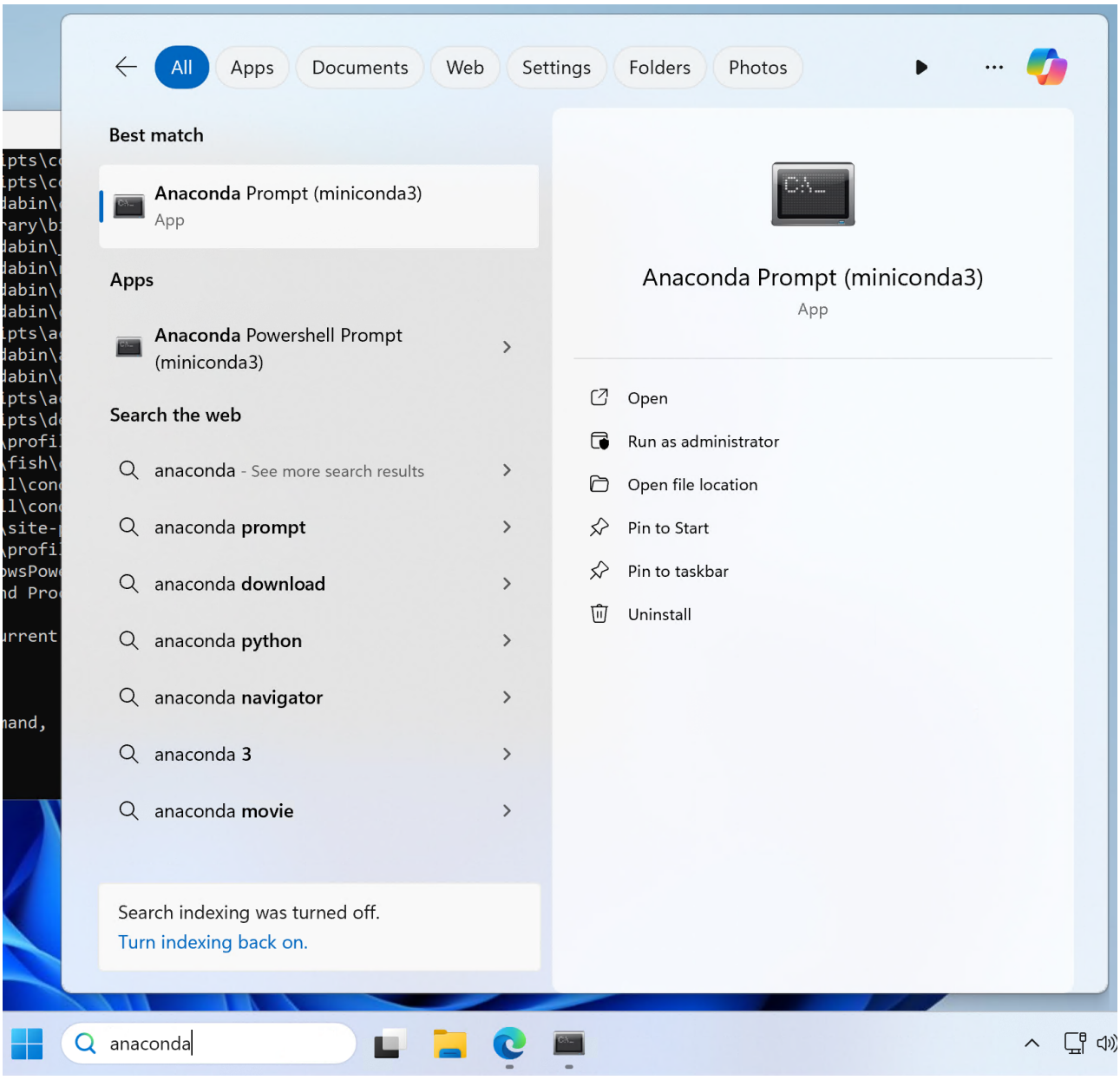
Python Setup Guide

Conda Installation

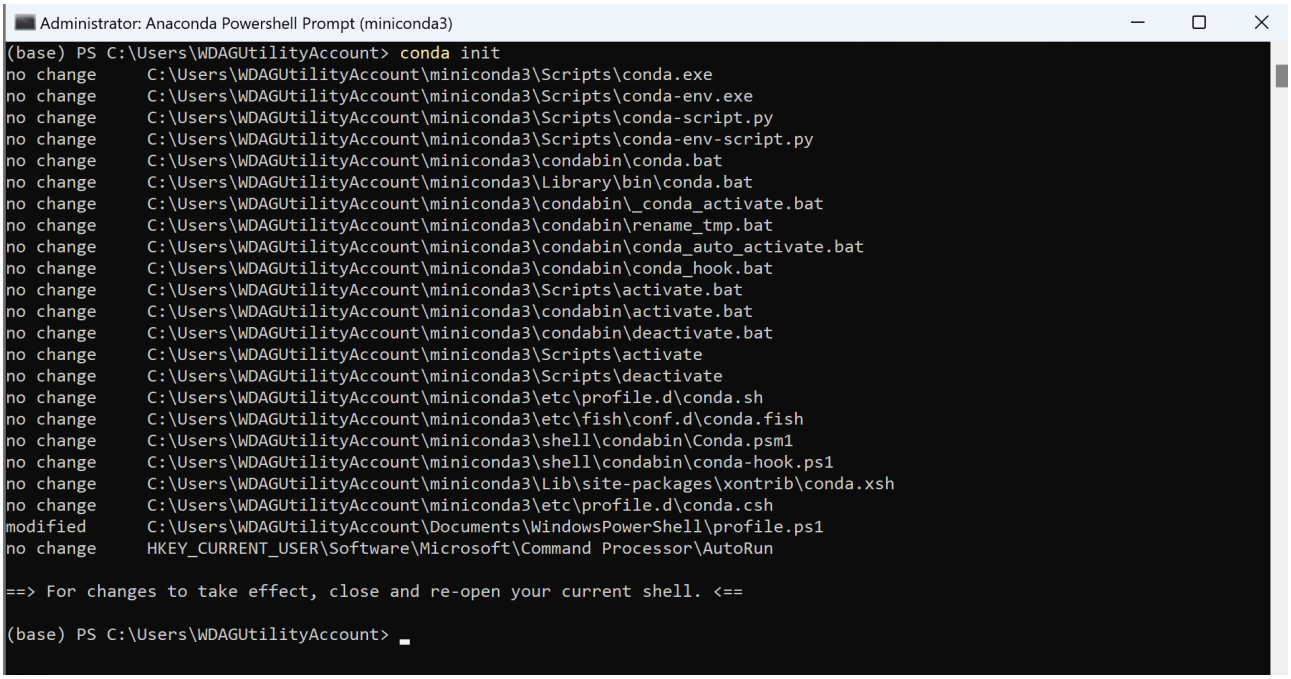
In this section, we'll cover how to set up your python environment with `conda`. You can also other methods if you know how.

We'll be using `miniconda`, which is a distribution with few packages preinstalled. You can download the packages [from their website] and manually run it.

- After your installation, Windows users should be able to search for `Anaconda Prompt` and `Anaconda Powershell Prompt` in their start menu.



- Click on either of those brings up a terminal with the `base` conda environment activated. You can interact with conda in this command
- *Optional: If you want to activate register conda in your local terminal, you can type `conda init` in the command window you just created. This will make sure conda is registered in your system terminals (e.g., `PowerShell` or `Terminal`) so you can directly interact with it.*

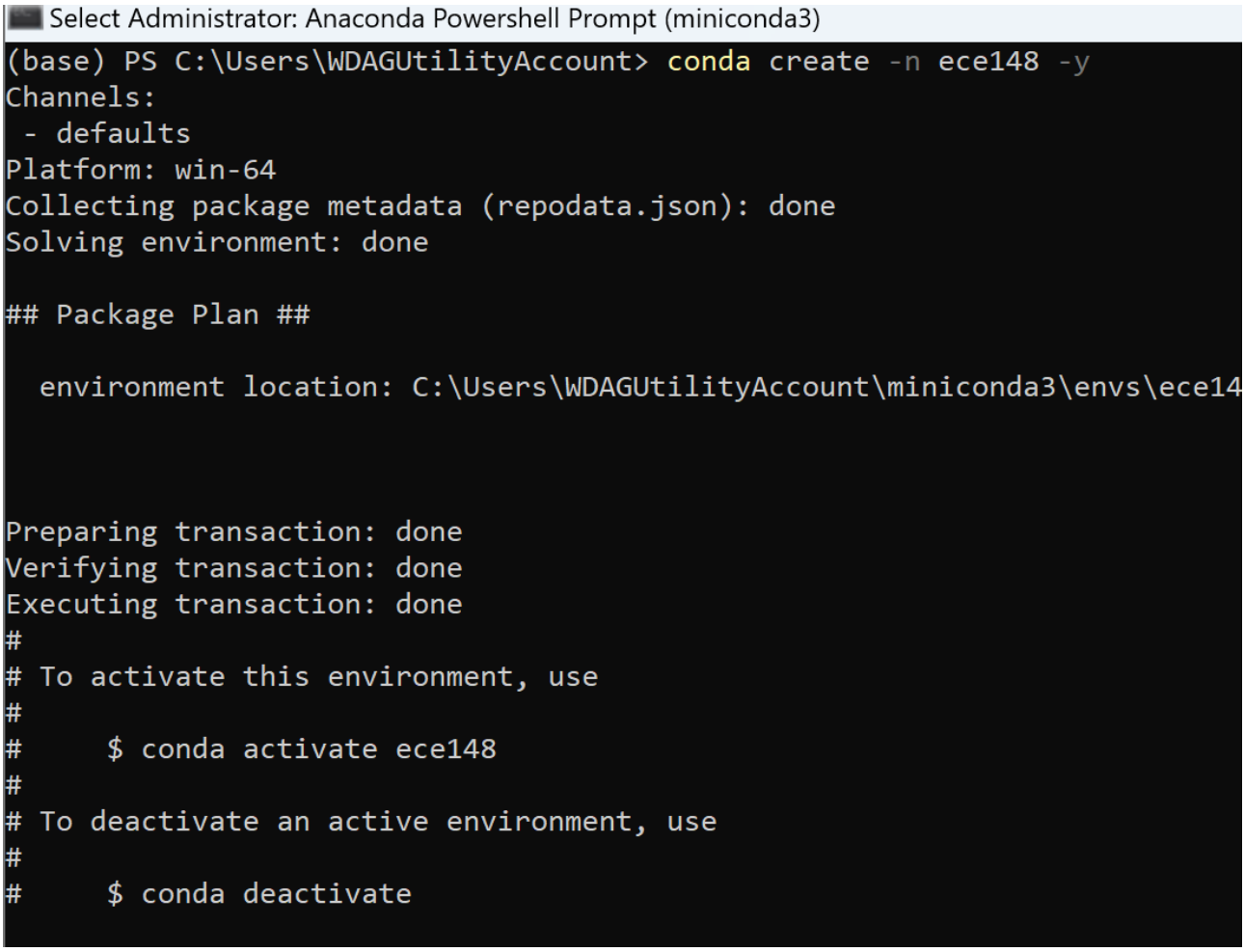


Environment creation and management

One of the biggest advantage of `conda` is environments management. You can create and manage separate python environments. This helps in isolating dependencies and avoiding conflicts between different project requirements.

As an example, we'll create a new environment called `ece148` (feel free to use any other name of your liking, but note that some terminals are case-sensitive).

- Type `conda create -n ece148 -y` in your conda terminal and wait for the installation to finish.



- After the environment is created, you can activate it by typing `conda activate ece148` in your conda terminal. This will switch you into the `ece148` environment, where you can install and manage packages separately from your base environment.

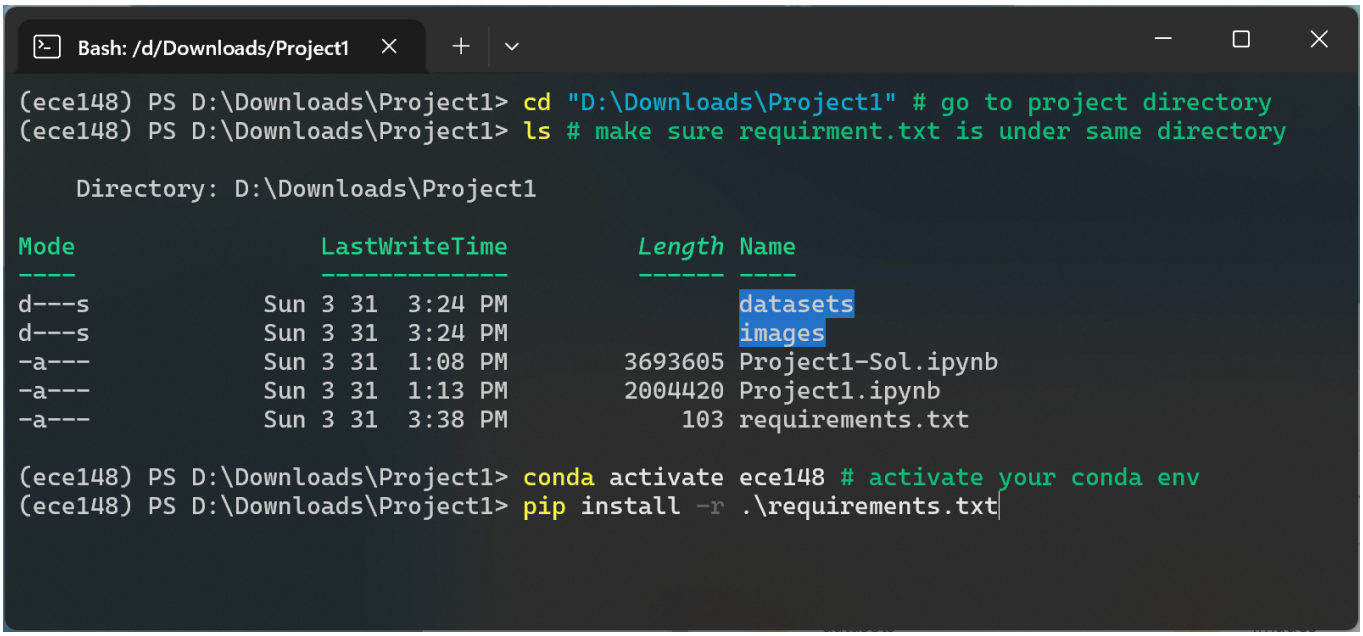
We recommend `python 3.11` and the following packages and versions for project 1:

```
scikit-learn==1.4.1.post1
numpy==1.26.4
scipy==1.12.0
pandas==2.2.1
matplotlib==3.8.3
jupyterlab==4.1.5
```

You can either use `pip install [package1] [package2] ...` to manually install the following packages, or you can use the `requirment.txt` file included in the project by `pip install -r requirment.txt` when your terminal is in the same directory in your conda terminal.

For example, you can use the following code if `requirment.txt` if your project path is `D:\Downloads\Project1`

```
cd \path\to\your\project
conda activate ece148
pip install -r requirment.txt
```



Jupyter Notebook

We'll be using `jupyter notebook` for all of our projects. To support it, make sure that `jupyterlab` package is installed in your python environment.

We will cover 2 ways to edit and run jupyter notebooks in this guide.

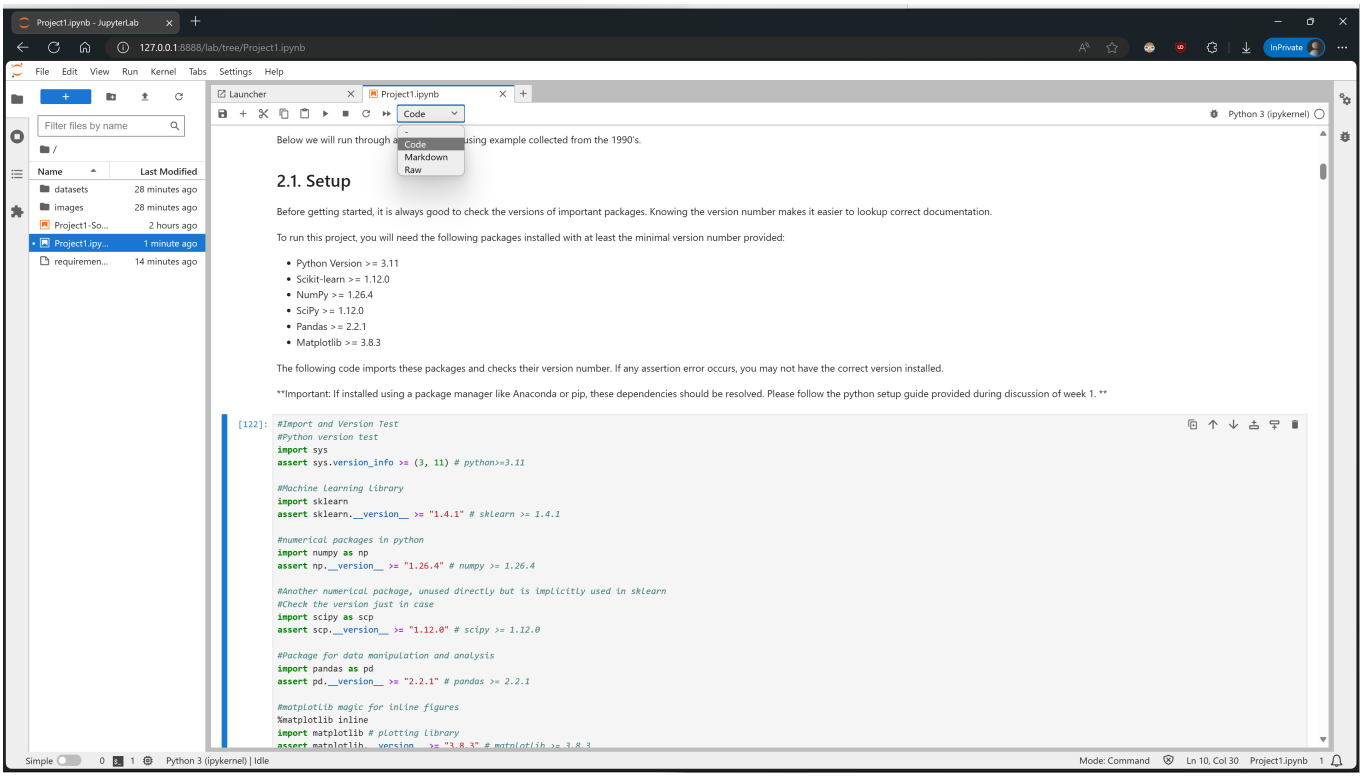
1. Jupyter Lab

To launch Jupyter Lab, type `jupyter lab` in your conda terminal while your environment is activated. This command will start the Jupyter Lab server and open a new tab in your default web browser with the Jupyter Lab interface, where you can create, edit, and run Jupyter notebooks.

You can access the web interface via the URL listed in the output such as the following:

```
[I 2024-03-31 15:47:02.500 ServerApp] Jupyter Server 2.13.0 is running at:
[I 2024-03-31 15:47:02.500 ServerApp] http://localhost:8888/lab?
token=50ce84a570293e01447a2a1a93332b597f9bb53f1c6e558e
```

The web interface looks like this:



A notebook is organized in cells. Each cell can be a different programming language. In our case, it can either be a `code` (`python`) cell which runs your code, or it can me a `markdown` cell which is used for annotations. You can refer to this guideif you are not familiar with `makrdown`.

You can run each cells independently. Note that all your variables, classes, etc. are stored in memory as long as the notebook is open. This gives you freedom to experiment with different things.

2. Use IDE/Text editor with Jupyter support

It might be easier for you to use an text editor/IDE with built in jupyter notebook support such as Visual Studio Code with Jupyter Extension or PyCharm. Please refer to their websites for details, but it should be as simple as just opening the notebook and things should work out of the box.