Ask aziz about AWS

Json and text :

API : making your code available to other in a structured way.

From fastapi import FastAPI

From fastapi.responses impoirt PlainTextResponse, JSONResponse

App = FastAPI()

/docs ( to see all the things we have done.

@app.get(‘/test’, response\_classs = JSONResponse) (decorator to tell where the out is)

Def test\_endpoijtn():

Return

Terminal :

Uvicorn main:app –reload

We are not going to return the plain text often, we would return the json much often.

Json is like a dict

Typing:

From typing import Union, Optional

Typing\_test.py

Def test\_function() -> Union[float,None]:

Optional(float) : same as union : if the retun can be converted to float then it returns float, otherwise it returns the same type as the basic variable.

Def dict\_unpack (val : dict[str,int])

The typing can help us to understand when the code become overwhelming and we need some structure to have it. -> it will be more readable for us.

Respone Models:

Pydantic : Check Type of data are correct

If we don’t initialize the Optional and we don’t pass the argument, it will give intenal server error!

Passing from int to str is fine but from str to int is not gonna work even with Optional

We can have a class type for an attribute inside a class.

Field name:

If we provide an alias, it means we are overriding the name and we have to make the necessary changes.

We can have inheritance of a field from another field.

We mass it just by Newinheredted ( \*\*User.dict(),\*\*new\_attr)

* For inheretence, it makes us able not to repeat ourselves and make code clean.

Path parameters:

@app.get('/user/{user\_id}',response\_model= User)

Adding the user\_id here,

We can modify our functions to have two things, a default one and the user\_id specific so that it returns what we want.

If you have a special case, make sure to code it before the {}

Request body

Get metod pof the decorator : It reads data

Post: it adds data > creation

See discord for the rest of this session

Testing

It’s a way to automate things to make sure that everything that was working previously continue to do so.

Pytest Intro

We want to do things at the lowest level to test everything.

For pytest, we have to have a **test prefix** in a function

We need to have assert statement.

In windows, just use pytest without \*

-vvv : as many v as you have increases the information**. -vv** is the best

If we have print statements, they are getting printed only when a test fails. Which helps during debugging.

Always test edge tests to test the bug. -> We want to get the test that reproduce the bug.

If our delete user is also async, we need also to await. But it’s complicated, all we need is to install pytest-asyncio to add the async capability to our tests.