Notes Taken By

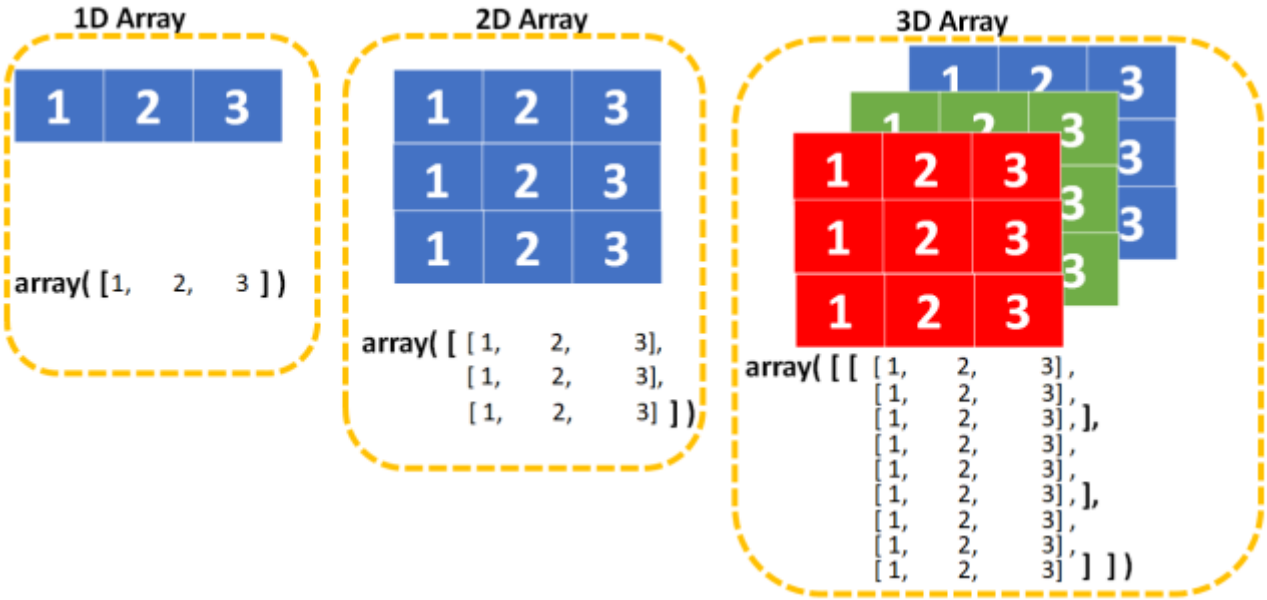
- Muhammad Raaid Khan
- Data Science and AI (Batch - 05)
- NED - CCEE

Vectorization

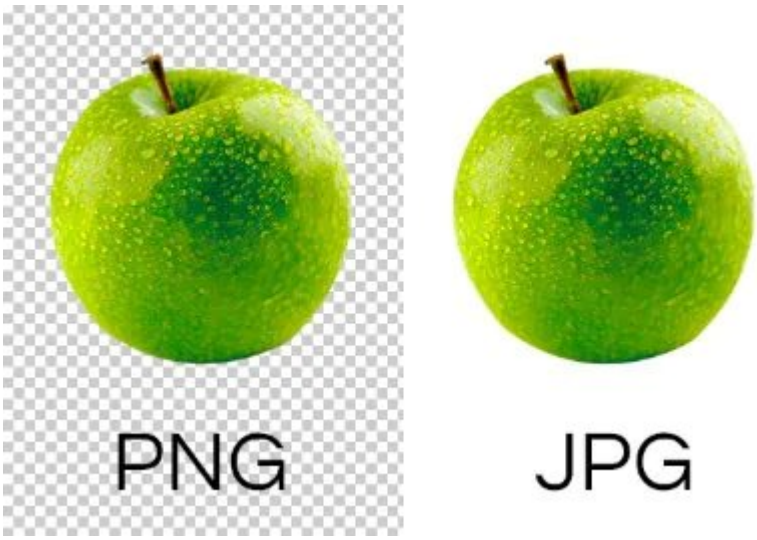
Conversion to an Raw Data into an array of numbers is called vectorization

An image is converted into 03 dimensions.

- X-Axis
- Y-Axis
- RGB Values



For PNG images, there is an additional channel i.e. **Alpha**. This indicates the transparency.



- To identify an image, the colors may or may not be required.

- If colors are not important, we can use *Grayscale* images as it will reduce Computation Power requirements.

OpenCV

OpenCV is a library of programming functions mainly for real-time computer vision.

- Used for Pre-Processing of Image Data.

Image Segmentation

In digital image processing and computer vision, image segmentation is the process of partitioning a digital image into multiple image segments, also known as image regions or image objects.

Tensorflow

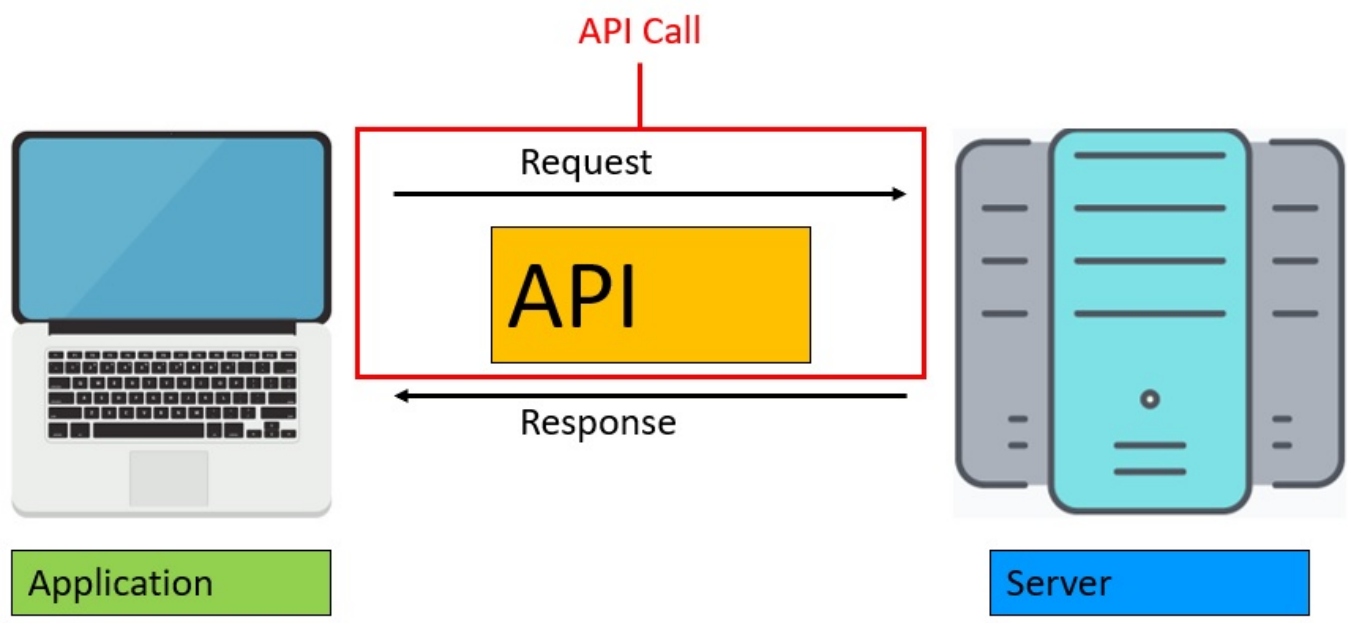
TensorFlow is a free and open-source software library for machine learning and artificial intelligence. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks.

Training Data Ratios

- 70% for training 30% for testing
- 80% for training 20% for testing
- 60% for training 40% for testing
- 60% for training, 20% validation, 20% for testing

API Calls

Application programming interfaces (APIs) are a way for one program to interact with another. API calls are the medium by which they interact. An API call, or API request, is a message sent to a server asking an API to provide a service or information.



API Standards

- JSON Input JSON Output

PyPI

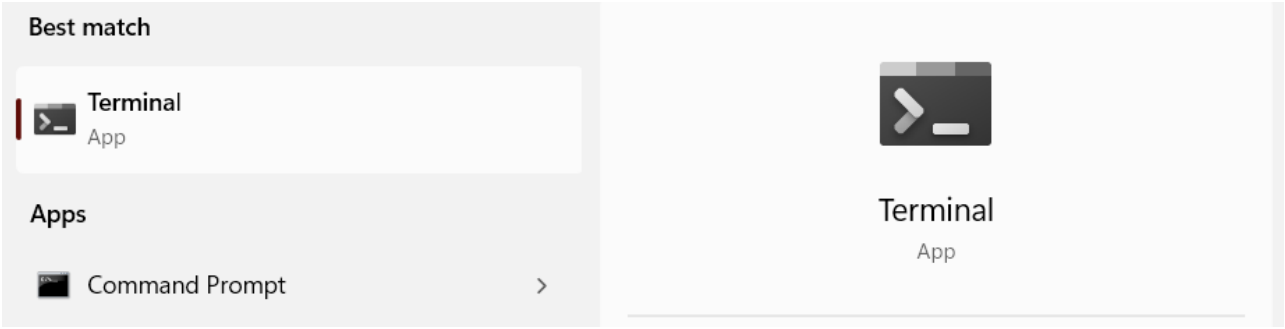
The Python Package Index (PyPI) is a repository of software for the Python programming language.

https://pypi.org/

Installation of Poetry Package

Install Scoop

- Open Terminal in Windows



- Run Following Commands in Sequence
  - `Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope CurrentUser`
  - `irm get.scoop.sh | iex`

```
PS C:\Users\raaid> Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope CurrentUser
PS C:\Users\raaid> irm get.scoop.sh | iex
Initializing...
Downloading...
Creating shim...
Adding ~\scoop\shims to your path.
Scoop was installed successfully!
Type 'scoop help' for instructions.
```

Installing Pipx

- Run following highlighted command in Terminal

```
PS C:\Users\raaid> scoop install pipx
Installing 'pipx' (1.5.0) [64bit] from 'main' bucket
pipx.pyz (311.8 KB) [=========================================================================================] 100%
Checking hash of pipx.pyz ... ok.
Running pre_install script...
Linking ~\scoop\apps\pipx\current => ~\scoop\apps\pipx\1.5.0
Creating shim for 'pipx'.
'pipx' (1.5.0) was installed successfully!
'pipx' suggests installing 'python'.
PS C:\Users\raaid> pipx ensurepath
Success! Added C:\Users\raaid\.local\bin to the PATH environment variable.

Consider adding shell completions for pipx. Run 'pipx completions' for instructions.

You will need to open a new terminal or re-login for the PATH changes to take effect.

Otherwise pipx is ready to go! ✨ 🌟 ✨
```

Installing Poetry

- Run following highlighted command in Terminal

```
PS C:\Users\raaid> pipx install poetry
'poetry' already seems to be installed. Not modifying existing installation in 'C:\Users\raaid\pipx\venvs\poetry'.
Pass '--force' to force installation.
PS C:\Users\raaid>
```

Using Poetry

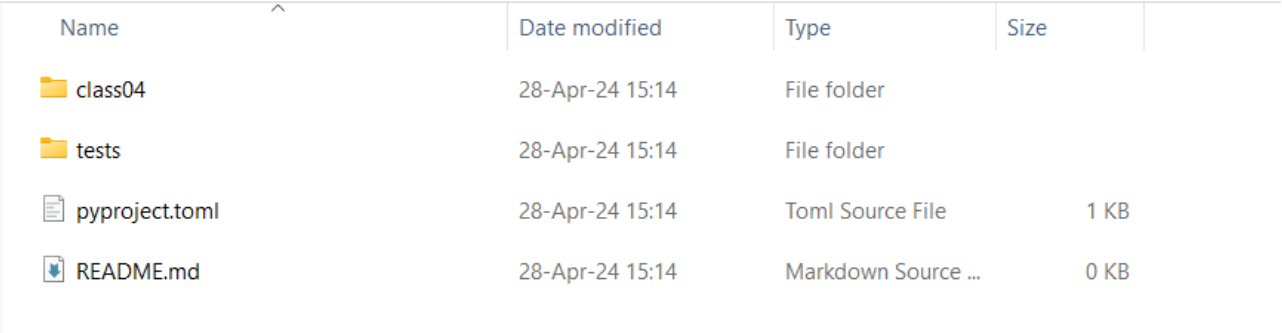
Creating Package

- Open Terminal in you Project directory and run below command

```
E:\PGD-CCEE\C04 - Machine Learning\Lecture Notes\L03-04\Code\Poetry>poetry new class04
Created package class04 in class04

E:\PGD-CCEE\C04 - Machine Learning\Lecture Notes\L03-04\Code\Poetry>
```

- After successful command, below directories will be created.



Name	Date modified	Type	Size
class04	28-Apr-24 15:14	File folder	
tests	28-Apr-24 15:14	File folder	
pyproject.toml	28-Apr-24 15:14	Toml Source File	1 KB
README.md	28-Apr-24 15:14	Markdown Source ...	0 KB

- Change Directory to Project Folder which Contains TOML (Tom's Obvious, Minimal Language) File.
    - If conda is installed, deactivate it using `conda deactivate`.
    - Run Poetry Shell in this folder using command `poetry shell`

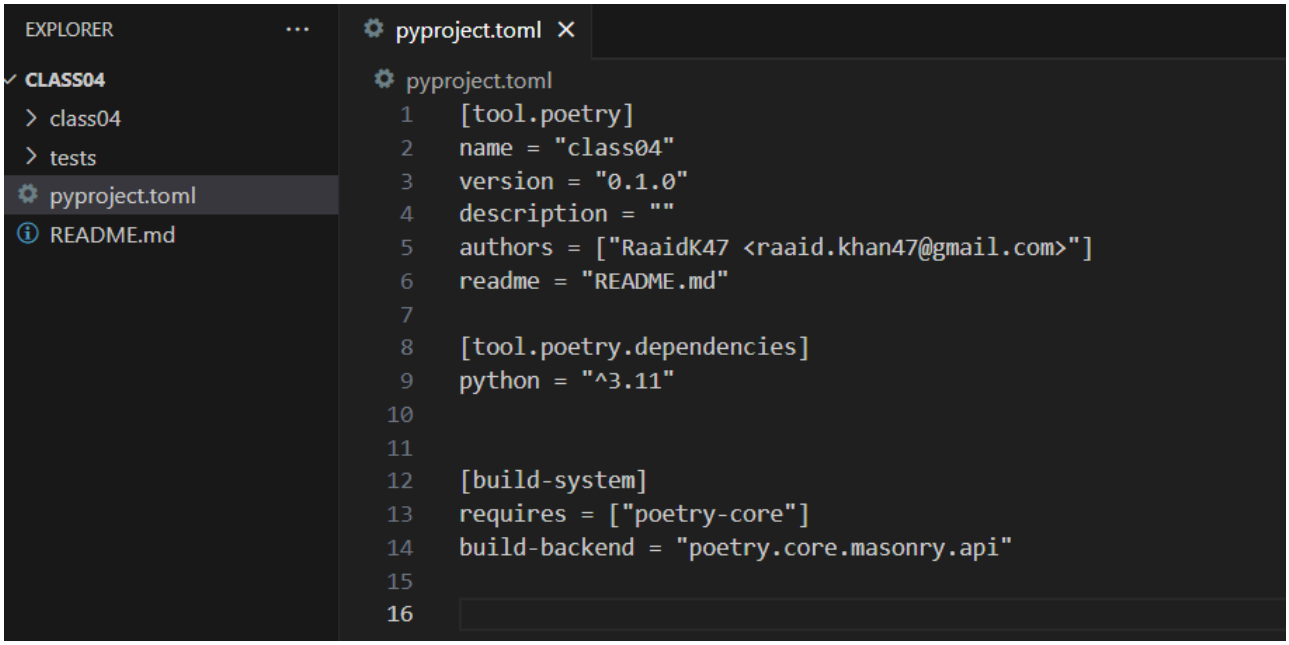
```
E:\PGD-CCEE\C04 - Machine Learning\Lecture Notes\L03-04\Code\Poetry\class04>poetry shell
Creating virtualenv class04-FLkWC2od-py3.11 in C:\Users\raaid\AppData\Local\pypoetry\Cache\virtualenvs
Spawning shell within C:\Users\raaid\AppData\Local\pypoetry\Cache\virtualenvs\class04-FLkWC2od-py3.11

E:\PGD-CCEE\C04 - Machine Learning\Lecture Notes\L03-04\Code\Poetry\class04>()

(class04-py3.11) E:\PGD-CCEE\C04 - Machine Learning\Lecture Notes\L03-04\Code\Poetry\class04>
```

TOML File

- Open Project in VS Code
- Open `pyproject.toml` file



EXPLORER

CLASS04
- class04
- tests
- pyproject.toml
- README.md

pyproject.toml

```toml
[tool.poetry]
name = "class04"
version = "0.1.0"
description = ""
authors = ["RaaidK47 <raaid.khan47@gmail.com>"]
readme = "README.md"

[tool.poetry.dependencies]
python = "^3.11"


[build-system]
requires = ["poetry-core"]
build-backend = "poetry.core.masonry.api"

```

- `.toml` contains MetaData (Data about Data) of our Project.
    - Author Details
    - Project Dependencies
    - Python `^3.11` (Version 3 is fixed (^), .11 can be changed)

Installing Dependencies in Project

- Open any terminal in Project Folder containing `.toml` file.

- Install dependencies with command `poetry add pandas`

```
E:\PGD-CCEE\C04 - Machine Learning\Lecture Notes\L03-04\Code\Poetry\class04>poetry add pandas
Using version ^2.2.2 for pandas

Updating dependencies
Resolving dependencies... (1.5s)

Package operations: 6 installs, 0 updates, 0 removals

  - Installing six (1.16.0)
  - Installing numpy (1.26.4)
  - Installing python-dateutil (2.9.0.post0)
  - Installing pytz (2024.1)
  - Installing tzdata (2024.1)
  - Installing pandas (2.2.2)

Writing lock file

E:\PGD-CCEE\C04 - Machine Learning\Lecture Notes\L03-04\Code\Poetry\class04>
```

- After Installation, `.toml` file will be change

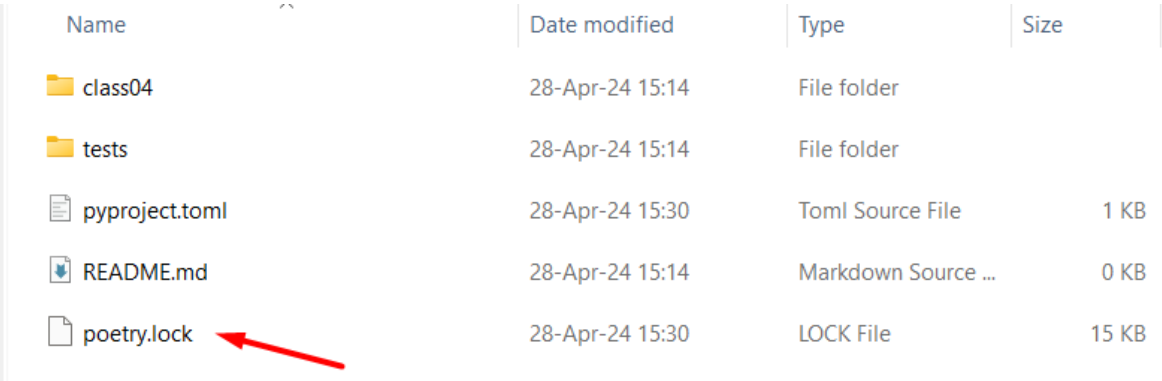
pyproject.toml

```toml
[tool.poetry]
name = "class04"
version = "0.1.0"
description = ""
authors = ["RaaidK47 <raaid.khan47@gmail.com>"]
readme = "README.md"

[tool.poetry.dependencies]
python = "^3.11"
pandas = "^2.2.2"


[build-system]
requires = ["poetry-core"]
build-backend = "poetry.core.masonry.api"
```

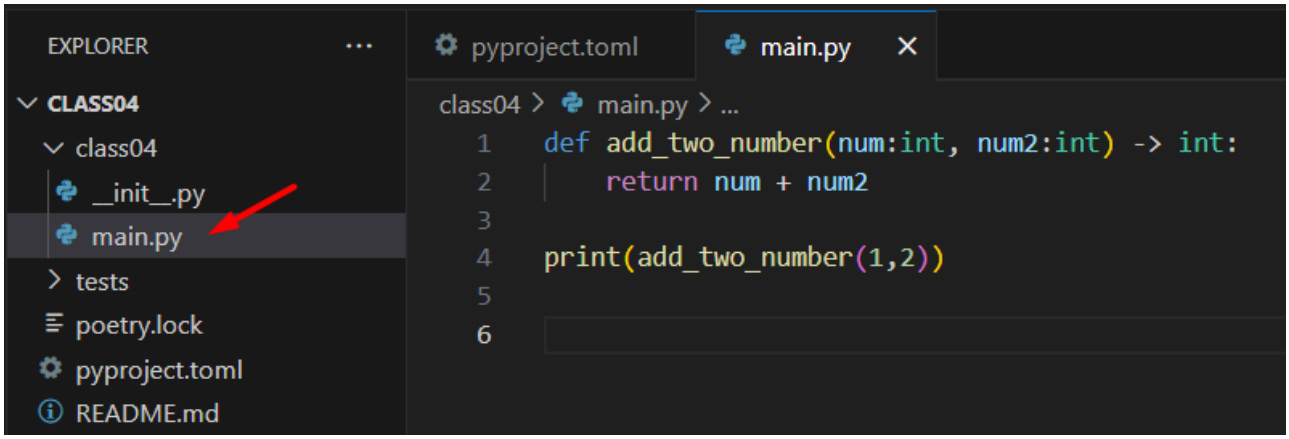
- A `poetry.lock` file will also be created.



Creating the Project

- Go to project folder i.e. `class04`

- Create `main.py` file.



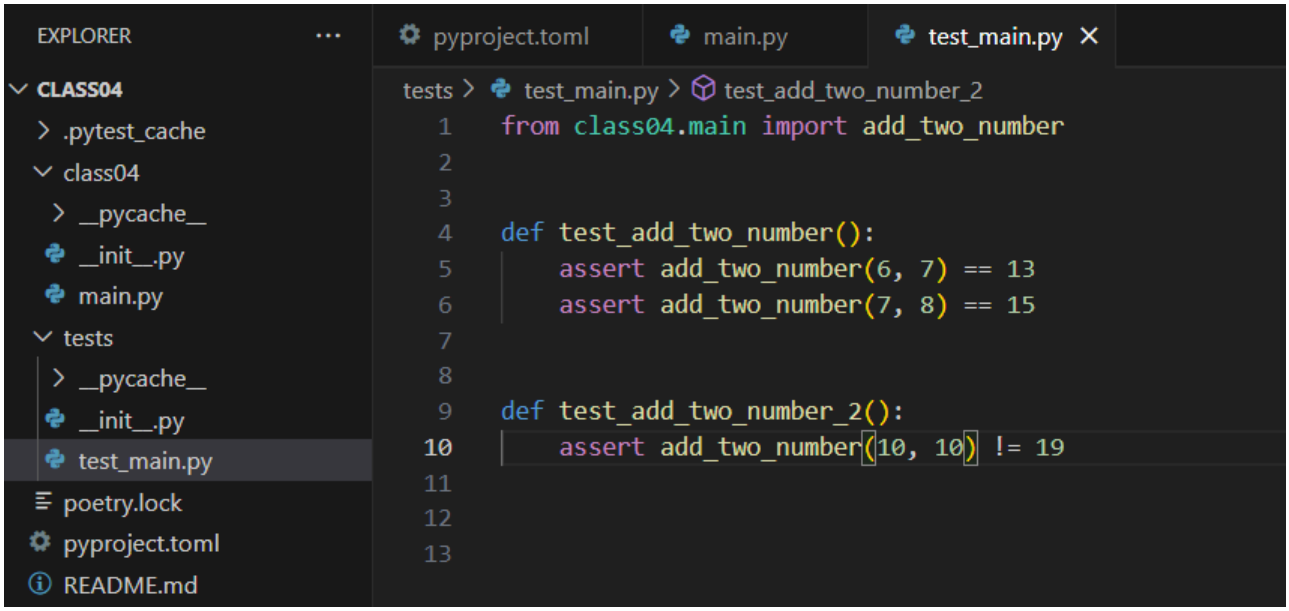
Running Project

- Open Terminal in Main Folder
- Run Following command

```
E:\PGD-CCEE\C04 - Machine Learning\Lecture Notes\L03-04\Code\Poetry\class04>poetry run python ./class04/main.py
3
```

Writing Tests

- Create a Test in `tests` folder.



- Test Application using `Poetry`

- First install `pytest` in Poetry

```
E:\PGD-CCEE\C04 - Machine Learning\Lecture Notes\L03-04\Code\Poetry\class04>poetry add pytest
Using version ^8.2.0 for pytest

Updating dependencies
Resolving dependencies... (1.2s)

Package operations: 5 installs, 0 updates, 0 removals

  - Installing colorama (0.4.6)
  - Installing iniconfig (2.0.0)
  - Installing packaging (24.0)
  - Installing pluggy (1.5.0)
  - Installing pytest (8.2.0)

Writing lock file
```

- Run tests using following command

```
E:\PGD-CCEE\C04 - Machine Learning\Lecture Notes\L03-04\Code\Poetry\class04>poetry run pytest -v
============================================= test session starts =============================================
platform win32 -- Python 3.11.5, pytest-8.2.0, pluggy-1.5.0 -- C:\Users\raaid\AppData\Local\pypoetry\Cache\virtualenvs\c
lass04-FLkWC2od-py3.11\Scripts\python.exe
cachedir: .pytest_cache
rootdir: E:\PGD-CCEE\C04 - Machine Learning\Lecture Notes\L03-04\Code\Poetry\class04
configfile: pyproject.toml
collected 2 items

tests/test_main.py::test_add_two_number PASSED                                                       [ 50%]
tests/test_main.py::test_add_two_number_2 PASSED                                                     [100%]

============================================== 2 passed in 0.02s ==============================================
```