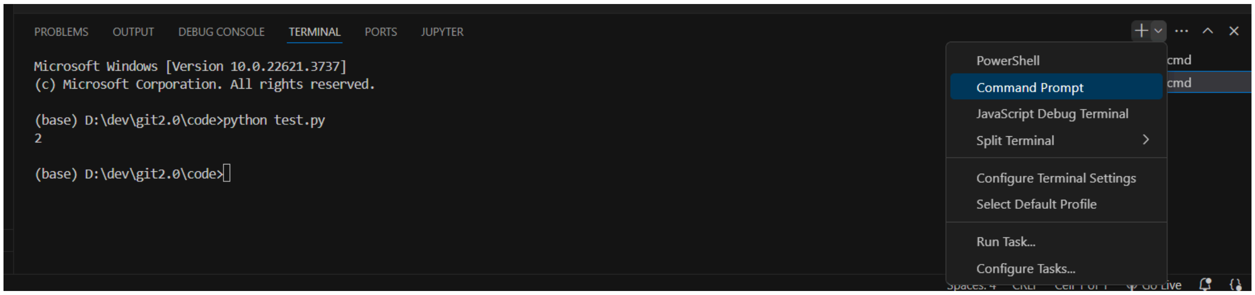
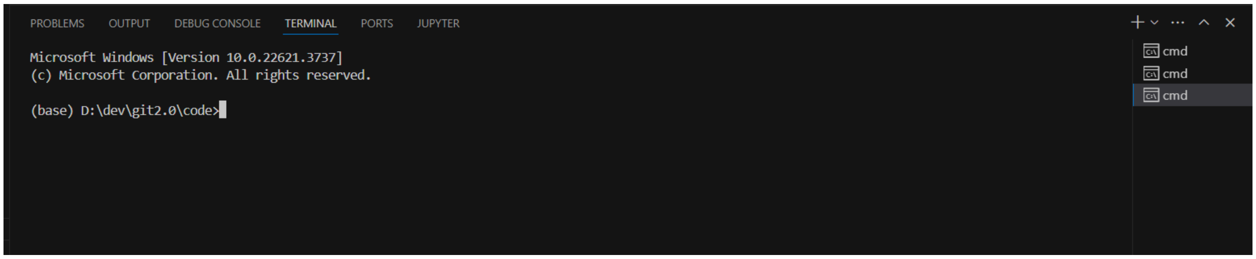
Creating a new environment in VSCode:

1) Create a new folder and open in vscode.
2) Create a new command prompt terminal.

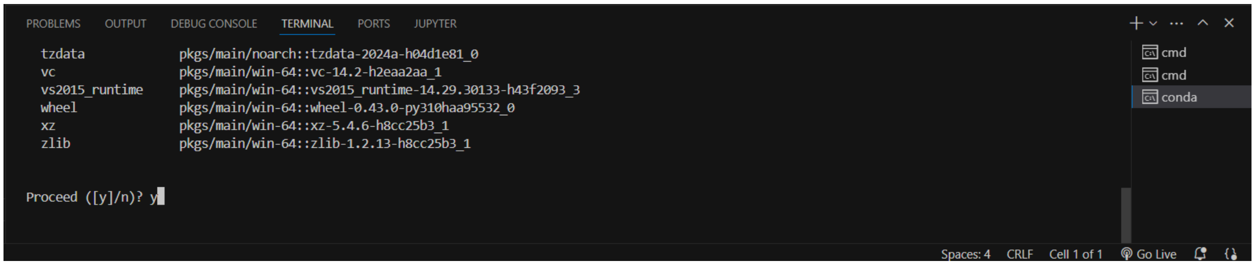


```
Microsoft Windows [Version 10.0.22621.3737]
(c) Microsoft Corporation. All rights reserved.

(base) D:\dev\git2.0\code>python test.py
2

(base) D:\dev\git2.0\code>
```



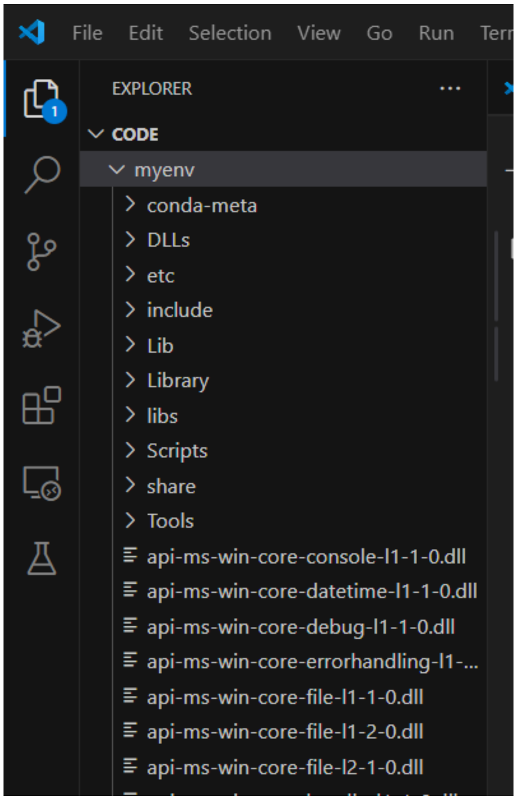
```
Microsoft Windows [Version 10.0.22621.3737]
(c) Microsoft Corporation. All rights reserved.

(base) D:\dev\git2.0\code>
```

3) Run the conda command to create a new environment.
>conda create -p myenv python=3.10



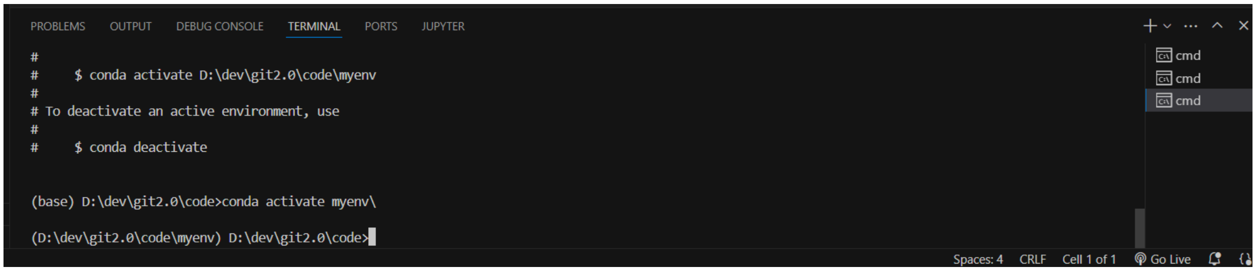
```
  tzdata             pkgs/main/noarch::tzdata-2024a-h04d1e81_0
  vc                 pkgs/main/win-64::vc-14.2-h2eaa2aa_1
  vs2015_runtime     pkgs/main/win-64::vs2015_runtime-14.29.30133-h43f2093_3
  wheel              pkgs/main/win-64::wheel-0.43.0-py310haa95532_0
  xz                 pkgs/main/win-64::xz-5.4.6-h8cc25b3_1
  zlib               pkgs/main/win-64::zlib-1.2.13-h8cc25b3_1


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

Press y when prompted.
This will download and extract the required packages and create a new environment with the name given in the command above.



```
CODE
  myenv
    conda-meta
    DLLs
    etc
    include
    Lib
    Library
    libs
    Scripts
    share
    Tools
    api-ms-win-core-console-l1-1-0.dll
    api-ms-win-core-datetime-l1-1-0.dll
    api-ms-win-core-debug-l1-1-0.dll
    api-ms-win-core-errorhandling-l1-...
    api-ms-win-core-file-l1-1-0.dll
    api-ms-win-core-file-l1-2-0.dll
    api-ms-win-core-file-l2-1-0.dll
```

4) Once the new env is created (myenv) we need to activate the new environment using the command

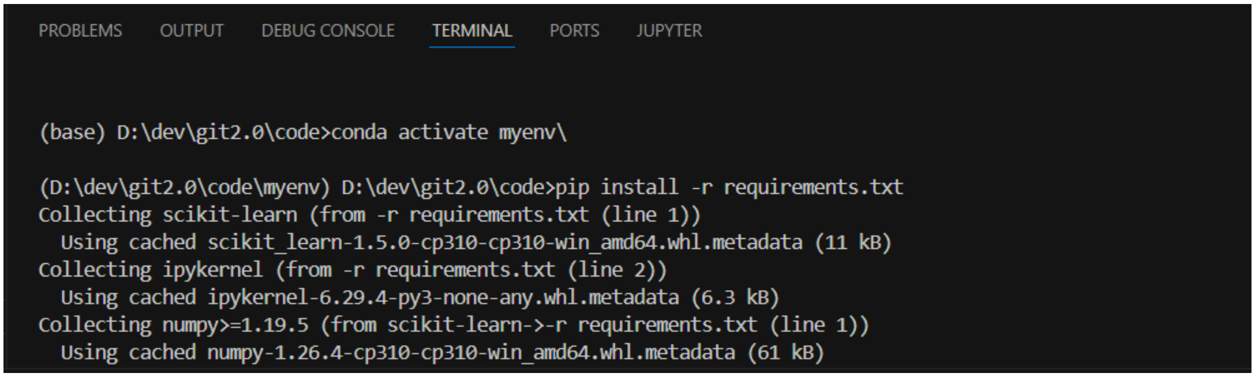


```
#
#     $ conda activate D:\dev\git2.0\code\myenv
#
# To deactivate an active environment, use
#
#     $ conda deactivate


(base) D:\dev\git2.0\code>conda activate myenv\

(D:\dev\git2.0\code\myenv) D:\dev\git2.0\code>
```

5) Now we have successfully created a new environment in vs code and have activated it , so that whatever libraries we will install , it will get installed in the myenv which is currently activated.

Creating and working with requirements.txt file in python

1) Whenever we have to install any dependency in python we run the pip install command.
2) The requirement.txt file contains all the required dependencies which are required for the project to run successfully.
3) We declare all the dependencies in the requirements.txt file and instead of installing the dependencies one by one through pip install <specific_dependency> , we run the installation through the requirements.txt file using the command below :
    - pip install -r requirements.txt



```
(base) D:\dev\git2.0\code>conda activate myenv\

(D:\dev\git2.0\code\myenv) D:\dev\git2.0\code>pip install -r requirements.txt
Collecting scikit-learn (from -r requirements.txt (line 1))
  Using cached scikit_learn-1.5.0-cp310-cp310-win_amd64.whl.metadata (11 kB)
Collecting ipykernel (from -r requirements.txt (line 2))
  Using cached ipykernel-6.29.4-py3-none-any.whl.metadata (6.3 kB)
Collecting numpy>=1.19.5 (from scikit-learn->-r requirements.txt (line 1))
  Using cached numpy-1.26.4-cp310-cp310-win_amd64.whl.metadata (61 kB)
```

4) Everytime a new dependency is used in the project we mention it is requirements.txt and then run the pip install -r requirements.txt to install it in the activated env.

Running python command in ipynb and py file extension type

1) Both the file extensions are used for running python code.
2) ipynb is jupyter file type which we can execute directly in the vscode editor using the ipykernel on clicking the run button.
   Note : We need to install the ipykernel for the commands to run in the ipynb file.
   For that we need to either mention ipykernel in the requirements.txt and pip install requirements.txt or run the pip install ipykernel directly from the command prompt.

3) py is the extension of a basic python source file which can be executed using the python file_name.py command.