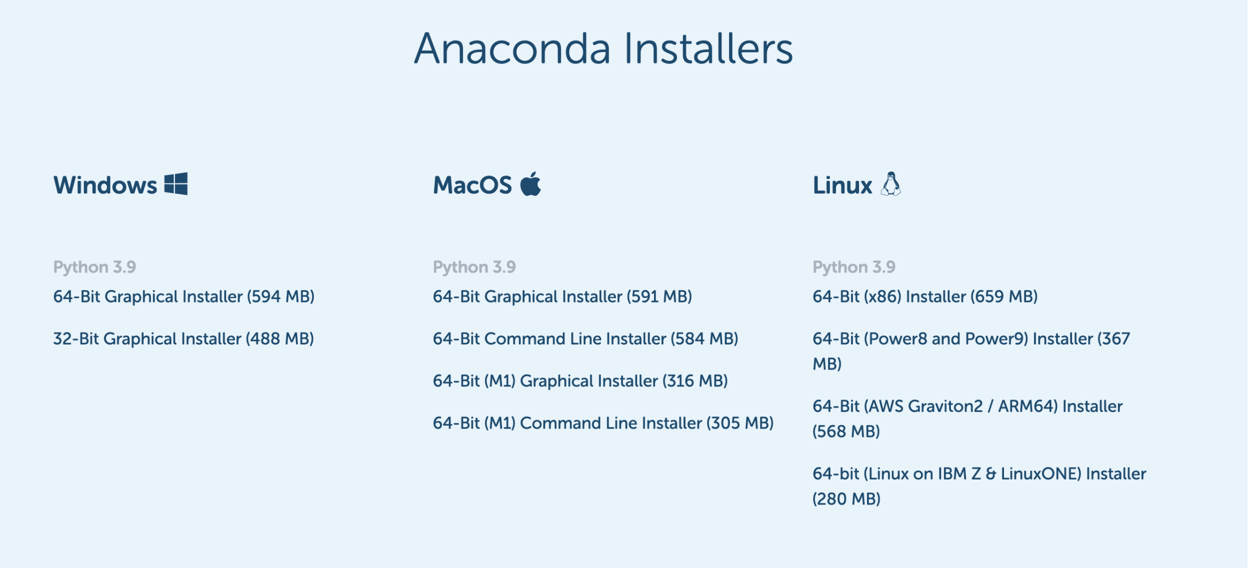
Setup Dev Environment - macOS

This tutorial walks you through the steps required to create a clean Python virtual environment for our project. Feel free to skip sections that you already know.

Conda

Conda is an open-source package management system compatible with macOS, Windows, and Linux. We will use it to install and manage all Python packages for this class.

The easiest way to install Conda is through the Anaconda installer. First, go to https://www.anaconda.com/products/distribution, then scroll down to the bottom and download the correct version based on your device.



Anaconda Installers

Windows

Python 3.9

64-Bit Graphical Installer (594 MB)

32-Bit Graphical Installer (488 MB)

MacOS

Python 3.9

64-Bit Graphical Installer (591 MB)

64-Bit Command Line Installer (584 MB)

64-Bit (M1) Graphical Installer (316 MB)

64-Bit (M1) Command Line Installer (305 MB)

Linux

Python 3.9

64-Bit (x86) Installer (659 MB)

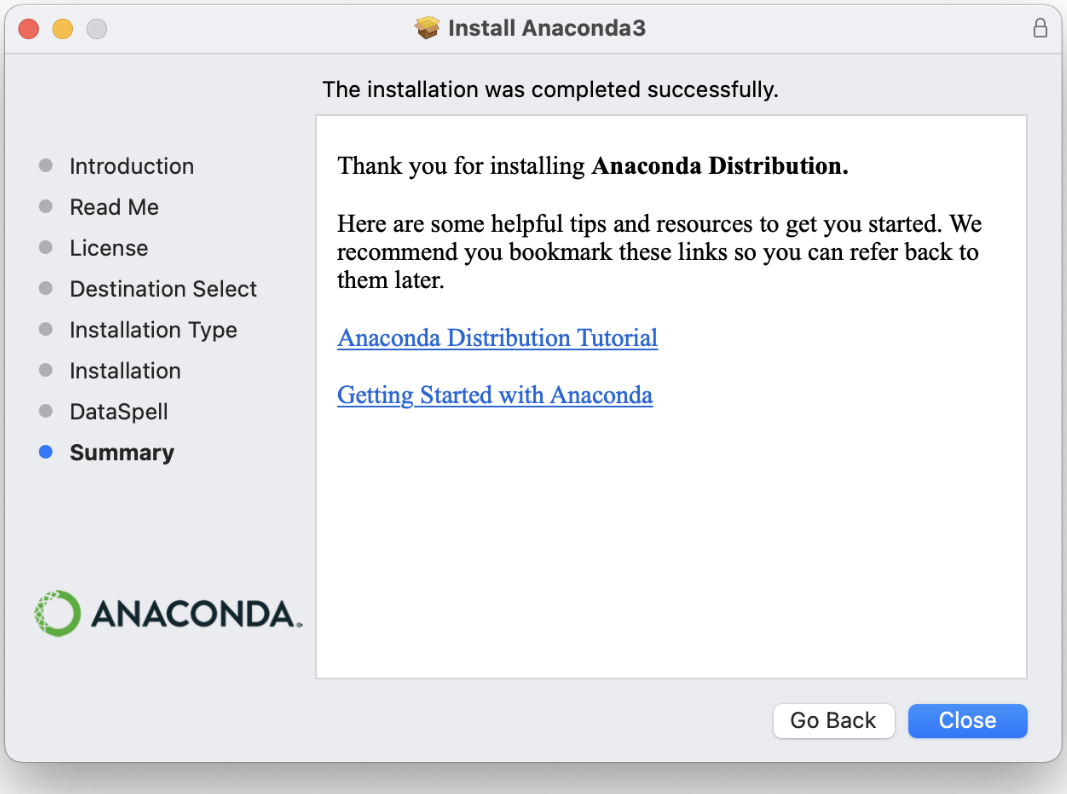
64-Bit (Power8 and Power9) Installer (367 MB)

64-Bit (AWS Graviton2 / ARM64) Installer (568 MB)

64-bit (Linux on IBM Z & LinuxONE) Installer (280 MB)

If your mac has Apple M Series CPU, download the “**64-Bit (M1) Graphical Installer**”. Otherwise, download “**64-Bit Graphical Installer.**”

Open the installer, and follow the instructions to install Anaconda. Once you’re done, you should see the following:



The installation was completed successfully.

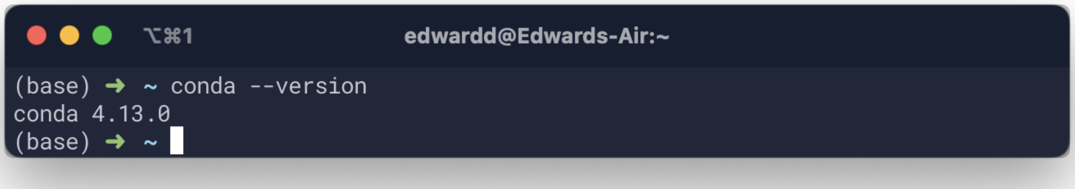
Thank you for installing **Anaconda Distribution.**

Here are some helpful tips and resources to get you started. We recommend you bookmark these links so you can refer back to them later.

Anaconda Distribution Tutorial

Getting Started with Anaconda

To verify Conda is installed successfully, open the **Terminal** app, and type in `conda --version`. The version info should be printed out.



```
(base) ➜ ~ conda --version
conda 4.13.0
(base) ➜ ~
```

Create a Virtual Environment

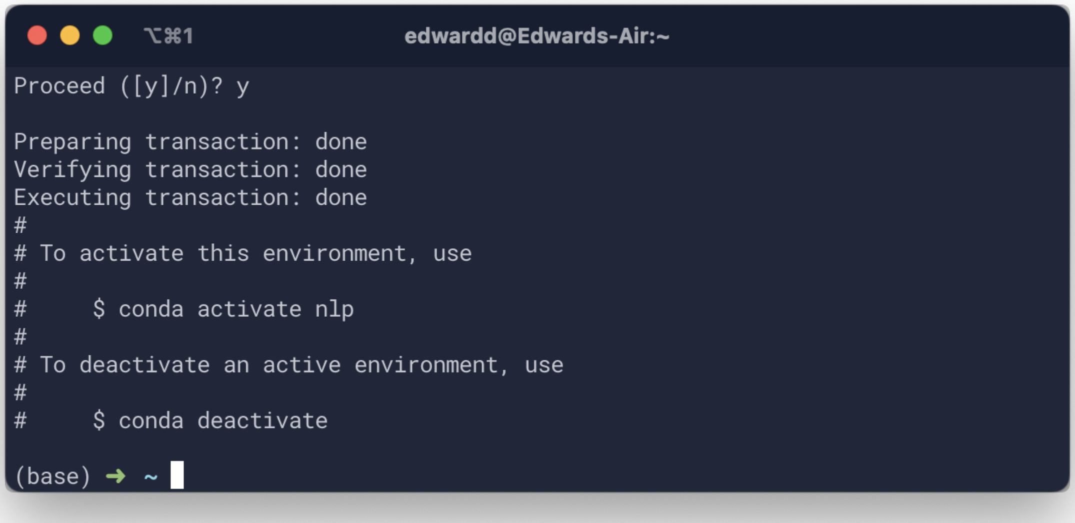
Conda uses virtual environments to manage packages. Virtual environments are separated so you can install different versions of packages in different virtual environments. You can also freeze and export a virtual environment so that other developers can clone the same setting on their device.

To create a new virtual environment, run the following command in your **Terminal** app.

```
conda create --name nlp python=3.9
```

When prompted `Proceed ([y]/n)?`, type `y` and hit return to proceed.

This will create a new virtual environment called `nlp`, which comes with a Python binary of version 3.9. Once the command finishes running, you should see something like the following:



```
Proceed ([y]/n)? y

Preparing transaction: done
Verifying transaction: done
Executing transaction: done
#
# To activate this environment, use
#
#     $ conda activate nlp
#
# To deactivate an active environment, use
#
#     $ conda deactivate

(base) ➜ ~ 
```

As prompted, in the future, whenever you want to use or make any changes to this virtual environment, you need to activate it first by `conda activate nlp`.

Once you're done, you can deactivate the virtual environment with `conda deactivate`. This will not remove the environment; just deactivate it. Alternatively, you could simply close the **Terminal** app.

Install Packages

To install packages into the virtual environment, run the following commands in your **Terminal**.

```
# This command activates the virtual environment we just created
conda activate nlp

# This command installs jupyter lib so that we can run jupyter notebooks
# later in VSCode
pip install jupyter
```

As the class proceed, we will install more packages as needed. Most of them can be installed in the same way.

Tensorflow

Intel based Mac

Simply use `pip`:

```
# Assuming the virtual environment is activate
pip install tensorflow
```

Apple M Series Macs

First we need to install some additional dependencies using `conda`

```
conda install -c apple tensorflow-deps -y
```

Then we can use `pip` to install Tensorflow similar as before. Note that the package name is `tensorflow-macos`. Optionally, we can install the Tensorflow Metal plugin to use the GPU on the M series chip.

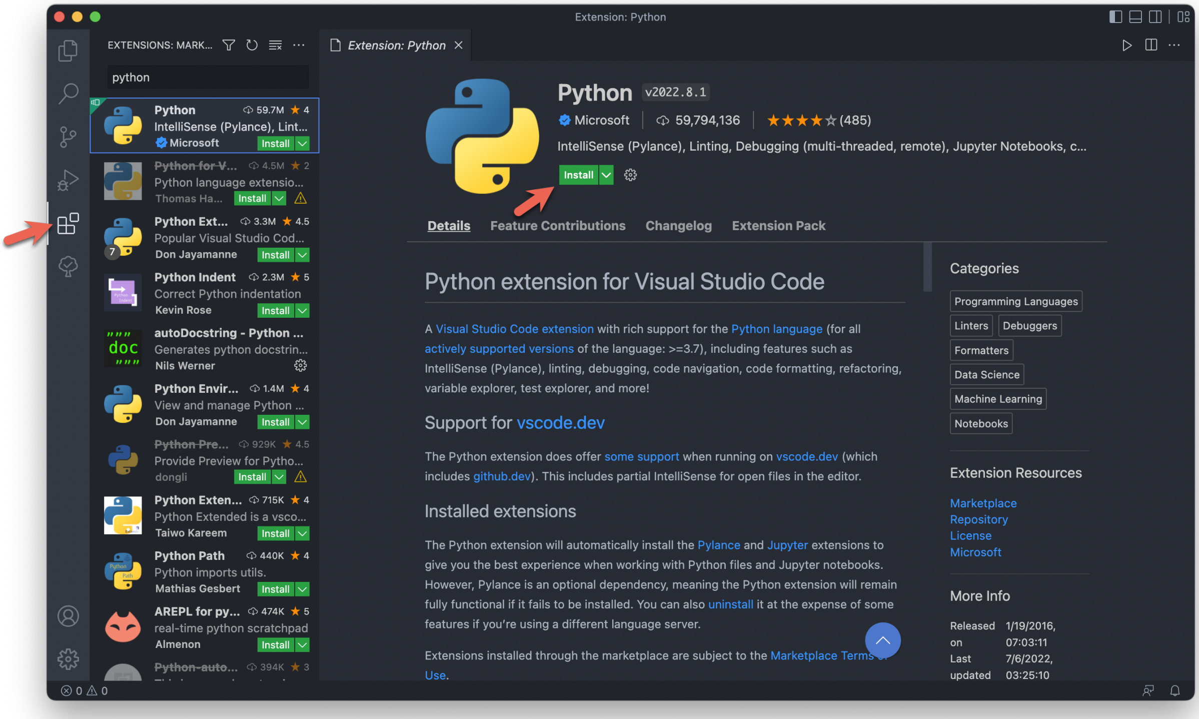
```
# Assuming the virtual environment is activate
pip install tensorflow-macos

# (Optional) Install metal plugin to speed up training with GPU
pip install tensorflow-metal
```

> 💡 We will not train large models on your laptop, so please don’t worry if your device does not come with a GPU or is not powerful enough.

VSCode

Go to https://code.visualstudio.com/, download and install VSCode. Then open VSCode and install the `Python`:



Python extension for Visual Studio Code

A Visual Studio Code extension with rich support for the Python language (for all actively supported versions of the language: >=3.7), including features such as IntelliSense (Pylance), linting, debugging, code navigation, code formatting, refactoring, variable explorer, test explorer, and more!

Support for vscode.dev

The Python extension does offer some support when running on vscode.dev (which includes github.dev). This includes partial IntelliSense for open files in the editor.

Installed extensions

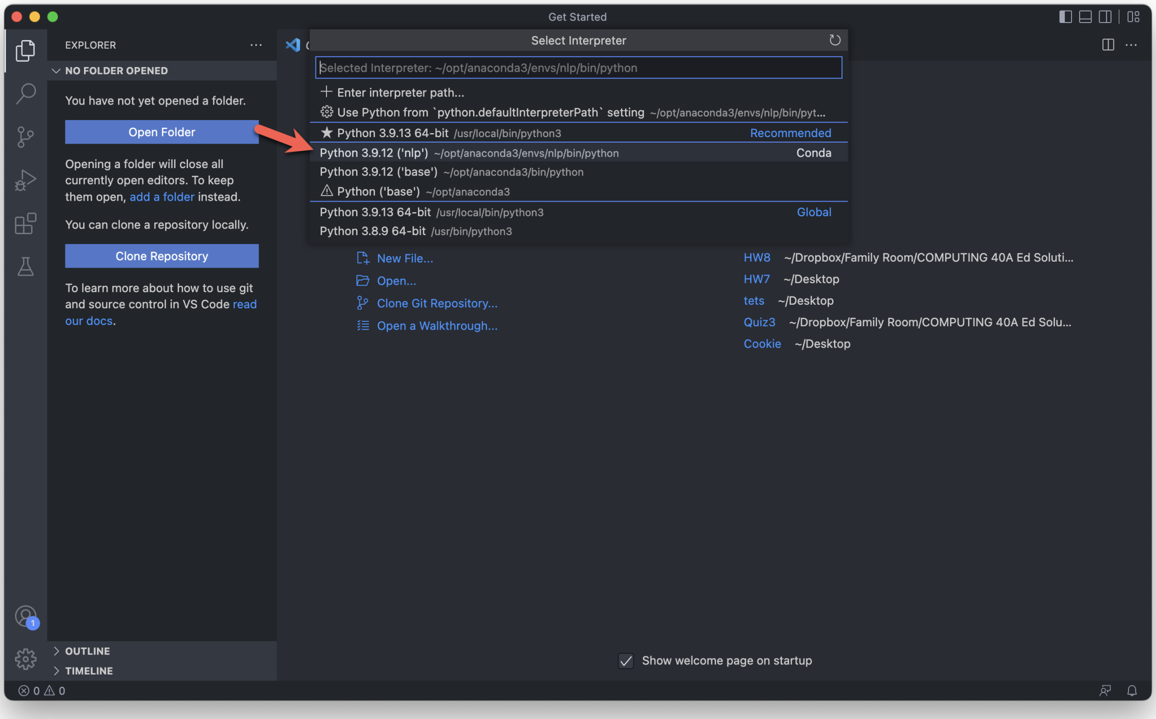
The Python extension will automatically install the Pylance and Jupyter extensions to give you the best experience when working with Python files and Jupyter notebooks. However, Pylance is an optional dependency, meaning the Python extension will remain fully functional if it fails to be installed. You can also uninstall it at the expense of some features if you're using a different language server.

Extensions installed through the marketplace are subject to the Marketplace Terms of Use.

💡 Here is a video walkthrough that you can refer to for each step

Create Your First Python Script

Press ⌘**+shift+p** to open the action panel, find **Python: Select Interpreter.** Choose the virtual environment you just created. This tells VSCode where to look for packages to import as we write codes.



Get Started

EXPLORER

NO FOLDER OPENED

You have not yet opened a folder.

Open Folder

Opening a folder will close all currently open editors. To keep them open, add a folder instead.

You can clone a repository locally.

Clone Repository

To learn more about how to use git and source control in VS Code read our docs.

Select Interpreter

Selected Interpreter: ~/opt/anaconda3/envs/nlp/bin/python

+ Enter interpreter path...
Use Python from `python.defaultInterpreterPath` setting ~/opt/anaconda3/envs/nlp/bin/pyt...
★ Python 3.9.13 64-bit /usr/local/bin/python3 Recommended
Python 3.9.12 ('nlp') ~/opt/anaconda3/envs/nlp/bin/python Conda
Python 3.9.12 ('base') ~/opt/anaconda3/bin/python
⚠ Python ('base') ~/opt/anaconda3
Python 3.9.13 64-bit /usr/local/bin/python3 Global
Python 3.8.9 64-bit /usr/bin/python3

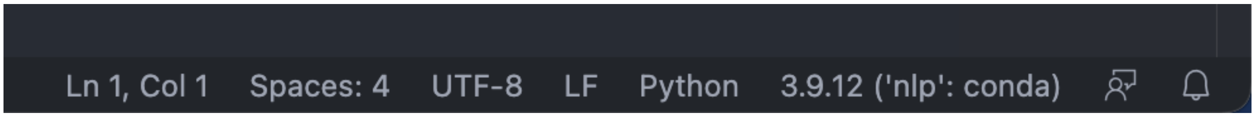
New File...
Open...
Clone Git Repository...
Open a Walkthrough...

HW8 ~/Dropbox/Family Room/COMPUTING 40A Ed Soluti...
HW7 ~/Desktop
tets ~/Desktop
Quiz3 ~/Dropbox/Family Room/COMPUTING 40A Ed Solu...
Cookie ~/Desktop

☑ Show welcome page on startup

OUTLINE
TIMELINE

Create a new empty Python file. Make sure the status bar at the lower right corner indicates the correct virtual environment is being used:



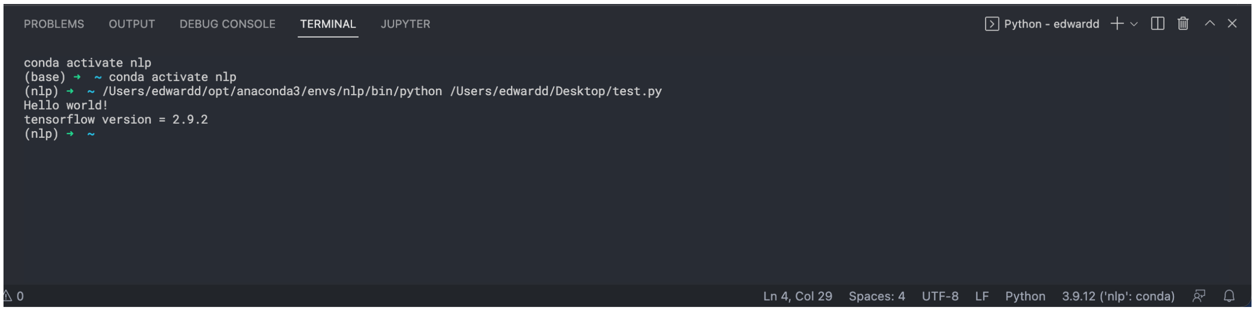
Ln 1, Col 1 Spaces: 4 UTF-8 LF Python 3.9.12 ('nlp': conda)

Let’s add some code to the empty file to verify that the Tensorflow installation is successful.

```
import tensorflow as tf

print("Hello world!")
print(f"tensorflow version = {tf.__version__}")
```

Save it as `test.py` and click the ▶️ button on the upper right corner to run the Python file. If everything is installed correctly, you should see something like the following:

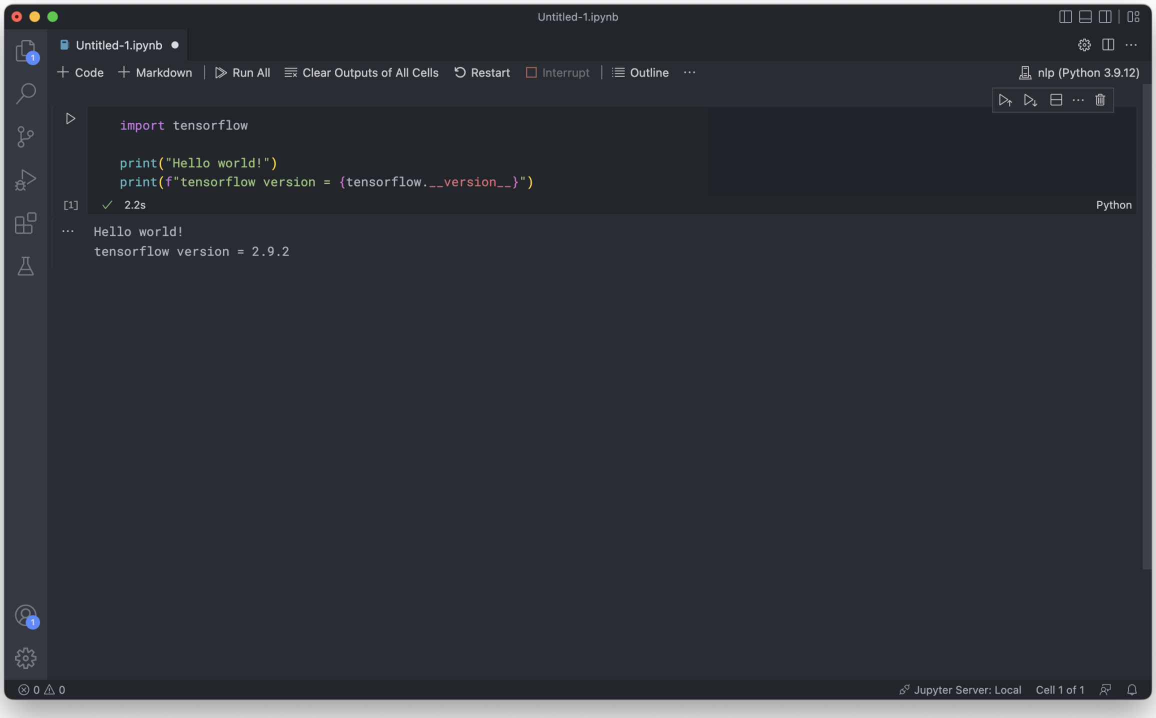


```
conda activate nlp
(base) ➜  ~ conda activate nlp
(nlp) ➜  ~ /Users/edwardd/opt/anaconda3/envs/nlp/bin/python /Users/edwardd/Desktop/test.py
Hello world!
tensorflow version = 2.9.2
(nlp) ➜  ~
```

Create Your First Notebook

Press ⌘**+shift+p** again, then choose **Create: New Jupyter Notebook** to create a new Jupyter notebook. Add the above code to the first cell, then click the **Run All** button.

You should see the output right below your code block.



```
import tensorflow

print("Hello world!")
print(f"tensorflow version = {tensorflow.__version__}")
```

```
Hello world!
tensorflow version = 2.9.2
```

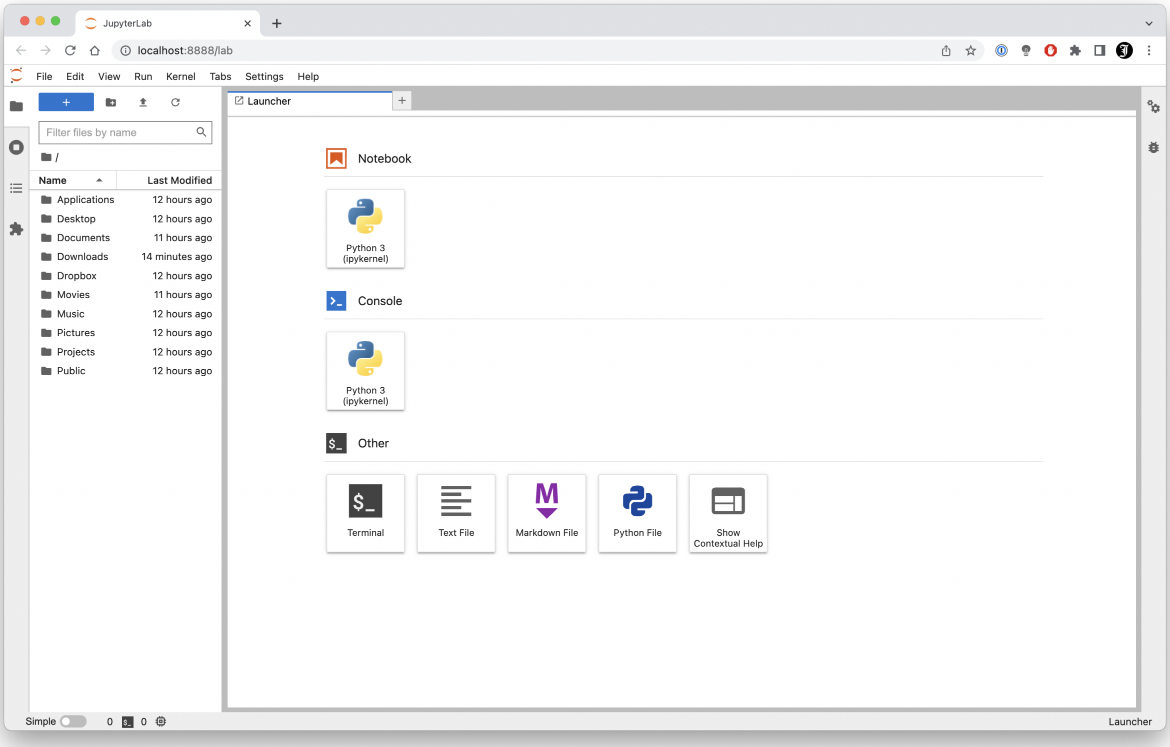
Jupyter Lab

Alternatively, if you're familiar with Jupyter notebook UI, you could also use Jupyter Lab to read and write your notebooks.

```
# Assuming the virtual environment is activate
pip install jupyterlab

# Launch Jupyter Hub UI
jupyter-lab
```

It will automatically open your browser and direct you to the landing page.



Name	Last Modified
Applications	12 hours ago
Desktop	12 hours ago
Documents	11 hours ago
Downloads	14 minutes ago
Dropbox	12 hours ago
Movies	11 hours ago
Music	12 hours ago
Pictures	12 hours ago
Projects	12 hours ago
Public	12 hours ago

> 💡 Do NOT close your Terminal app when using the Jupyter Hub, otherwise, the kernel will be shut down.

🎉🎉🎉 Congratulations! You’ve now successfully setup your local dev environment!