1. sudo vim query\_test.py를 새로 만든다

from fastapi import FastAPI

app = FastAPI()

fake\_items\_db = [{"item\_name": "Foo"}, {"item\_name": "Bar"}, {"item\_name": "Baz"}]

@app.get("/items/")

async def read\_item(skip: int = 0, limit: int = 10):

return fake\_items\_db[skip : skip + limit]

1. [http://3.80.156.43/items/](http://3.80.156.43/items/?skip=1&limit=2) 를 확인해보면 위 코드에서 설정한 skip==0, limit=10인것을 확인할 수 있다.

그러나 <http://3.80.156.43/items/?skip=1&limit=2> 처럼 주소에 값을 넣어주면 skip==1, limit==2 로 값이 들어간 것을 볼 수 있다.

1. 코드를 추가해준다
   1. from typing import Optional
   2. @app.get("/items/{item\_id}")

async def read\_item(item\_id: str, q: Optional[str] = None):

if q:

return {"item\_id": item\_id, "q": q}

return {"item\_id": item\_id}

* 1. items 뒤에 str을 넣으면 이 코드가 실행되고, q가 있다면 q까지 출력된다.
     1. <http://3.80.156.43/items/item?q=sex> 이렇게 ?로 q의 값을 붙혀 전달해야 한다

1. 코드를 추가해준다

@app.get("/items/{item\_id}")

async def read\_user\_item(

item\_id: str, q: Optional[str] = None, short: bool = False

):

item = {"item\_id": item\_id}

if q:

item.update({"q": q})

if not short:

item.update(

{"description": "This is an amazing item that has a long description"}

)

return item

위 코드에서 item\_id는 기존과 같지만 short: bool은 기본값이 False이며

<http://3.80.156.43/items2/test?short=true> 와 같이 bool값을 줄 수 있다.

이때

참 : 1(0이 아닌 정수), true, on, yes등으로 줄 수 있고,

거짓 : 0, false, off, no 등으로 줄 수 있다.

1. HTML 사용하기

from fastapi import FastAPI

from fastapi.responses import HTMLResponse

app = FastAPI()

@app.get("/items/", response\_class=HTMLResponse)

async def read\_items():

return

"""

<html>

<head>

<title>Some HTML in here</title>

</head>

<body>

<h1>Look ma! HTML!</h1>

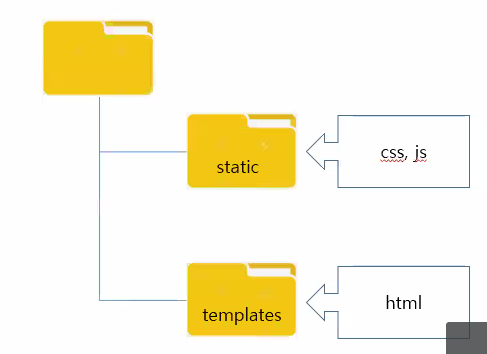
</body>

</html>

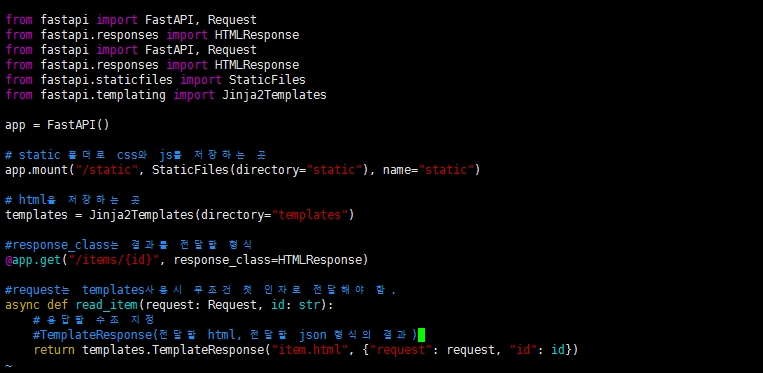
"""

# “”” 가운데에 있는 html을 보여준다.

# @app.get에 있는 response\_class를 지우면 html형식이 아니라 그냥 text로만 나오게 된다.



1. static
   1. app.mount(경로, 실 디렉토리 이름, 구불할 이름)
   2. app.mount(“/static”, StaticFiles(directory=”static”), name=”static”)
2. templates
   1. templates = Jinja2Templates(directory=”templates”)
3. tmp.py



app.mount로 css나 js파일이 들어있는 디렉토리를 정의한다.

또 Jinja2templates로 html 파일들이 들어있는 폴더를 정의한다.

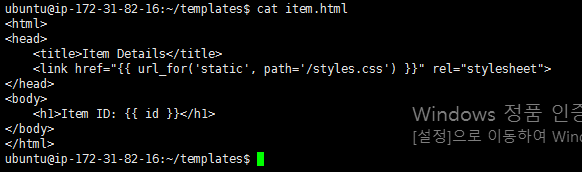
@app.get에서 response\_class로 HTML결과를 전달할 형식을 정해주고

temlates.TemlatesResponse로 결과를 리턴한다. 이때 반드시 Reauest가 먼저 나와야 한다.

1. styles 폴더와 그안에 css파일



1. templates 폴더와 그 안에 item.html파일



1. 그리고 결과

