

How to run a notebook Locally?



ZerocostDL4Mic | an open platform to use Deep Learning in microscopy

GitHub page <https://github.com/HenriquesLab/ZeroCostDL4Mic/wiki>

Video Tutorial: <https://github.com/HenriquesLab/ZeroCostDL4Mic/wiki>

Google Colab lets users connect to a local runtime using Jupyter notebook on. This allows you to run Zerocost4DLMic notebooks on your local machine and have access to your local file system. This can be helpful for those who wish to train models using their own GPU power.

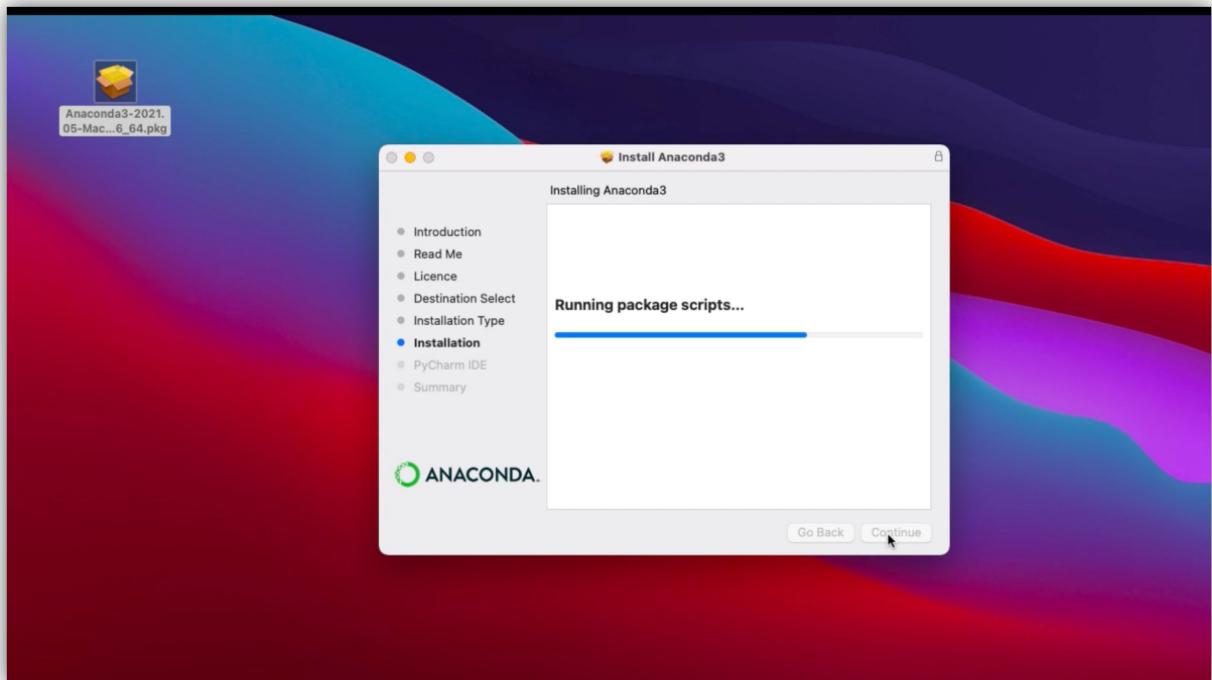
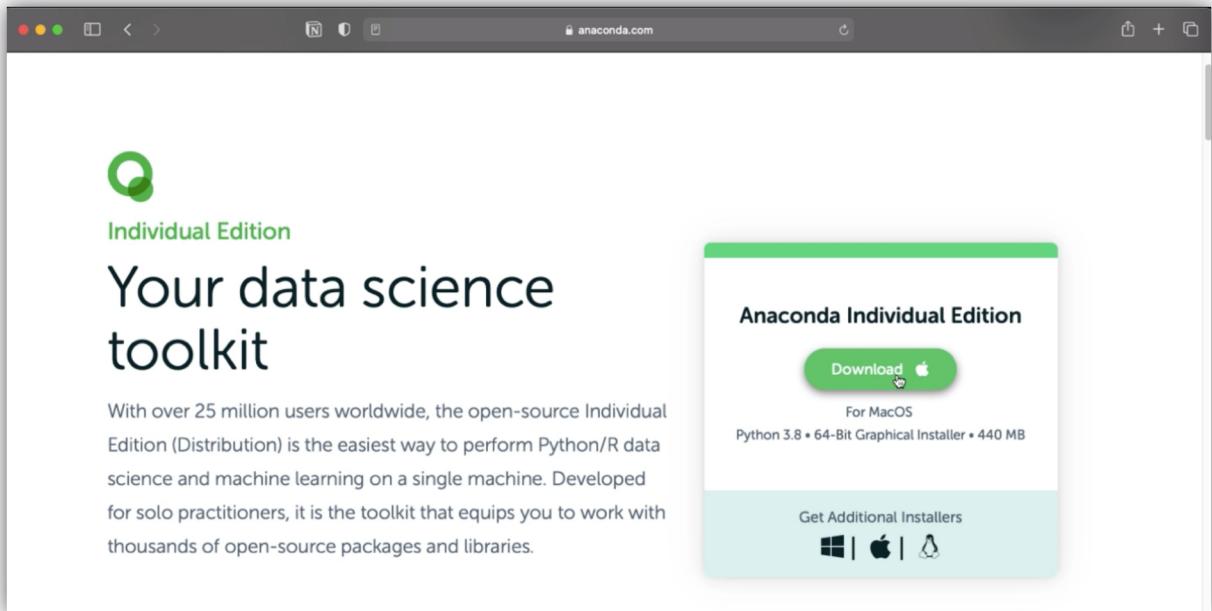
To run a notebook locally, first you should create an environment on which the notebook can work. This environment should include all the necessary packages and dependencies that each notebook requires for running. Here is a summary of all steps that you should take for connecting a notebook to local runtime, followed by detailed instructions of each step.

- 1. Installing Anaconda*
- 2. Creating a new Conda environment in Anaconda*
- 3. Connecting Google Colab to Jupyter notebook*
- 4. Installing Requirements.txt file
(notebook specific packages and dependencies)*
- 5. Running the notebook*

Step 1: Download and Install Anaconda on your machine.

In this step you need to go to download the individual edition for your operating system. Then simply follow several steps for installation and you should have it on your machine.

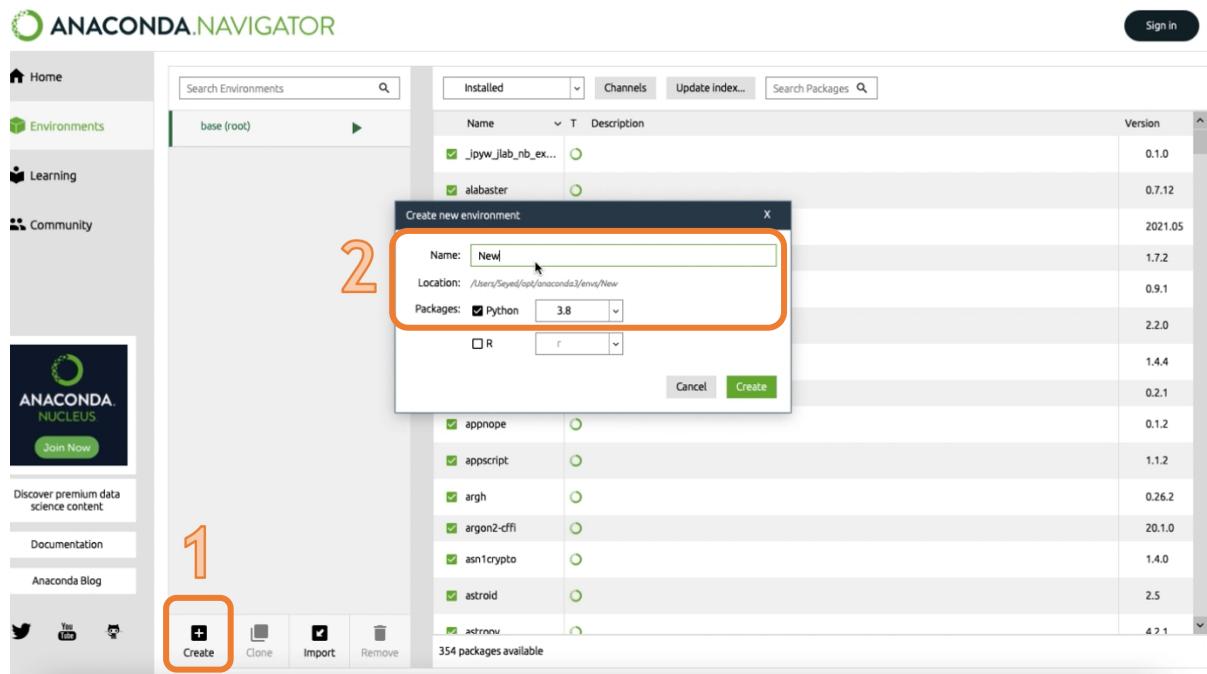
Download link: <https://www.anaconda.com/products/individual>



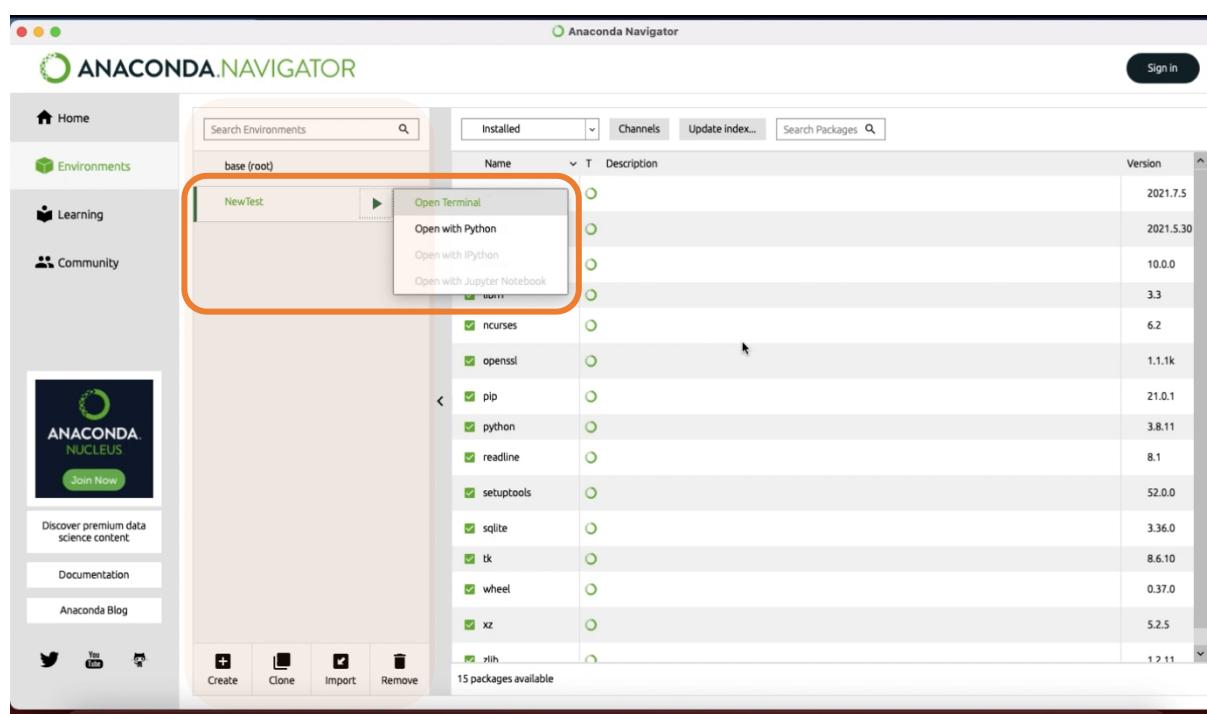
Step 2: Create a new environment in Anaconda.

In this step you should create an environment for your notebook to be able to run on your local machine. Here, you need to pick a name for your new environment and more importantly to identify what version of Python you want to install on your environment.

(you can have several environments with specific versions of Python, depending on the packages and dependencies of each notebook or any other codes you want to run.)



After having your new Conda environment created you should be able to open its terminal for writing code and to see it under the environment list.



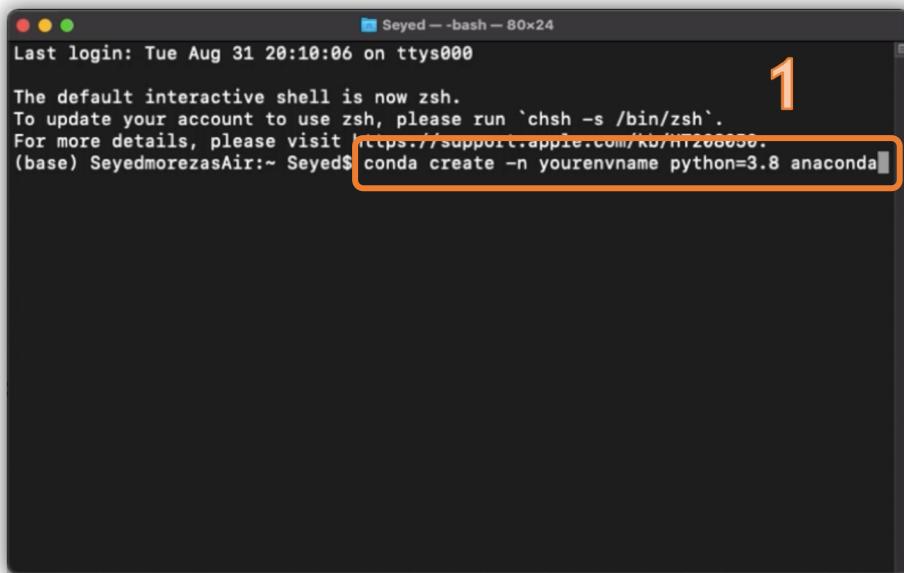
Another way to create a new Conda environment is to write a function in Terminal client. In the terminal client enter the following where *yourenvname* is the name you want to call your environment, and **replace x.x with the Python version** you want to use (*i.e 3.6.9*).

```
conda create -n yourenvname python=x.x anaconda
```

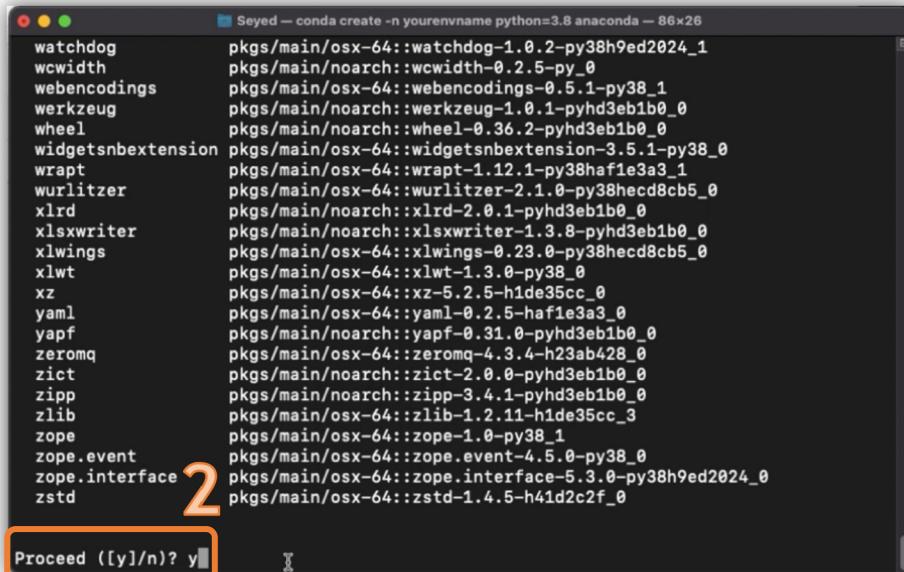
Press **Y** to proceed. This will install the Python version and all the associated anaconda packaged libraries at “*path_to_your_anaconda_location/anaconda/envs/yourenvname*”

After creating the new Conda environment, now you should activate it in the Terminal client. For this, You should run the following code:

```
conda activate yourenvname
```



The terminal window shows the command being typed. An orange box highlights the command 'conda create -n yourenvname python=3.8 anaconda'. A large orange number '1' is positioned above the terminal window.



The terminal window displays the output of the 'conda create' command, listing various packages and their versions. An orange box highlights the package list. A large orange number '2' is positioned below the terminal window. At the bottom, a prompt 'Proceed ([y]/n)? y' is shown in an orange box.

Package	Version
watchdog	pkgs/main/osx-64::watchdog-1.0.2-py38h9ed2024_1
wcwidth	pkgs/main/noarch::wcwidth-0.2.5-py_0
webencodings	pkgs/main/osx-64::webencodings-0.5.1-py38_1
werkzeug	pkgs/main/noarch::werkzeug-1.0.1-pyhd3eb1b0_0
wheel	pkgs/main/noarch::wheel-0.36.2-pyhd3eb1b0_0
widgetsnbextension	pkgs/main/osx-64::widgetsnbextension-3.5.1-py38_0
wrapt	pkgs/main/osx-64::wrapt-1.12.1-py38hafie3a3_1
wurlitzer	pkgs/main/osx-64::wurlitzer-2.1.0-py38hecd8cb5_0
xlrd	pkgs/main/noarch::xlrd-2.0.1-pyhd3eb1b0_0
xlsxwriter	pkgs/main/noarch::xlsxwriter-1.3.8-pyhd3eb1b0_0
xlwings	pkgs/main/osx-64::xlwings-0.23.0-py38hecd8cb5_0
xlwt	pkgs/main/osx-64::xlwt-1.3.0-py38_0
xz	pkgs/main/osx-64::xz-5.2.5-h1de35cc_0
yaml	pkgs/main/osx-64::yaml-0.2.5-hafie3a3_0
yapf	pkgs/main/noarch::yapf-0.31.0-pyhd3eb1b0_0
zeromq	pkgs/main/osx-64::zeromq-4.3.4-h23ab428_0
zict	pkgs/main/noarch::zict-2.0.0-pyhd3eb1b0_0
zipp	pkgs/main/noarch::zipp-3.4.1-pyhd3eb1b0_0
zlib	pkgs/main/osx-64::zlib-1.2.11-h1de35cc_3
zope	pkgs/main/osx-64::zope-1.0-py38_1
zope.event	pkgs/main/osx-64::zope.event-4.5.0-py38_0
zope.interface	pkgs/main/osx-64::zope.interface-5.3.0-py38h9ed2024_0
zstd	pkgs/main/osx-64::zstd-1.4.5-h41d2c2f_0

```
Seyed -- bash -- 86x26
yapf      pkgs/main/noarch::yapf-0.31.0-pyhd3eb1b0_0
zeromq   pkgs/main/osx-64::zeromq-4.3.4-h23ab428_0
zict      pkgs/main/noarch::zict-2.0.0-pyhd3eb1b0_0
zipp      pkgs/main/noarch::zipp-3.4.1-pyhd3eb1b0_0
zlib      pkgs/main/osx-64::zlib-1.2.11-h1de35cc_3
zope      pkgs/main/osx-64::zope-1.0-py38_1
zope.event pkgs/main/osx-64::zope.event-4.5.0-py38_0
zope.interface pkgs/main/osx-64::zope.interface-5.3.0-py38h9ed2024_0
zstd      pkgs/main/osx-64::zstd-1.4.5-h41d2c2f_0

Proceed ([y]/n)? y

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

(base) SeyedmorezasAir:~ Seyed$ conda activate yourenvname
```

3

After activating your new Conda environment you should be able to see it replaced the (base) environment.

```
Seyed -- bash -- 86x26
zeromq   pkgs/main/osx-64::zeromq-4.3.4-h23ab428_0
zict      pkgs/main/noarch::zict-2.0.0-pyhd3eb1b0_0
zipp      pkgs/main/noarch::zipp-3.4.1-pyhd3eb1b0_0
zlib      pkgs/main/osx-64::zlib-1.2.11-h1de35cc_3
zope      pkgs/main/osx-64::zope-1.0-py38_1
zope.event pkgs/main/osx-64::zope.event-4.5.0-py38_0
zope.interface pkgs/main/osx-64::zope.interface-5.3.0-py38h9ed2024_0
zstd      pkgs/main/osx-64::zstd-1.4.5-h41d2c2f_0

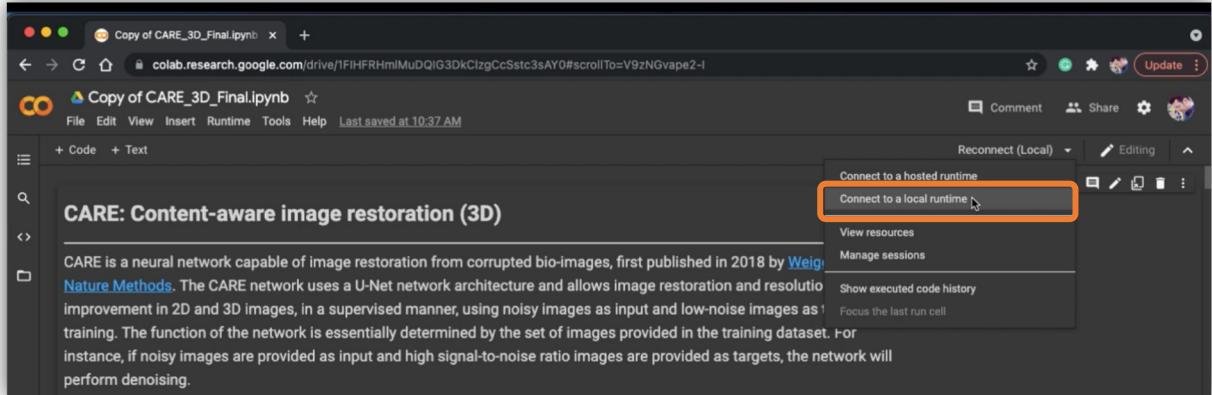
Proceed ([y]/n)? y

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

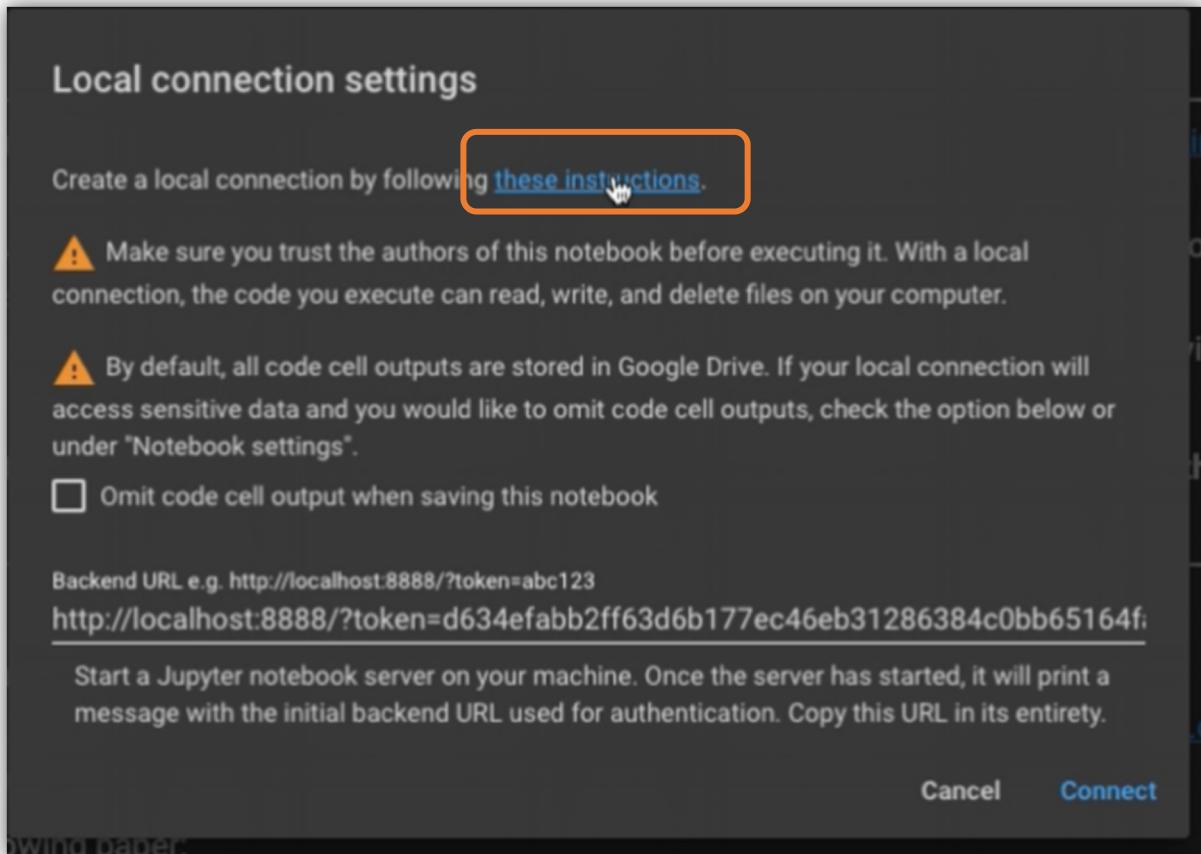
(base) SeyedmorezasAir:~ Seyed$ conda activate yourenvname
(yourenvname) SeyedmorezasAir:~ Seyed$
```

Step 3: Connecting Google Colab to Local Run time.

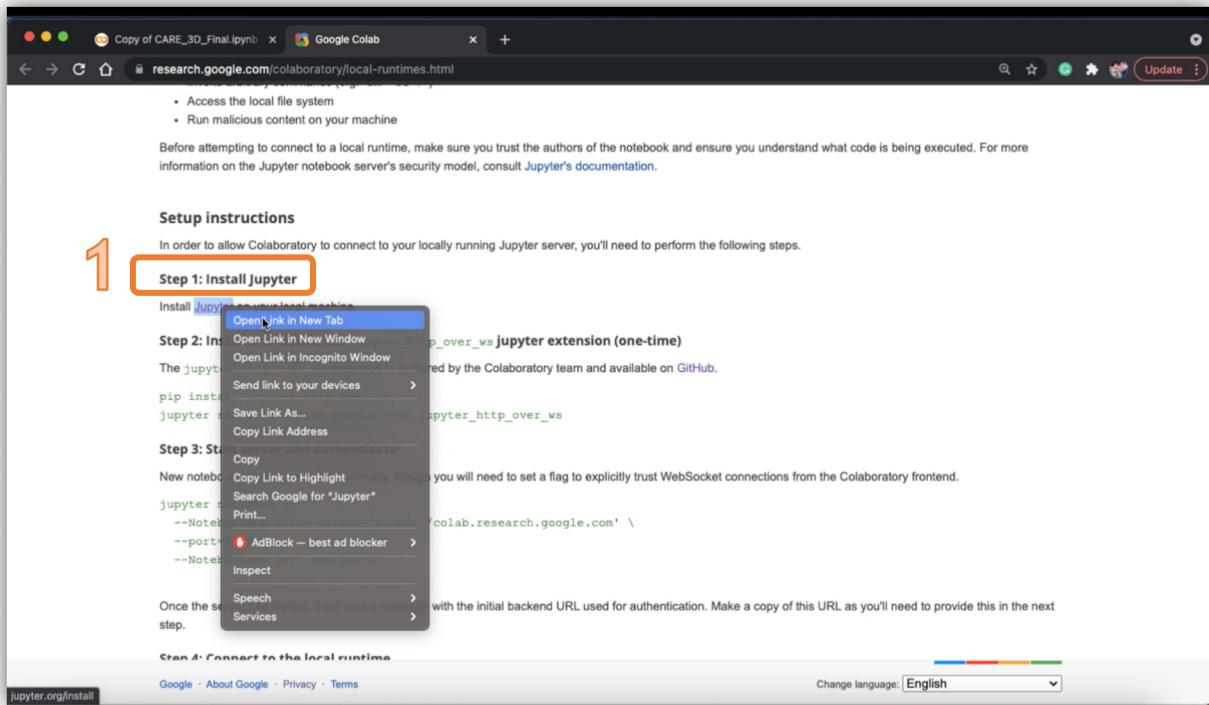
In this step you should connect the Google Colab (as front-end) to a Jupyter notebook server (as back-end) on your local machine. For this, open the drop down menu shown in the picture and choose “connect to local run time”.



Now you should see below pop-up window including a link to Google Colab instructions for local run time connection. Click on the link and follow the next steps.

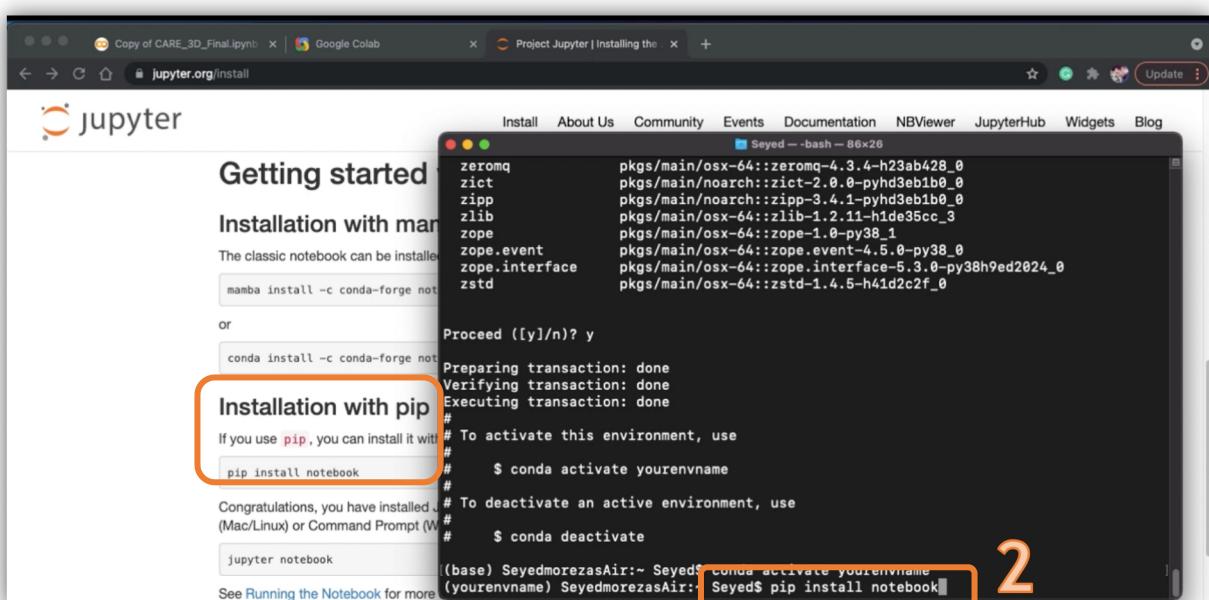


As you can read in the instruction webpage, you should first install Jupyter notebook on your previously made Conda environment. After that, you have to install a jupyter extension (for setting up a server on your machine) and activate it. Finally, you need to start the server and authenticate it to have your back-end ready for Google Colab to connect.



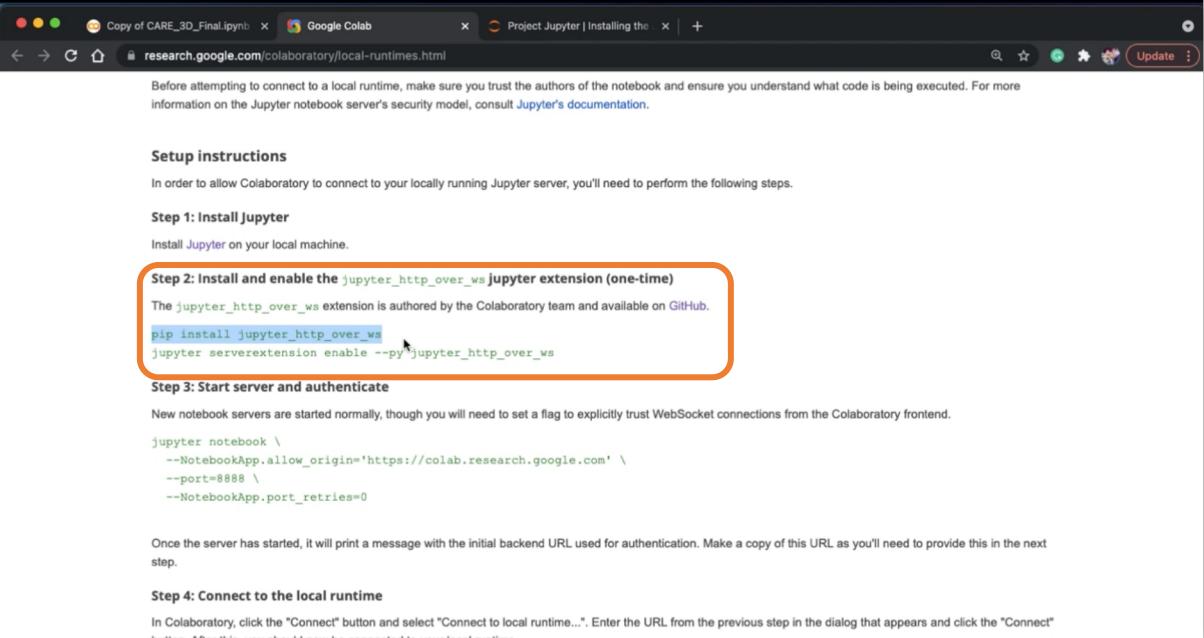
As you can see in the Jupyter website, one of the easiest ways to install it is using [pip](#) function.

```
pip install notebook
```

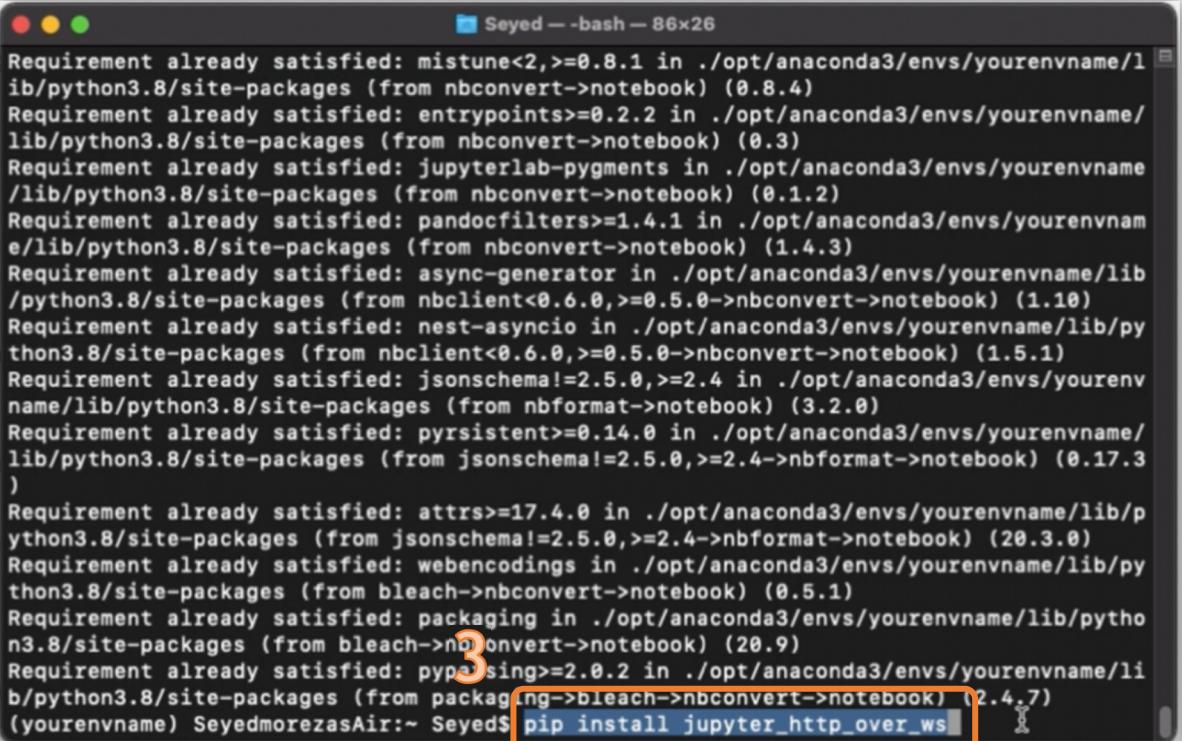


Next, install the extension and enable it using the following function:

```
pip install jupyter_http_over_ws
```



The screenshot shows a browser window titled "Project Jupyter | Installing the" with the URL "research.google.com/colaboratory/local-runtimes.html". The page contains setup instructions for connecting Colaboratory to a local Jupyter server. Step 2, "Install and enable the `jupyter_http_over_ws` jupyter extension (one-time)", is highlighted with an orange box. It includes the command `pip install jupyter_http_over_ws` and `jupyter serverextension enable --py jupyter_http_over_ws`. A cursor arrow points to the first part of the command.



The screenshot shows a terminal window titled "Seyed - -bash - 86x26". The user has run the command `pip install jupyter_http_over_ws`, which is highlighted with an orange box. The output shows various dependency requirements already satisfied, such as `mistune<2,>=0.8.1`, `nbconvert>notebook`, and `jupyterlab-pygments`. A large orange number "3" is placed over the terminal window near the bottom left.

Now you should be able to enable the extension using this function:

```
jupyter serverextension enable --py jupyter_http_over_ws
```

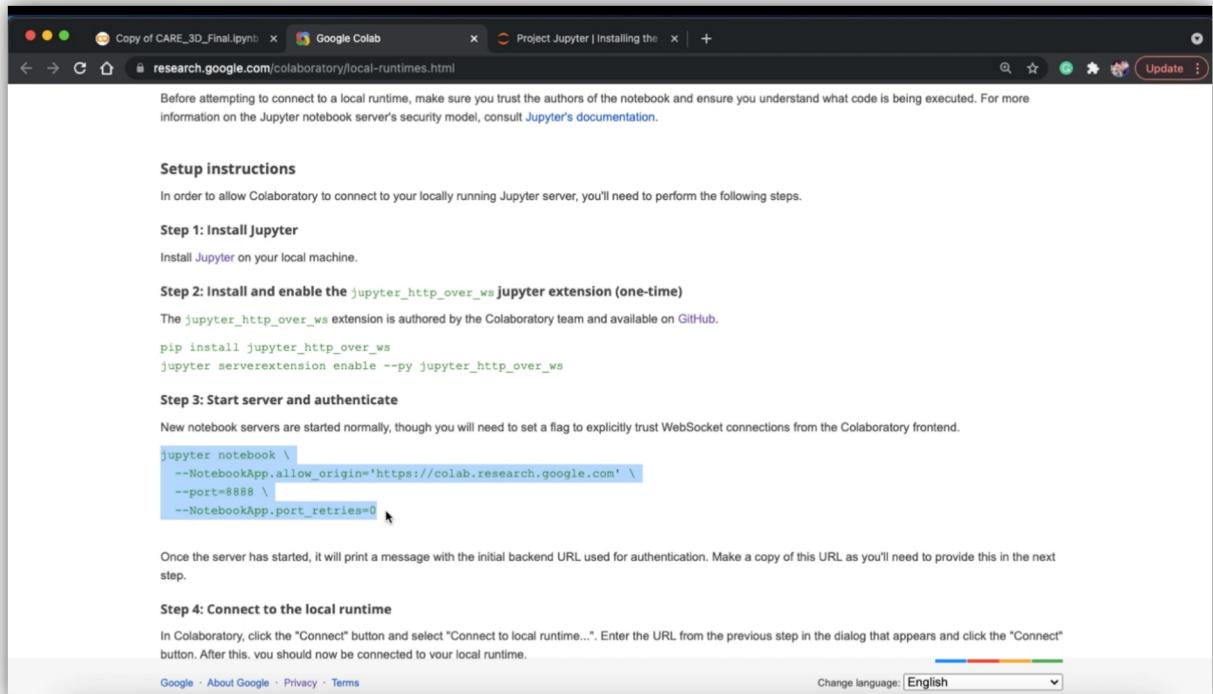
The screenshot shows a web browser window with the URL research.google.com/collaboratory/local-runtimes.html. The page contains setup instructions for connecting Colaboratory to a locally running Jupyter server. It includes steps for installing Jupyter, enabling the `jupyter_http_over_ws` extension, and starting the server. A terminal command `pip install jupyter_http_over_ws` is highlighted in blue, indicating it was copied from the page.

The screenshot shows a terminal window titled "Seyed – jupyter-serverextension enable --py jupyter_http_over_ws – 86x26". The output of the command is displayed, showing that several requirements were already satisfied. The last few lines of the output are highlighted with an orange box and a large orange number "4" is overlaid on the right side of the box.

```
Requirement already satisfied: nest-asyncio in ./opt/anaconda3/envs/yourenvname/lib/python3.8/site-packages (from nbclient<0.6.0,>=0.5.0->nbconvert->notebook>=5.0->jupyter_http_over_ws) (1.5.1)
Requirement already satisfied: jsonschema!=2.5.0,>=2.4 in ./opt/anaconda3/envs/yourenvname/lib/python3.8/site-packages (from nbformat->notebook>=5.0->jupyter_http_over_ws) (3.2.0)
Requirement already satisfied: pyrsistent>=0.14.0 in ./opt/anaconda3/envs/yourenvname/lib/python3.8/site-packages (from jsonschema!=2.5.0,>=2.4->nbformat->notebook>=5.0->jupyter_http_over_ws) (0.17.3)
Requirement already satisfied: attrs>=17.4.0 in ./opt/anaconda3/envs/yourenvname/lib/python3.8/site-packages (from jsonschema!=2.5.0,>=2.4->nbformat->notebook>=5.0->jupyter_http_over_ws) (20.3.0)
Requirement already satisfied: webencodings in ./opt/anaconda3/envs/yourenvname/lib/python3.8/site-packages (from bleach->nbconvert->notebook>=5.0->jupyter_http_over_ws) (0.5.1)
Requirement already satisfied: packaging in ./opt/anaconda3/envs/yourenvname/lib/python3.8/site-packages (from bleach->nbconvert->notebook>=5.0->jupyter_http_over_ws) (20.9)
Requirement already satisfied: pyparsing>=2.0.2 in ./opt/anaconda3/envs/yourenvname/lib/python3.8/site-packages (from packaging->bleach->nbconvert->notebook>=5.0->jupyter_http_over_ws) (2.4.7)
Installing collected packages: jupyter-http-over-ws
Successfully installed jupyter-http-over-ws-0.0.8
(yourenvname) SeyedmorezasAir:~ Seyed$ jupyter serverextension enable --py jupyter_htt
p_over_ws
```

Now that you have your Jupyter notebook server ready, your last step is to start it and authenticate it using the following function:

```
jupyter notebook \
--NotebookApp.allow_origin='https://colab.research.google.com' \
--port=8888 \
--NotebookApp.port_retries=0
```



The screenshot shows a terminal window titled "Seyed — jupyter-notebook --NotebookApp.allow_origin=https://colab.research.google.com --port=8888 --N...". The terminal displays the output of the Jupyter command, which includes requirements for various packages like jupyter_http_over_ws, webencodings, packaging, and pyparsing. It shows the successful installation of the jupyter_http_over_ws package and the configuration of the jupyter server. A large orange box highlights the command line and its output. A large orange number "5" is overlaid on the right side of the terminal window.

```
jupyter notebook \
--NotebookApp.allow_origin='https://colab.research.google.com' \
--port=8888 \
--NotebookApp.port_retries=0
```

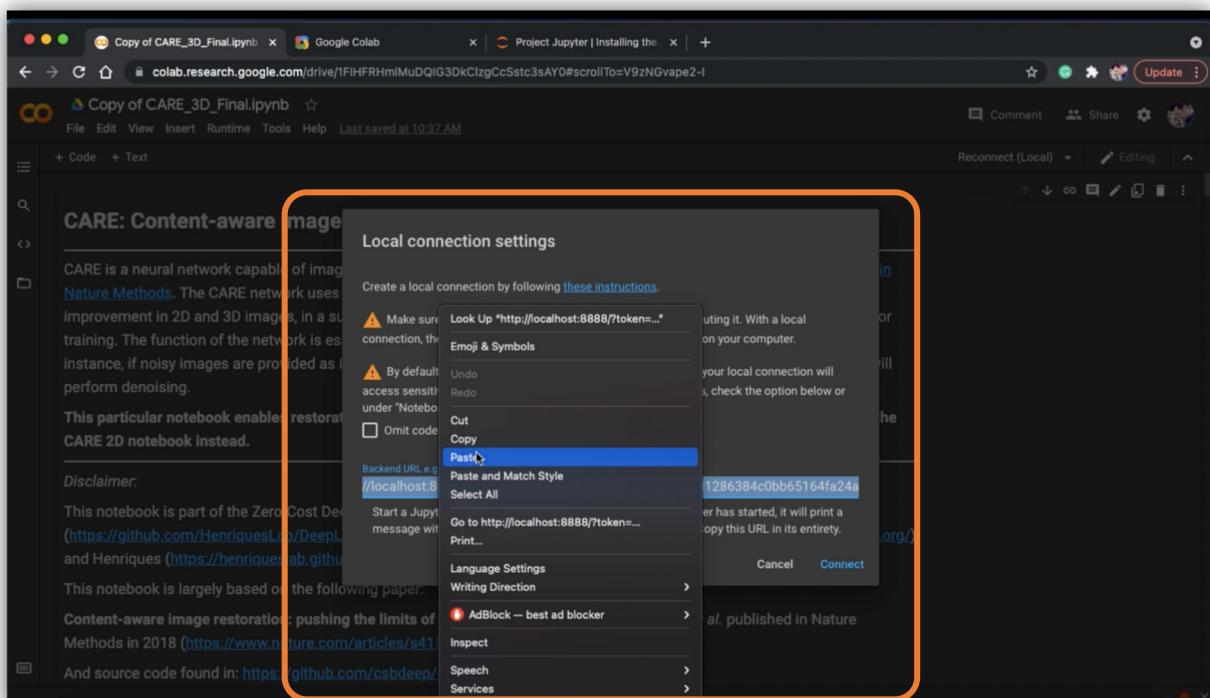
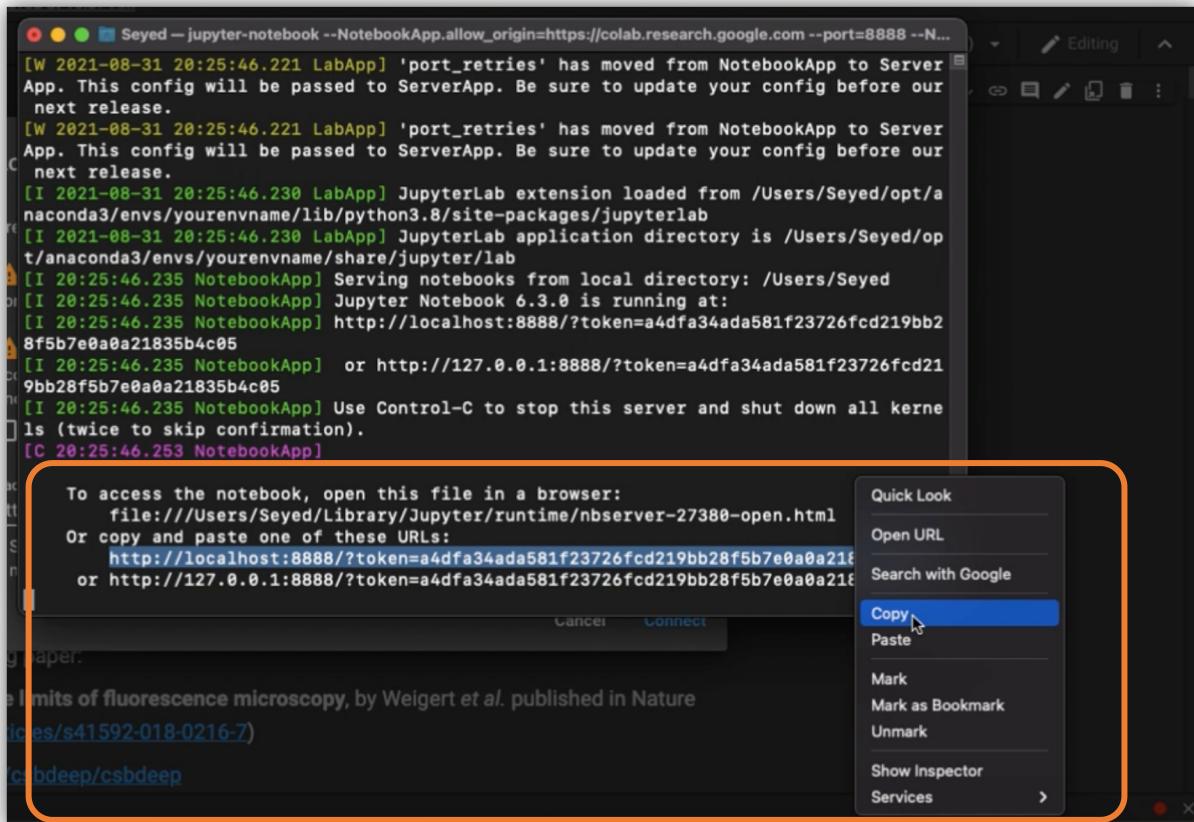
After starting the server it will automatically open a page like this on your browser. This page can be closed and is not needed during the next steps.

The screenshot shows a web-based file browser interface for a Jupyter Notebook server. The address bar at the top displays "localhost:8888/tree". The main area is titled "jupyter" and contains tabs for "Files", "Running", and "Clusters". A sidebar on the left lists standard system directories like Applications, Desktop, Documents, Downloads, Movies, Music, OneDrive, opt, Parallels, Pictures, and Public. The main content area is a table listing files with columns for Name, Last Modified, and File size. The table includes entries for "CARE_3D_ZeroCostDL4Mic.ipynb", "Untitled.ipynb", "≈0.7", and "Seyed Hosseini.pdf".

Coming back to your Terminal client, you should see your server address which should be **copied** into the pop-up window of Google Colab.

The terminal window shows the startup logs for a Jupyter Notebook server. It includes messages about configuration changes, the loading of the JupyterLab extension, and the start of the notebook at port 8888. A specific line in the log is highlighted in orange: "[C 20:25:46.253 NotebookApp] or http://127.0.0.1:8888/?token=a4dfa34ada581f23726fc...". This line is labeled "6- Copy this line" in large orange text. Below the log, instructions tell the user to open the file in a browser or copy and paste one of the URLs provided.

```
[W 2021-08-31 20:25:46.221 LabApp] 'port_retries' has moved from NotebookApp to ServerApp. This config will be passed to ServerApp. Be sure to update your config before our next release.
[W 2021-08-31 20:25:46.221 LabApp] 'port_retries' has moved from NotebookApp to ServerApp. This config will be passed to ServerApp. Be sure to update your config before our next release.
[I 2021-08-31 20:25:46.230 LabApp] JupyterLab extension loaded from /Users/Seyed/opt/anaconda3/envs/yourenvname/lib/python3.8/site-packages/jupyterlab
[I 2021-08-31 20:25:46.230 LabApp] JupyterLab application directory is /Users/Seyed/opt/anaconda3/envs/yourenvname/share/jupyter/lab
[I 20:25:46.235 NotebookApp] Serving notebooks from local directory: /Users/Seyed
[I 20:25:46.235 NotebookApp] Jupyter Notebook 6.3.0 is running at:
[I 20:25:46.235 NotebookApp] http://localhost:8888/?token=a4dfa34ada581f23726fc219bb28f5b7e0a0a21835b4c05
[I 20:25:46.235 NotebookApp] or http://127.0.0.1:8888/?token=a4dfa34ada581f23726fc219bb28f5b7e0a0a21835b4c05
[I 20:25:46.235 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 20:25:46.253 NotebookApp]
6- Copy this line
To access the notebook, open this file in a browser:
file:///Users/Seyed/Library/Jupyter/runtime/nbserver-27380-open.html
Or copy and paste one of these URLs:
http://localhost:8888/?token=a4dfa34ada581f23726fc219bb28f5b7e0a0a21835b4c05
or http://127.0.0.1:8888/?token=a4dfa34ada581f23726fc219bb28f5b7e0a0a21835b4c05
```



Finally, you can click on “Connect” to have your Zerocost4DLMic connected to local runtime.

