# IDEs & Notebooks

Development environments and notebook servers available on the cluster.

- Jupyter on Slurm
- PowerIDE User Guide

# Jupyter on Slurm

Run JupyterLab interactively on a compute node via SSH port forwarding. For a simpler browser-based experience, see **PowerIDE**.

# Step 1: Start an Interactive Job

```
srun -p <partition-name> -A <account-name> --pty bash
```

Replace `<partition-name>` and `<account-name>` with your values from `check_my_partitions`. Note the compute node name you land on (e.g. `compute-0-62`) — you'll need it later.

# Step 2: Load the Jupyter Module

```
module load mamba-env158/jupyter
```

# Step 3: Start JupyterLab

```
jupyter lab --ip=* --port=8892 --no-browser
```

Once started, you'll see output containing a URL with a token:

```
http://localhost:8892/lab?token=<token>
```

Copy the full URL including the token.

# Step 4: SSH Port Forwarding

Open a new terminal on your local machine and run:

```
ssh -N -L 8892:<compute-node-name>:8892 <username>@slurmlogin.tau.ac.il
```

Replace `<compute-node-name>` with the node from Step 1 (e.g. `compute-0-62`) and `<username>` with your TAU username. Keep this terminal open.

# Step 5: Open in Browser

Paste the URL from Step 3 into your browser:

```
http://localhost:8892/lab?token=<token>
```

# Step 6: Closing

When finished, press `Ctrl+C` in the interactive session to stop JupyterLab, then:

```
exit
```

Also close the SSH tunnel terminal.

# Notes

- If port 8892 is in use, pick any other free port — use the same number in both the `jupyter lab` command and the SSH tunnel
- The SSH tunnel must stay open while you work
- For a simpler experience without SSH tunneling, use **PowerIDE** instead

# PowerIDE User Guide

PowerIDE provides interactive access to the HPC cluster through a web browser. Run Jupyter notebooks, VS Code, and RStudio directly on compute nodes without needing SSH access.

Access PowerIDE at: https://poweride.tau.ac.il/jupyter

# Getting Started

Log in with your TAU university credentials (same as email and other university services).

After logging in, click **Start My Server**. You'll see a Server Options form to configure your compute resources.

When you start your server, PowerIDE submits a Slurm job to the PowerSlurm cluster. Your session runs on a compute node — not on the PowerIDE server itself. This means:

- You get dedicated resources (CPUs, memory, GPUs) on a compute node
- Your job runs through the same Slurm scheduler as other HPC jobs
- Your session will queue if the cluster is busy

# Configuring Resources

## Partition

Select which partition to run on. The dropdown shows only partitions you have access to. Common options:

- `power-general-shared-pool` — general purpose computing
- `gpu-general-pool` — GPU-enabled nodes

Check with your PI or HPC admin if unsure which partition to use.

## QOS

Controls priority and resource limits. **Default (owner)** is usually the right choice. Only valid QOS options for your selected partition are shown.

# GPUs

Appears only when a GPU partition is selected. Specify how many GPUs you need (0 if none).

# Time (D-HH:MM:SS)

How long your session should run. Default: `04:00:00`. Your session is terminated when time runs out — save your work regularly.

- `02:30:00` — 2.5 hours
- `1-12:00:00` — 1 day and 12 hours

# CPUs per task

Default: 1. Increase for multi-threaded code.

# Memory

Default: 1G. Examples: `2G`, `8G`, `500M`. Start small and increase if needed — over-requesting delays job start.

# Working Directory

Default: your home directory. Change to your project directory to save navigation time after launch.

# Stdout / Stderr Directory

Where job logs are written. Default (home directory) is fine for most users.

# Starting Your Session

Click the orange **Start** button. PowerIDE submits a Slurm job and shows a progress page. Once a compute node is available (usually 10–60 seconds), you're automatically redirected to JupyterLab.

If the cluster is busy, you can close the browser and come back — your session will start when resources are available.

# Using JupyterLab

- **Left sidebar** — file browser, running kernels, extensions
- **Main area** — notebooks, text files, terminals
- **+** button — opens the launcher for new tools

Common tasks:

- **New notebook** — click **+** → choose a Python kernel
- **Terminal** — click **+** → click Terminal (bash shell on the compute node)
- **Upload files** — drag and drop into the file browser
- **Download files** — right-click file → Download

# Using VS Code

PowerIDE includes VS Code running in your browser:

1. Click **+** to open the launcher
2. Click the **VS Code** icon
3. VS Code opens in a new tab with access to all your files and the same resources as JupyterLab

# Using RStudio

RStudio runs as a separate service on a dedicated compute node. It has its own launch form with R-specific options:

1. Click **+** to open the launcher
2. Click the **RStudio** icon
3. Fill in the resource form and select your R environment
4. Click **Start** — RStudio opens in a new tab once the job is running

# R Environment

Select the R environment to load. Each environment is a named module (e.g. `webR-genomics-2024`) with R and a pre-installed set of packages. Contact HPC support if you need a package that isn't available.

# R Library Path (optional)

If you have a personal R package library installed in a directory on the cluster, enter its full path here (e.g. `/home/user/R/library`). R will search this directory first, before the environment's default library.

# Stopping RStudio

Use the **Stop** button in the PowerIDE topbar to terminate your RStudio job. **Do not use File → Quit Session** — that ends the R session but leaves the Slurm job running, continuing to consume resources.

# Python Environments

PowerIDE provides one default kernel: **Python 3.12 (Base)**.

You can register your own conda/mamba environments as kernels:

```
module load mamba/mamba-2.1.1
mamba create -n my-project python=3.11 pandas matplotlib
mamba activate my-project
mamba install ipykernel

# Register as kernel (only visible to you)
python -m ipykernel install --user --name my-project --display-name "My Project (Python 3.11)"
```

Refresh your browser — the new kernel appears in the launcher. To remove a kernel:

```
jupyter kernelspec uninstall kernel-name
```

# Stopping Your Server

**Always stop your server when done** to free resources for others.

- **From JupyterLab** — File → Hub Control Panel → Stop My Server
- **From PowerIDE home** — navigate to https://poweride.tau.ac.il/jupyter/hub/home → Stop My Server
- **VS Code / RStudio** — use the **Stop** button in the PowerIDE topbar

# Best Practices

- Request only what you need — over-requesting delays your job and others
- Set a realistic time limit; restart if you need more
- Only request GPUs if your code actually uses them
- Store large datasets in scratch space, not your home directory
- Use Git for code — not for large data files