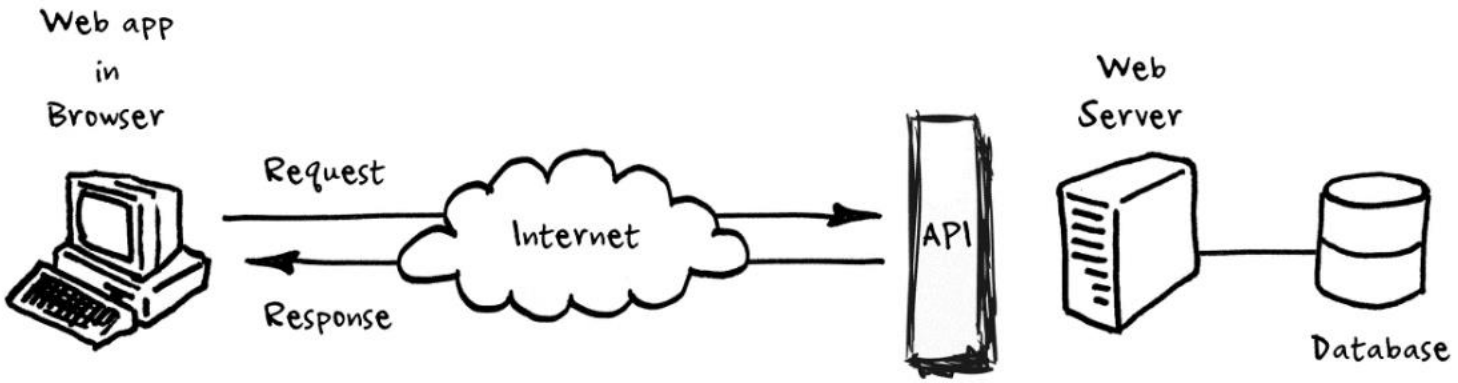
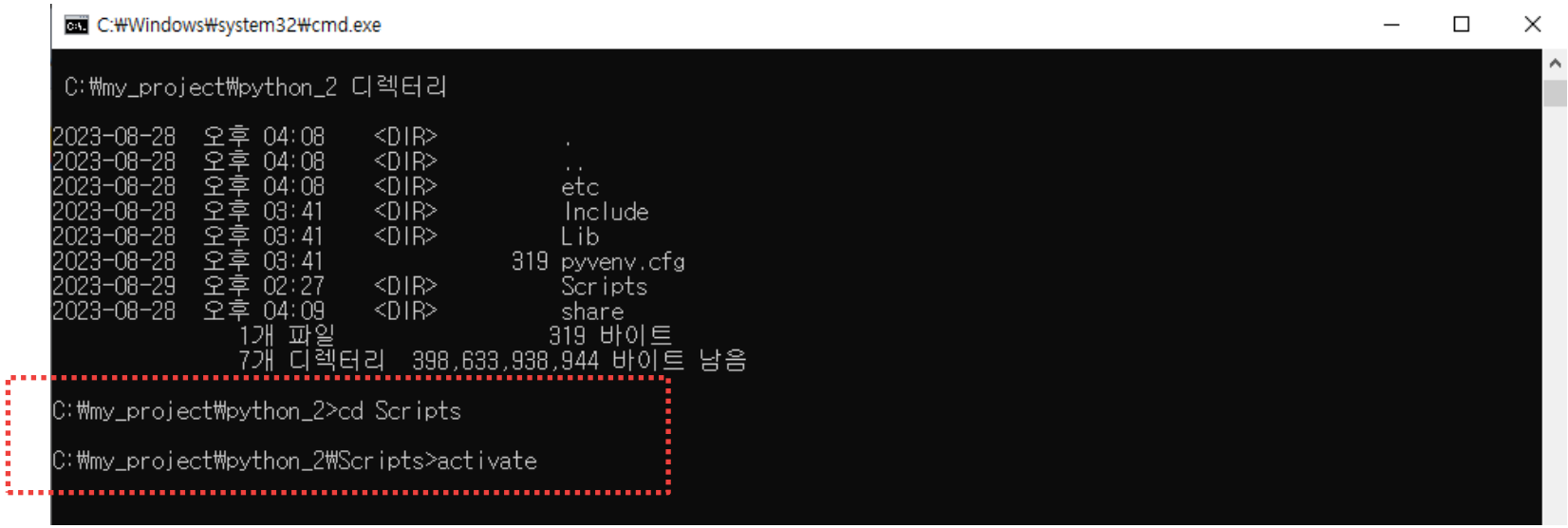
Fast API

https://replit.com/~ 에서 실행



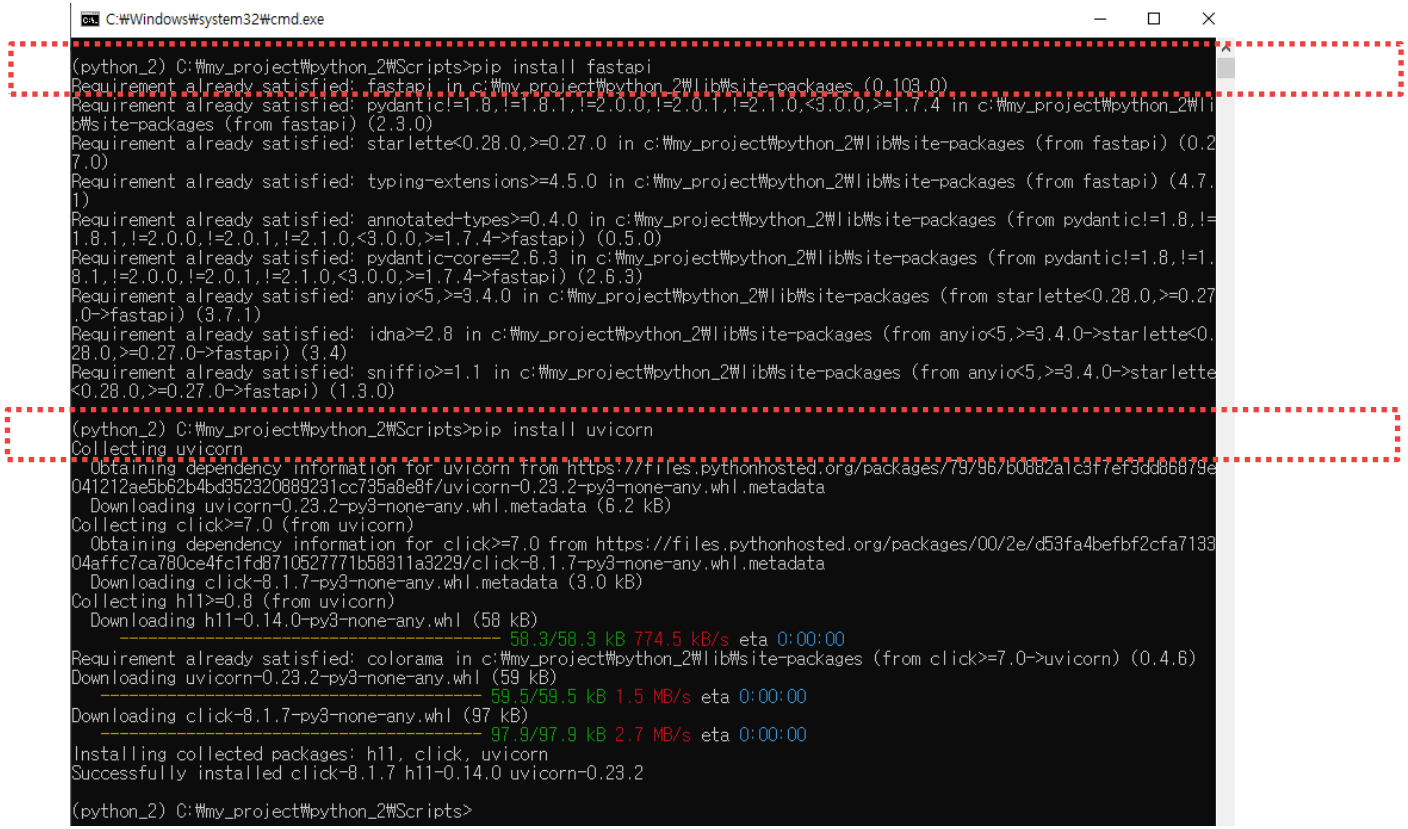
python 가상환경 디렉토리 접근

- **1. 'C:\my_project\python_2\Scripts'로 이동하여 'activate'입력하여 가상환경 생성**



python 가상환경에 FastAPI와 uvicorn 설치

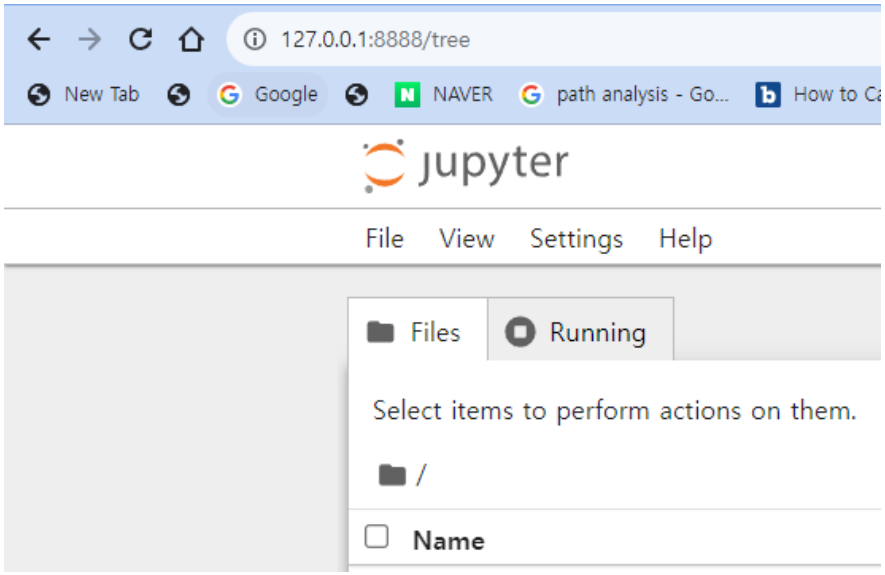
- **2. pip install fastapi**
- **3. pip install uvicorn**



python 가상환경에서 노트북 열기

- **4. ‘jupyter notebook’을 입력하고 실행한다.**

  (python_2) C:\my_project\python_2\Scripts> jupyter notebook



FastAPI 따라하기

- **5. FstAPI 사이트'https://fastapi.tiangolo.com/ko/'를 따라한다.**

- `main.py` 파일을 만드십시오:

```python
from typing import Union

from fastapi import FastAPI

app = FastAPI()


@app.get("/")
def read_root():
    return {"Hello": "World"}


@app.get("/items/{item_id}")
def read_item(item_id: int, q: Union[str, None] = None
    return {"item_id": item_id, "q": q}
```

실행하기

서버를 실행하십시오:

```bash
$ uvicorn main:app --reload

INFO:     Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)
INFO:     Started reloader process [28720]
INFO:     Started server process [28722]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
```

`uvicorn main:app --reload` 명령에 관하여...

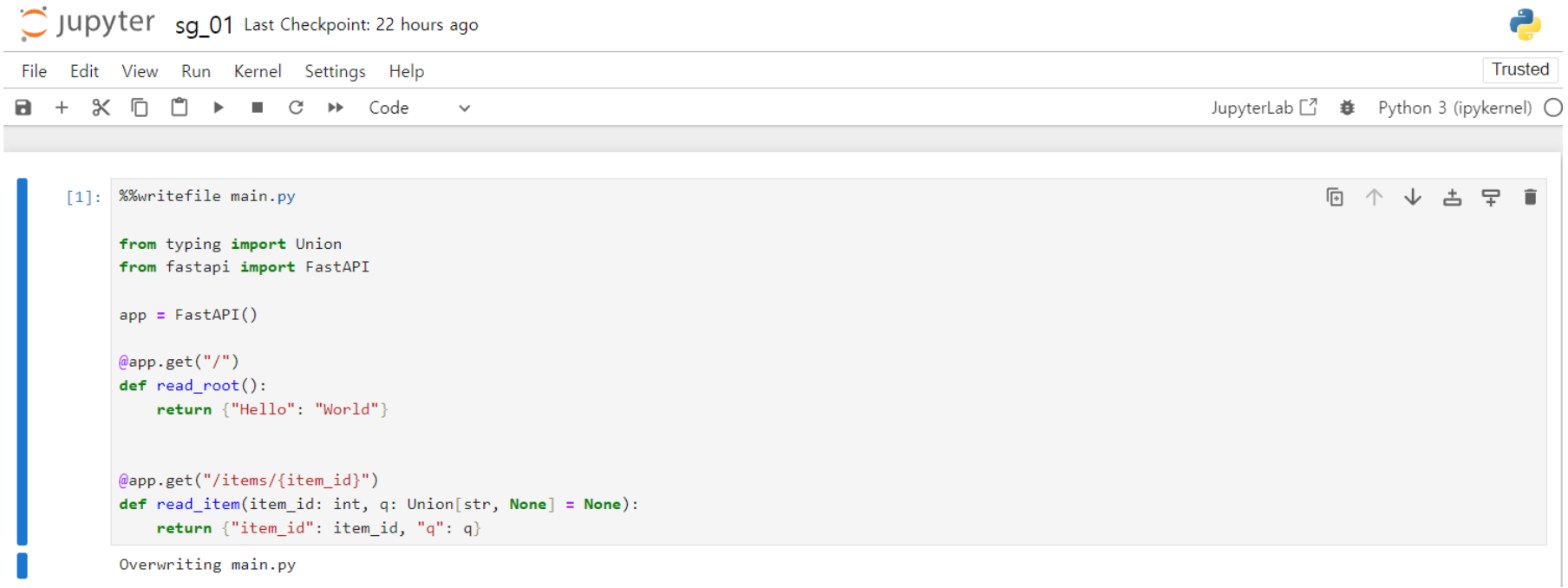
명령 `uvicorn main:app` 은 다음을 나타냅니다:

- `main` : `main.py` 파일 (파이썬 "모듈").
- `app` : the object created inside of `main.py` with the line `app = FastAPI()` .
- `--reload` : 코드가 변경된 후 서버 재시작하기. 개발환경에서만 사용하십시오.

FastAPI 실행을 위해 파이썬파일 만들기

- **6. FstAPI 사이트'https://fastapi.tiangolo.com/ko/' 요래**

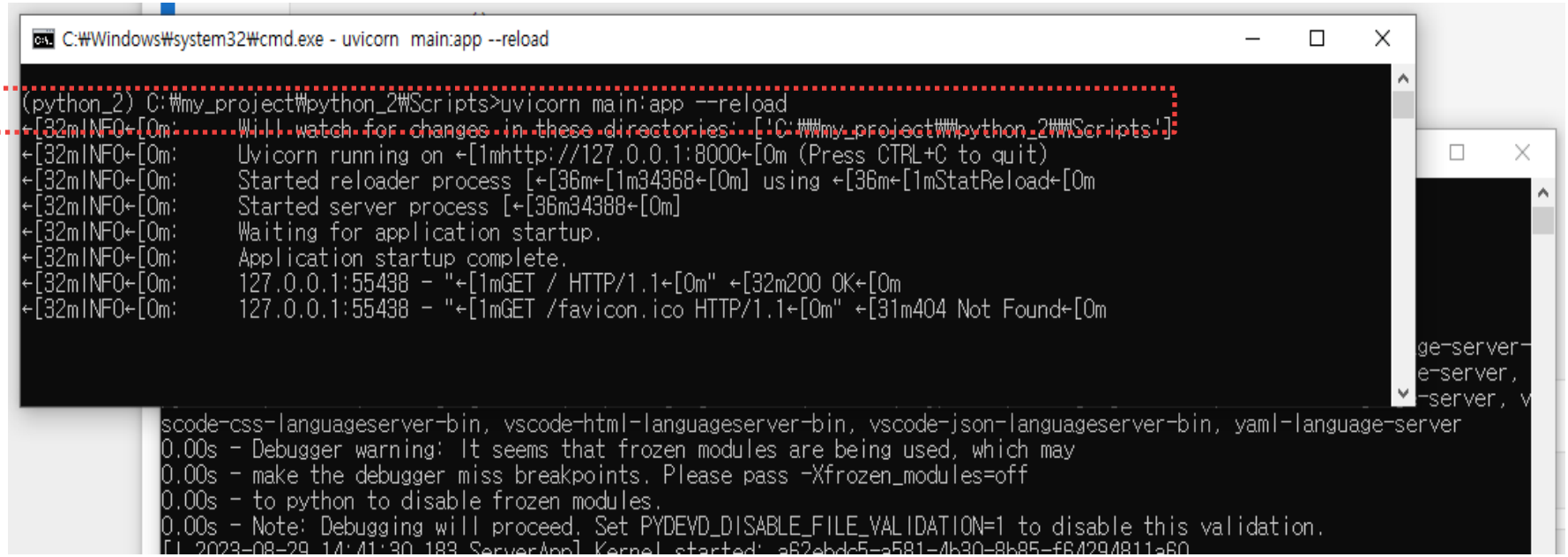
%%writefile main.py # 현재 디렉토리에 main.py라는 이름으로 파일 생성



Fast API와 웹서버(uvicorn) 실행하기

- **7. 도스화면을 하나 더 열고, 가상환경하에서**

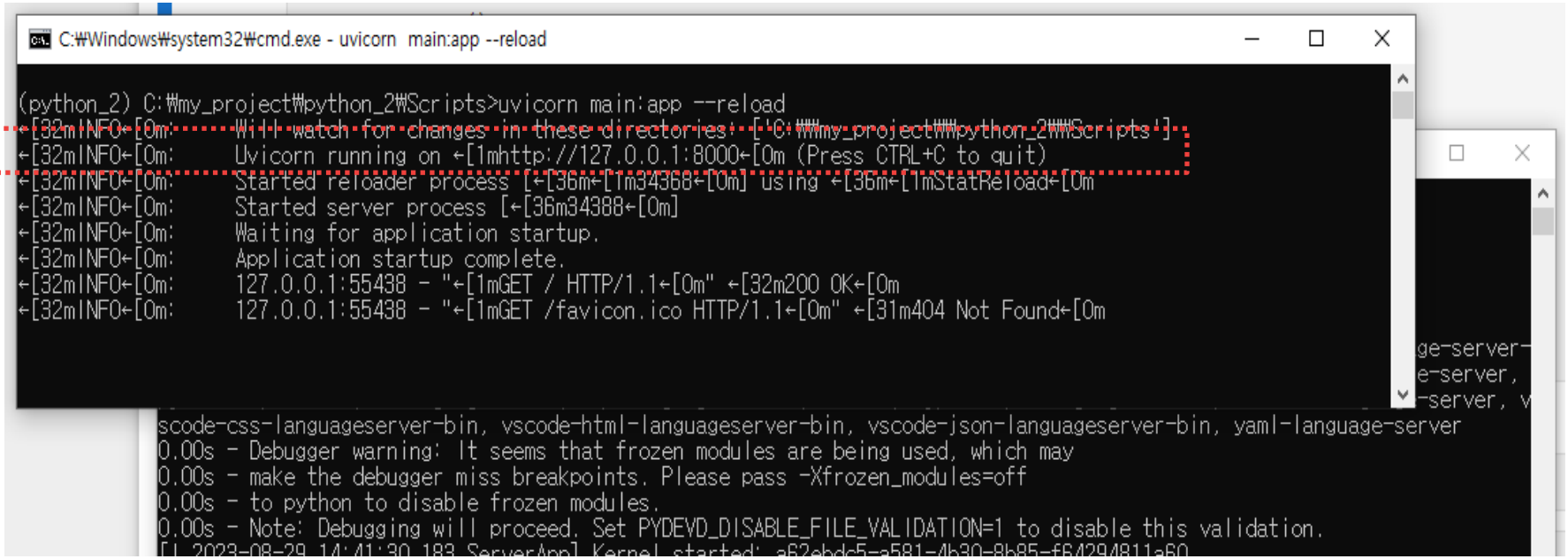
  도스 prompt에서 'uvicorn main:app –reload' 명령어 입력(main.py를 실행)



실행 결과 확인하기

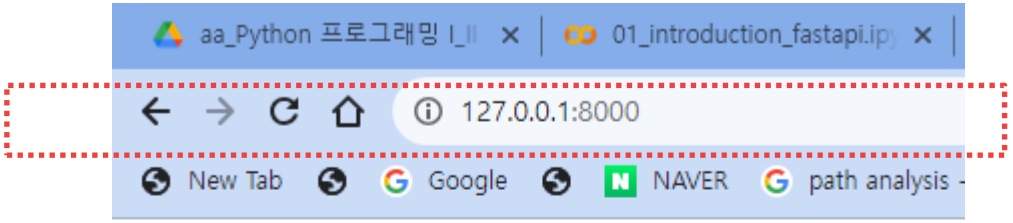
- **8. 'http://127.0.0.1:8000' 주소로 가서 화면을 열어본다**

  Uvicorn running on [1mhttp://127.0.0.1:8000[0m (Press CTRL+C to quit)



실행 결과 확인하기

- **8. 'http://127.0.0.1:8000' 주소로 가서 화면을 열어본다**



{"Hello":"World"}

확인하기

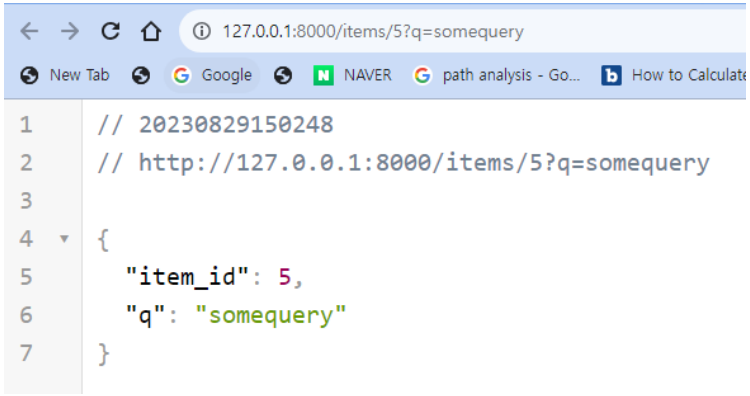
브라우저로 http://127.0.0.1:8000/items/5?q=somequery [↪]를 열어보십시오.

아래의 JSON 응답을 볼 수 있습니다:

```
{"item_id": 5, "q": "somequery"}
```

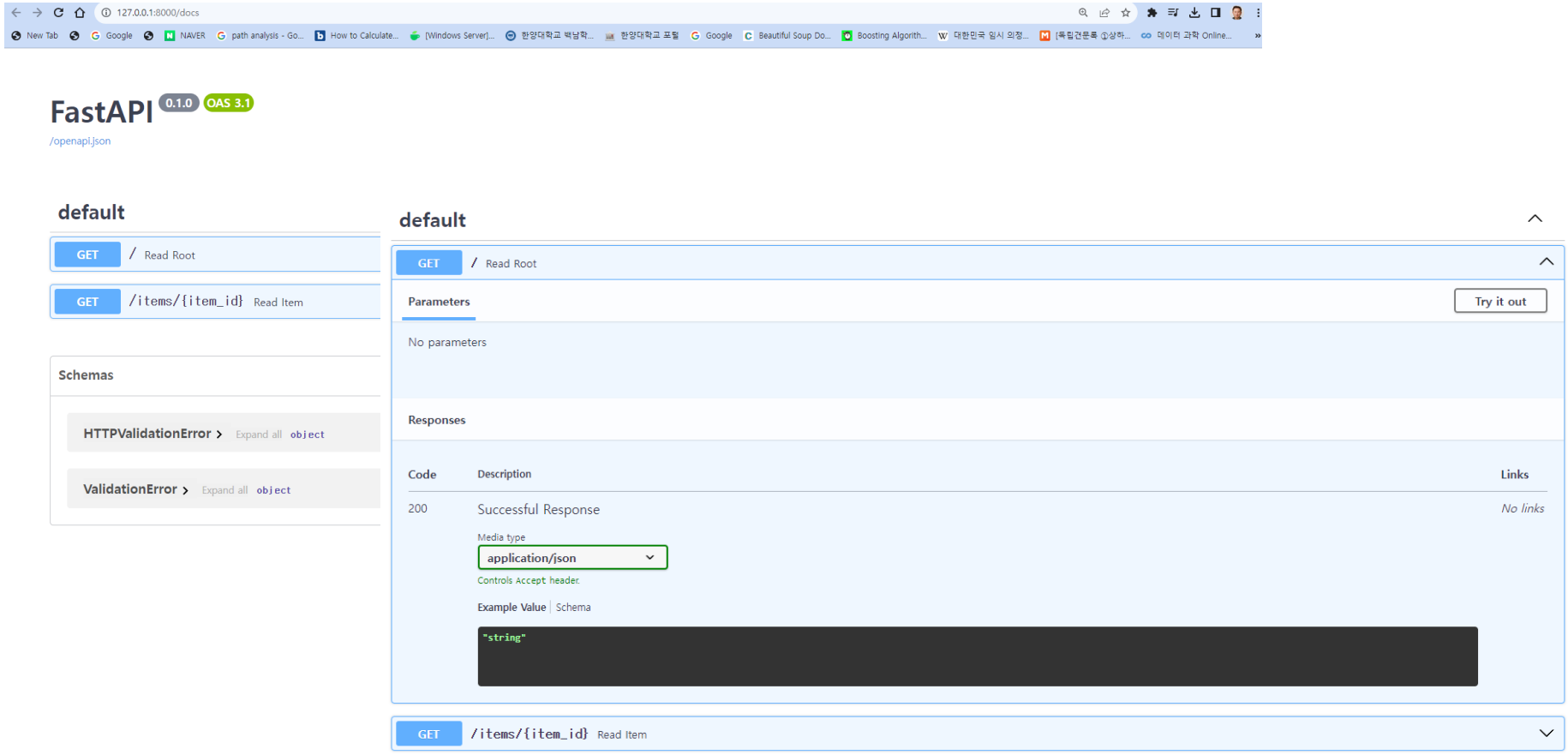
여러분은 벌써 API를 만들었습니다:

- *경로* `/` 및 `/items/{item_id}`에서 HTTP 요청 받기.
- 두 *경로*모두 `GET` *연산*(HTTP *메소드*로 알려진)을 받습니다.
- *경로* `/items/{item_id}`는 *경로 매개변수* `int`형 이어야 하는 `item_id`를 가지고 있습니다.
- *경로* `/items/{item_id}`는 선택적인 `str`형 이어야 하는 *경로 매개변수* `q`를 가지고 있습니다.



실행 결과 확인하기

- 9. http://127.0.0.1:8000/docs end point로 가라



Pycaret with FastAPI

https://www.datacamp.com/tutorial/introduction-fastapi-tutorial

CONTENTS

SHARE

```
best = compare_models()
```

Explain code

Powered by OpenAI

	Model	MAE	MSE	RMSE	R2	RMSLE	MAPE	TT (Sec)
gbr	Gradient Boosting Regressor	2520.6229	20769829.9785	4525.8866	0.8568	0.3947	0.2709	0.0230
catboost	CatBoost Regressor	2711.2094	22678972.2643	4730.0674	0.8437	0.4191	0.2886	0.4740
lightgbm	Light Gradient Boosting Machine	2855.9507	23314394.3696	4791.7314	0.8394	0.4880	0.3225	0.2960
rf	Random Forest Regressor	2689.3003	23521544.1910	4826.4161	0.8380	0.4269	0.2863	0.0870
ada	AdaBoost Regressor	3833.7662	25143449.8335	4987.6102	0.8265	0.5675	0.6121	0.0080
et	Extra Trees Regressor	2690.2509	26168744.7162	5092.3057	0.8195	0.4448	0.2821	0.0790
xgboost	Extreme Gradient Boosting	3108.6635	28860047.6000	5336.6700	0.7998	0.4893	0.3525	0.2740
llar	Lasso Least Angle Regression	4213.1579	35666189.6567	5947.2977	0.7551	0.5901	0.4213	0.0060
ridge	Ridge Regression	4231.2149	35681675.4000	5949.1723	0.7550	0.5831	0.4252	0.0050
lasso	Lasso Regression	4218.5314	35675973.2000	5948.1665	0.7550	0.5968	0.4235	0.0060
br	Bayesian Ridge	4228.1661	35684770.8810	5949.2847	0.7550	0.5846	0.4248	0.0050
lr	Linear Regression	4219.3180	35688562.0000	5949.2035	0.7549	0.6007	0.4236	0.6830
lar	Least Angle Regression	4276.7246	36316676.7397	5995.9336	0.7502	0.6042	0.4403	0.0050
dt	Decision Tree Regressor	2987.3224	39904963.5364	6267.2071	0.7280	0.4740	0.2795	0.0110
huber	Huber Regressor	3483.0094	46557209.1038	6773.2554	0.6803	0.4893	0.2349	0.2230
omp	Orthogonal Matching Pursuit	5756.4046	57504547.0291	7551.4122	0.6063	0.7200	0.8625	0.0060
par	Passive Aggressive Regressor	4927.7735	64161499.7943	7976.0997	0.5625	0.6802	0.4169	0.0070
en	Elastic Net	7229.1871	87894292.8000	9364.7893	0.4006	0.7166	0.8866	0.0070
knn	K Neighbors Regressor	7753.2965	124523234.4000	11119.9351	0.1540	0.8098	0.8460	0.0150
dummy	Dummy Regressor	9117.1966	148041516.8000	12143.8482	-0.0043	0.9955	1.5027	0.0070