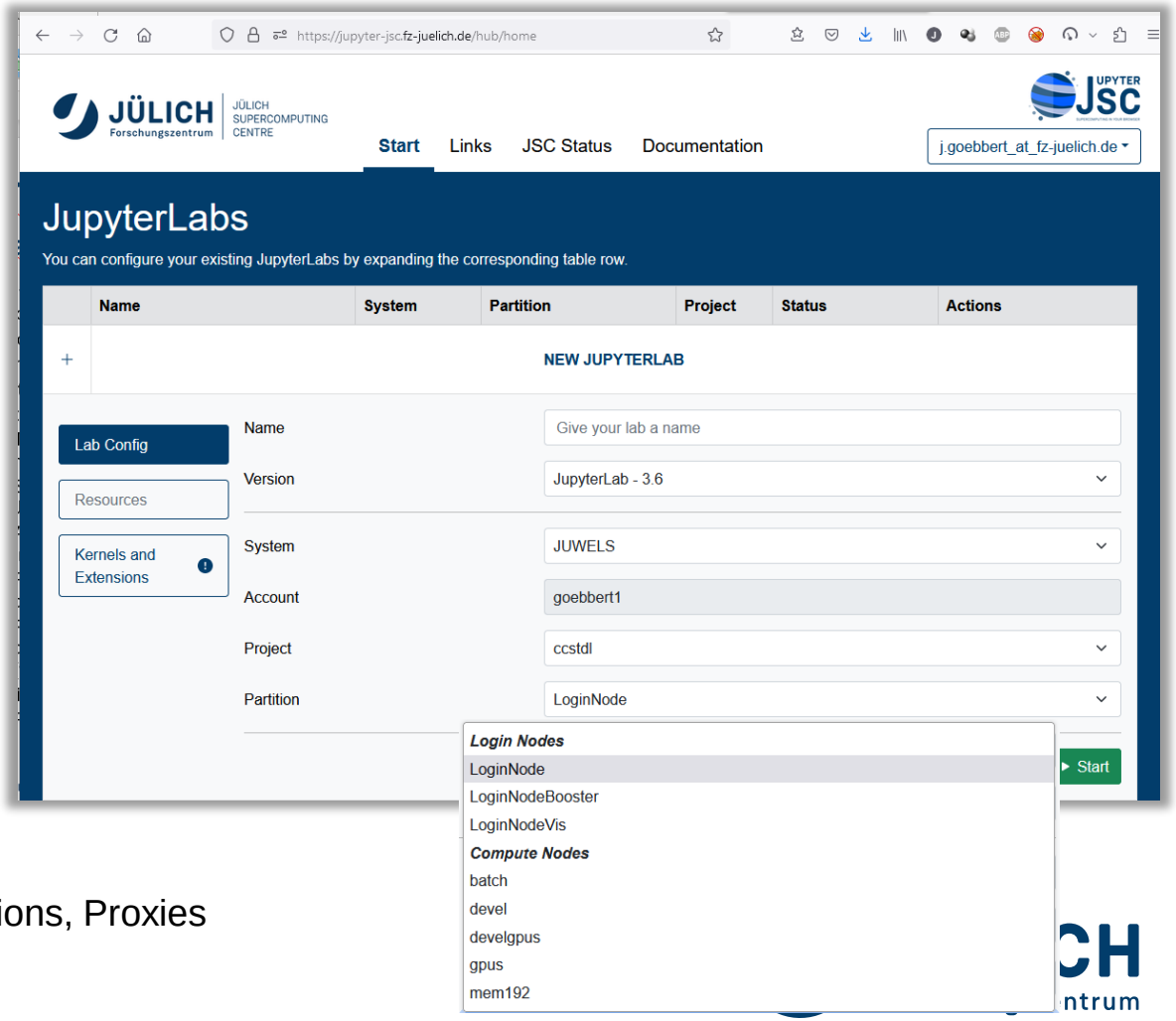
- user account settings visible in **judoor.fz-juelich.de**
- system specific usage agreement on JuDoor is signed (!!!)
- currently available systems in all of your projects

**Basic options**

- Version:
  multiple versions of JupyterLab are installed
- System:
  JUWELS, JURECA, JUSUF, DEEP, HDFML, HDF-Cloud
- Account:
  In general users only have a single account
- Project:
  project which have access to the selected system
- Partition:
  partition which are accessible by the project
  (this includes the decision for LoginNode and ComputeNode)

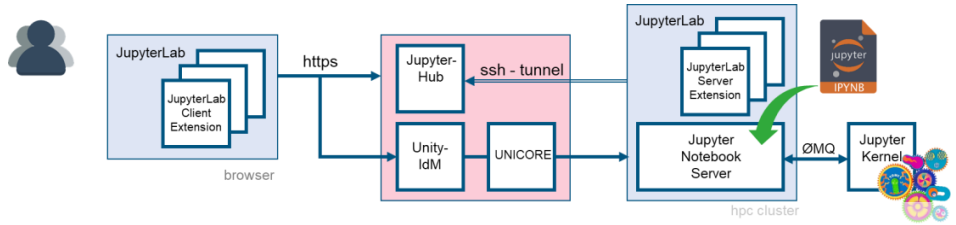
**Extra options**

- Partition == compute  Resources
- Kernel and Extensions  non-default JupyterKernel, Extensions, Proxies



# JUPYTER-JSC WEBSERVICE

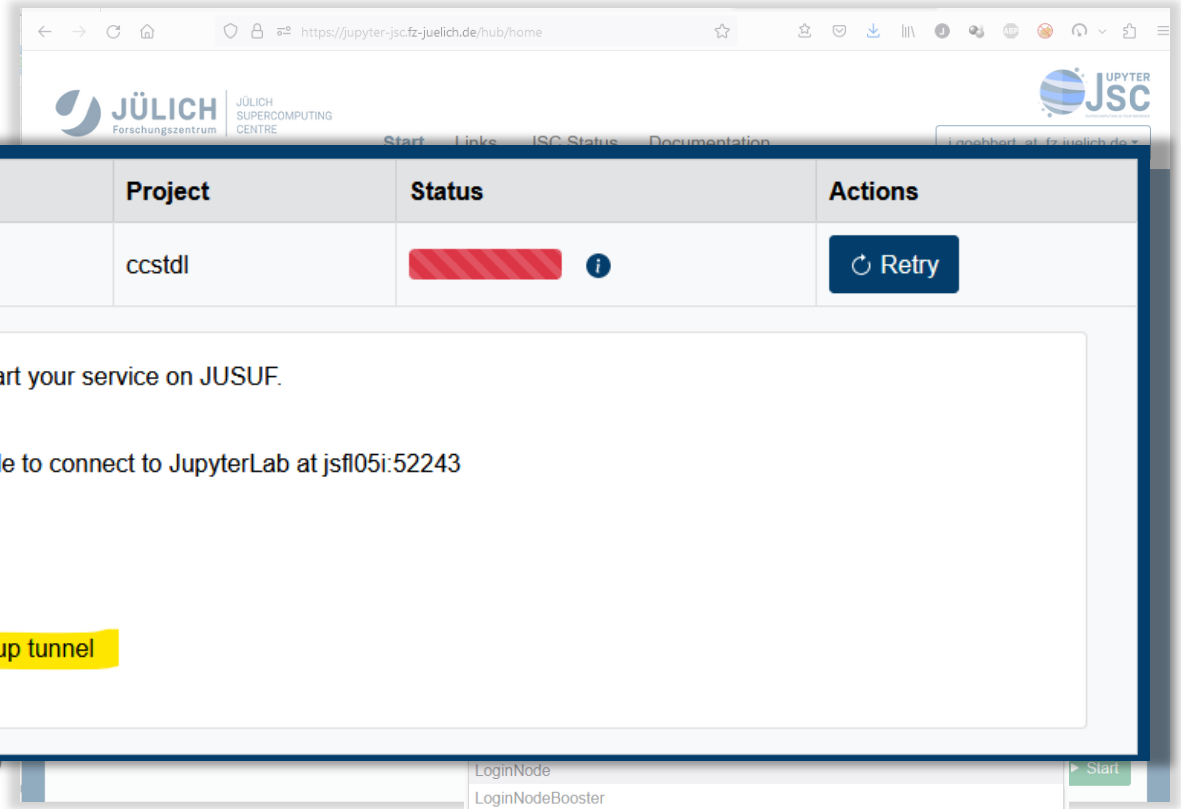
## JupyterLab Configuration




### Jupyter-JSC – Configuration

Available options **depend on**

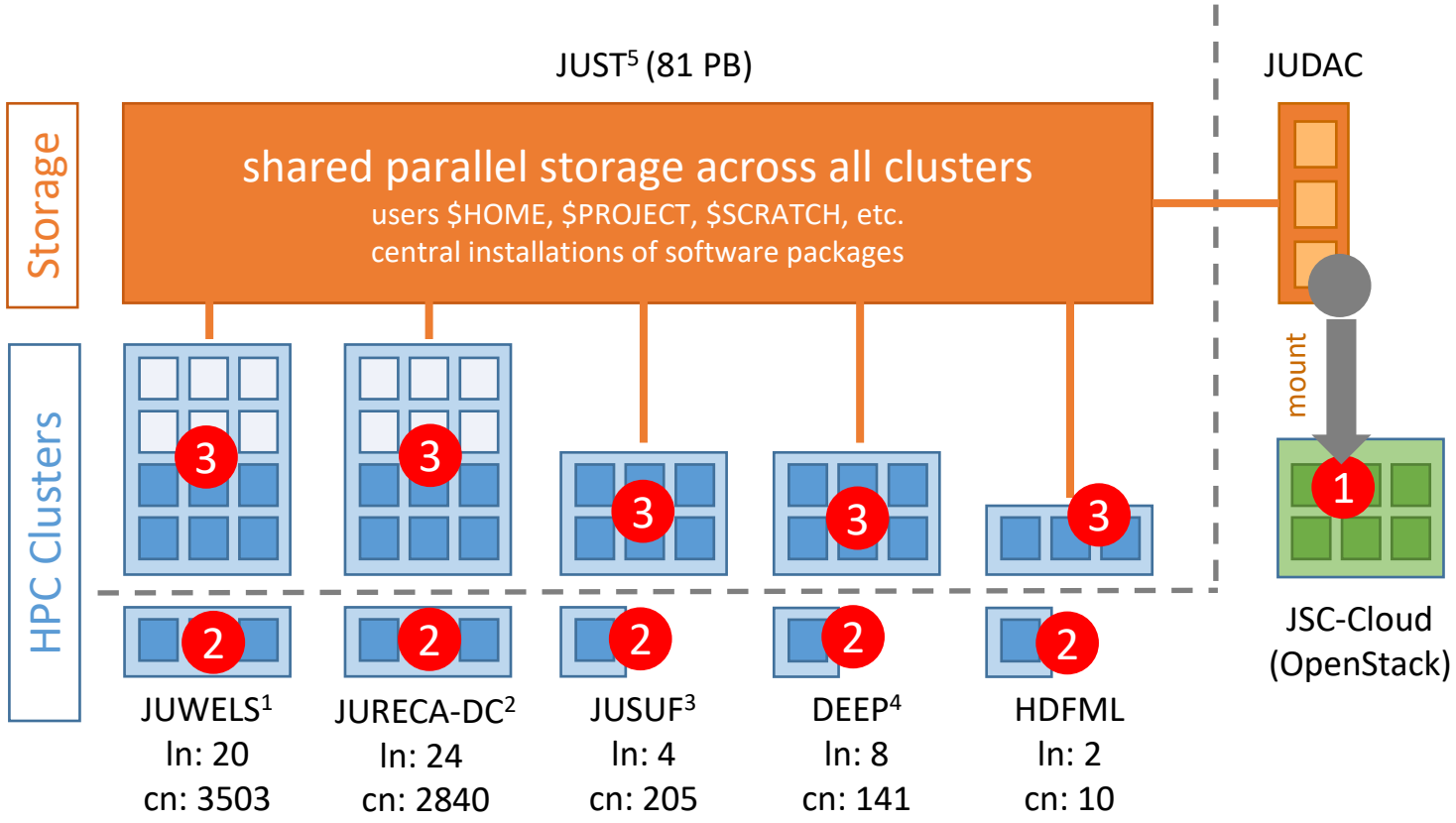
- user account settings visible in **judoor.fz-juelich.de**
- system specific usage agreement on JuDoor is signed (!!!)




### Extra options

- Partition == compute — Resources
- Kernel and Extensions — non-default JupyterKernel, Extensions, Proxies

# JUPYTERLAB EVERYWHERE




no. login nodes = ln
no. compute nodes = cn

**JupyterLab everywhere**

1. JupyterLab on cloud
2. JupyterLab on login nodes
3. JupyterLab on compute nodes

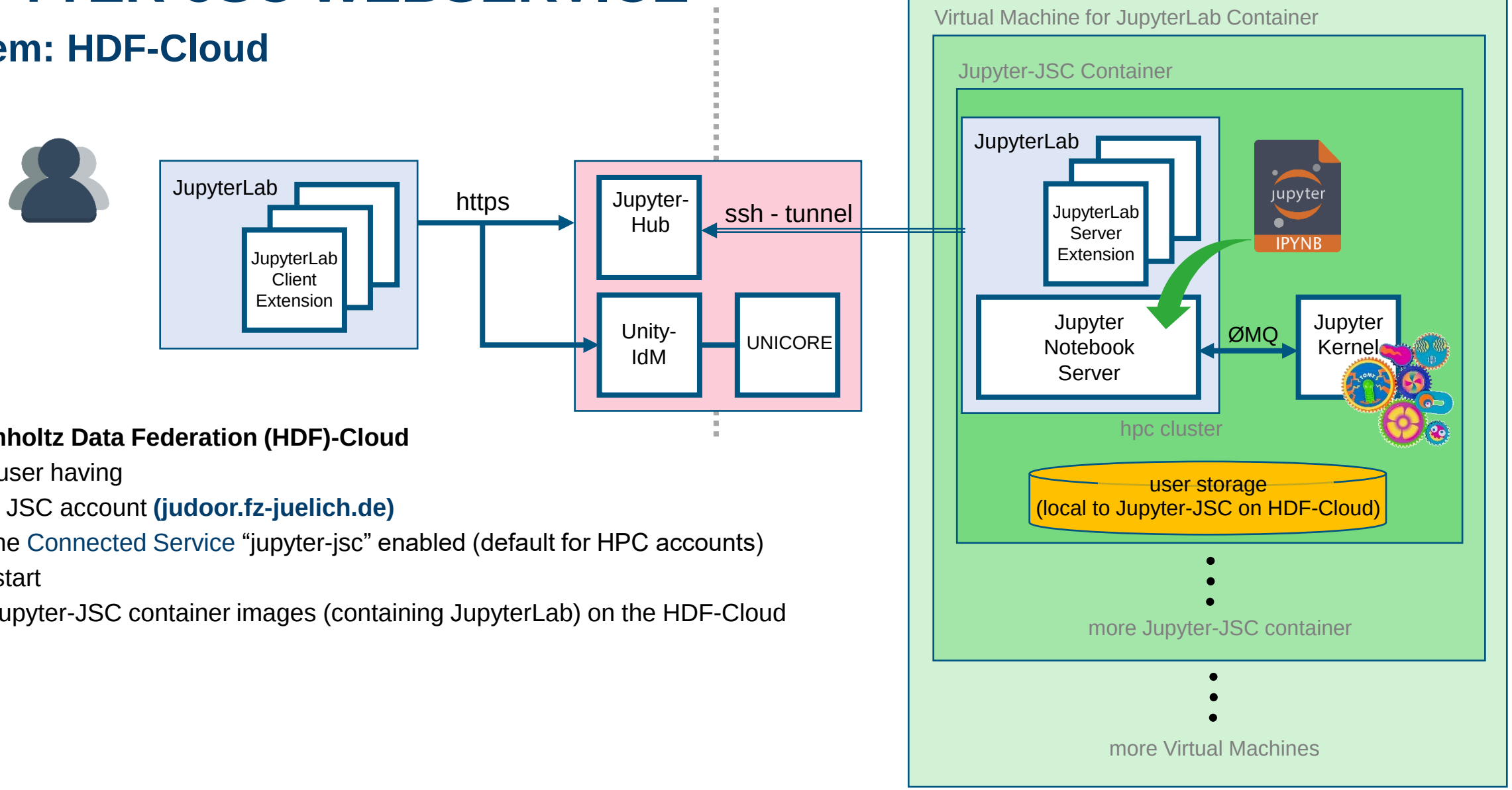
[1] https://apps.fz-juelich.de/jsc/hps/juwels/configuration.html
[2] https://apps.fz-juelich.de/jsc/hps/jureca/configuration.html
[3] https://apps.fz-juelich.de/jsc/hps/jusuf/cluster/configuration.html
[4] https://www.fz-juelich.de/ias/jsc/EN/Expertise/Supercomputers/DEEP-EST/_node.html
[5] https://www.fz-juelich.de/ias/jsc/EN/Expertise/Datamanagement/OnlineStorage/JUST/Configuration/Configuration_node.html



JÜLICH Forschungszentrum

# JUPYTER-JSC WEBSERVICE

## System: HDF-Cloud




**Helmholtz Data Federation (HDF)-Cloud**

Any user having

- a JSC account **(judoor.fz-juelich.de)**
- the Connected Service “jupyter-jsc” enabled (default for HPC accounts)

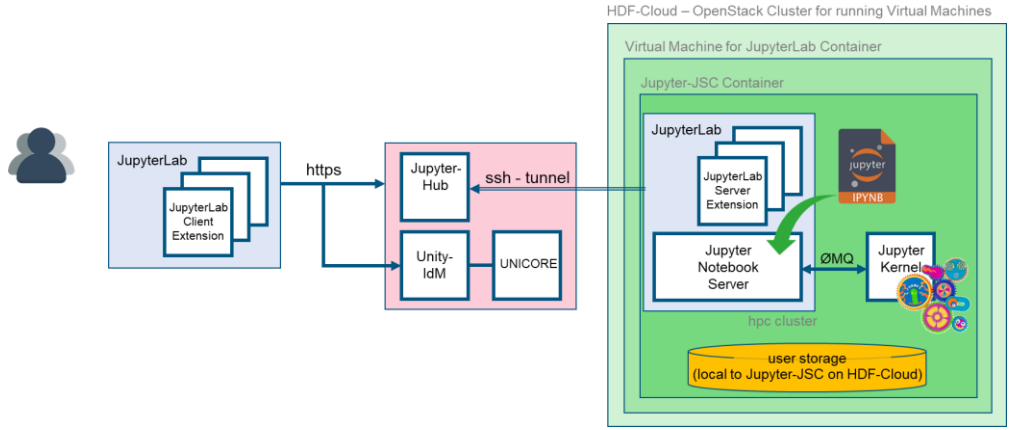
can start

- Jupyter-JSC container images (containing JupyterLab) on the HDF-Cloud

https://www.fz-juelich.de/ias/jsc/EN/Expertise/SciCloudServices/HDFCloud/_node.html

# JUPYTER-JSC WEBSERVICE

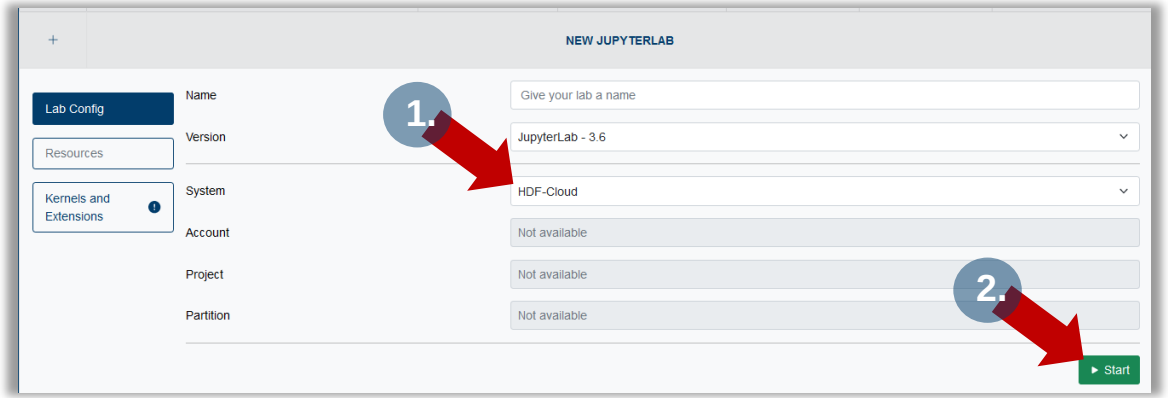
## System: HDF-Cloud




**Start JupyterLab on HDF-Cloud**

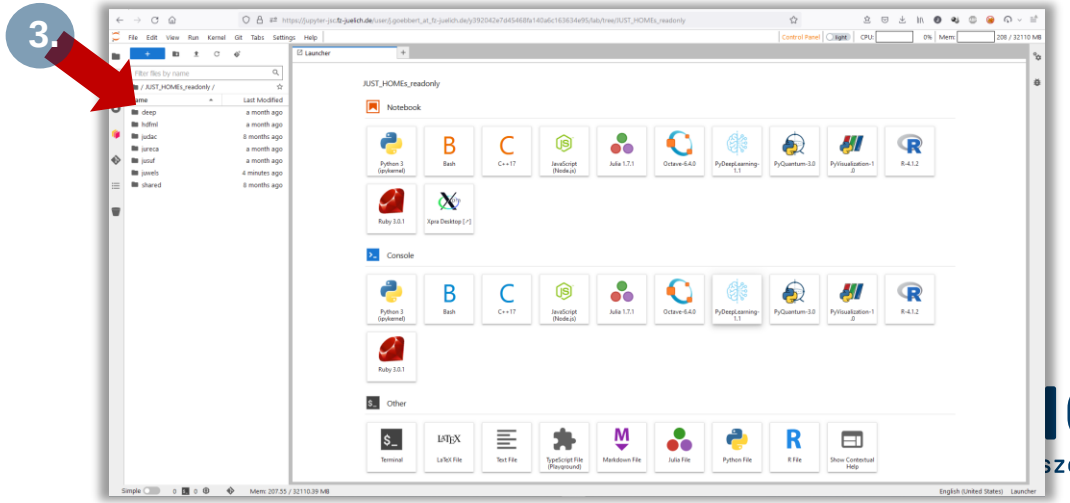
- Requirements:
  - Registered JSC account at https://judoor.fz-juelich.de
  - Logged in to Jupyter-JSC at https://jupyter-jsc.fz-juelich.de
  - Named a new JupyterLab configuration

- Start a JupyterLab:
  - Version == “JupyterLab 3.6”
  - System == “HDF-Cloud”

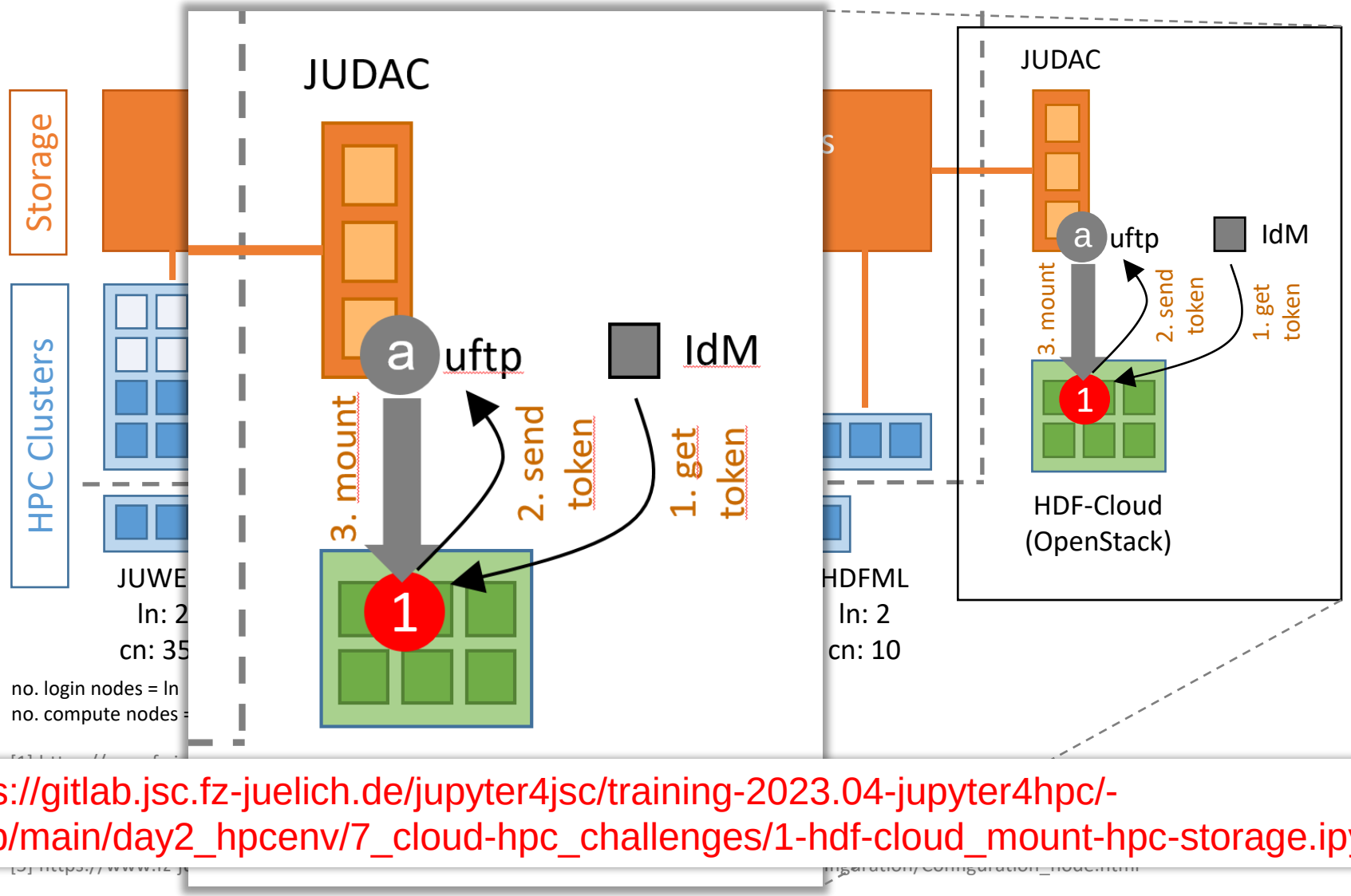



**Limitations on JupyterLab on HDF-Cloud**

- max. **4 GB** memory
  - ATTENTION:
    the container automatically stops, when this limit is reached.
- Storage in Jupyter-JSC container
  - **is local** to the HDF-Cloud
  - HPC $HOMEs are mounted read-only
  - only accessible from a Jupyter-JSC container
- HDF-Cloud has **no GPUs**



# HOW TO MOUNT GPFS ON HDF-CLOUD

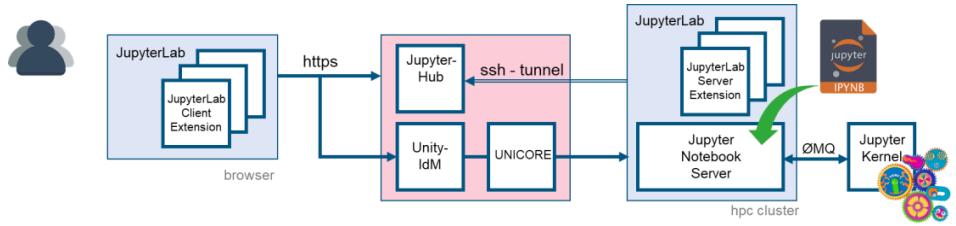



https://gitlab.jsc.fz-juelich.de/jupyter4jsc/training-2023.04-jupyter4hpc/-/blob/main/day2_hpcenv/7_cloud-hpc_challenges/1-hdf-cloud_mount-hpc-storage.ipynb

JÜLICH Forschungszentrum

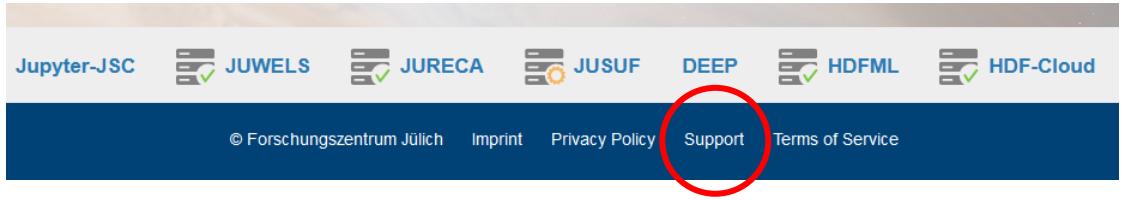
# JUPYTER-JSC SECRETS

## Very important to know



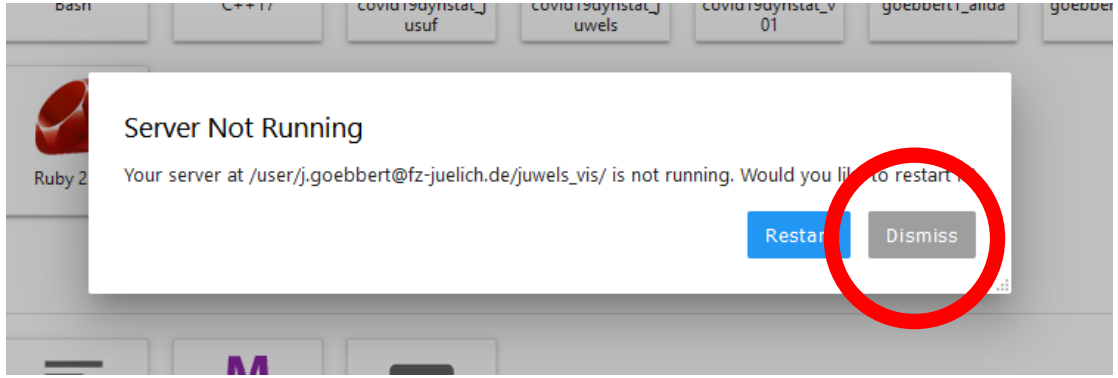

**Secret 1: Support button**

- Let us know, if something does not work.
  We can only fix it, if we know it.

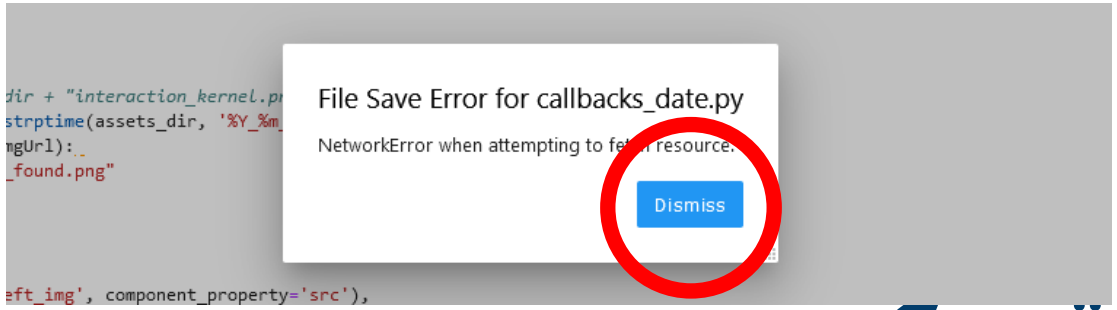



**Secret 2: Reload on connection loss**

- "Server Not Running"
  means, that your browser just lost connection
  => **Just hit "Dismiss" !!!**
  (as soon as you are online again)




- "File Save Error for <…>"
  means, that your browser just lost connection
  => **Just hit "Dismiss" !!!**
  (as soon as you are online again)

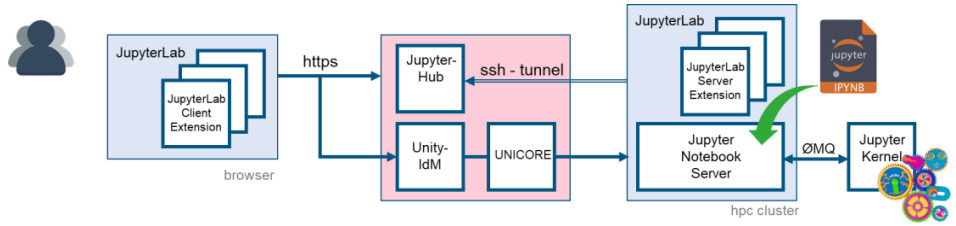



You can **always** safely hit the "Reload" button of your browser, if the connection to JupyterLab ever gets lost.
(it will just restart JupyterLab on the browser-site)

JÜLICH Forschungszentrum

# JUPYTER-JSC SECRETS

## For experts only ☺




**Secret 3: Jupyter-JSC logs**

- Jupyter-Lab gets started by UNICORE on our HPC systems
- On startup UNICORE created the directory `$SCRATCH_<project>/unicore-jobs/<random-hash>/`
  - In the terminal of a running JupyterLab, this directory is `$JUPYTER_LOG_DIR`
- In this directory you find
  - `stdout` -> terminal output of jupyterlab messages
  - `stderr` -> terminal output of jupyterlab error messages
  - `.start` -> details how your JupyterLab got started

**Secret 4: change to a different JupyterLab version**

- In `.start` you can see, that
  - `$HOME/.jupyter/start_jupyter-jsc.sh`

  is used to prepare the environment for JupyterLab.
  This script must ensure that the command `jupyter` is available in $PATH.

```
#!/bin/bash

module purge
module load Stages/2023
module load GCCcore/.11.3.0
module load JupyterCollection/2023.3.6
```

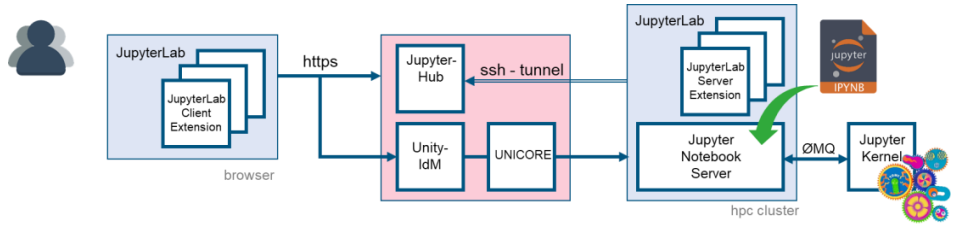
Switch to a customized JupyterLab with
`$HOME/.jupyter/start_jupyter-jsc.sh`

It enables you to switch to an older/newer/other version of JupyterLab,
if the default one gives you trouble or is missing features.

JÜLICH Forschungszentrum

# JUPYTER-JSC WEBSERVICE

## Some comments about the UI




open filebrowser

open launcher

type of active notebook cell

no close, but go back to Jupyter-JSC´s controll panel

memory consuption (keep an eye on that!)

tutorials & examples

Type of Jupyter kernel this notebook is connected to (click to change)

sidebar with core and extentions features

[*] indicates that cell was send to Jupyter kernel for execution

notebook cell

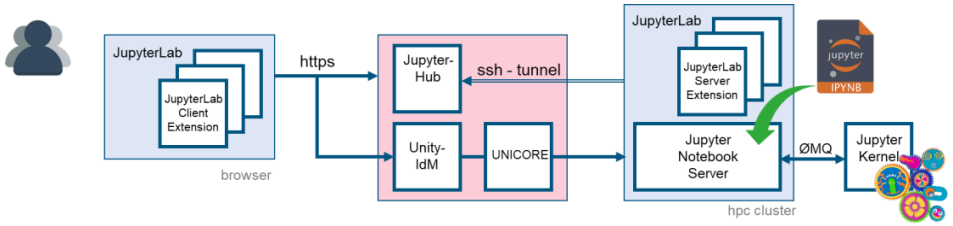
indicates active notebook cell

[ ] indicates that cell has never been executed by the connected Jupyter kernel

# JUPYTERLAB EXTENSIONS

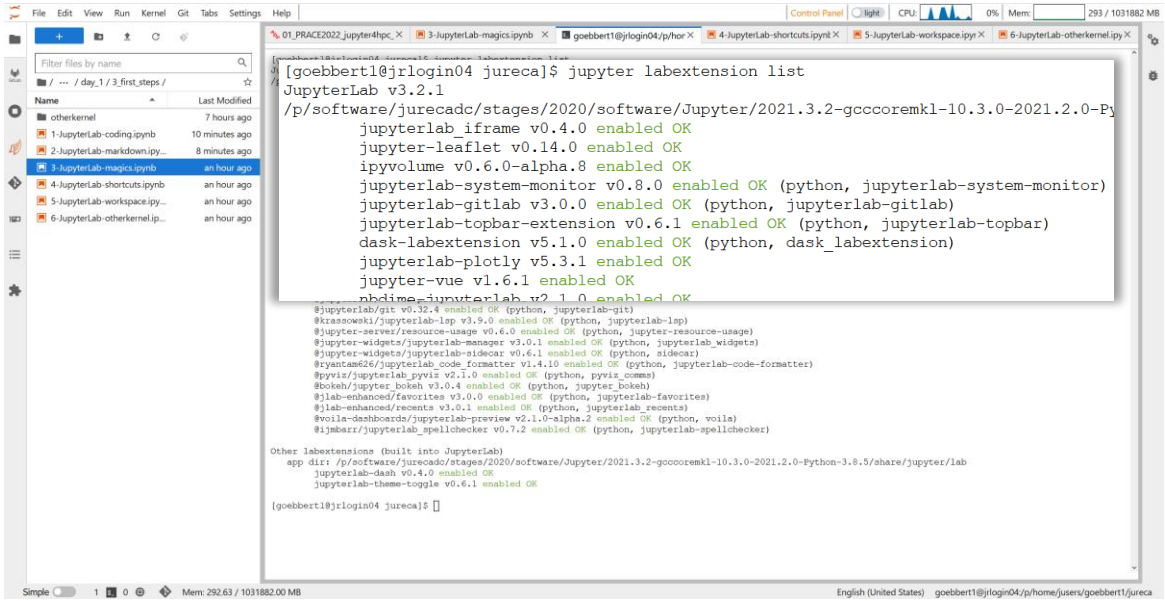
# JUPYTERLAB EXTENSIONS

## Some general information




**List the installed JupyterLab extensions**

- Open the Launcher
- Start a Terminal
- Run command `jupyter labextension list`




**Extensions are installed in JupyterLab´s Application Directory, which**

- stores any information that JupyterLab persists
  - including settings and built assets of extensions
- default location is `<sys-prefix>/share/jupyter/lab`
- can be relocated by setting $JUPYTERLAB_DIR
  - contains the JupyterLab static assets
    - (e.g. `static/index.html`)
  - **JupyterLab < 3:**
    any change requires a rebuild of the whole JupyterLab to take effect!
  - **JupyterLab >= 3:**
    introduced prebuild extensions, which are loaded at startup time

https://jupyterlab.readthedocs.io/en/stable/user/extensions.html

**Hint: JupyterLab Playground**

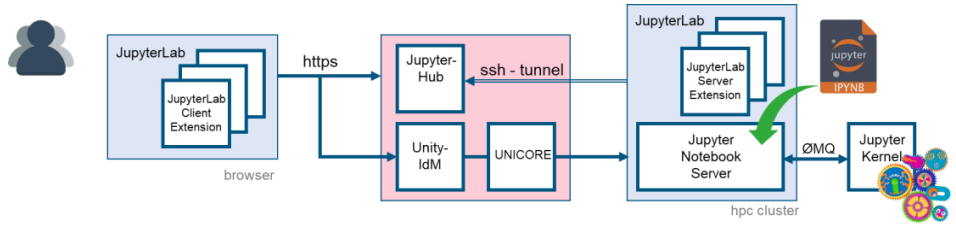
A JupyterLab extension to write and load simple JupyterLab plugins inside JupyterLab.

https://github.com/jupyterlab/jupyterlab-plugin-playground

JÜLICH Forschungszentrum

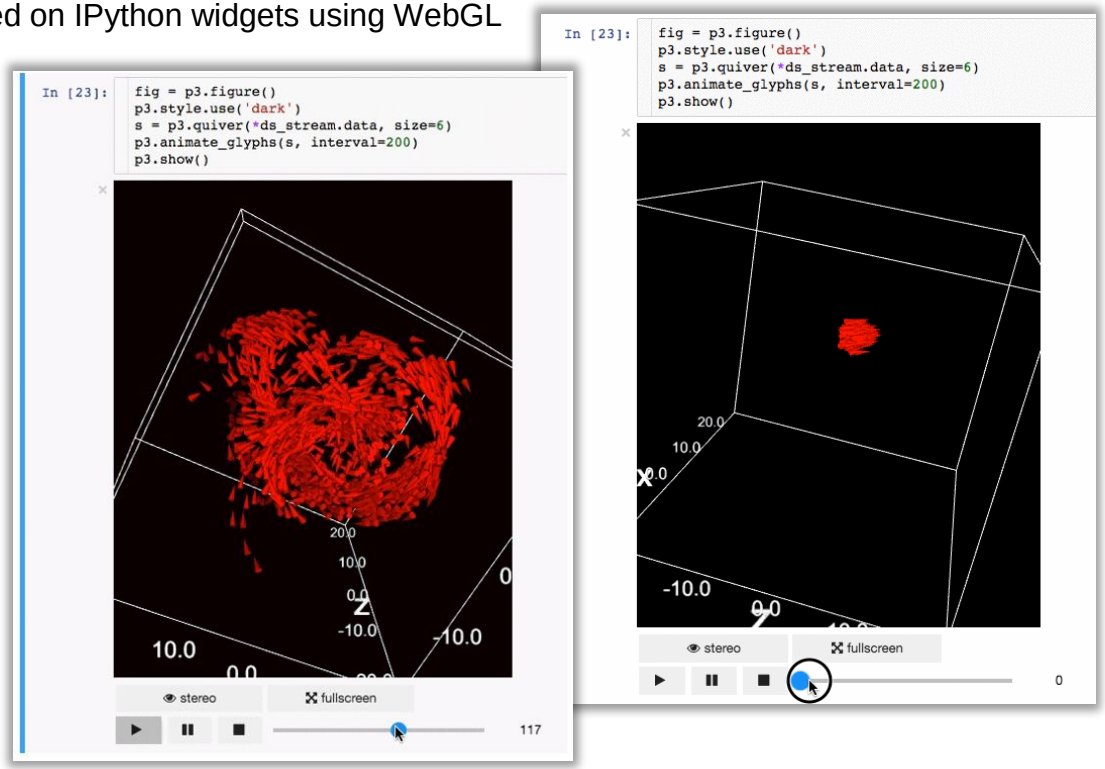
# JUPYTERLAB EXTENSIONS

## Installed by default at Jupyter-JSC




**IPyVolume**

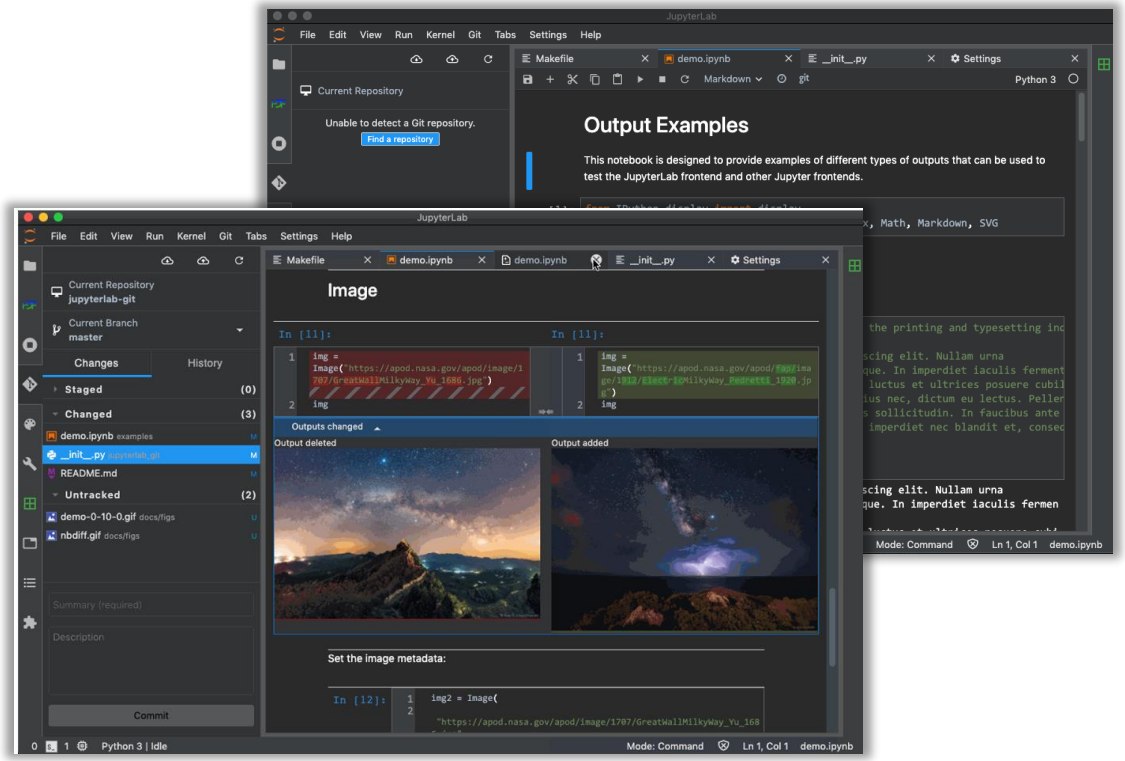
3d plotting for Python in the Jupyter notebook based on IPython widgets using WebGL




https://github.com/maartenbreddels/ipyvolume

**JupyterLab-Git**

JupyterLab extension for version control using Git

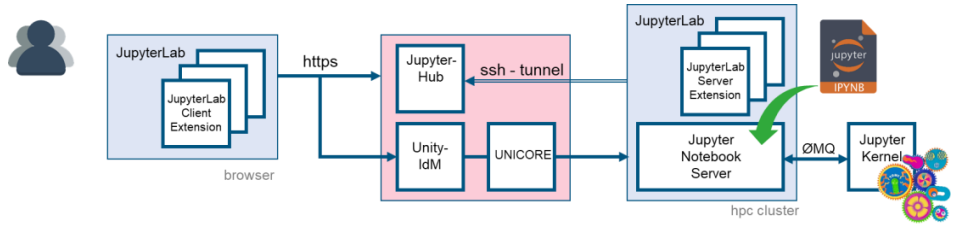



https://github.com/jupyterlab/jupyterlab-git

JÜLICH Forschungszentrum

# JUPYTERLAB EXTENSIONS

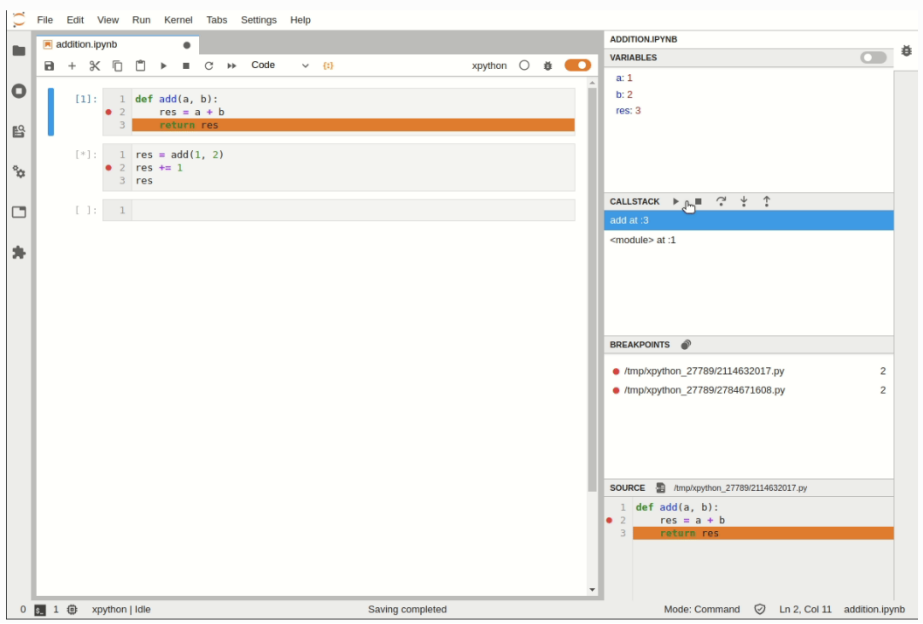
## Installed by default at Jupyter-JSC




**JupyterLab - Visual Debugger**

JupyterLab >= 3 ships with a Debugger front-end by default.

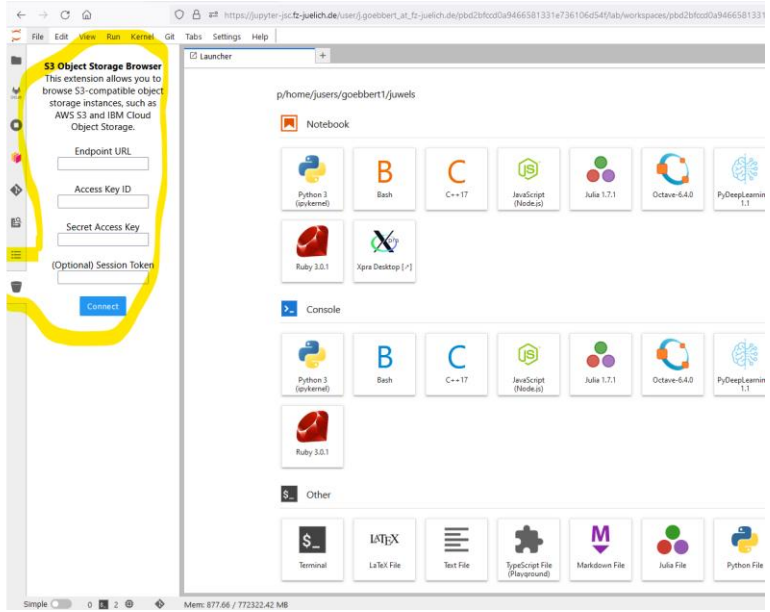
This means that notebooks, code consoles and files can now be debugged from JupyterLab directly! For the debugger to be enabled and visible, a kernel with support for debugging is required.




https://jupyterlab.readthedocs.io/en/stable/user/debugger.html

**JupyterLab-S3-browser**

A JupyterLab extension for browsing S3-compatible object storage




https://github.com/IBM/jupyterlab-s3-browser

JÜLICH Forschungszentrum

# JUPYTERLAB EXTENSIONS

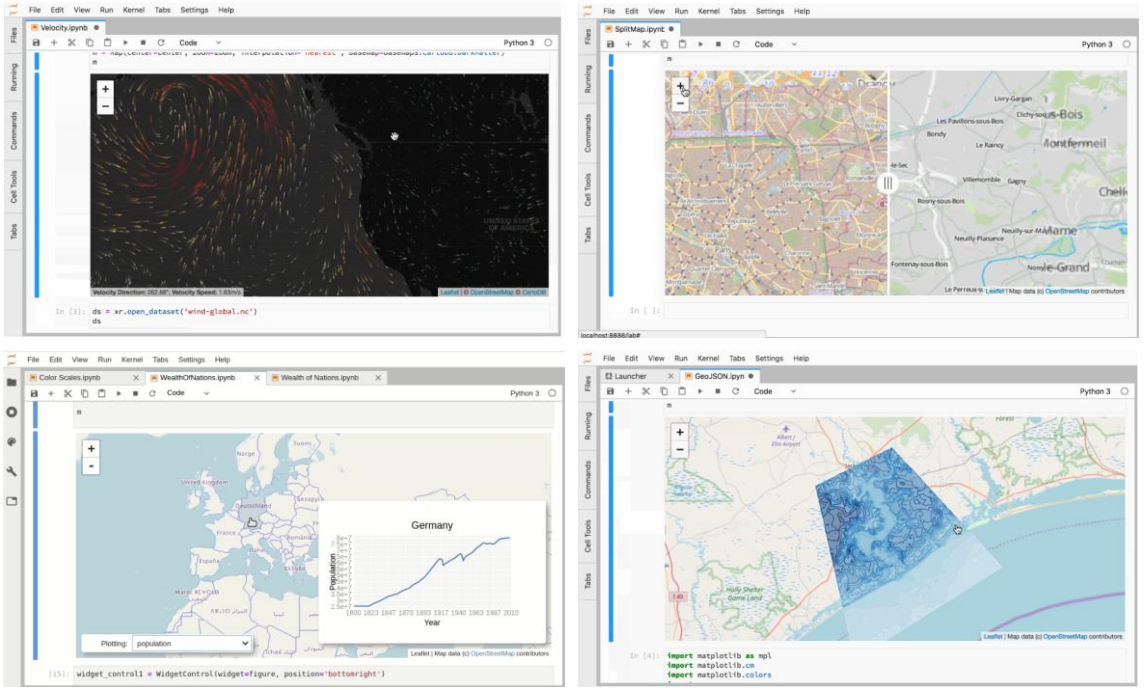
## Installed by default at Jupyter-JSC

**PyThreeJS**
A Python / ThreeJS bridge utilizing the Jupyter widget infrastructure.
https://threejs.org - lightweight, 3D library with a default WebGL renderer.

```
In [9]: f = """
function f(origu,origv) {
    // scale u and v to the ranges I want: [0, 2*pi]
    var u = 2*Math.PI*origu;
    var v = 2*Math.PI*origv;

    var x = Math.sin(u);
    var y = Math.cos(v);
    var z = Math.cos(u+v);

    return new THREE.Vector3(x,y,z)
}
"""
surf_g = ParametricGeometry(func=f);

surf = Mesh(geometry=surf_g, material=LambertMaterial(color='green', side='FrontSide'))
surf2 = Mesh(geometry=surf_g, material=LambertMaterial(color='yellow', side='BackSide'))
scene = Scene(children=[surf, surf2, AmbientLight(color='#777777')])
c = PerspectiveCamera(position=[5, 5, 3], up=[0, 0, 1],
                      children=[DirectionalLight(color='white',
                                                 position=[3, 5, 1],
                                                 intensity=0.6)])
renderer = Renderer(camera=c, scene=scene, controls=[OrbitControls(controlling=c)])
display(renderer)
```

https://github.com/jupyter-widgets/pythreejs

**IPyLeaflet**
A Jupyter / Leaflet bridge enabling interactive maps in the Jupyter notebook.



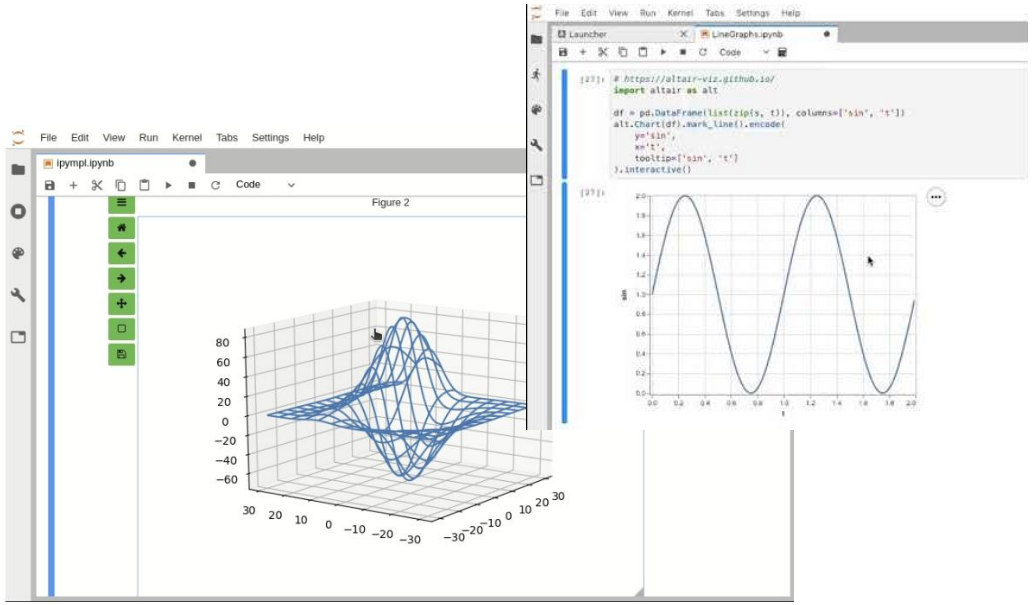

https://github.com/jupyter-widgets/ipyleaflet

JÜLICH
Forschungszentrum

# JUPYTERLAB EXTENSIONS

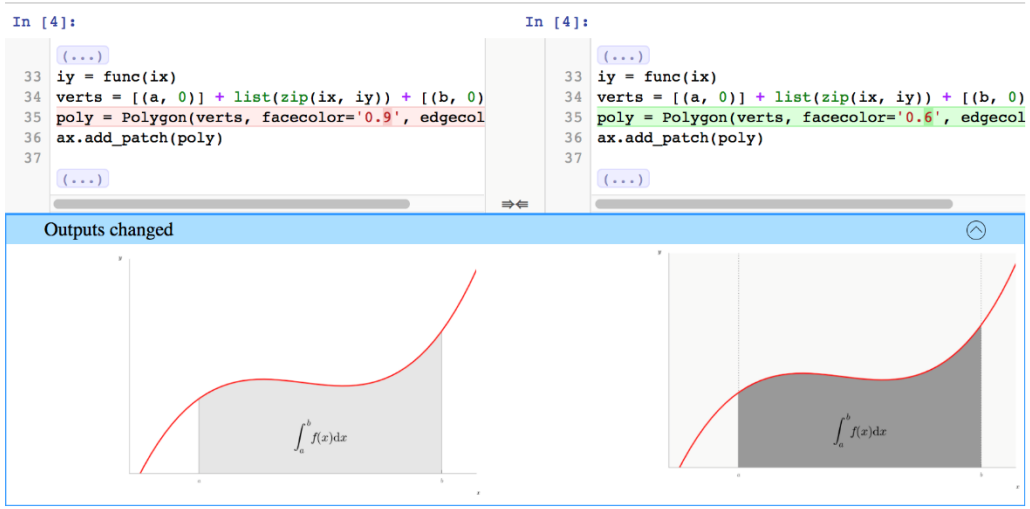
## Installed by default at Jupyter-JSC

**IPyMPL - matplotlib**
Leveraging the Jupyter interactive widgets framework, ipympl enables the interactive features of matplotlib in the Jupyter notebook and in JupyterLab.




https://github.com/matplotlib/ipympl

**NBDime**
Tools for diffing and merging of Jupyter notebooks.



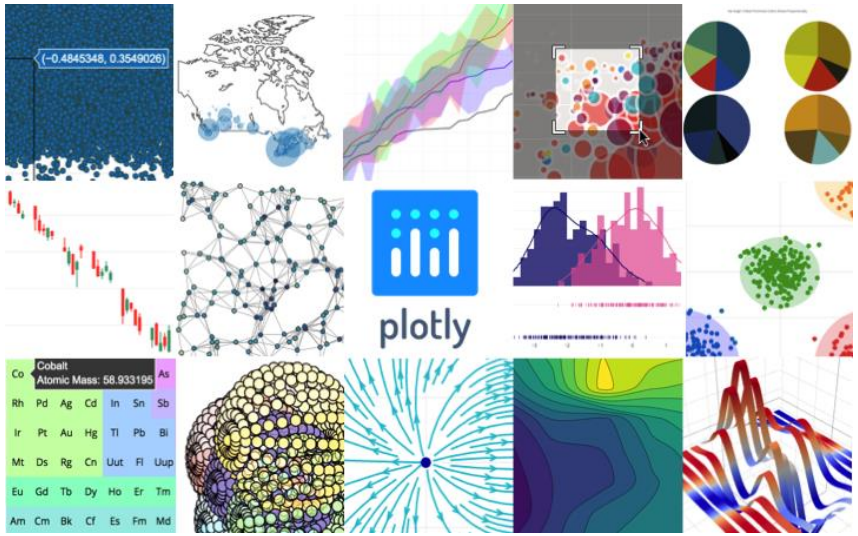

https://github.com/jupyter/nbdime

JÜLICH Forschungszentrum

# JUPYTERLAB EXTENSIONS

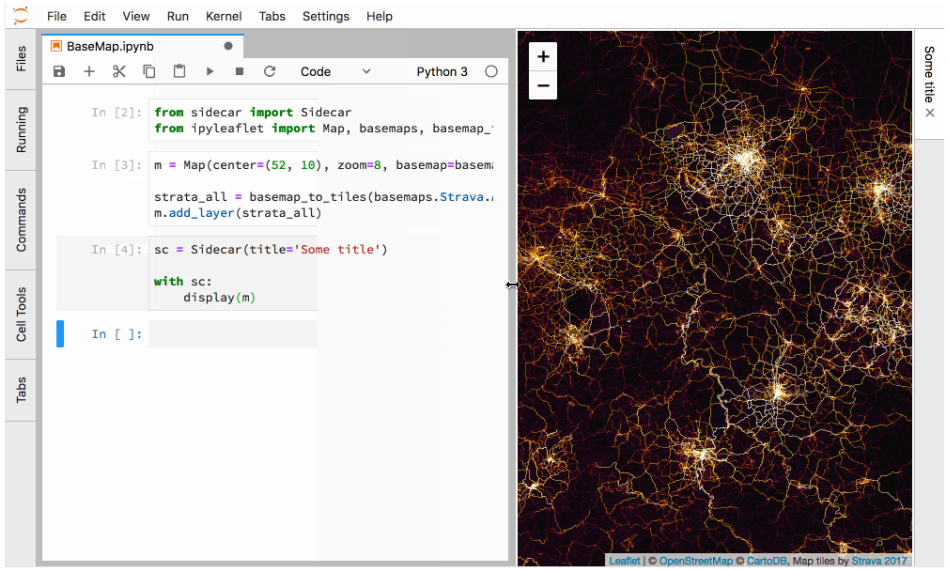
## Installed by default at Jupyter-JSC

**Plotly**
JupyterLab extension for the interactive and browser-based graphing library Plotly.
https://plotly.com/python/




https://github.com/plotly/plotly.py

**JupyterLab-Sidecar**
A sidecar output widget for JupyterLab.

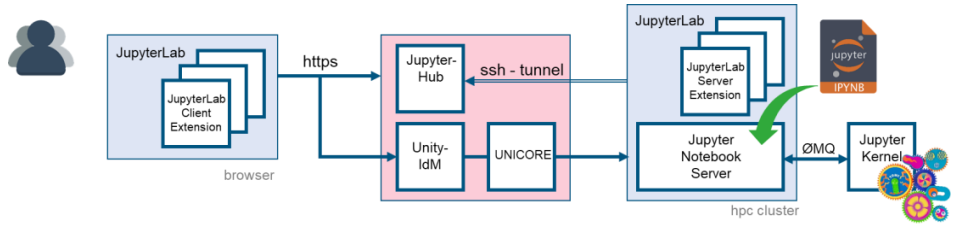



https://github.com/jupyter-widgets/jupyterlab-sidecar

JÜLICH
Forschungszentrum

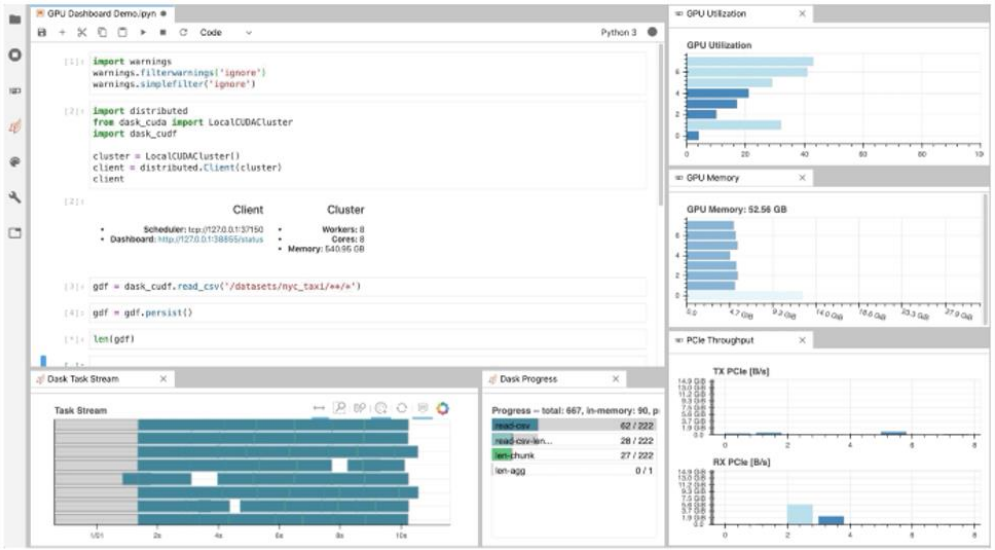
# JUPYTERLAB EXTENSIONS

## Installed by default at Jupyter-JSC




**NVDashboard**

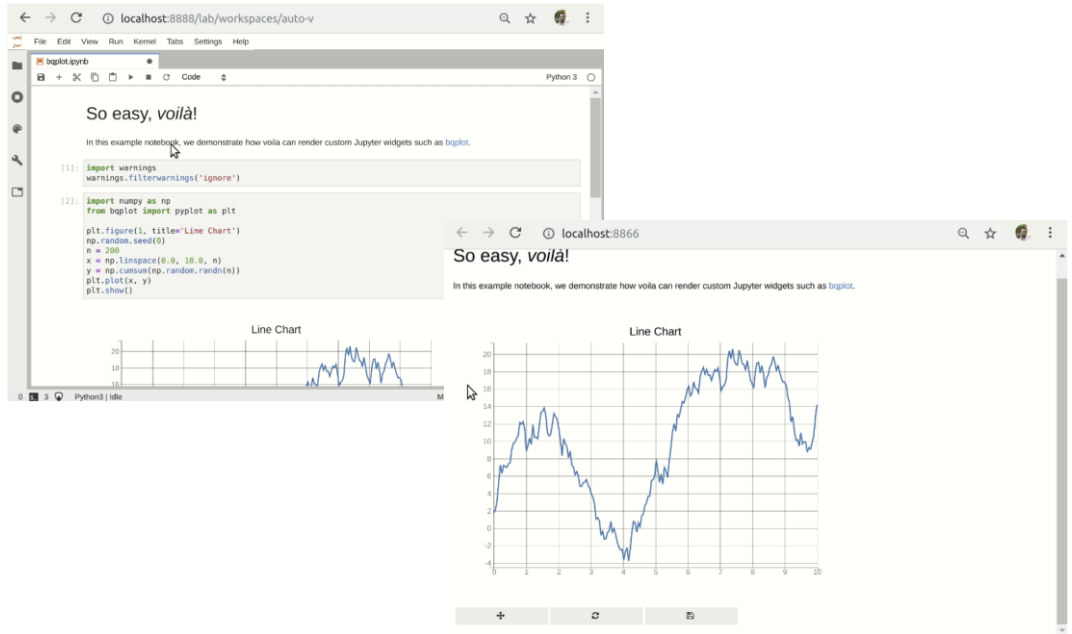
NVDashboard is an open-source package for the real-time visualization of NVIDIA GPU metrics in interactive Jupyter Lab environments.




https://github.com/rapidsai/jupyterlab-nvdashboard
https://developer.nvidia.com/blog/gpu-dashboards-in-jupyter-lab/

**Voilà**

Voilà turns Jupyter notebooks into standalone web applications.




https://github.com/voila-dashboards/voila

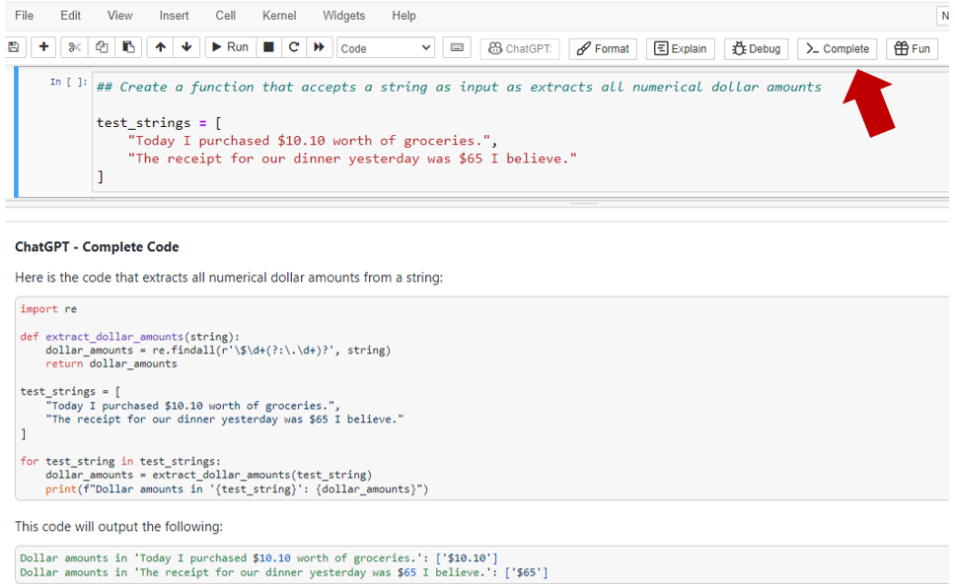
JÜLICH Forschungszentrum

# JUPYTERLAB EXTENSIONS

## … more useful extensions

**ChatGPT for Jupyter**

A browser extension to provide various helper functions in Jupyter Notebooks and Jupyter Lab, powered by ChatGPT.




https://github.com/TiesdeKok/chat-gpt-jupyter-extension

JÜLICH Forschungszentrum

# JUPYTERLAB EXTENSIONS

## Installed by default at Jupyter-JSC

| Core packages | Version | Link | Description |
|---|---|---|---|
| jupyterlab | 3.6.5 | https://jupyterlab.readthedocs.io/en/3.6.x/ | - |
| nbclassic | 1.0.0 | https://nbclassic.readthedocs.io | Jupyter Notebook as a Jupyter Server extension |
| jupyterlab_server | 2.23.0 | https://jupyterlab-server.readthedocs.io | Server components for JupyterLab applications |
| jupyterhub | 3.1.1 | https://jupyterhub.readthedocs.io | Multi-user server for Jupyter notebooks |

| Optional extension | Version | Link | Description |
|---|---|---|---|
| jupyterlab-nvdashboard | 0.8.0 | https://github.com/rapidsai/jupyterlab-nvdashboard , eb-file | A JupyterLab extension for displaying dashboards of GPU usage. |
| jupyter-slurm-provisioner | 0.6.0 | https://github.com/FZJ-JSC/jupyter-slurm-provisioner , eb-file | Allows to start Jupyter kernels as a SLURM job remote from the Jupyter server |
| nglview | 3.0.6 | http://nglviewer.org/nglview/latest/ , eb-file | Jupyter widget to interactively view molecular structures and trajectories |
| jupyter-ai | 0.9.0 | https://jupyter-ai.readthedocs.io/ , eb-file | A generative AI extension for JupyterLab |

| Core Kernels | Version | Link | Description |
|---|---|---|---|
| Bash | 0.9.0 | https://github.com/takluyver/bash_kernel , eb-file | A bash kernel for IPython |
| Cling (C++) | 20230205 | https://github.com/root-project/cling/ , eb-file | Jupyter kernel for the C++ programming language |
| Julia | 1.8.5 | https://github.com/JuliaPy/pyjulia , eb-file | python interface to julia |
| LFortran | 0.19.0 | https://lfortran.org/ , eb-file | Modern interactive LLVM-based Fortran compiler |
| Octave | 8.2.0 | https://www.octave.org/ , eb-file | Scientific Programming Language - Powerful mathematics-oriented syntax with built-in 2D/3D plotting and visualization tools |
| R | 4.2.1 | https://irkernel.github.io , eb-file | R kernel for Jupyter |
| Ruby | 3.0.5 | https://github.com/SciRuby/iruby , eb-file | Ruby kernel for Jupyter |

| Community Kernels | Version | Link | Description |
|---|---|---|---|
| DeepLearning | 2023.5 | eb-file | Python kernel incl. a collection of extra modules/packages for Deep Learning |
| PyEarthSystem | 2023.5 | eb-file | Python kernel incl. a collection of extra modules/packages for the Earth System community |
| QuantumComputing | 2023.5 | eb-file | Python kernel incl. a collection of extra modules/packages for the quantum computing community |
| Visualization | 2023.5 | eb-file | Python kernel incl. a collection of extra modules/packages for visualization |

| JupyterLab Applications | Version | Link | Description |
|---|---|---|---|
| Xpra | 4.4.6 | https://xpra.org eb-file | Remote desktop for X11 applications in the browser |

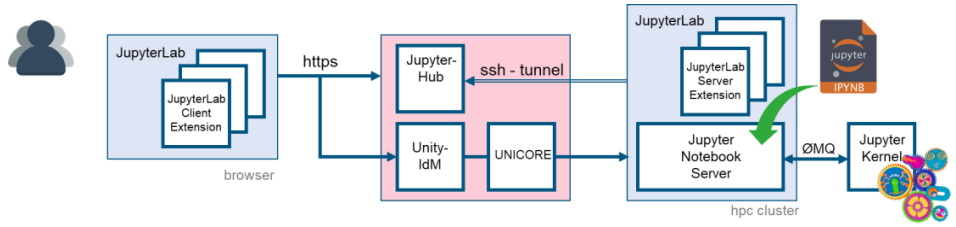
| Core Extensions | Version | Link | Description |
|---|---|---|---|
| jupyter-server-proxy | 4.0.0 | https://jupyter-server-proxy.readthedocs.io | Jupyter notebook server extension to proxy web services. |
| jupyterlab-lsp | 4.2.0 | https://github.com/jupyter-lsp/jupyterlab-lsp | Coding assistance for JupyterLab using Language Server Protocol |
| ipympl | 0.9.3 | https://matplotlib.org/ipympl/ | Interactive features of matplotlib in Jupyter |
| ipyleaflet | 0.17.3 | https://ipyleaflet.readthedocs.io | Interactive maps in the Jupyter notebook |
| bqplot | 0.12.39 | https://bqplot.github.io/bqplot/ | Plotting library for IPython/Jupyter notebooks |
| ipyvolume | 0.6.3 | https://github.com/widgetti/ipyvolume | 3d plotting for Python in the Jupyter notebook based on IPython widgets using WebGL |
| jupyterlab_gitlab | 3.0.0 | https://gitlab.com/beenje/jupyterlab-gitlab | A JupyterLab extension for browsing GitLab repositories |
| jupyterlab_git | 0.41.0 | https://github.com/jupyterlab/jupyterlab-git | A Git extension for JupyterLab |
| nbdime | 3.2.1 | https://nbdime.readthedocs.io/ | Tools for diffing and merging of Jupyter notebooks. |
| jupyterlab_latex | 3.2.0 | https://github.com/jupyterlab/jupyterlab-latex | JupyterLab extension for live editing of LaTeX documents |
| jupyterlab_s3_browser | 0.13.0 | https://github.com/IBM/jupyterlab-s3-browser | A JupyterLab extension for browsing S3-compatible object storage |
| plotly | 5.15.0 | https://plotly.com/python/ | Python graphing library for interactive, publication-quality graphs. |
| jupyter_bokeh | 3.0.4 | https://github.com/bokeh/jupyter_bokeh | An extension for rendering Bokeh content in JupyterLab notebooks |
| panel | 0.14.4 | https://panel.holoviz.org/ | The powerful data exploration & web app framework for Python |
| holoviews | 1.16.0 | https://holoviews.org/ | With Holoviews, your data visualizes itself. |
| jupyterlab_hdf | 1.3.0 | https://github.com/jupyterlab/jupyterlab-hdf5 | Open and explore HDF5 files in JupyterLab. Can handle very large (TB) sized files, and datasets of any dimensionality |
| ipyparallel | 8.6.1 | https://ipyparallel.readthedocs.io | IPython Parallel: Interactive Parallel Computing in Python |
| dask_labextension | 6.1.0 | https://github.com/dask/dask-labextension | JupyterLab extension for Dask |
| voila | 0.5.0a4 | https://voila.readthedocs.io | Voilà turns Jupyter notebooks into standalone web applications |
| nbdev | 2.3.12 | https://nbdev.fast.ai/ | Create delightful software with Jupyter Notebooks |
| sidecar | 0.5.2 | https://github.com/jupyter-widgets/jupyterlab-sidecar | A sidecar output widget for JupyterLab |
| dash | 2.11.1 | https://plotly.com/dash | Data Apps & Dashboards for Python. No JavaScript Required. |
| ipyvue | 1.9.2 | https://github.com/widgetti/ipyvue | Jupyter widgets base for Vue libraries |
| ipywebrtc | 0.6.0 | https://github.com/maartenbreddels/ipywebrtc | WebRTC for Jupyter notebook/lab |
| jupyterlab-spellchecker | 0.7.3 | https://github.com/jupyterlab-contrib/spellchecker | Spellchecker for JupyterLab notebook markdown cells and file editor. |
| jupyterlab_code_formatter | 1.6.1 | https://github.com/ryantam626/jupyterlab_code_formatter | A JupyterLab plugin to facilitate invocation of code formatters. |
| jupyterlab_recents | 3.2.0 | https://github.com/NERSC/jupyterlab-recents | A JupyterLab extension that tracks recent files and directories. |
| jupyterlab-favorites | 3.1.1 | https://github.com/NERSC/jupyterlab-favorites | Add the ability to save favorite folders to JupyterLab for quicker browsing |
| jupyterlab-system-monitor | 0.8.0 | https://github.com/jtpio/jupyterlab-system-monitor | JupyterLab extension to display system metrics |
| jupyterlab_iframe | 0.4.4 | https://github.com/timkpaine/jupyterlab_iframe | View html as an embedded iframe in JupyterLab |
| jupyterlab-tour | 3.1.4 | https://github.com/jupyterlab-contrib/jupyterlab-tour | A JupyterLab UI tour built on jupyterlab-tutorial and react-joyride. |
| papermill | 2.4.0 | https://papermill.readthedocs.io | Parameterize, execute, and analyze notebooks |
| pyunicore | 0.15.0 | https://github.com/HumanBrainProject/pyunicore | UNICORE REST bindings for python |

https://docs.jupyter-jsc.fz-juelich.de/github/FZJ-JSC/jupyter-jsc-notebooks/blob/documentation/05-News%26Updates/Announcement-2023-07_JupyterLab3.6-Upgrade.ipynb#JupyterLab-specific-Packages

# JUPYTER KERNEL

# JUPYTER KERNEL

## How to create your own Juypter Kernel

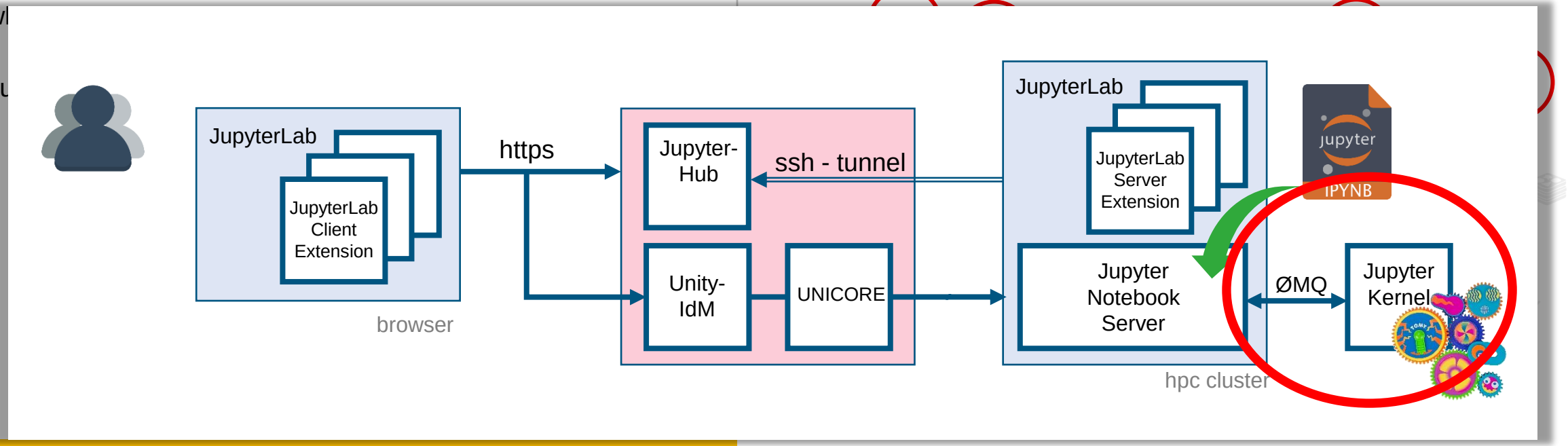



**Jupyter Kernel**

A “kernel” refers to the separate process wh

Ju

- 
- 
- 
- 



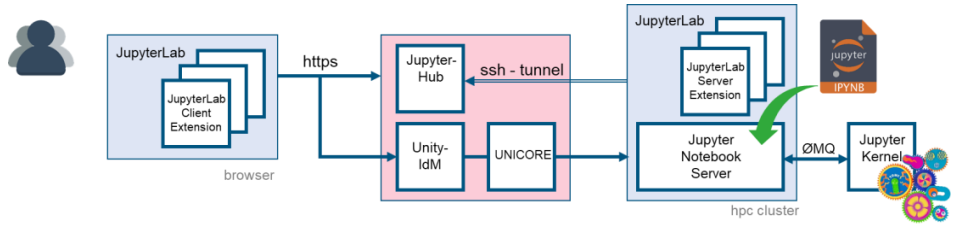

You can easily **create your own kernel** which for example runs your specialized virtual Python environment.

https://github.com/jupyter/jupyter/wiki/Jupyter-kernels

JÜLICH Forschungszentrum

# JUPYTER KERNEL

## How to create your own Juypter Kernel



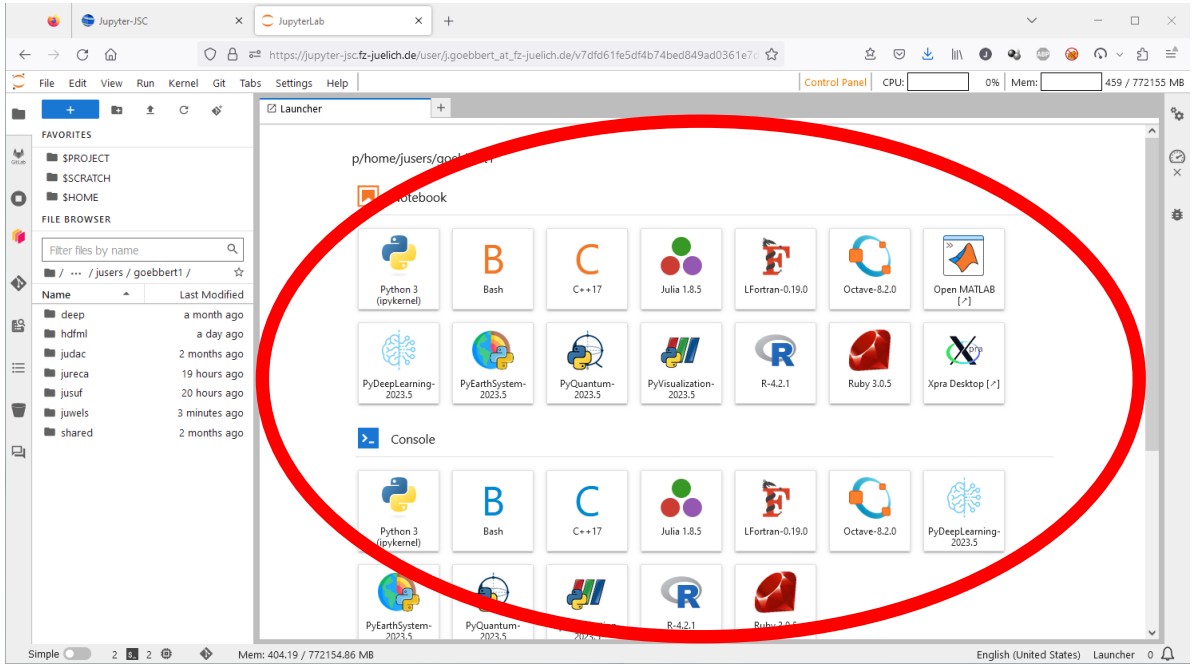

**Jupyter Kernel**

A “kernel” refers to the separate process which executes code cells within a Jupyter notebook.

Jupyter Kernel

- run code in different programming languages **and environments**.
- can be connected to a notebook (one at a time).
- communicates via ZeroMQ with the JupyterLab.

- Multiple **preinstalled** Jupyter Kernels can be found on our clusters
  - Python, R, Julia, Bash, C++, Ruby, JavaScript
  - Specialized kernels for visualization, quantum computing



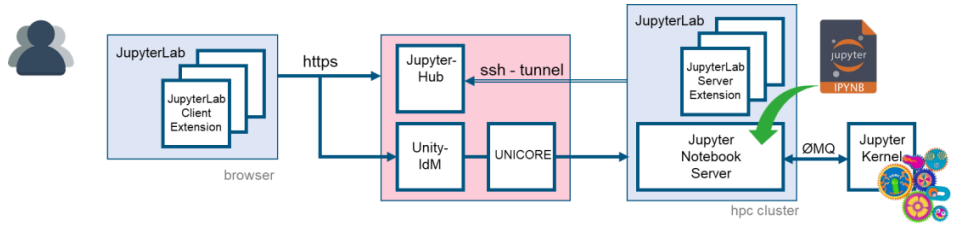

You can easily **create your own kernel** which for example runs your specialized virtual Python environment.

https://github.com/jupyter/jupyter/wiki/Jupyter-kernels

JÜLICH Forschungszentrum

# JUPYTER KERNEL

## How to create your own Juypter Kernel




**Jupyter Kernel**

A “kernel” refers to the separate process which executes code cells within a Jupyter notebook.

Jupyter Kernel

- run code in different programming languages **and environments**.
- can be connected to a notebook (one at a time).
- communicates via ZeroMQ with the JupyterLab.

- Multiple **preinstalled** Jupyter Kernels can be found on our clusters
  - Python, R, Julia, Bash, C++, Ruby, JavaScript
  - Specialized kernels for visualization, quantum computing

You can easily **create your own kernel** which for example runs your specialized virtual Python environment.

**Building your own Jupyter kernel is a three step process**

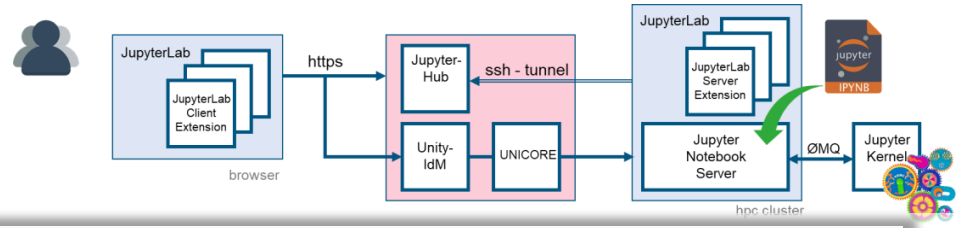
1. Create/Pimp new **virtual Python environment**
   `venv`
2. Create/Edit **launch script** for the Jupyter kernel
   `kernel.sh`
3. Create/Edit Jupyter **kernel configuration**
   `kernel.json`

https://github.com/jupyter/jupyter/wiki/Jupyter-kernels

JÜLICH Forschungszentrum

# JUPYTER KERNEL

## How to create your own Juypter Kernel

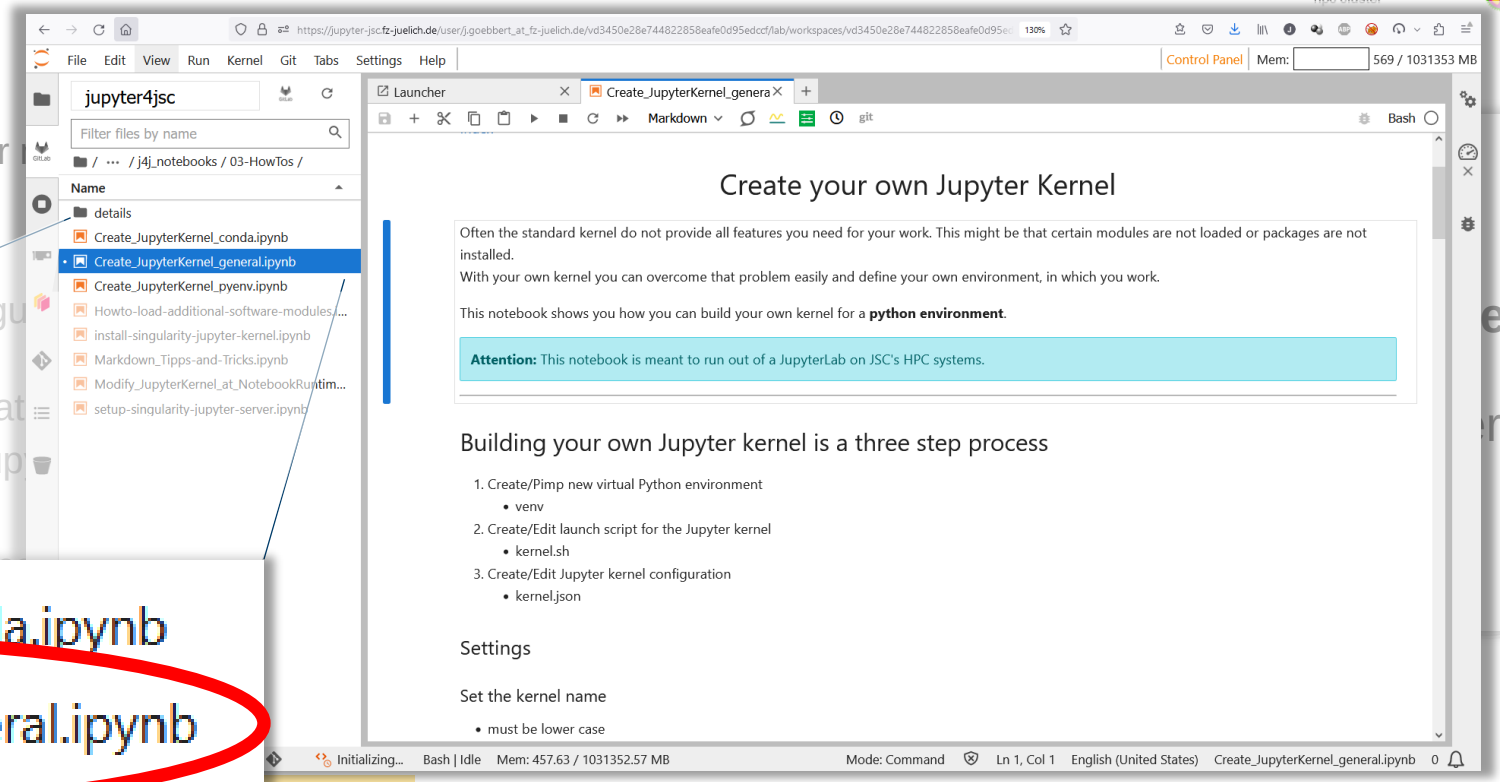


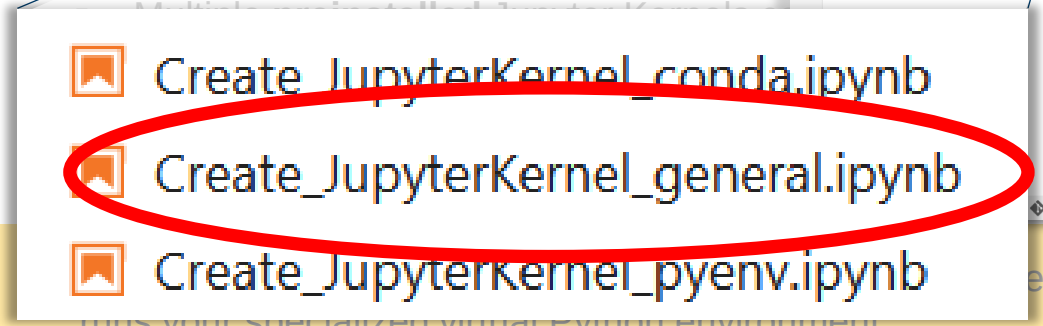

### Jupyter Kernel

A “kernel” refers to the separate process which executes code cells within a Jupyter

Jupyter Kernel

- run code in different programming langu **and environments**.
- can be connected to a notebook (one at
- communicates via ZeroMQ with the Jup




runs your specialized virtual Python environment.

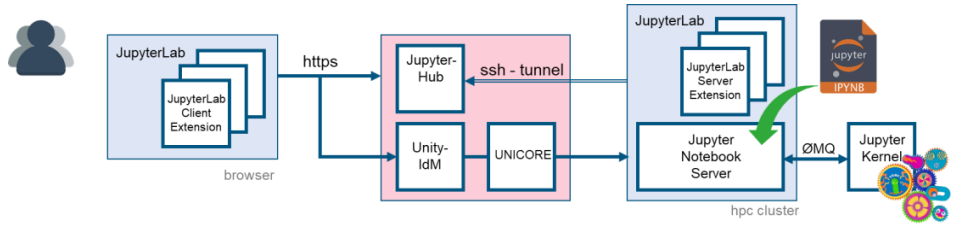
https://github.com/jupyter/jupyter/wiki/Jupyter-kernels

https://gitlab.version.fz-juelich.de/jupyter4jsc/j4j_notebooks/-/blob/master/001-Jupyter/Create_JupyterKernel_general.ipynb

JÜLICH Forschungszentrum

# JUPYTER KERNEL

## Run your Jupyter kernel configuration

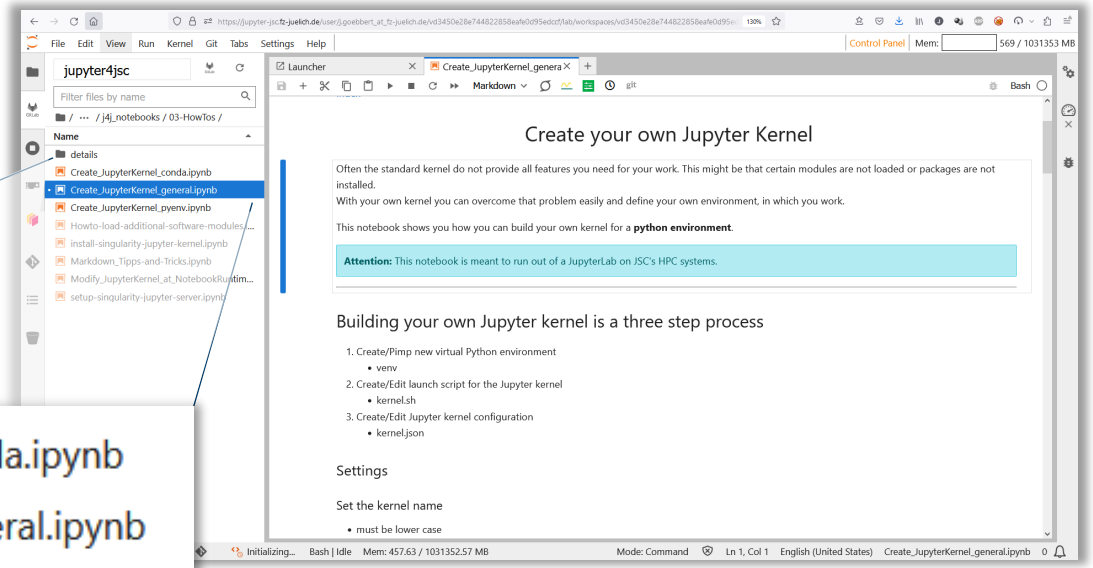


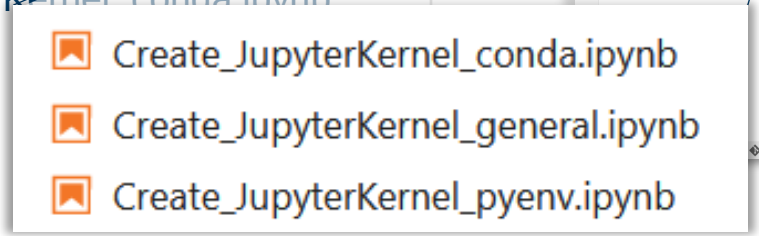

**Run your Jupyter Kernel**

1. https://jupyter-jsc.fz-juelich.de
2. Choose system where your Jupyter kernel is installed in `~/.local/share/jupyter/kernels`
3. Select your kernel in the launch pad or click the kernel name.

**One of the many alternatives: Conda**

Base your Jupyter Kernel on a Conda environment.
https://gitlab.version.fz-juelich.de/jupyter4jsc/j4j_notebooks/-/blob/master/001-Jupyter/Create_JupyterKernel_conda.ipynb




Jupyter kernel are **NOT limited** to Python at all!
The kernel-endpoint just needs to talk the Jupyter's kernel protocol (in general over ZeroMQ).
E.g.
- IRkernel for R (https://github.com/IRkernel/IRkernel)
- IJulia.jl (https://github.com/JuliaLang/IJulia.jl)

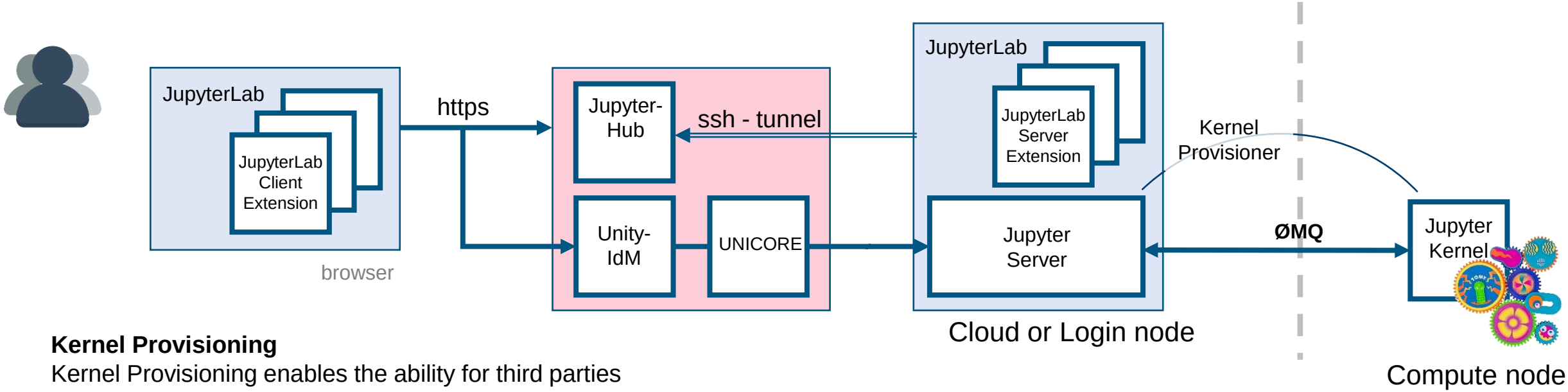
JÜLICH Forschungszentrum

# SLURM WRAPPED KERNELS WITH SLURM-PROVISIONER

JÜLICH
Forschungszentrum

# REMOTE JUPYTER KERNELS

## Running multiple Jupyter kernels separate on the HPC system




**Kernel Provisioning**
Kernel Provisioning enables the ability for third parties
to **manage the lifecycle of a kernel's runtime environment**.

By implementing and configuring a *kernel provisioner*,
third parties have the ability to **provision kernels for different environments**,
typically managed by resource managers like Kubernetes, Hadoop YARN, Slurm, etc.

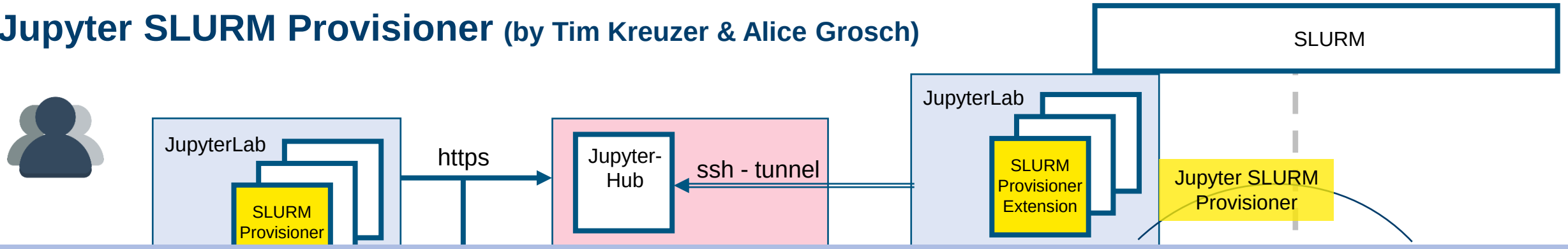
The kernel provisioner optionally extends the current **metadata stanza within the kernel.json**
to include the specification of the kernel provisioner name, along with an optional config stanza

```
[..]
"metadata": {
    "kernel_provisioner": {
        "provisioner_name": "slurm-provisioner",
        "config": {
            "kernel_argv": "Python",
            "project": "zam",
            "partition": "batch",
            "nodes": 1,
            "runtime": 3600,
        }
    }
},
```

https://jupyter-client.readthedocs.io/en/stable/provisioning.html

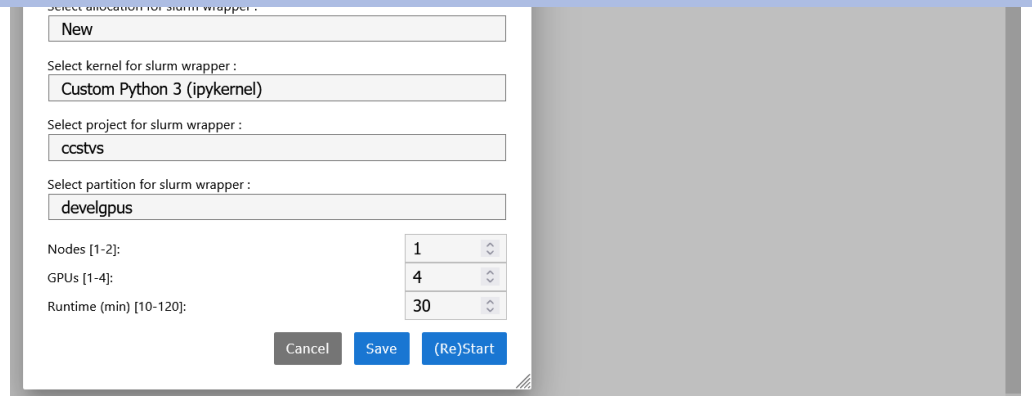
# REMOTE JUPYTER KERNELS

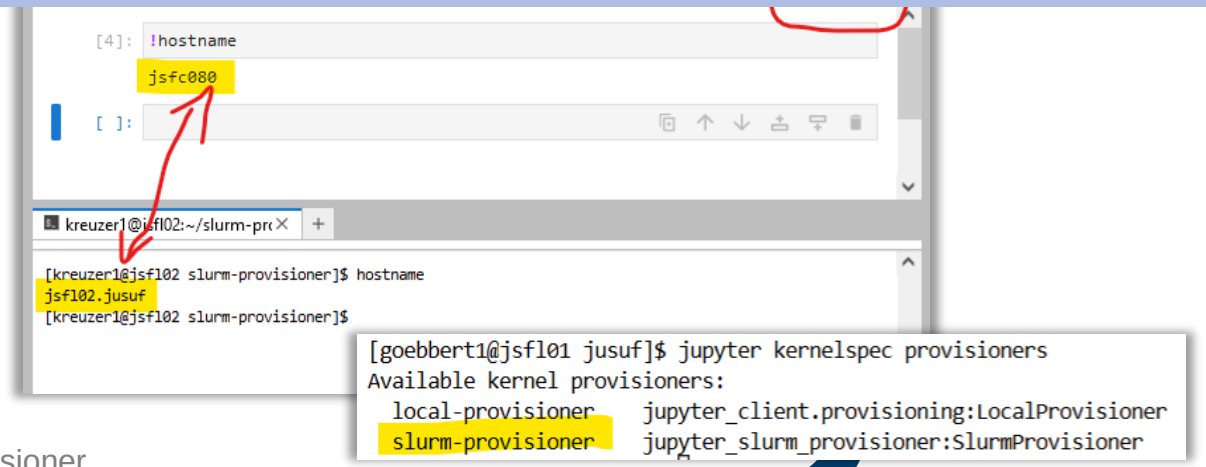
## Jupyter SLURM Provisioner (by Tim Kreuzer & Alice Grosch)




**Slurm wrapped kernels allow you to run kernels on compute nodes while your Jupyter Server runs on a login node.**

This has the advantage that when your allocation on the compute node(s) ends, **only the kernel is stopped**, but your JupyterLab server keeps running. You will only have to restart the kernel, not your entire JupyterLab instance.

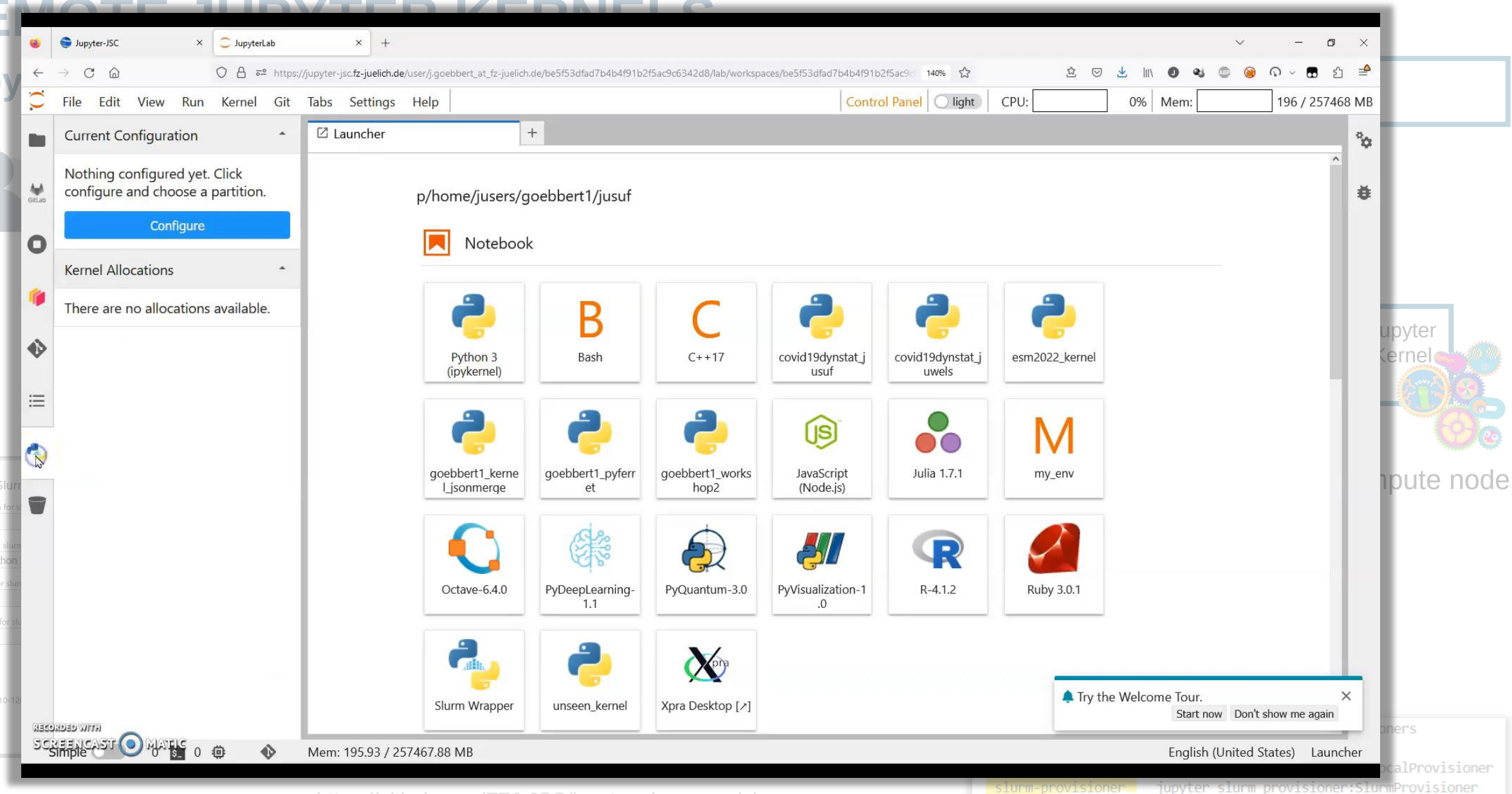



https://github.com/FZJ-JSC/jupyter-slurm-provisioner
https://github.com/FZJ-JSC/jupyter-slurm-provisioner-extension

# REMOTE JUPYTER KERNELS

Jupy




upyter
Kernel

npute node

https://github.com/FZJ-JSC/jupyter-slurm-provisioner
https://github.com/FZJ-JSC/jupyter-slurm-provisioner-extension

slurm-provisioner jupyter_slurm_provisioner:SlurmProvisioner

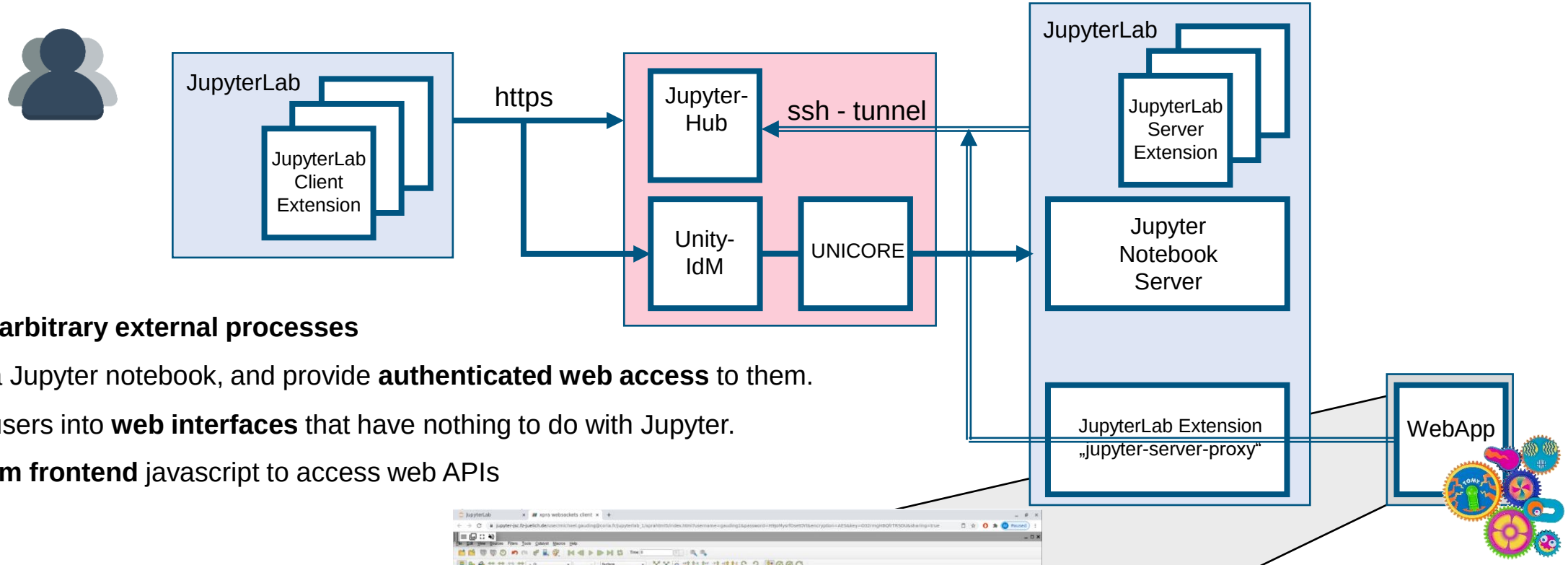
Forschungszentrum

# JUPYTER SERVER PROXY

JÜLICH
Forschungszentrum

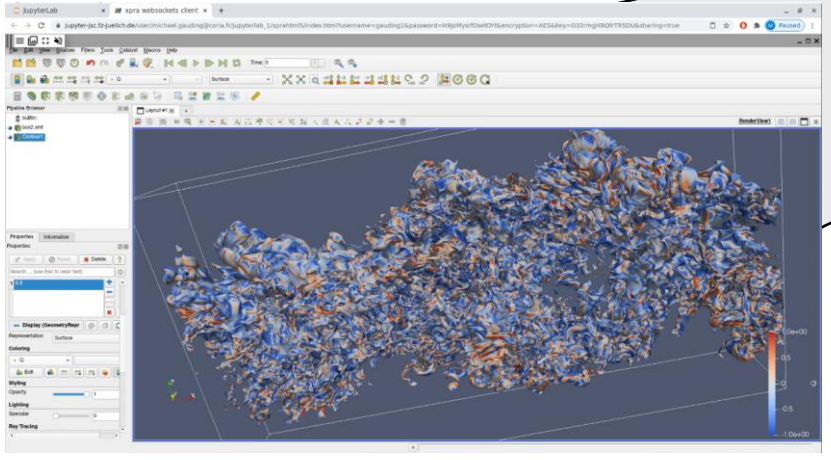
# JUPYTERLAB – WEBSERVICE PROXY

## Extension: jupyter-server-proxy




Allows to run **arbitrary external processes**

- alongside a Jupyter notebook, and provide **authenticated web access** to them.
- launching users into **web interfaces** that have nothing to do with Jupyter.
- **access from frontend** javascript to access web APIs



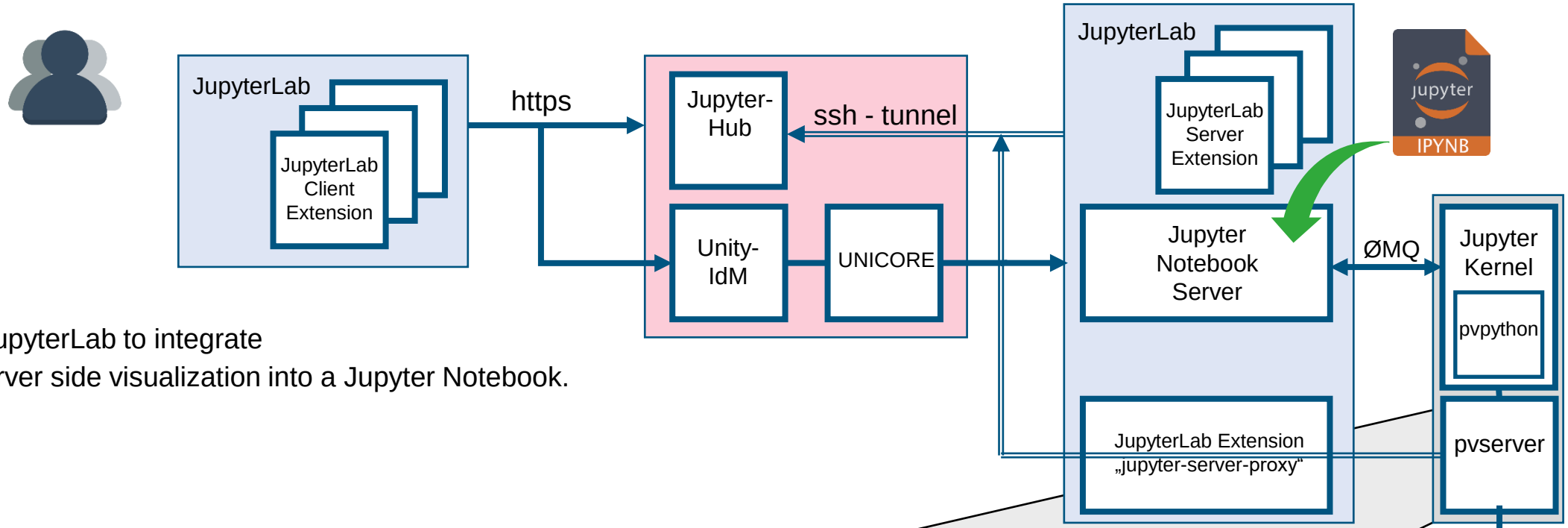
Turbulent mixing with variable density,
subset of 1939x600x3584 grid points, Michael Gauding, CORIA

https://github.com/jupyterhub/jupyter-server-proxy

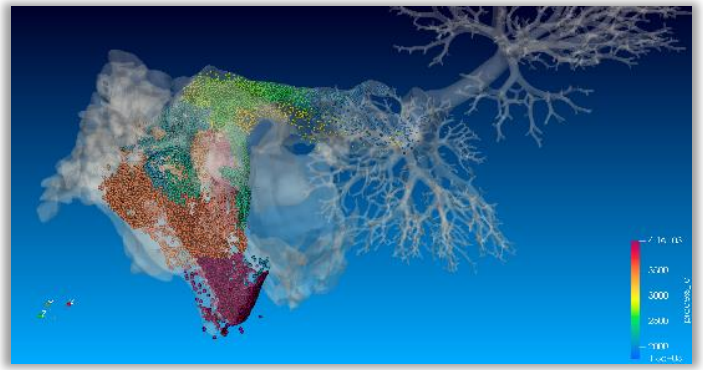
JÜLICH Forschungszentrum

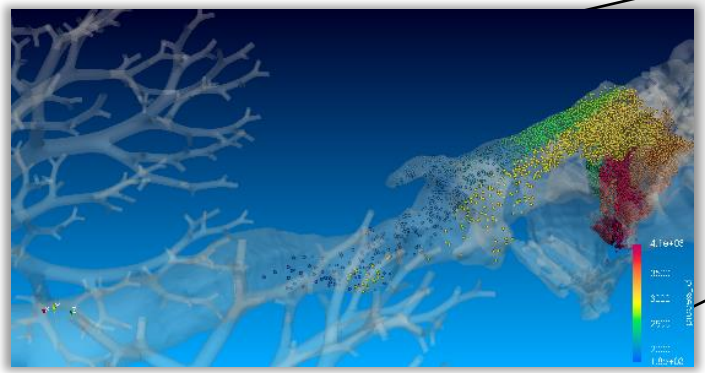
# JUPYTERLAB – WEBSERVICE PROXY

## Extension: jupyter-server-proxy




How to use JupyterLab to integrate
interactive server side visualization into a Jupyter Notebook.

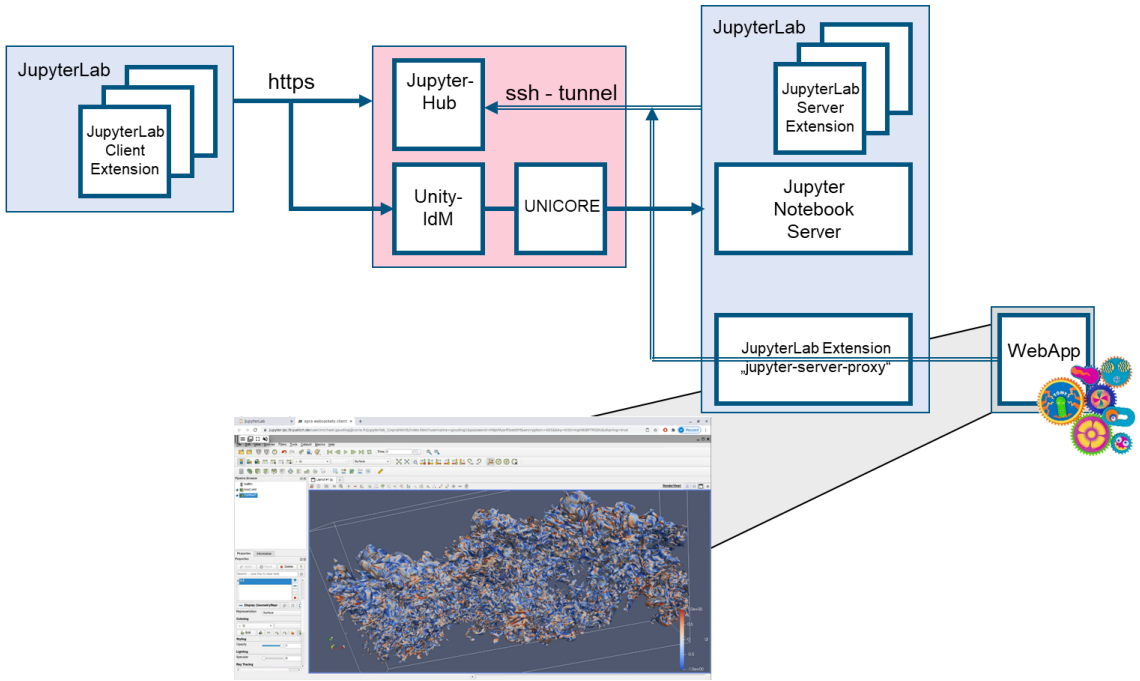




JÜLICH Forschungszentrum

# PORT TUNNELING – WEBSERVICE PROXY

## Extension: jupyter-server-proxy




**Accessing Arbitrary Ports or Hosts from the Browser**

If you have a web-server running on the server
listening on <port>, you can access it through the notebook at
**<notebook-base>/proxy/<port>**
The URL will be rewritten to remove the above prefix.

You can disable URL rewriting by using
**<notebook-base>/proxy/absolute/<port>**
so your server will receive the full URL in the request.

This works for all ports listening on the local machine.

**Example:**

https://jupyter-jsc.fz-juelich.de/user/j.goebbert@fz-juelich.de/juwels_login/**proxy/<port>**
https://jupyter-jsc.fz-juelich.de/user/j.goebbert@fz-juelich.de/juwels_login/**proxy/<host>:<port>**

**Upcoming:** Support proxying to a server process via a Unix socket (#337)

https://jupyter-server-proxy.readthedocs.io/en/latest/arbitrary-ports-hosts.html

JÜLICH Forschungszentrum

# JUPYTER SERVER PROXY EXAMPLES

# JUPYTERLAB – REMOTE DESKTOP

## Run your X11-Applications in the browser

**Jupyter-JSC gives you easy access to a remote desktop**

1. https://jupyter-jsc.fz-juelich.de
2. Click on “Xpra”

**Xpra - X Persistent Remote Applications**
is a tool which runs X clients on a remote host and directs their display to the local machine.

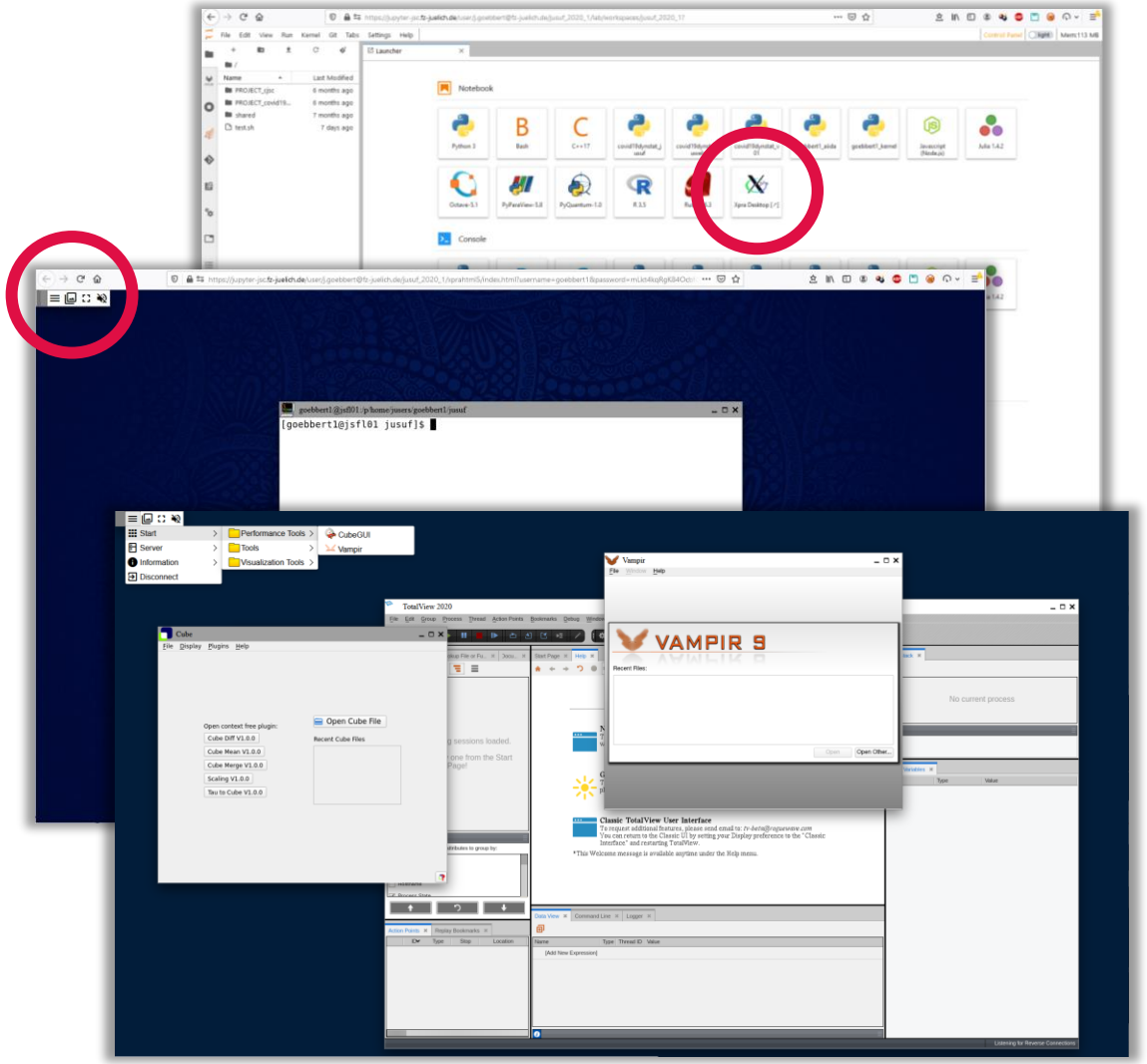
- Runs in a browser
- allows dis-/reconnection without disrupting the forwarded application
- https://xpra.org

**The remote desktop will run on the same node as your JupyterLab does (this includes compute nodes).**

**It gets killed, when you stop your JupyterLab session.**

**Hint:**

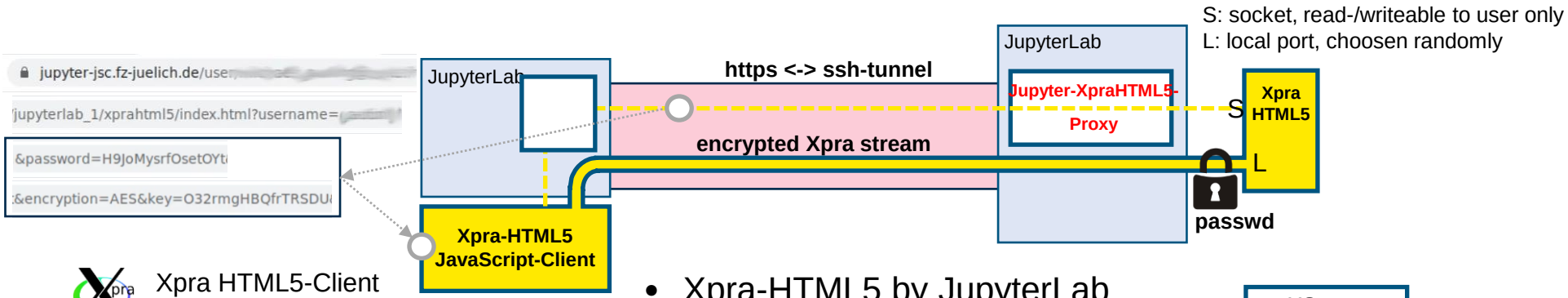
- `CTRL + C -> CTRL + Insert`
- `CTRL + V -> SHIFT + Insert`




JÜLICH
Forschungszentrum

# JUPYTERLAB – REMOTE DESKTOP

## Run your X11-Applications in the browser




- Xpra-HTML5 by JupyterLab
  - Access through **JupyterLab-URL** by configurable https proxy
  - Auto-generated **one-time password & encryption key** is communicated through https-proxy

XServer (dummy driver)

https://github.com/FZJ-JSC/jupyter-xprahtml5-proxy

JÜLICH Forschungszentrum

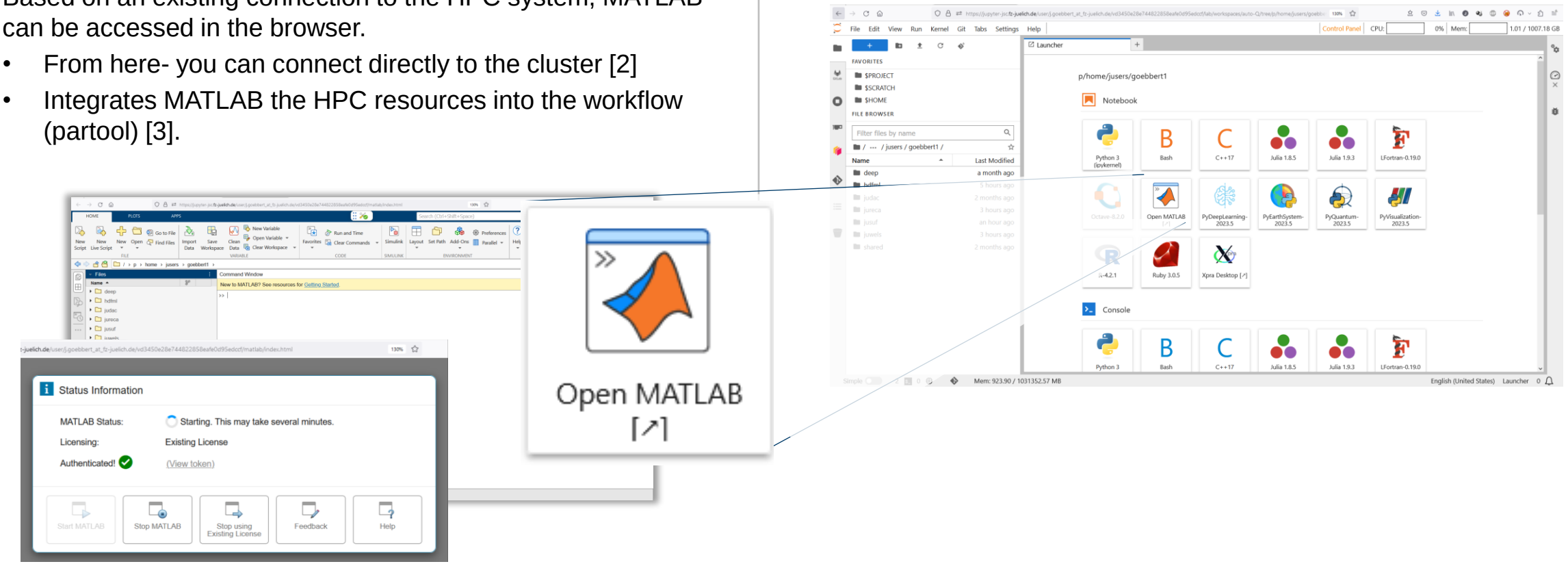
# JUPYTERLAB – MATLAB

## Web-based GUI for MATLAB

**MATLAB – Web-based GUI**

Based on an existing connection to the HPC system, MATLAB can be accessed in the browser.

- From here- you can connect directly to the cluster [2]
- Integrates MATLAB the HPC resources into the workflow (partool) [3].




[1] https://www.fz-juelich.de/en/ias/jsc/services/user-support/software-tools/matlab
[2] https://de.mathworks.com/help/parallel-computing/remoteclusteraccess.html
[3] https://de.mathworks.com/products/parallel-computing.html

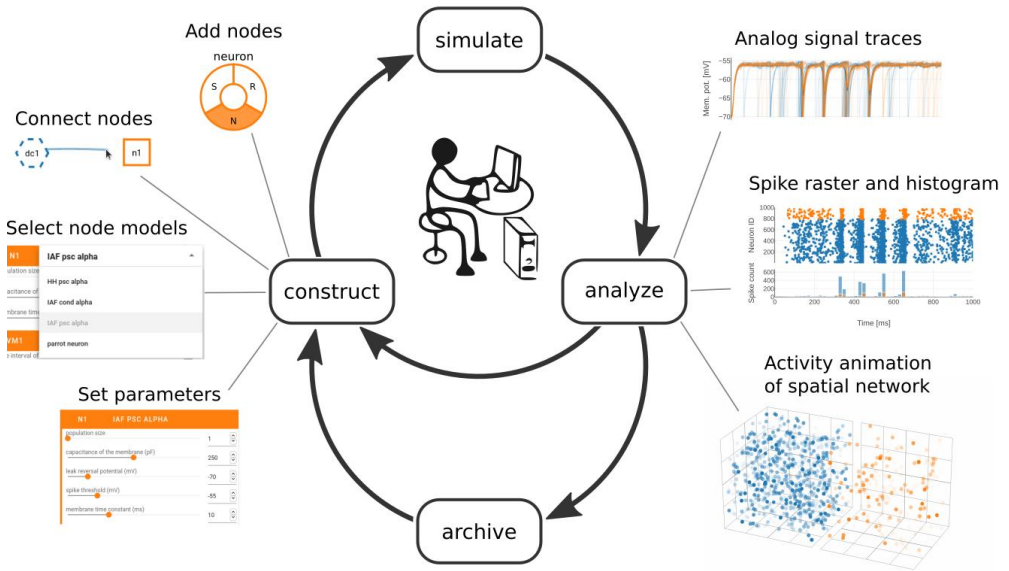
JÜLICH Forschungszentrum

# JUPYTERLAB – NEST DESKTOP

## Web-based GUI for Neuroscientists using NEST

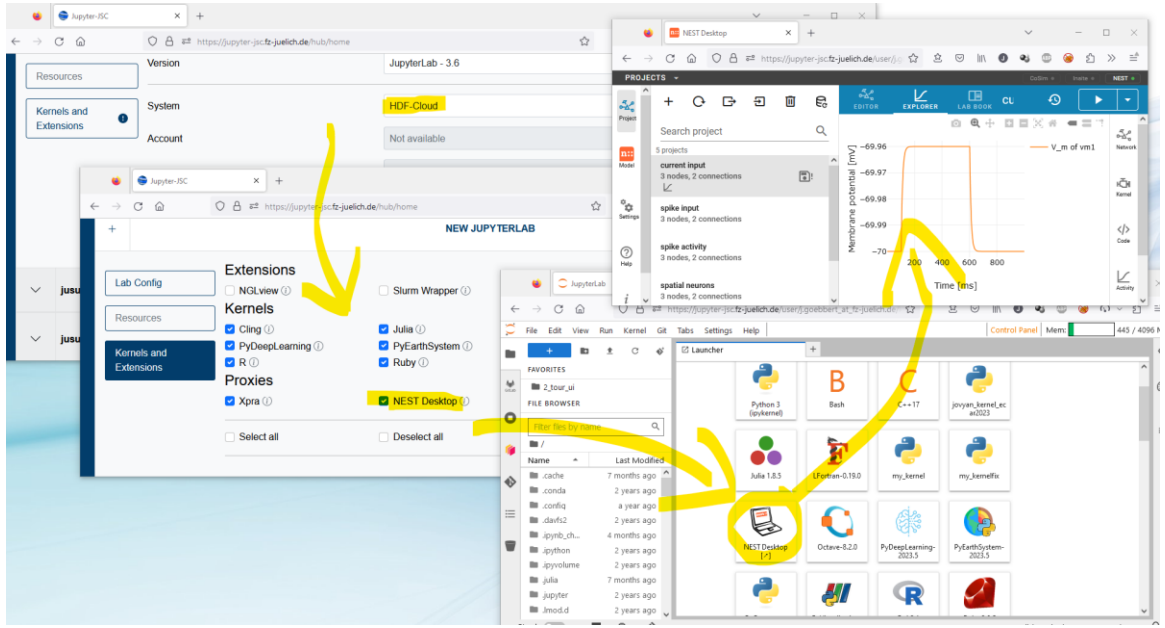
### NEST-Desktop

NEST Desktop is a web-based GUI application for NEST Simulator, an advanced simulation tool for the computational neuroscience.




### Jupyter-JSC gives you easy access to a NEST-Desktop

With Jupyter-JSC using Jupyter-Server-Proxy authenticated & authorized users get secure access to the WebUI of NEST-Desktop running NEST-simulations on HPC.




**Plugin for Jupyter-Server-Proxy: jupyter-xprahtml5-proxy**
https://github.com/jhgoebbert/jupyter-nestdesktop-proxy

[1] https://nest-desktop.readthedocs.io
[2] https://www.nest-simulator.org

# CONCLUSION

## Why Jupyter is so popular among Data Scientists

JupyterLab …

… is a **web-based platform for interactive computing and data analysis**
that is well-suited to the needs of research software engineers.

… provides researchers with a **comprehensive environment** for working with
code, text, multimedia, and data, making it an ideal tool for a wide range of research tasks.

… is designed to be **flexible and customizable**,
and can be modified to suit the specific needs and workflows of individual researchers.

… supports the creation of **reproducible research** through its support for Jupyter notebooks.

… supports **collaboration and sharing** of research work
through its support for sharing notebooks, dashboards, and other elements of a research project.

… provides a wide range of **extensions and plugins**
that can be used to integrate other tools and services into the environment.

… is an **open-source project**, which means that researchers
have access to the source code and can contribute to its development.

# QUESTIONS?


**More details:**
**https://gitlab.jsc.fz-juelich.de/jupyter4jsc/training-2023.04-jupyter4hpc**

JÜLICH
Forschungszentrum