

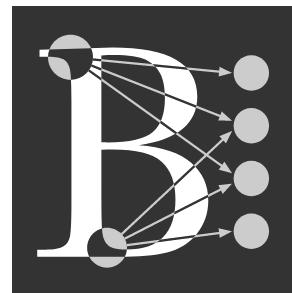
Baskerville Portal

EPSRC Tier 2 HPC System
University of Birmingham

Overview

1. What is Baskerville Portal?
2. Connect and Configure Baskerville Portal
3. JupyterLab
 - a) Module Loading
 - b) Conda Environments
4. Final Task: Build PyTorch in a Conda Environment

1. What is Baskerville Portal?



The Baskerville portal provides web-based access to the Baskerville Tier 2 system

This service is operated by Advanced Research Computing at the University of Birmingham and is funded by EPSRC
Grant EP/T022221/1

portal.baskerville.ac.uk/

2. Connect and Configure Baskerville Portal

BASKERVILLE

Sign in to your account

Username or email

Password

[New User / Forgot Password?](#)

[Sign In](#)

Or sign in with (do not use for first time login)

[University of Birmingham](#)

[CILogon Federated Identity](#)

[Alan Turing Institute](#)

[ORCID](#)

Open a Terminal

Click **Clusters > Baskerville Tier 2 HPC Shell Access**, enter your Password and OTP to access a Terminal.



```
[username@bask-pg0310u16a ~]$ my_baskerville
```

to see information about your Baskerville projects and available QoS.

This information is also available at admin.baskerville.ac.uk/.

Your home folder | 20GB



```
[username@bask-pg0310u16a ~]$ pwd  
/bask/homes/u/username  
[username@bask-pg0310u16a ~]$ my_quota  
Your home directory on Baskerville (/bask/homes  
/u/username):  
You are using 97.30 percent ( 19.48 GB ) of your total  
quota 20.02 GB
```

Your project folder | 1TB



```
[username@bask-pg0310u16a ~]$ cd /bask/projects/m/myproject  
[username@bask-pg0310u16a myproject]$ df -h .  
Filesystem      Size  Used Avail Use% Mounted on  
bask            1.0T   64M  1.0T   1% /bask
```

Symlink your home and project folder

- For JupyterLab, only the home folder is accessible from the GUI
- Solution - Access data in your project folder from your home folder via a symbolic link. Configure this with:



```
[username@bask-pg0310u16a ~]$ ln -s /bask/projects  
/m/myproject ~/myproject
```

baskstatus - Query Available GPUs



```
username@bask-pg0310u16a myproject]$ baskstatus
Current Baskerville GPU availability:
* 1 node with 3 x A100-40 available
* 10 nodes with 1 x A100-40 available
* 2 nodes with 4 x A100-80 available
* 1 node with 2 x A100-80 available
```

3. JupyterLab

Launch a JupyterLab server

- Click **Interactive Apps > JupyterLab**
- Load Python kernel
- Request Resources - number of hours and number of GPUs
- Select Project and Queue
- Click **Launch** to launch JupyterLab server

Home / My Interactive Sessions / JupyterLab

Interactive Apps

- [JupyterLab](#)
- [GUIs](#)
- [CST Studio Suite](#)
- [Fiji](#)
- [RELION](#)

JupyterLab version: 70aad71

This app will launch a [JupyterLab](#) server (supporting [Python](#) and [Julia](#) kernels) on Baskerville.

Kernel to load

Python 3.8.6 (2020b / GCCcore-10.2.0)

This defines the kernel you wish to load. The extra packages for use in Python can be selected from inside Jupyter using the Lmod extension. For further information please refer to the [JupyterLab documentation](#).

Show Conda Environments

Include kernels for compatible Conda environments found in: `~/.conda/environments.txt`. For further information please see our [Conda environment documentation](#).

Number of hours

4

Number of GPUs

1

Number of GPUs to request. For each GPU you will get 25% of a node's total cores.

Baskerville Project

myproject

Please select the Baskerville Project to which the job will be attached.

Queue

myqueue

Please select the Queue/QoS on which your job will run.

[Launch](#)

* The JupyterLab session data for this session can be accessed under the [data root directory](#).

Launch a JupyterLab server

- Wait time depends on number of GPUs requested and time requested
- Each node has 4 GPUs available, with 36 CPU cores per GPU
- Once loaded, click **Connect to Jupyter**

Baskerville OnDemand Files ▾ Jobs ▾ Clusters ▾ Interactive Apps ▾   

Session was successfully created.

Home / My Interactive Sessions

Interactive Apps	1 node 36 cores Running
JupyterLab	Host: > _bask-pg0308u10a.cluster.baskerville.ac.uk
GUIs	Created at: 2022-10-20 16:39:38 BST
CST Studio Suite	Time Remaining: 1 hour and 58 minutes
Fiji	Session ID: 8394c5ff-8cc6-4906-b461-1a1e2fb79474
RELION	Connect to Jupyter

 Delete

+

Filter files by name

/

Name	Last Modified
myproject	5 months ago
ondemand	9 months ago

Symlink to project folder

Launcher +

Notebook

Python

Console

Python

Other

Terminal

Text File

Markdown File

Python File

Show Contextual Help

Launcher

+ + ↑ ⌂ ⌂+

Filter files by name

/ myproject /

Name	Last Modified
myfolder1	21 days ago
myfolder2	21 days ago
other	7 hours ago

Simple 4 \$_ 0 ⌂

Launcher

myproject

Notebook

New Jupyter Notebook

Python

Console

Python

Other

Terminal

Text File

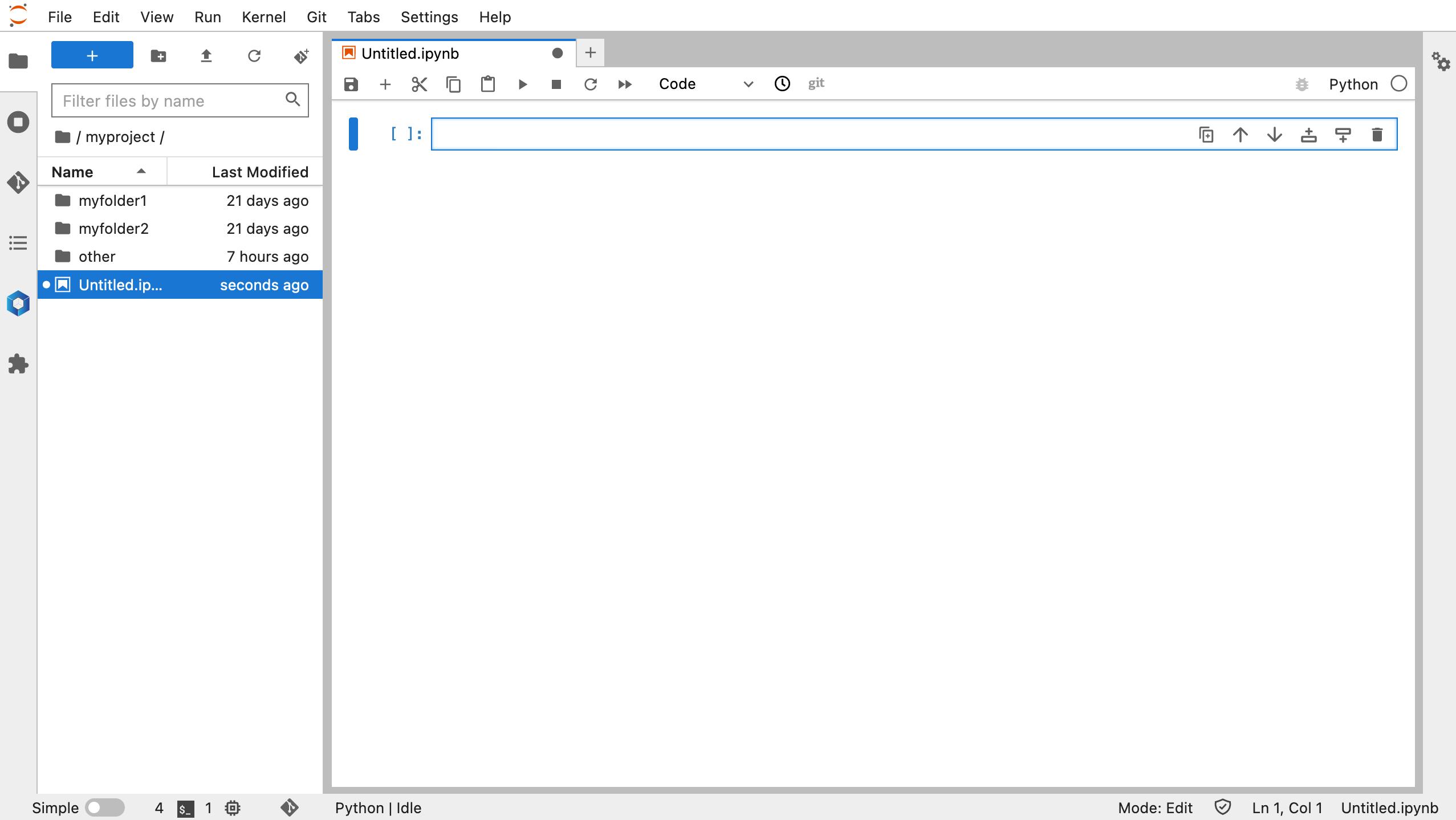
Markdown File

Python File

Show Contextual Help

⚙️

Launcher



A screenshot of a Jupyter Notebook interface. The top navigation bar includes File, Edit, View, Run, Kernel, Git, Tabs, Settings, and Help. On the left, there's a sidebar with icons for file operations like +, folder, up, down, and search, followed by a 'myproject' folder structure. A table lists files: myfolder1 (21 days ago), myfolder2 (21 days ago), other (7 hours ago), and Untitled.ipynb (2 minutes ago). The main workspace shows an open notebook titled 'Untitled.ipynb'. In the first code cell, the command `print("Hello world!")` is run, resulting in the output `Hello world!`. Below it is an empty cell labeled []. The bottom status bar shows 'Simple' mode, a toggle switch, and the path 'Python | Idle'.

File Edit View Run Kernel Git Tabs Settings Help

+ Untitled.ipynb X +

Filter files by name

/ myproject /

Name	Last Modified
myfolder1	21 days ago
myfolder2	21 days ago
other	7 hours ago
Untitled.ip... (selected)	2 minutes ago

[1]: `print("Hello world!")`

Hello world!

[]:

Mode: Edit X Ln 1, Col 1 Untitled.ipynb

The screenshot shows a Jupyter Notebook interface with the following elements:

- Top Bar:** File, Edit, View, Run, Kernel, Git, Tabs, Settings, Help.
- Toolbar:** Buttons for New, Open, Save, Run, Cell, Kernel, Help, and a gear icon for settings.
- File Explorer Sidebar:** Shows a folder structure under "/myproject/".
 - myfolder1 (21 days ago)
 - myfolder2 (21 days ago)
 - other (7 hours ago)
 - mynotebook.ipynb (seconds ago)A blue arrow points from the "mynotebook.ipynb" entry to a "Right click > Rename" option.
- Search Bar:** Filter files by name.
- Code Cell:** [1]: `print("Hello world!")`
Output:
Hello world!
- Input Cell:** []: (empty)
- Language Selection:** Python (Python 3.6.9)
- Help Icon:** A gear icon in the top right corner.
- Bottom Status Bar:** Simple, Mode: Edit, Ln 1, Col 1, mynotebook.ipynb.

Text Overlay: • Remember to save and rename your notebooks

a) Module Loading

- Module applications available include
 - PyTorch
 - TensorFlow
 - Keras
 - Dask
 - ... and many more!
- See apps.baskerville.ac.uk/applications/ for a complete list

The screenshot shows a Jupyter Notebook interface with the following components:

- Top Bar:** File, Edit, View, Run, Kernel, Git, Tabs, Settings, Help.
- Left Sidebar:**
 - Search available modules... input field.
 - LOADED MODULES section with a '+' button highlighted by a blue box.
 - AVAILABLE MODULES section listing various software environments:
 - bask-apps/live/live
 - bask-container-conf/live
 - bask-licences/live
 - bask-variables/live
 - baskdefault
 - baskerville/live
 - openmpi-env/live
 - Lmod extension icon with a blue arrow pointing to the 'Available Modules' list.
- Main Area:**
 - Tab bar: mynotebook.ipynb (selected), +, X.
 - Toolbar: file, new, close, back, forward, Code, git.
 - Code cell [2]: `print("Hello world!")` output: Hello world!
 - Code cell [1]: (empty).
 - Bottom toolbar: copy, up, down, plus, minus, trash.
- Bottom Status Bar:** Simple, Python | Idle, Mode: Command, Ln 1, Col 1, mynotebook.ipynb.

List of Loaded Modules:

- ASE/3.22.0-foss-2021a
- ATK/2.36.0-GCCcore-10....
- AlphaFold/2.2.2-foss-202...
- Arm-Forge/22.0.3-foss-2...
- Autoconf/2.69-GCCcore-...
- Autoconf/2.71-GCCcore-1...
- Automake/1.16.2-GCCcor...
- Automake/1.16.3-GCCcor...
- Autotools/20200321-GC...
- Autotools/20210128-GCC...
- BLIS/0.8.1-GCC-10.3.0
- Bazel/3.7.2-GCCcore-10....
- Bazel/3.7.2-GCCcore-10.3...
- Biopython/1.79-foss-2021a
- Bison/3.5.3
- Bison/3.7.1-GCCcore-10.2.0

List of Available Modules (highlighted in blue):

- **Loaded Modules** appear at the top of the menu
- Default environment is currently loaded
- Save current collection as **default**

File Edit View Run Kernel Git Tabs Settings Help

Search available modules...

LOADED MODULES + ⌛ 🗑

bask-apps/live/live
bask-container-conf/live
bask-licences/live
bask-variables/live
baskdefault
baskerville/live
openmpi-env/live

AVAILABLE MODULES

ASE/3.22.0-foss-2021a
ATK/2.36.0-GCCcore-10.2.0
AlphaFold/2.2.2-foss-2021a-CUDA...
Arm-Forge/22.0.3-foss-2021a
Autoconf/2.69-GCCcore-10.2.0
Autoconf/2.71-GCCcore-10.3.0
Automake/1.16.2-GCCcore-10.2.0
Automake/1.16.3-GCCcore-10.3.0
Autotools/20200321-GCCcore-10.2...
Autotools/20210128-GCCcore-10.3.0
BLIS/0.8.1-GCC-10.3.0
Bazel/3.7.2-GCCcore-10.2.0
Bazel/3.7.2-GCCcore-10.3.0
Biopython/1.79-foss-2021a
Bison/3.5.3
Bison/3.7.1-GCCcore-10.2.0

mynotebook.ipynb • +

File + ✎ 🗑️ 📁 ➡️ ➤ Code ⌄ git 🐧 pytorch_gpu_conda_env (Conda)

[1]: `print("Hello world!")`
Hello world!

[]:

Save collection
Enter a new collection name:

Cancel Save

- **Loaded Modules** appear at the top of the menu
- Default environment is currently loaded
- Save current collection as **default**

Simple 1 \$ 1 ⚙️ ⌛ pytorch_gpu_conda_env (Conda) | Idle Mode: Command 🛡️ Ln 1, Col 1 mynotebook.ipynb

The screenshot shows a Jupyter Notebook interface with the following components:

- Top Bar:** File, Edit, View, Run, Kernel, Git, Tabs, Settings, Help.
- Left Sidebar:** A sidebar for managing modules. It includes a search bar ("Search available modules..."), a "LOADED MODULES" section with a "+" button and a refresh icon, and an "AVAILABLE MODULES" section listing various software environments. The "openmpi-env/live" entry is highlighted with a blue arrow pointing to the text "Lmod extension".
- Central Area:** A notebook tab titled "mynotebook.ipynb" is active. The interface includes standard notebook controls (New, Close, etc.) and a toolbar with Code, git, and other icons.
- Code Cell Output:** The notebook displays a single code cell output:

```
[2]: print("Hello world!")  
Hello world!
```
- Bottom Status Bar:** Shows "Simple" mode, a cell identifier (4 \$ 1), a Python Idle icon, "Mode: Command", a shield icon, "Ln 1, Col 1", and the file name "mynotebook.ipynb".

List of Available Modules (from bottom to top):

- Simple
- 4 \$ 1
- Python | Idle
- Mode: Command
- Ln 1, Col 1
- mynotebook.ipynb

Key Features:

- **Loaded Modules** appear at the top of the menu
- Default environment is currently loaded
- Save current collection as **default**
- **Available Modules** appear below

The screenshot shows a Jupyter Notebook interface. On the left, there is a sidebar with various icons and a list of available modules. The module 'PyTorch/1.8.1-fosscuda-2...' is highlighted with a blue border. The main area displays a notebook titled 'mynotebook.ipynb'. A single cell is visible, containing the Python code `[1]: print("Hello world!")`. The output of this cell is 'Hello world!'. The interface includes standard Jupyter Notebook controls like code, run, and git buttons, and a Python kernel selection.

- Load **PyTorch/1.8.1-fosscuda-2020b**
- It's a good idea to load modules built using compatible toolchains, e.g., fosscuda-2020b is compatible with Python 3.8.6 (2020b/GCCcore-10.2.0)
- <https://docs.baskerville.ac.uk/apps/#toolchains>
- Module loading can take a few minutes

Simple 0 \$ 1 Python | Idle Mode: Command ✓ Ln 1, Col 1 mynotebook.ipynb

The screenshot shows a Jupyter Notebook interface. On the left, there's a sidebar with various icons and a search bar for available modules. The main area has a tab for 'mynotebook.ipynb'. Below the tabs, a toolbar includes a refresh icon, a plus sign for new cells, and other standard notebook controls. The main workspace contains a code cell with the command `print("Hello world!")`, which has been executed and printed the output "Hello world!". A large blue callout box points to the "Loaded Modules" section in the sidebar, listing numerous system and library modules. The status bar at the bottom indicates the mode is "Command", and the file name is "mynotebook.ipynb".

- Check modules are loaded in the **Loaded Modules** section
- Restart the kernel (this can also take a few minutes)

Simple 0 \$ 1 Python | Idle Mode: Command 0 1 mynotebook.ipynb

File Edit View Run Kernel Git Tabs Settings Help

Search available modules...

LOADED MODULES + 🔍 🗑

- CUDA/11.1-GCC-10.2.0
- CUDAcore/11.1
- Check/0.15.2-GCCcore...
- FFTW/3.3.8-gompic-20...
- FFmpeg/4.3.1-GCCcore...
- FriBidi/1.0.10-GCCcore...
- GCC/10.2.0
- GCCcore/10.2.0
- GDRCopy/2.1-GCCcore...
- GMP/6.2.0-GCCcore-1...
- LAME/3.100-GCCcore-...
- LibTIFF/4.1.0-GCCcore-...
- MPFR/4.1.0-GCCcore-1...
- NASM/2.15.05-GCCcor...
- NCCL/2.8.3-GCCcore-...
- Ninja/1.10.1-GCCcore-1...
- OpenBLAS/0.3.12-GCC...
- OpenMPI/4.0.5-gcccud...
- PMLx/3.1.5-GCCcore-1...
- Pillow/8.0.1-GCCcore-1...
- PyTorch/1.8.1-fosscuda...
- PyYAML/5.3.1-GCCcor...
- Python/3.8.6-GCCcore...
- SQLite/3.33.0-GCCcor...
- ScalAPACK/2.1.0-aom...

mynotebook.ipynb × +

Code git

Python

[1]: `print("Hello world!")`
Hello world!

[2]: `import torch`

[]:

↑ ↓ ± ↻

- Check modules are loaded in the **Loaded Modules** section
- Restart the kernel (this can also take a few minutes)

Simple 0 \$ 1 🔍 Python | Idle Mode: Edit 🛡️ Ln 1, Col 1 mynotebook.ipynb

The screenshot shows a Jupyter Notebook interface with the following details:

- File Bar:** File, Edit, View, Run, Kernel, Git, Tabs, Settings, Help.
- Toolbar:** Search available modules..., LOADED MODULES, +, - (refresh), Python, gear icon.
- Left Sidebar (Loaded Modules):** A list of loaded modules including CUDA, CUDAcore, Check, FFTW, FFmpeg, FriBidi, GCC, GCCcore, GDRCopy, GMP, LAME, LibTIFF, MPFR, NASM, NCCL, Ninja, OpenBLAS, OpenMPI, PMIx, Pillow, PyTorch, PyYAML, Python, SQLite, and ScaLAPACK.
- Central Area:** A notebook tab titled "mynotebook.ipynb" is active. The code cell [1] contains `print("Hello world!")` and outputs "Hello world!". The code cell [2] contains `import torch`. The code cell [3] contains `torch.__version__` and outputs '1.8.1'. A new code cell [4] is currently being typed with the placeholder `[4]:` .
- Bottom Status Bar:** Shows Simple, 0 \$ 1, Python | Idle, Mode: Edit, shield icon, Ln 1, Col 1, mynotebook.ipynb.

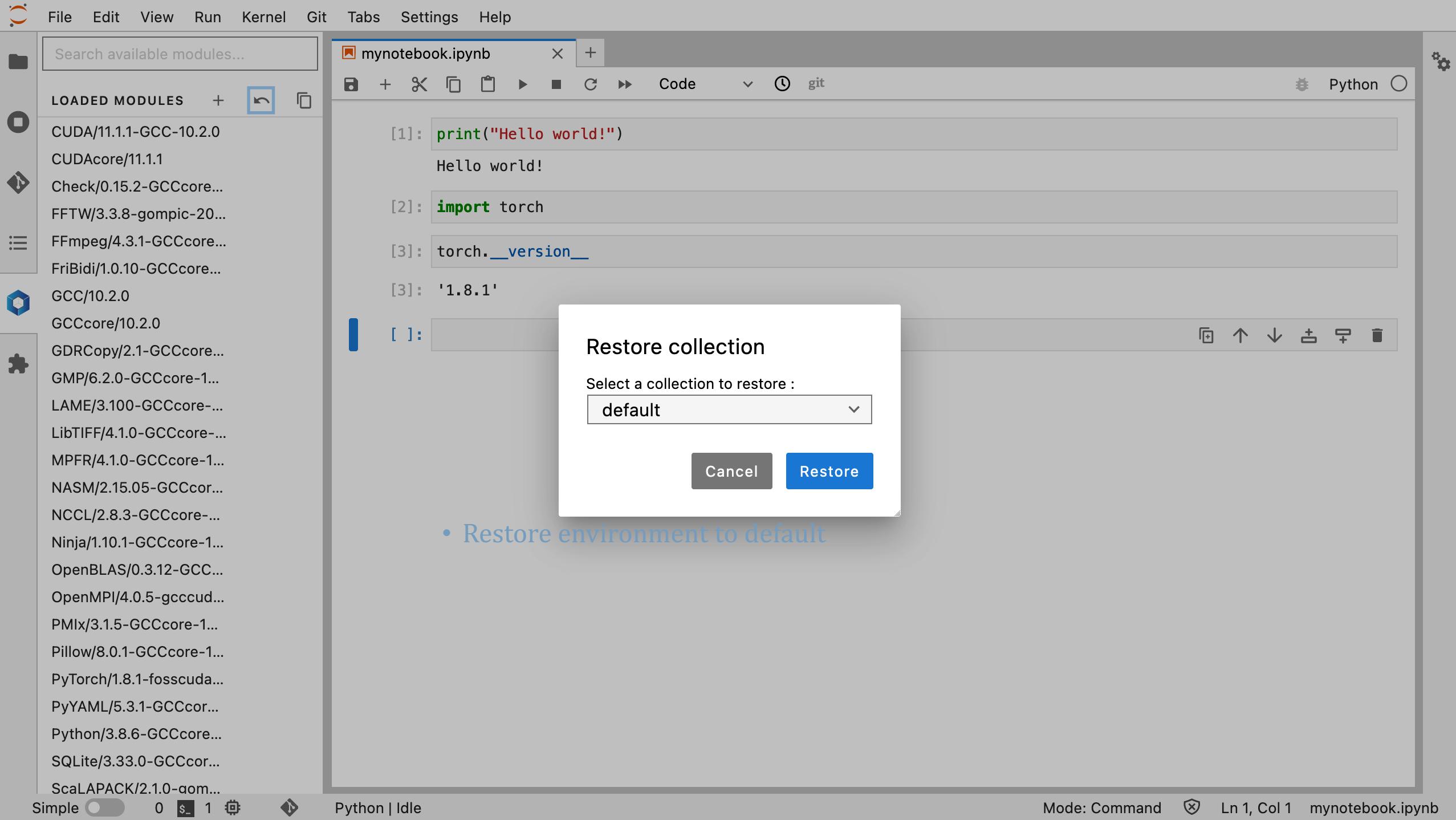
List-Group

- Check modules are loaded in the **Loaded Modules** section
- Restart the kernel (this can also take a few minutes)

The screenshot shows a Jupyter Notebook interface with the following components:

- Top Bar:** File, Edit, View, Run, Kernel, Git, Tabs, Settings, Help.
- Left Sidebar:** A sidebar titled "LOADED MODULES" containing a list of loaded modules, each with a small icon and a "Remove" button. The list includes:
 - CUDA/11.1-GCC-10.2.0
 - CUDAcore/11.1
 - Check/0.15.2-GCCcore...
 - FFTW/3.3.8-gompic-20...
 - FFmpeg/4.3.1-GCCcore...
 - FriBidi/1.0.10-GCCcore...
 - GCC/10.2.0
 - GCCcore/10.2.0
 - GDRCopy/2.1-GCCcore...
 - GMP/6.2.0-GCCcore-1...
 - LAME/3.100-GCCcore-...
 - LibTIFF/4.1.0-GCCcore-...
 - MPFR/4.1.0-GCCcore-1...
 - NASM/2.15.05-GCCcor...
 - NCCL/2.8.3-GCCcore-...
 - Ninja/1.10.1-GCCcore-1...
 - OpenBLAS/0.3.12-GCC...
 - OpenMPI/4.0.5-gccud...
 - PMLx/3.1.5-GCCcore-1...
 - Pillow/8.0.1-GCCcore-1...
 - PyTorch/1.8.1-fosscuda...
 - PyYAML/5.3.1-GCCcor...
 - Python/3.8.6-GCCcore...
 - SQLite/3.33.0-GCCcor...
 - ScalAPACK/2.1.0-aom...
- Central Area:** A notebook tab titled "mynotebook.ipynb" is active. The code cell [1] contains the command `print("Hello world!")`, which has been executed and printed "Hello world!". The code cell [2] contains the command `import torch`. The code cell [3] contains the command `torch.__version__`, which has been executed and printed "1.8.1". Below these cells is an empty code cell [4] with a blue border, ready for input. To the right of the cells are standard Jupyter notebook toolbar icons for cell selection, execution, and file operations.
- Bottom Status Bar:** Shows the status "Python | Idle", the current mode "Mode: Edit", and the file name "mynotebook.ipynb".

A blue callout box points to the text "• Restore environment to default" located in the bottom left area of the central workspace.



Your turn

- Restore **default** environment and restart kernel
- Module load **TensorFlow**
- **import tensorflow as tf**
- Check the software version number with **tf._version_**
- Query visible GPU devices with the command



```
tf.config.list_physical_devices(  
    device_type=None  
)
```

The screenshot shows a Jupyter Notebook interface with the following components:

- Top Bar:** File, Edit, View, Run, Kernel, Git, Tabs, Settings, Help.
- Left Sidebar:** A sidebar titled "LOADED MODULES" containing a list of loaded modules, each with a small icon to its left:
 - CUDA/11.1-GCC-10.2.0
 - CUDAcore/11.1.1
 - Check/0.15.2-GCCcore...
 - FFTW/3.3.8-gompic-20...
 - GCC/10.2.0
 - GCCcore/10.2.0
 - GDRCopy/2.1-GCCcore...
 - GMP/6.2.0-GCCcore-1...
 - HDF5/1.10.7-gompic-2...
 - ICU/67.1-GCCcore-10.2.0
 - JsonCpp/1.9.4-GCCcor...
 - LMDB/0.9.24-GCCcore...
 - NASM/2.15.05-GCCcor...
 - NCCL/2.8.3-GCCcore-1...
 - OpenBLAS/0.3.12-GCC...
 - OpenMPI/4.0.5-gcccu...
 - PCRE/8.44-GCCcore-1...
 - PMLx/3.1.5-GCCcore-10...
 - Python/3.8.6-GCCcore...
 - SQLite/3.33.0-GCCcor...
 - ScaLAPACK/2.1.0-gom...
 - SciPy-bundle/2020.11-f...
 - Szip/2.1.1-GCCcore-10....
 - Tcl/8.6.10-GCCcore-10....
 - TensorFlow/2.5.0-fossc...
- Central Area:** A tab titled "mynotebook.ipynb" is active. The notebook contains the following code cells:
 - [2]: `print("Hello world!")`
Output: Hello world!
 - [3]: `import tensorflow as tf`
 - [4]: `tf.__version__`
Output: '2.5.0'
 - [5]: `tf.config.list_physical_devices(device_type=None)`
Output: [PhysicalDevice(name='/physical_device:CPU:0', device_type='CPU'), PhysicalDevice(name='/physical_device:GPU:0', device_type='GPU')]
 - []: (An empty cell with a blue vertical bar on the left.)
- Right Sidebar:** Includes a "Python" language selector, a gear icon for settings, and a status bar at the bottom indicating "Mode: Command".

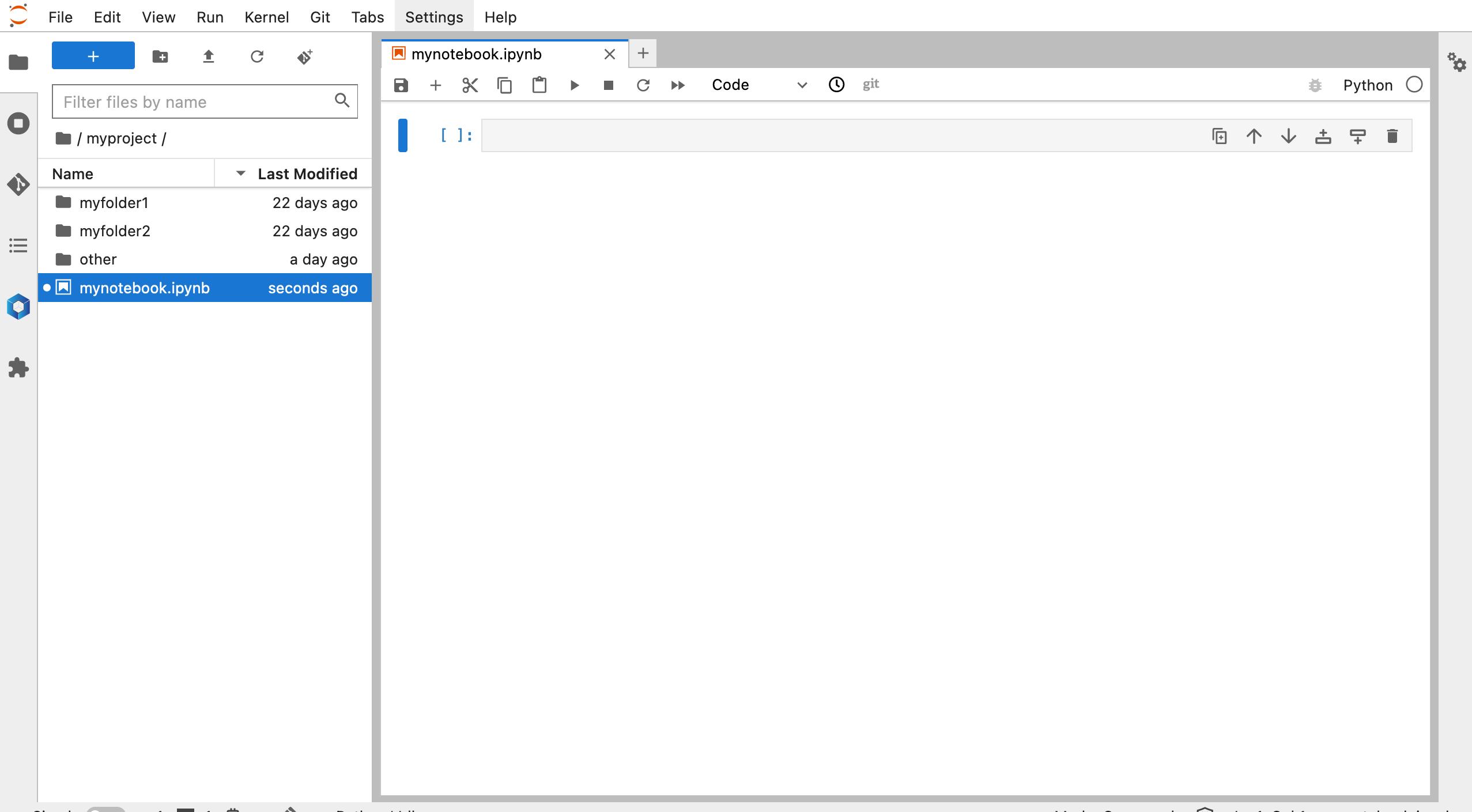
b) Conda Environments

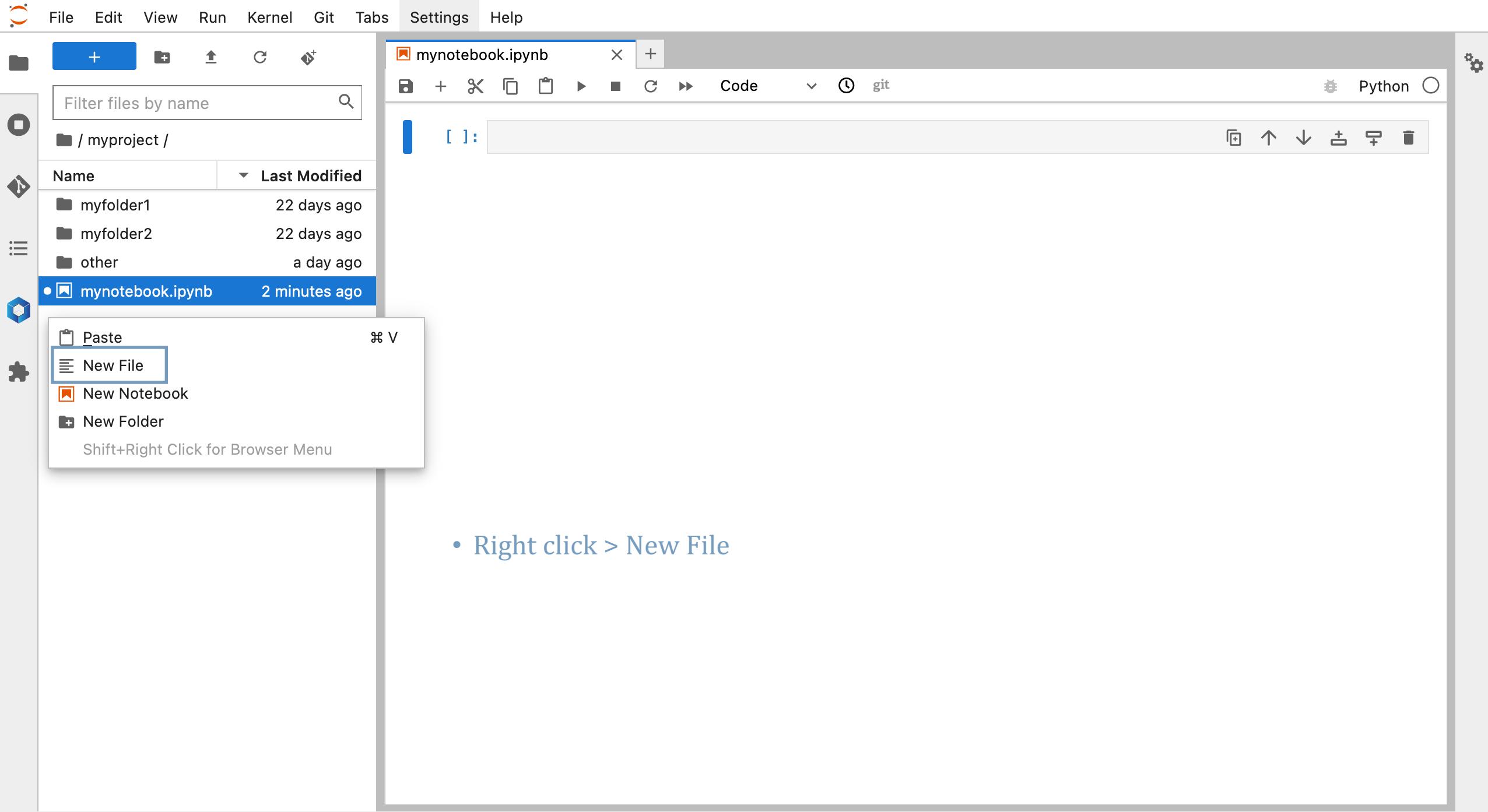
- Conda is a package and environment manager for Python
- Steps to create a Conda Environment for JupyterLab:
 - i. Create a Conda environment in your **project folder** using a batch script in the Terminal
 - ii. Relaunch the JupyterLab server to load the new Conda kernel
 - iii. (Optional) Use the same JupyterLab session to modify the environment

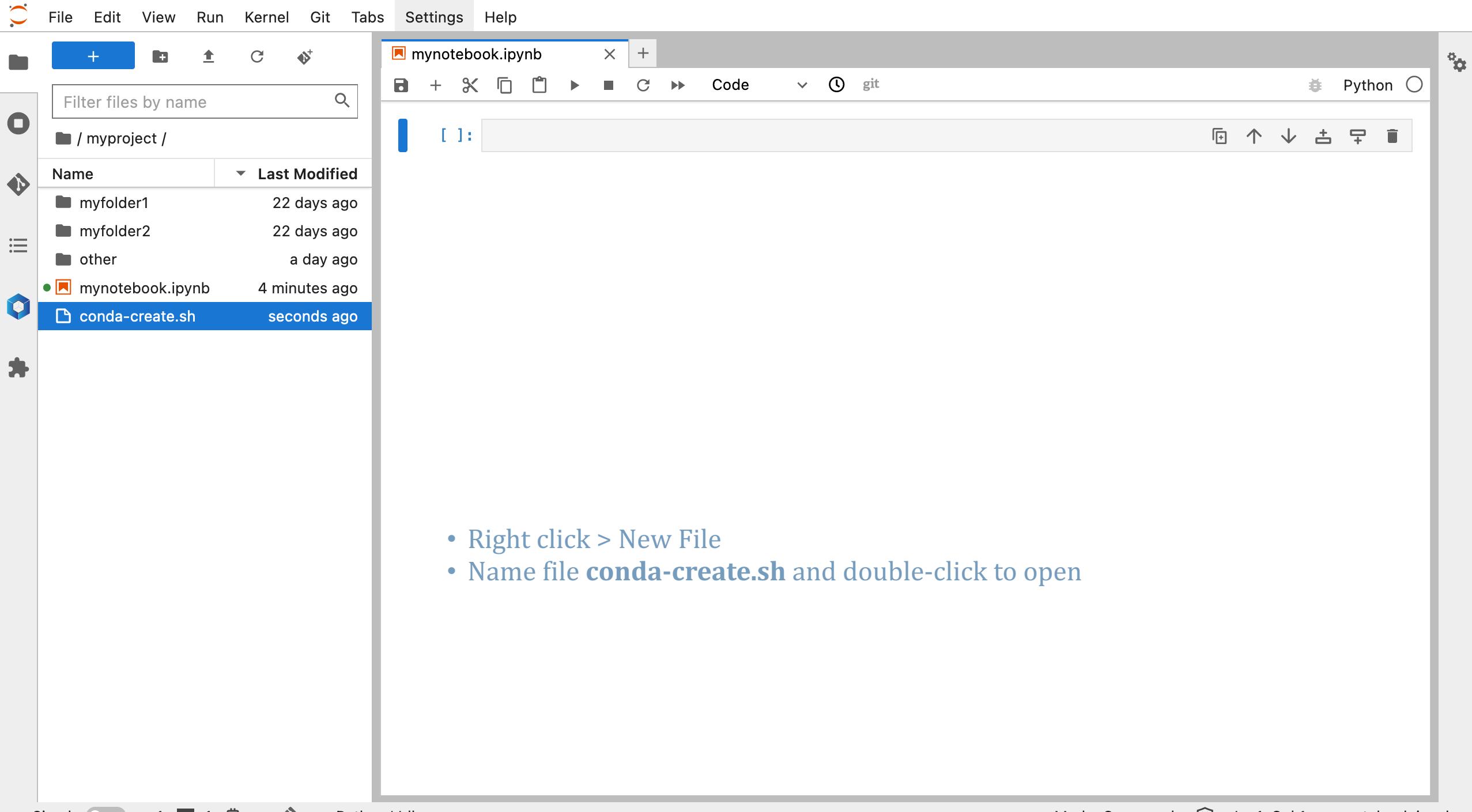
3. JupyterLab

b) Conda Environments

(i) Create your Conda Environment







The screenshot shows a Jupyter Notebook interface. On the left, there is a sidebar with various icons for file operations like creating new files, uploading, and running cells. Below the sidebar is a file browser titled "myproject /". It lists several items:

Name	Last Modified
myfolder1	22 days ago
myfolder2	22 days ago
other	a day ago
mynotebook.ipynb	6 minutes ago
conda-create.sh	2 minutes ago

The file "conda-create.sh" is currently selected and highlighted with a blue background. In the main workspace, there are two tabs open: "mynotebook.ipynb" and "conda-create.sh". The "conda-create.sh" tab is active, showing a single line of text: "1". The status bar at the bottom indicates "Simple" mode, "Ln 1, Col 1", "Spaces: 4", and the file name "conda-create.sh".

- Right click > New File
- Name file **conda-create.sh** and double-click to open
- This opens a text editor in which you can write a batch script to create your conda environment

File Edit View Run Kernel Git Tabs Settings Help

+ 📂 ⏚ ⏚ ⏚ +

Filter files by name

/ myproject /

Name	Last Modified
myfolder1	22 days ago
myfolder2	22 days ago
other	a day ago
mynotebook.ipynb	11 minutes ago
conda-create.sh	seconds ago

conda-create.sh

```
1 #!/bin/bash
2 #SBATCH --account myproject
3 #SBATCH --qos myqueue
4 #SBATCH --time 01:00:00
5 #SBATCH --nodes 1
6 #SBATCH --gpus 1
7 #SBATCH --cpus-per-gpu 36
```

• **account** and **qos** details can be double checked with **my_baskerville** command in the Terminal

• request a modest amount of time for an increased chance of your job starting straight away

• 1 node has 4 GPUs and 144 CPU cores, therefore $144/4 = 36$ cores and $4/4 = 1$ GPU are the minimum resources available to build the environment (even if we requested less cores, this still takes 1 GPU out of service due to distribution of system architecture)

Simple 2 \$ 1 Shell Ln 7, Col 26 Spaces: 4 conda-create.sh

+ + ↑ ⌂ ⌂+

Filter files by name

/ myproject /

Name	Last Modified
myfolder1	22 days ago
myfolder2	22 days ago
other	a day ago
mynotebook.ipynb	an hour ago
conda-create.sh	in a few seconds

Simple 3 \$ 1 Shell

mynotebook.ipynb conda-create.sh

```
1 #!/bin/bash
2 #SBATCH --account myproject
3 #SBATCH --qos myqueue
4 #SBATCH --time 01:00:00
5 #SBATCH --nodes 1
6 #SBATCH --gpus 1
7 #SBATCH --cpus-per-gpu 36
8
9 # Module loading
10 module purge # unloads and loaded modules and resets the environment
11 module load baskerville # loads the default Baskerville environment
12 module load bask-apps/live # load the stable, default application stack
13 module load Miniconda3/4.10.3 # load the Conda package manager
14
15 set -e # exit immediately if there is an error
16 eval "$( ${EBROOTMINICONDA3}/bin/conda shell.bash hook)" # initialise Conda
```

Saving completed

Ln 16, Col 75 Spaces: 4 conda-create.sh

+ + ↑ ⌂ ⌂+

Filter files by name

/ myproject /

Name	Last Modified
my_conda_env	23 minutes ago
myfolder1	40 minutes ago
myfolder2	40 minutes ago
other	seconds ago
conda-create.sh	38 minutes ago
mynotebook.ipynb	6 minutes ago

mynotebook.ipynb conda-create.sh

```
1 #!/bin/bash
2 #SBATCH --account yearwoog-baskerville
3 #SBATCH --qos bham
4 #SBATCH --time 01:00:00
5 #SBATCH --nodes 1
6 #SBATCH --gpus 1
7 #SBATCH --cpus-per-gpu 36
8
9 # Module loading
10 module purge # unloads and loaded modules and resets the environment
11 module load baskerville # loads the default Baskerville environment
12 module load bask-apps/live # load the stable, default application stack
13 module load Miniconda3/4.10.3 # load the Conda package manager
14
15 set -e # exit immediately if there is an error
16 eval "${EBROOTMINICONDA3}/bin/conda shell.bash hook" # initialise Conda
17
18 # Conda environment
19
20 # Define the path to your Conda environment (modify as appropriate)
21 # N.B. the environment directory's name must be lowercase only
22 # N.B. this path will be created by the subsequent commands if it doesn't already exist
23 export CONDA_ENV_PATH="/bask/projects/e/edmondac-rsg/wongj/my_conda_env"
24
25 export CONDA_PKGS_DIRS="/tmp" # Download packages to temporary storage
26
27 # Create the environment. Only required once.
28 conda create --yes --prefix "${CONDA_ENV_PATH}"
29 # Activate the environment
30 conda activate "${CONDA_ENV_PATH}"
31 # Choose your version of Python
32 conda install --yes python=3.10
33
34 # Continue to install any further items as required.
35 # For example:
36 conda install --yes numpy
```

+

Filter files by name

/ myproject /

Name	Last Modified
myfolder1	22 days ago
myfolder2	22 days ago
other	a day ago
mynotebook.ipynb	an hour ago
conda-create.sh	2 minutes ago

mynotebook.ipynb conda-create.sh Launcher +

myproject

Notebook

Python

Console

Python

\$ Other New Terminal

Terminal Text File Markdown File Python File Show Contextual Help

Simple 0 1 Launcher

The screenshot shows a Jupyter Notebook interface with a sidebar on the left containing various icons for file operations like creating new files, opening, saving, and running cells.

The main area has three tabs:

- mynotebook.ipynb**: The active tab, showing a file browser with the current directory set to `/myproject/`. The browser lists several items:
 - `myfolder1` (22 days ago)
 - `myfolder2` (22 days ago)
 - `other` (a day ago)
 - `mynotebook.ipynb` (an hour ago)
 - `conda-create.sh` (5 minutes ago)
- conda-create.sh**: A terminal window showing the command `pwd` being run, which outputs the path `/bask/projects/m/myproject`.
- Terminal 1**: An empty terminal window.

A large callout box highlights the terminal output in the `conda-create.sh` tab, with the following text:

- **pwd** – check current directory is the myproject folder
- (if not, **cd /bask/projects/m/myproject**)

At the bottom, there are buttons for "Simple" mode, a cell run button, and a terminal switcher showing "Terminal 1".

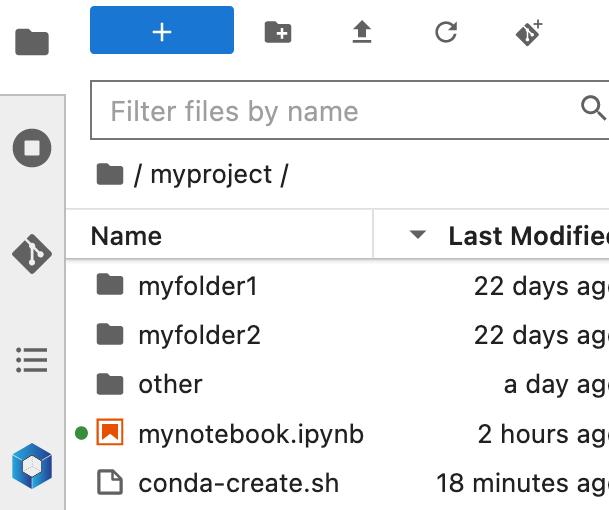
The screenshot shows a Jupyter Notebook interface with the following components:

- File Bar:** File, Edit, View, Run, Kernel, Git, Tabs, Settings, Help.
- File Browser:** A sidebar on the left showing the directory structure of `/myproject/`. It includes a search bar labeled "Filter files by name". The contents are:
 - Folder: `/ myproject /`
 - Table: Name (Column 1) and Last Modified (Column 2). Items listed are:
 - `myfolder1` 22 days ago
 - `myfolder2` 22 days ago
 - `other` a day ago
 - `mynotebook.ipynb` an hour ago
 - `conda-create.sh` 7 minutes ago
- Terminal:** Three tabs are open:
 - `mynotebook.ipynb`
 - `conda-create.sh`
 - `Terminal 1` (highlighted)The Terminal 1 tab contains the following command history:

```
Singularity> pwd  
/bask/projects/m/myproject  
Singularity> ls  
conda-create.sh  myfolder1  myfolder2  mynotebook.ipynb  other  
Singularity> [REDACTED]
```
- Bottom Status Bar:** Shows "Simple" mode, a toggle switch, and the number "1" followed by a terminal icon.
- Bottom Right:** "Terminal 1" label.

Listed Instructions:

- **pwd** – check current directory is the `myproject` folder
- (if not, `cd /bask/projects/m/myproject`)
- **ls** – list files and folders



mynotebook.ipynb conda-create.sh Terminal 1

```
Singularity> pwd  
/bask/projects/m/myproject  
Singularity> ls  
conda-create.sh myfolder1 myfolder2 mynotebook.ipynb other  
Singularity> sbatch conda-create.sh  
Submitted batch job 273422  
Singularity>
```

- **pwd** – check current directory is the myproject folder
- (if not, **cd /bask/projects/m/myproject**)
- **ls** – list files and folders
- **sbatch conda-create.sh** – submit batch job

The screenshot shows a Jupyter Notebook interface with a terminal window and a file browser.

File Browser:

- Path: / myproject /
- Items:
 - myfolder1 (22 days ago)
 - myfolder2 (22 days ago)
 - other (a day ago)
 - mynotebook.ipynb (2 hours ago) - selected
 - conda-create.sh (18 minutes ago)

Terminal Window:

```
Singularity> pwd  
/bask/projects/m/myproject  
Singularity> ls  
conda-create.sh myfolder1 myfolder2 mynotebook.ipynb other  
Singularity> sbatch conda-create.sh  
Submitted batch job 273422  
Singularity> squeue -u $USER  
JOBID PARTITION NAME USER ST TIME NODES NODELIST(REASON)  
273376 baskervil sys/dash wongj R 2:43:28 1 bask-pg0309u03a  
273422 baskervil conda-cr wongj PD 0:00 1 (Priority)  
Singularity>
```

Terminal Output Summary:

- pwd** – check current directory is the myproject folder
- (if not, **cd /bask/projects/m/myproject**)
- ls** – list files and folders
- sbatch conda-create.sh** – submit batch job
- squeue -u \$USER** – check your job status

Bottom Status Bar:

- Simple (button)
- 1 \$ 1 (button)
- Terminal 1 (button)

The screenshot shows a Jupyter Notebook interface with a terminal tab and a file browser.

Terminal Tab:

```
Singularity> pwd  
/bask/projects/m/myproject  
Singularity> ls  
conda-create.sh myfolder1 myfolder2 mynotebook.ipynb other  
Singularity> sbatch conda-create.sh  
Submitted batch job 273422  
Singularity> squeue -u $USER  
JOBID PARTITION NAME USER ST TIME NODES NODELIST(REASON)  
273376 baskervil sys/dash wongj R 2:43:28 1 bask-pg0309u03a  
273422 baskervil conda-cr wongj PD 0:00 1 (Priority)  
Singularity> squeue -u $USER  
JOBID PARTITION NAME USER ST TIME NODES NODELIST(REASON)  
273376 baskervil sys/dash wongj R 2:43:43 1 bask-pg0309u03a  
273422 baskervil conda-cr wongj R 0:05 1 bask-pg0308u24a  
Singularity> 
```

File Browser:

Filter files by name

/ myproject /

Name	Last Modified
myfolder1	22 days ago
myfolder2	22 days ago
other	a day ago
mynotebook.ipynb	2 hours ago
conda-create.sh	18 minutes ago
slurm-273422.st...	seconds ago

Simple 1 \$ 1 ⚙ Terminal 1

- **pwd** – check current directory is the myproject folder
- (if not, **cd /bask/projects/m/myproject**)
- **ls** – list files and folders
- **sbatch conda-create.sh** – submit batch job
- **squeue -u \$USER** – check your job status

The screenshot shows a Jupyter Notebook interface with a terminal window and a file browser.

Terminal Window:

```
Singularity> pwd  
/bask/projects/m/myproject  
Singularity> ls  
conda-create.sh myfolder1 myfolder2 mynotebook.ipynb other  
Singularity> sbatch conda-create.sh  
Submitted batch job 273422  
Singularity> squeue -u $USER  
JOBID PARTITION NAME USER ST TIME NODES NODELIST(REASON)  
273376 baskervil sys/dash wongj R 2:43:28 1 bask-pg0309u03a  
273422 baskervil conda-cr wongj PD 0:00 1 (Priority)  
Singularity> squeue -u $USER  
JOBID PARTITION NAME USER ST TIME NODES NODELIST(REASON)  
273376 baskervil sys/dash wongj R 2:43:43 1 bask-pg0309u03a  
273422 baskervil conda-cr wongj R 0:05 1 bask-pg0308u24a  
Singularity> squeue -u $USER  
JOBID PARTITION NAME USER ST TIME NODES NODELIST(REASON)  
273376 baskervil sys/dash wongj R 2:47:43 1 bask-pg0309u03a  
Singularity>
```

File Browser:

- Filter files by name: myproject /
- Table of contents:

Name	Last Modified
myfolder1	22 days ago
myfolder2	22 days ago
other	a day ago
my_conda_env	3 minutes ago
mynotebook.ipynb	2 hours ago
conda-create.sh	22 minutes ago
slurm-273422.out	3 minutes ago
slurm-273422.st...	3 minutes ago

Bottom Status Bar:

Simple 1 \$ 1 ⚡ Terminal 1

- **pwd** – check current directory is the myproject folder
- (if not, **cd /bask/projects/m/myproject**)
- **ls** – list files and folders
- **sbatch conda-create.sh** – submit batch job
- **squeue -u \$USER** – check your job status
- **my_conda_env** – environment location

The screenshot shows a Jupyter Notebook interface with three tabs: "mynotebook.ipynb", "conda-create.sh", and "Terminal 1". The "Terminal 1" tab is active, displaying a series of Singularity commands:

```
Singularity> pwd  
/bask/projects/m/myproject  
Singularity> ls  
conda-create.sh myfolder1 myfolder2 mynotebook.ipynb other  
Singularity> sbatch conda-create.sh  
Submitted batch job 273422  
Singularity> squeue -u $USER  
JOBID PARTITION NAME USER ST TIME NODES NODELIST(REASON)  
273376 baskervil sys/dash wongj R 2:43:28 1 bask-pg0309u03a  
273422 baskervil conda-cr wongj PD 0:00 1 (Priority)  
Singularity> squeue -u $USER  
JOBID PARTITION NAME USER ST TIME NODES NODELIST(REASON)  
273376 baskervil sys/dash wongj R 2:43:43 1 bask-pg0309u03a  
273422 baskervil conda-cr wongj R 0:05 1 bask-pg0308u24a  
Singularity> squeue -u $USER  
JOBID PARTITION NAME USER ST TIME NODES NODELIST(REASON)  
273376 baskervil sys/dash wongj R 2:47:43 1 bask-pg0309u03a  
Singularity>
```

The left sidebar shows a file browser with the current directory set to "/myproject/". The contents of the directory are:

- myfolder1 (modified 22 days ago)
- myfolder2 (modified 22 days ago)
- other (modified a day ago)
- my_conda_env (selected, modified 3 minutes ago)
- mynotebook.ipynb (modified 2 hours ago)
- conda-create.sh (modified 22 minutes ago)
- slurm-273422.out (modified 3 minutes ago)
- slurm-273422.stats (modified 3 minutes ago)

A list of terminal commands is displayed below the terminal window:

- **pwd** – check current directory is the myproject folder
- (if not, **cd /bask/projects/m/myproject**)
- **ls** – list files and folders
- **sbatch conda-create.sh** – submit batch job
- **squeue -u \$USER** – check your job status
- **my_conda_env** – environment location
- **slurm-xxxxxx.out** – output file
- **slurm-xxxxxx.stats** – stats file

File Edit View Run Kernel Git Tabs Settings Help

New New Launcher Open from Path... Open from URL... New View for New Console for Activity Close Tab Shutdown Terminal Close All Tabs Save Save As... Save All Reload from Disk Revert to Checkpoint Rename... Download Save and Export Notebook As... Save Current Workspace As... Save Current Workspace Print... Log Out Shut Down Restart session

mynotebook.ipynb × conda-create.sh × Terminal 1 × +

```
singularity> pwd
/bask/projects/m/myproject
singularity> ls
conda-create.sh myfolder1 myfolder2 mynotebook.ipynb other
singularity> sbatch conda-create.sh
Submitted batch job 273422
singularity> squeue -u $USER
      JOBID PARTITION      NAME     USER ST      TIME  NODES NODELIST(REASON)
          273376 baskervil sys/dash   wongj R  2:43:28      1 bask-pg0309u03a
          273422 baskervil conda-cr   wongj PD    0:00      1 (Priority)
singularity> squeue -u $USER
      JOBID PARTITION      NAME     USER ST      TIME  NODES NODELIST(REASON)
          273376 baskervil sys/dash   wongj R  2:43:43      1 bask-pg0309u03a
          273422 baskervil conda-cr   wongj R  0:05      1 bask-pg0308u24a
singularity> squeue -u $USER
      JOBID PARTITION      NAME     USER ST      TIME  NODES NODELIST(REASON)
          273376 baskervil sys/dash   wongj R  2:47:43      1 bask-pg0309u03a
singularity>
```

- **pwd** – check current directory is the myproject folder
- (if not, **cd /bask/projects/m/myproject**)
- **ls** – list files and folders
- **sbatch conda-create.sh** – submit batch job
- **squeue -u \$USER** – check your job status
- **my_conda_env** – environment location
- **slurm-xxxxxx.out** – output file
- **slurm-xxxxxx.stats** – stats file

3. JupyterLab

b) Conda Environments

(ii) Relaunch JupyterLab and load new kernel

Relaunch JupyterLab and load new kernel

This time, tick to enable

Show Conda Environments

The screenshot shows the Baskerville OnDemand interface for launching an interactive session. The top navigation bar includes links for Home, My Interactive Sessions, and JupyterLab. The main content area is titled "Interactive Apps" and lists several options: JupyterLab (selected), GUIs, CST Studio Suite, Fiji, and RELION. To the right, the "JupyterLab" section is detailed:

- JupyterLab version:** 70aad71
- Description:** This app will launch a [JupyterLab](#) server (supporting [Python](#) and [Julia](#) kernels) on Baskerville.
- Kernel to load:** Python 3.8.6 (2020b / GCCcore-10.2.0)
- Show Conda Environments:** A checked checkbox with a descriptive text explaining it includes kernels for compatible Conda environments found in `~/.conda/environments.txt`. It links to the [Conda environment documentation](#).
- Number of hours:** A dropdown menu set to 4.
- Number of GPUs:** A dropdown menu set to 1.
- Baskerville Project:** A dropdown menu set to myproject, with a note about selecting a project.
- Queue:** A dropdown menu set to myqueue, with a note about selecting a queue/QoS.
- Launch:** A large blue button at the bottom.

A small note at the bottom states: * The JupyterLab session data for this session can be accessed under the [data root directory](#).

+ ☰ ⌂ ⌄ ⌅

Filter files by name

/ myproject /

Name	Last Modified
my_conda_env	2 minutes ago
myfolder1	an hour ago
myfolder2	an hour ago
other	an hour ago
conda-create.sh	an hour ago
mynotebook.ipynb	an hour ago
slurm-275558.out	an hour ago
slurm-275558.stats	an hour ago

Simple 1 \$ 1 ⌂ ⌄

mynotebook.ipynb conda-create.sh Terminal 1 +

Singularity>

+ X C ⌂

Filter files by name

/ myproject /

Name	Last Modified
my_conda_env	2 minutes ago
myfolder1	an hour ago
myfolder2	an hour ago
other	an hour ago
conda-create.sh	an hour ago
mynotebook.ipynb	an hour ago
slurm-275558.out	an hour ago
slurm-275558.stats	an hour ago

mynotebook.ipynb conda-create.sh Terminal 1 Launcher

myproject

Notebook

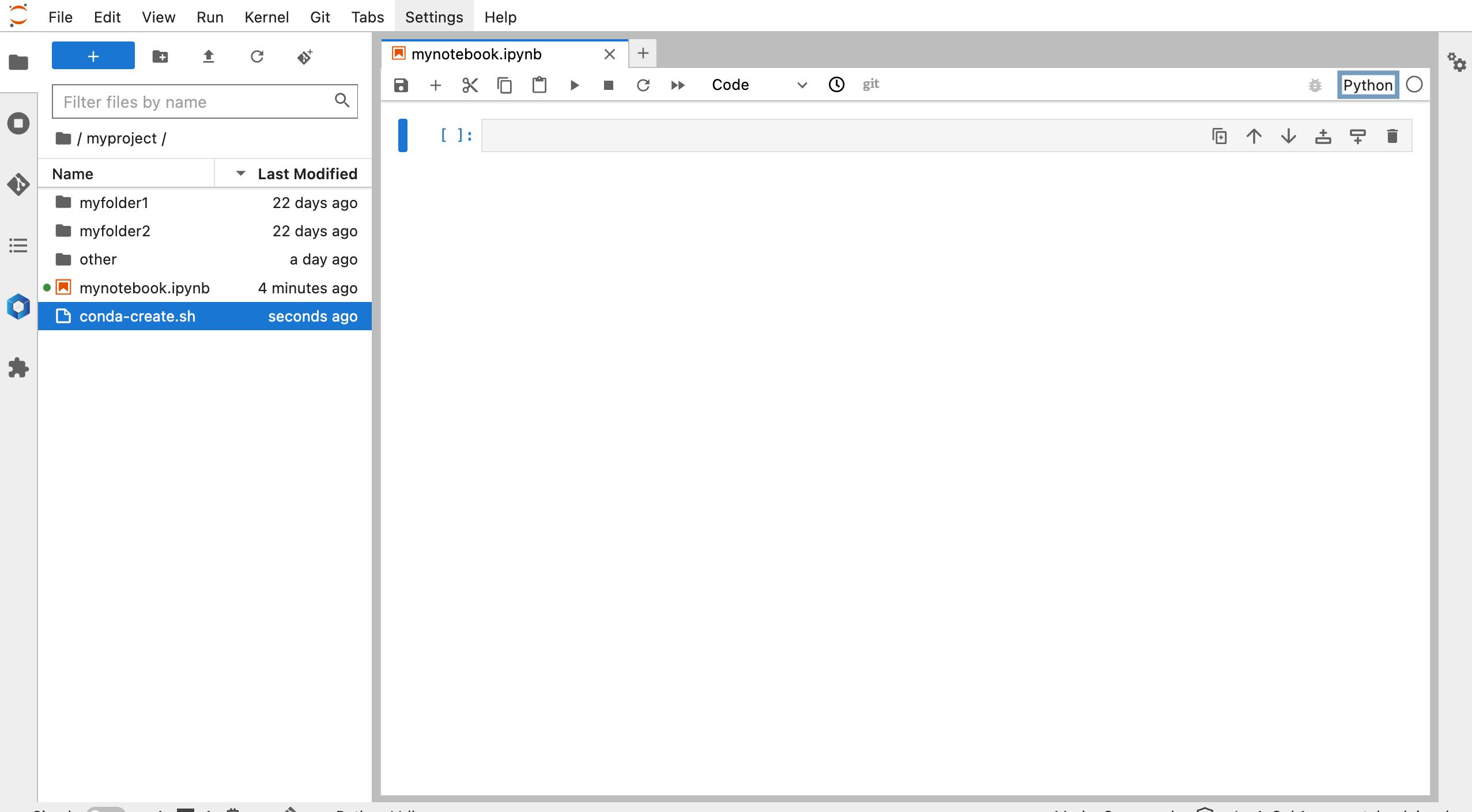
Python my_conda_env (Conda)

Console

Python my_conda_env (Conda)

Other

Terminal Text File Markdown File Python File Show Contextual Help



+ ☰ ⌂ ⌄ ⌅

Filter files by name

/ myproject /

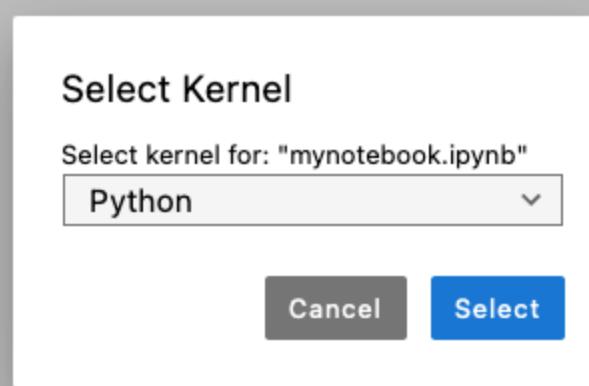
Name	Last Modified
my_conda_env	4 minutes ago
myfolder1	an hour ago
myfolder2	an hour ago
other	an hour ago
conda-create.sh	an hour ago
mynotebook.ipynb	an hour ago
slurm-275558.out	an hour ago
slurm-275558.stats	an hour ago

mynotebook.ipynb conda-create.sh Terminal 1 Launcher Python

[]:

Filter files by name

Up Down Add Remove



The screenshot shows a Jupyter Notebook interface with the following elements:

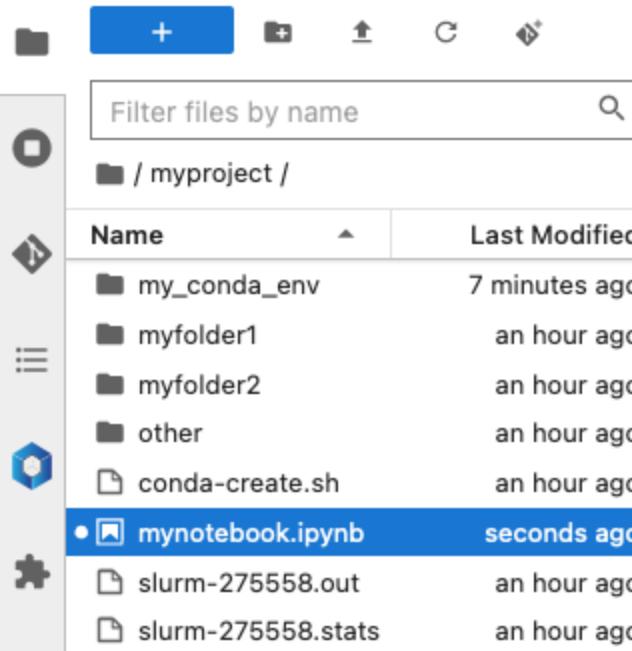
- File Browser:** On the left, a sidebar displays a tree view of a directory structure under "/ myproject /". The current path is "/ myproject /". The contents include:
 - my_conda_env (modified 4 minutes ago)
 - myfolder1 (modified an hour ago)
 - myfolder2 (modified an hour ago)
 - other (modified an hour ago)
 - conda-create.sh (modified an hour ago)
 - mynotebook.ipynb** (selected, modified an hour ago)
 - slurm-275558.out (modified an hour ago)
 - slurm-275558.stats (modified an hour ago)
- Terminal Tabs:** At the top, there are three tabs: "mynotebook.ipynb", "conda-create.sh", and "Terminal 1". The "Terminal 1" tab has "Code" and "git" sub-tabs.
- Launcher:** A tab labeled "Launcher" is visible at the top right.
- Select Kernel Dialog:** A modal dialog titled "Select Kernel" is centered on the screen. It asks "Select kernel for: "mynotebook.ipynb"" and shows a dropdown menu with "my_conda_env (Conda)" selected. There are "Cancel" and "Select" buttons at the bottom.
- Bottom Status Bar:** The status bar at the bottom shows "Simple" mode, a toggle switch, and the text "Python | Idle". It also indicates "Mode: Command" and "Ln 1, Col 1" with the file name "mynotebook.ipynb".

The screenshot shows a Jupyter Notebook interface with the following components:

- File Browser:** On the left, there's a sidebar with icons for file operations like creating, deleting, and moving files. Below it is a search bar labeled "Filter files by name". The current directory is "/ myproject /". A table lists files and folders:

Name	Last Modified
my_conda_env	a minute ago
myfolder1	an hour ago
myfolder2	an hour ago
other	an hour ago
conda-create.sh	an hour ago
mynotebook.ipynb	7 minutes ago
slurm-275558.out	an hour ago
slurm-275558.stats	an hour ago
- Code Editor:** The main area contains three tabs: "mynotebook.ipynb" (selected), "conda-create.sh", and "Terminal 1". The "mynotebook.ipynb" tab shows a single cell with the code:

```
[ ]:
```
- Terminal:** The "Terminal 1" tab shows the command "git" and the environment "my_conda_env (Conda)".
- Bottom Status Bar:** It displays "Simple" mode, a toggle switch, and the path "my_conda_env (Conda) | Idle". It also shows the mode as "Command" and the status as "Ln 1, Col 1" with the file name "mynotebook.ipynb".



File browser tabs: mynotebook.ipynb, conda-create.sh, Terminal 1.

Code editor tabs: my_conda_env (Conda).

```
[1]: import numpy
```

[]:

+ ☰ ⌂ ⌄ ⌅

Filter files by name

/ myproject /

Name	Last Modified
my_conda_env	8 minutes ago
myfolder1	an hour ago
myfolder2	an hour ago
other	an hour ago
conda-create.sh	an hour ago
mynotebook.ipynb	a minute ago
slurm-275558.out	an hour ago
slurm-275558.stats	an hour ago

mynotebook.ipynb conda-create.sh Terminal 1 + my_conda_env (Conda)

```
[1]: import numpy
[2]: numpy.__file__
[2]: '/bask/homes/w/wongj/myproject/my_conda_env/lib/python3.10/site-packages/numpy/__init__.py'
[ ]:
```

3. JupyterLab

b) Conda Environments

(iii) Modifying your environment



Filter files by name



/ myproject /

Name ▾ Last Modified

my_conda_env	3 minutes ago
myfolder1	3 minutes ago
myfolder2	3 minutes ago
other	3 minutes ago
conda-create.sh	a minute ago
mynotebook.ipynb	3 minutes ago
slurm-275558.out	3 minutes ago
slurm-275558.stats	3 minutes ago

mynotebook.ipynb



conda-create.sh



Terminal 1



Singularity> □





+

Filter files by name

/ myproject /

Name ▾ Last Modified

my_conda_env	2 minutes ago
myfolder1	2 minutes ago
myfolder2	2 minutes ago
other	2 minutes ago
conda-create.sh	seconds ago
• mynotebook.ipynb	2 minutes ago
slurm-275558.out	2 minutes ago
slurm-275558.stats	2 minutes ago

mynotebook.ipynb X conda-create.sh X Terminal 1 +

```
Singularity> module purge
Singularity> module load baskerville
Detected OS: Rocky 8.6
Singularity> module load bask-apps/live
Singularity> module load Miniconda3/4.10.3
Miniconda3/4.10.3
Singularity> eval "$(${EBROOTMINICONDA3}/bin/conda shell.bash hook)" # initialise Conda
(base) Singularity> export CONDA_ENV_PATH="/bask/projects/e/edmondac-rsg/wongj/my_conda_env"
(base) Singularity> export CONDA_PKGS_DIRS="/tmp"
(base) Singularity> 
```





+

Filter files by name

/ myproject /

Name ▾ Last Modified

my_conda_env 6 minutes ago

myfolder1 6 minutes ago

myfolder2 6 minutes ago

other 6 minutes ago

conda-create.sh 4 minutes ago

mynotebook.ipynb 6 minutes ago

slurm-275558.out 6 minutes ago

slurm-275558.stats 6 minutes ago

mynotebook.ipynb X conda-create.sh X Terminal 1 X +

```
Singularity> module purge
Singularity> module load baskerville
Detected OS: Rocky 8.6
Singularity> module load bask-apps/live
Singularity> module load Miniconda3/4.10.3
Miniconda3/4.10.3
Singularity> eval "$(${EBROOTMINICONDA3}/bin/conda shell.bash hook)" # initialise Conda
(base) Singularity> export CONDA_ENV_PATH="/bask/projects/e/edmondac-rsg/wongj/my_conda_env"
(base) Singularity> export CONDA_PKGS_DIRS="/tmp"
(base) Singularity> conda activate "${CONDA_ENV_PATH}"
(/bask/projects/e/edmondac-rsg/wongj/my_conda_env) Singularity> 
```



+ + ↑ ⌂ ⌂+

Filter files by name

/ myproject /

Name	Last Modified
my_conda_env	6 minutes ago
myfolder1	14 minutes ago
myfolder2	14 minutes ago
other	14 minutes ago
conda-create.sh	12 minutes ago
mynotebook.ipynb	14 minutes ago
slurm-275558.out	14 minutes ago
slurm-275558.stats	14 minutes ago

mynotebook.ipynb conda-create.sh Terminal 1

```
Singularity> module purge
Singularity> module load baskerville
Detected OS: Rocky 8.6
Singularity> module load bask-apps/live
Singularity> module load Miniconda3/4.10.3
Miniconda3/4.10.3
Singularity> eval "$(${EBROOTMINICONDA3}/bin/conda shell.bash hook)" # initialise Conda
(base) Singularity> export CONDA_ENV_PATH="/bask/projects/e/edmondac-rsg/wongj/my_conda_env"
(base) Singularity> export CONDA_PKGS_DIRS="/tmp"
(base) Singularity> conda activate "${CONDA_ENV_PATH}"
(/bask/projects/e/edmondac-rsg/wongj/my_conda_env) Singularity> conda list
# packages in environment at /bask/projects/e/edmondac-rsg/wongj/my_conda_env:
#
# Name           Version      Build  Channel
_libgcc_mutex   0.1          main
_openmp_mutex   5.1          1_gnu
asttokens       2.0.5        pyhd3eb1b0_0
backcall        0.2.0        pyhd3eb1b0_0
blas            1.0          mkl
bzip2           1.0.8        h7b6447c_0
ca-certificates 2022.10.11  h06a4308_0
certifi          2022.9.24   pypi_0    pypi
debugpy          1.5.1        pypi_0    pypi
decorator        5.1.1        pyhd3eb1b0_0
entrypoints      0.4          pypi_0    pypi
executing        0.8.3        pyhd3eb1b0_0
intel-openmp     2021.4.0   h06a4308_3561
ipykernel        6.15.2       pypi_0    pypi
ipython          8.4.0        pypi_0    pypi
jedi             0.18.1       pypi_0    pypi
jupyter-client   7.3.5        pypi_0    pypi
jupyter-core     4.11.1       pypi_0    pypi
jupyter_client   7.3.5        py310h06a4308_0
jupyter_core     4.11.1       py310h06a4308_0
ld_impl_linux-64 2.38         h1181459_1
```

File Edit View Run Kernel Git Tabs Settings Help

+ 📂 ⏚ ⏵ ⏴ ⏵ +

Filter files by name

/ myproject /

Name	Last Modified
my_conda_env	6 minutes ago
myfolder1	15 minutes ago
myfolder2	15 minutes ago
other	15 minutes ago
conda-create.sh	13 minutes ago
mynotebook.ipynb	15 minutes ago
slurm-275558.out	15 minutes ago
slurm-275558.stats	15 minutes ago

mynotebook.ipynb conda-create.sh Terminal 1

Package	Version	Build	Source	Platform
mkl_random	1.2.2	py310h00e6091_0		
ncurses	6.3	h5eee18b_3		
nest-asyncio	1.5.5	pypi_0	pypi	
numpy	1.23.3	pypi_0	pypi	
numpy-base	1.23.3	py310h8e6c178_0		
openssl	1.1.1q	h7f8727e_0		
packaging	21.3	pyhd3eb1b0_0		
parso	0.8.3	pyhd3eb1b0_0		
pexpect	4.8.0	pyhd3eb1b0_3		
pickleshare	0.7.5	pyhd3eb1b0_1003		
pip	22.2.2	pypi_0	pypi	
prompt-toolkit	3.0.20	pyhd3eb1b0_0		
psutil	5.9.0	pypi_0	pypi	
ptyprocess	0.7.0	pyhd3eb1b0_2		
pure_eval	0.2.2	pyhd3eb1b0_0		
pygments	2.11.2	pyhd3eb1b0_0		
pyparsing	3.0.9	pypi_0	pypi	
python	3.10.6	haa1d7c7_1		
python-dateutil	2.8.2	pyhd3eb1b0_0		
pyzmq	23.2.0	pypi_0	pypi	
readline	8.2	h5eee18b_0		
setuptools	65.5.0	pypi_0	pypi	
six	1.16.0	pyhd3eb1b0_1		
sqlite	3.39.3	h5082296_0		
stack_data	0.2.0	pyhd3eb1b0_0		
tk	8.6.12	h1ccaba5_0		
tornado	6.2	pypi_0	pypi	
traitlets	5.1.1	pyhd3eb1b0_0		
tzdata	2022f	h04d1e81_0		
wcwidth	0.2.5	pyhd3eb1b0_0		
wheel	0.37.1	pyhd3eb1b0_0		
xz	5.2.6	h5eee18b_0		
zeromq	4.3.4	h2531618_0		
zlib	1.2.13	h5eee18b_0		

(/bask/projects/e/edmondac-rsg/wongj/my_conda_env) Singularity>

Simple 1 \$ 1 ⚡ Terminal 1

+ + ↑ ⌂ ⌂+

Filter files by name

/ myproject /

Name	Last Modified
my_conda_env	7 minutes ago
myfolder1	16 minutes ago
myfolder2	16 minutes ago
other	16 minutes ago
conda-create.sh	14 minutes ago
mynotebook.ipynb	16 minutes ago
slurm-275558.out	16 minutes ago
slurm-275558.stats	16 minutes ago

mynotebook.ipynb conda-create.sh Terminal 1

Package	Version	Build	Source	Platform
mkl_random	1.2.2	py310h00e6091_0		
ncurses	6.3	h5eee18b_3		
nest-asyncio	1.5.5	pypi_0	pypi	
numpy	1.23.3	pypi_0	pypi	
numpy-base	1.23.3	py310h8e6c178_0		
openssl	1.1.1q	h7f8727e_0		
packaging	21.3	pyhd3eb1b0_0		
parso	0.8.3	pyhd3eb1b0_0		
pexpect	4.8.0	pyhd3eb1b0_3		
pickleshare	0.7.5	pyhd3eb1b0_1003		
pip	22.2.2	pypi_0	pypi	
prompt-toolkit	3.0.20	pyhd3eb1b0_0		
psutil	5.9.0	pypi_0	pypi	
ptyprocess	0.7.0	pyhd3eb1b0_2		
pure_eval	0.2.2	pyhd3eb1b0_0		
pygments	2.11.2	pyhd3eb1b0_0		
pyparsing	3.0.9	pypi_0	pypi	
python	3.10.6	haa1d7c7_1		
python-dateutil	2.8.2	pyhd3eb1b0_0		
pyzmq	23.2.0	pypi_0	pypi	
readline	8.2	h5eee18b_0		
setuptools	65.5.0	pypi_0	pypi	
six	1.16.0	pyhd3eb1b0_1		
sqlite	3.39.3	h5082296_0		
stack_data	0.2.0	pyhd3eb1b0_0		
tk	8.6.12	h1ccaba5_0		
tornado	6.2	pypi_0	pypi	
traitlets	5.1.1	pyhd3eb1b0_0		
tzdata	2022f	h04d1e81_0		
wcwidth	0.2.5	pyhd3eb1b0_0		
wheel	0.37.1	pyhd3eb1b0_0		
xz	5.2.6	h5eee18b_0		
zeromq	4.3.4	h2531618_0		
zlib	1.2.13	h5eee18b_0		

```
(/bask/projects/e/edmondac-rsg/wongj/my_conda_env) Singularity> conda install pandas --yes
```

+ X + ↑ C ⌂

Filter files by name

/ myproject /

Name	Last Modified
my_conda_env	seconds ago
myfolder1	17 minutes ago
myfolder2	17 minutes ago
other	17 minutes ago
conda-create.sh	15 minutes ago
mynotebook.ipynb	17 minutes ago
slurm-275558.out	17 minutes ago
slurm-275558.stats	17 minutes ago

mynotebook.ipynb X conda-create.sh X Terminal 1 +

```
environment location: /bask/projects/e/edmondac-rsg/wongj/my_conda_env

added / updated specs:
- pandas

The following packages will be downloaded:

      package          | build
-----+-----
bottleneck-1.3.5 | py310ha9d4c09_0   274 KB
numexpr-2.8.3     | py310hcea2de6_0   321 KB
pandas-1.4.4      | py310h6a678d5_0   25.5 MB
pytz-2022.1       | py310h06a4308_0   196 KB
-----+-----
                                         Total:    26.3 MB

The following NEW packages will be INSTALLED:

bottleneck      pkgs/main/linux-64::bottleneck-1.3.5-py310ha9d4c09_0
numexpr          pkgs/main/linux-64::numexpr-2.8.3-py310hcea2de6_0
pandas           pkgs/main/linux-64::pandas-1.4.4-py310h6a678d5_0
pytz              pkgs/main/linux-64::pytz-2022.1-py310h06a4308_0

Downloading and Extracting Packages
bottleneck-1.3.5 | 274 KB    | #####| 100%
numexpr-2.8.3     | 321 KB    | #####| 100%
pandas-1.4.4      | 25.5 MB   | #####| 100%
pytz-2022.1       | 196 KB    | #####| 100%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
(/bask/projects/e/edmondac-rsg/wongj/my_conda_env) Singularity> 
```

File Edit View Run Kernel Git Tabs Settings Help

+ Filter files by name

/ myproject /

Name	Last Modified
my_conda_env	a minute ago
myfolder1	an hour ago
myfolder2	an hour ago
other	an hour ago
conda-create.sh	an hour ago
mynotebook.ipynb	14 minutes ago
slurm-275558.out	an hour ago
slurm-275558.stats	an hour ago

mynotebook.ipynb conda-create.sh Terminal 1 + my_conda_env (Conda)

[1]: `import numpy`

[2]: `numpy.__file__`

[2]: '/bask/homes/w/wongj/myproject/my_conda_env/lib/python3.10/site-packages/numpy/__init__.py'

[]:

Simple 1 \$ 1 my_conda_env (Conda) | Idle Mode: Command Ln 1, Col 1 mynotebook.ipynb

The screenshot shows a Jupyter Notebook interface with the following components:

- Top Bar:** File, Edit, View, Run, Kernel, Git, Tabs, Settings, Help.
- Left Sidebar:** Includes icons for file operations (+, folder, up, refresh, etc.), a search bar ("Filter files by name"), and a list of files and folders in the current directory (`/myproject/`). The file `mynotebook.ipynb` is selected, indicated by a blue background.
- File Explorer:** A sidebar showing the project structure with items like `myconda_env`, `myfolder1`, `myfolder2`, `other`, `conda-create.sh`, `mynotebook.ipynb`, `slurm-275558.out`, and `slurm-275558.stats`.
- Code Editor:** The main workspace displays the following code execution history:
 - [1]: `import numpy`
 - [2]: `numpy.__file__`
 - [2]: '/bask/homes/w/wongj/myproject/myconda_env/lib/python3.10/site-packages/numpy/__init__.py'
 - [3]: `import pandas`
 - [4]: `pandas.__file__`
 - [4]: '/bask/homes/w/wongj/myproject/myconda_env/lib/python3.10/site-packages/pandas/__init__.py'
 - []: (An empty cell with a blue border, likely where the next command will be entered.)
- Terminal:** A terminal tab titled "Terminal 1" is visible at the top right.
- Status Bar:** At the bottom, it shows "Simple" mode, a toggle switch, the terminal name "myconda_env (Conda) | Idle", "Mode: Edit", a shield icon, "Ln 1, Col 1", and the notebook name "mynotebook.ipynb".

More Conda commands

Consult the Cheat Sheet

<https://docs.conda.io/projects/conda/en/latest/user-guide/cheatsheet.html>

<https://github.com/baskerville-hpc/baskerville-portal>

QUICK START	
Tip: It is recommended to create a new environment for any new project or workflow.	
verify conda install and check version	conda info
update conda in base environment	conda update -n base conda
install latest anaconda distribution (see release notes)	conda install anaconda=2022.05
create a new environment (tip: name environment descriptively)	conda create --name ENVNAME
activate environment (do this before installing packages)	conda activate ENVNAME
CHANNELS AND PACKAGES	
Tip: Package dependencies and platform specifics are automatically resolved when using conda.	
list installed packages	conda list
list installed packages with source info	conda list --show-channel-urls
update all packages	conda update --all
install a package from specific channel	conda install -c CHANNELNAME PKG1 PKG2
install specific version of package	conda install PKGNAME=3.1.4
install a package from specific channel	conda install CHANNELNAME::PKGNAME
install package with AND logic	conda install "PKGNAME>2.5,<3.2"
install package with OR logic	conda install "PKGNAME [version='2.5 3.2']"
uninstall package	conda uninstall PKGNAME
view channel sources	conda config --show-sources
add channel	conda config --add channels CHANNELNAME
set default channel for pkg fetching (targets first channel in channel sources)	conda config --set channel_priority strict
WORKING WITH CONDA ENVIRONMENTS	
Tip: List environments at the beginning of your session. Environments with an asterisk are active.	
list all environments and locations	conda env list
list all packages + source channels	conda list -n ENVNAME --show-channel-urls
install packages in environment	conda install -n ENVNAME PKG1 PKG2
remove package from environment	conda uninstall PKGNAME -n ENVNAME
update all packages in environment	conda update --all -n ENVNAME

Tips

- Do **not** mix Module Loading and Conda Environments
 - This can result in version conflicts and incompatible software
- Module loading is preferred where possible
 - These system-wide applications are built and optimised specifically for Baskerville, resulting in better performance and compatibility
 - Not easily customisable if the module you want does not exist, although we are happy to install more module applications at your request
- Conda Environments are customisable, but can be large
 - When modifying your Conda environment, add the **conda install** commands into your shell script for reproducibility (helpful for troubleshooting)
 - Run **conda env remove --prefix <\$CONDA_ENV_PATH>** to delete an environment
 - Run **conda clean --all** to clear your Conda cache in your home folder
- Remember to shutdown your JupyterLab notebook when you are done
- Consult <https://docs.baskerville.ac.uk/portal/portal/> for more advice

Final Task – PyTorch Conda Environment

- Create another Conda environment using a shell script
- **conda install --yes pytorch torchvision pytorch-cuda=11.6 -c pytorch -c nvidia**
- Launch a JupyterLab server
 - **import torch**
 - Check **torch.__path__**
 - Check **torch.__version__**
 - Query visible GPU devices with the commands from **pytorch-gpu.py**



```
import torch

use_cuda = torch.cuda.is_available()

if use_cuda:
    print('__CUDNN VERSION:', torch.backends.cudnn.version())
    print('__Number CUDA Devices:', torch.cuda.device_count())
    print('__CUDA Device Name:', torch.cuda.get_device_name(0))
    print('__CUDA Device Total Memory [GB]:', torch.cuda.get_device_properties(0).total_memory/1e9)
```

pytorch-gpu.py

Congratulations!

Useful links:

- <https://docs.baskerville.ac.uk/>
- <https://admin.baskerville.ac.uk/>
- <https://apps.baskerville.ac.uk/>
- <https://portal.baskerville.ac.uk/>
- <https://github.com/baskerville-hpc/baskerville-portal>

Email:

- baskerville-tier2-support@contacts.bham.ac.uk

