@@@그냥 REST API의 개념에 대한 공부 @@@

REST: REpresentational State Transfer  
API: Application Programming Interface

# REST의 구성: 자원-URI, 행위-HTTP METHOD, 표현

# REST API 중심 규칙

1. URI는 정보의 자원을 표현해야 한다. (리소스명은 동사가 아닌 명사를 사용)  
GET /members/delete/1 이런 식으로 쓰지 않는다. delete이런거 쓰지 말라고.  
DELETE /members/1 이게 맞는거겠지.

2. 자원에 대한 행위는 HTTP Method(GET, POST, PUT, DELETE 등)로 표현  
정보 가져올 때 GET, 회원 추가할 때 POST 이런식으로 쓰면 되겠지  
ex) 회원 정보 가져오는 URI  
GET /members/show/1 (x)  
GET /members/1 (o)

회원 추가  
GET /members/insert/2 (x)  
POST /members/2

POST -> 리소스 생성 GET -> 리소스 조회하고 정보 가져오기  
PUT -> 리소스 수정 DELETE -> 리소스 삭제

# URI 설계 시 주의점

1. 슬래시 구분자(/)는 계층 관계를 나타내는 데 사용.  
$ /animals/mammals/whales

2) URI 마지막 문자로 슬래시(/)를 포함하지 않는다.  
$ animals/mammals/whales/ (X) $ animals/mammals/whales (O)

3) 하이픈(-) 쓸라면 쓰셈. 대신 언더바( \_ ) 는 쓰지 마라.  
4) 가급적이면 소문자만 써라. 대문자 쓰면 다른 걸로 구분되니까 소문자로 통일 rr  
5) 파일 확장자는 URI에 포함 안된다.

# 리소스간 관계 표현하기  
/리소스명/리소스 ID/관계가 있는 다른 리소명  
ex) GET : /users/{userid}/devices  
 GET : /users/{userid}/likes/devices

# HTTP response 코드

200 : 클라이언트의 요청을 정상적으로 수행함  
201 : 클라이언트가 어떤 리소스 생성을 요청했을 때, 해당 리소스가 성공적으로 생성됨.  
400 : 클라의 요청이 부적절합니다. 이상한거 시켰네요.  
401 : 클라가 인증되지 않았는데, 보호된 리소스를 요청했습니다.  
(클라 로그인 안하셨는데, 로그인 해야 요청 가능한 리소스 요청하셨습니다.)  
403 : 요청 리소스가 있긴 한데 인증과는 관계없이 그냥 응답하기 싫습니다. (잘 안씀 가급적 404쓰셈) (404는 아예 없어서 응답 못한다는 뜻)  
405 : 클라이언트가 요청한 리소스에서는 사용 불가능한 Method를 이용했을 경우 사용하는 응답코드  
301 : 클라가 요청한 리소스에 대한 URI가 변경되었습니다. (Location header에 변경된 URI를 적어줘야합니다.)  
500 : 서버 이상!

@@@ django REST API @@@

우선 설치해줘야 할게 있다. 가상환경을 단디 키자.  
pip install ~  
djangorestframework: django rest api쓰기 위한 핵심적인 녀석  
django-rest-auth: 계정관리 할 때 쓰는 놈  
django-allauth: registration 할 때 쓰더 놈  
Pillow: 장고에서 미디어파일 다룰 때 있어야함. 따로 INSTALLED 안해도 됨

- INSTALLED\_APPS에 추가해야 하는 것들

#djangorestframework -> ‘rest\_framework’   
#django-rest-auth -> ‘rest\_auth’, ‘rest\_framework.authtoken’  
#django-allauth ->

    'django.contrib.sites',

    'allauth',

    'allauth.account',

    'rest\_auth.registration',

별도로 INSTALLED\_APPS 밖에 SITE\_ID = 1 변수 추가

~~~homework\_20210105~~~

setting.py 건드리고 이런거 다 생략한다. 앞으로 setting 건드리는건 알아서 해라. 언급 안한다. 앱 이름은 upload로 했다.

@@@사진 업로드@@@

루트 폴더에 ‘media’ 폴더를 만들었고, 이걸 MEDIA\_URL로 쓸거다. 이하 작업 알아서 해주자.

#upload/models.py

Writing라는 모델을 만들었다. 유저가 올리는 글임 ㅇㅇ. 구조는 나중에 제대로 짜도록 하고 일단은 사진 올리는거나 구경하도록 하자. 일단은 제목이랑 내용, 파일만 갖도록 했다.

    image = models.ImageField(upload\_to="", null=True)

좀 주목해야할 부분인데, upload to=”경로” 할 수도 있고 “%Y/%m/%d”해서 알아서 넣게 할 수도 있다. 참고로 upload to=의 디폴트는 MEDIA\_ROOT로 썼던 그 폴더다. 공란으로 두면 media바로 쏙 드간다. 저기다가 “media”한번 더 하면 뇌절이라는 뜻.  
뒤에 null=True 안하면 makemigrations 할 때 ‘저거 못 받을 수도 있잖아 ㅠㅠ’하면서 땡깡피운다.  
편집하고 migrate하는거 잊지 말자.

#upload/serializers.py

Django REST API에서는 serializer라는걸 쓰더라. 쿼리랑 모델인스턴스 같은걸 json이랑 xml데이터로 바꿔주는 놈이라 함. 당연히 처음 만들면 없으니 serializers.py 만들어줘야 함.

from rest\_framework import serializers

다른 놈들이 다 머리에 뭔가 달고 다니듯이 이게 있어야 시리얼라이저라는 뜻. 특히 이제 프레임워크를 쓸거라서 from rest\_framework는 앞으로도 계속 보인다.

from .models import Writing

class WritingSerializer(serializers.ModelSerializer):

    image = serializers.ImageField(use\_url=True)

    class Meta:

        model = Writing

        fields = ('title', 'text', 'image')

Writing모델에 대한 시리얼라이저를 만들어주자.

#upload/views.py

from rest\_framework import viewsets

serializers.py 처럼 얘도 rest\_framework쓰는게 생겼다.

from .serializers import WritingSerializer

from .models import Writing

얘네 쓸거니까 선언해주고 ㅇㅇ   
모델 만들어둔건 보통 뷰에서 이렇게 가져와서 쓴다. url은 view를 가져오고, view는 model을 가져오고 이런 관계인듯.

class WritingViewset(viewsets.ModelViewSet):

    queryset = Writing.objects.all()

    serializer\_class = WritingSerializer

Post에 대한 뷰셋이다. 쿼리는 Post한테 있는거 싹 다 긁어오고, 시리얼라이저는 아까 만든거.

#project/urls.py

각 앱에서 urls 안만들고 프로젝트에서 하나봄?

from rest\_framework import routers

urls.py도 rest\_framework의 작품을 쓴다. 라우터? URL을 맵핑해준댄다. 뭔진 모르겠는데 일단 써

from django.conf.urls import url, include

원래는 project/urls.py에서

from django.urls import path, include

이렇게 썼었다. 프레임워크 쓸땐 이렇게 하나보다. 그냥 ㄹㅇㅋㅋ나 치는 수 밖에 없다.

from upload import views

이것도 이제 설명 안한다. url이니까 views를 가져와 쓰는거다.

router = routers.DefaultRouter()

router.register(r'writings', views.WritingViewset)

urlpatterns = [

    url(r'^',include(router.urls)),

    url(r'^api-auth/',

include('rest\_framework.urls', namespace='rest\_framework'))

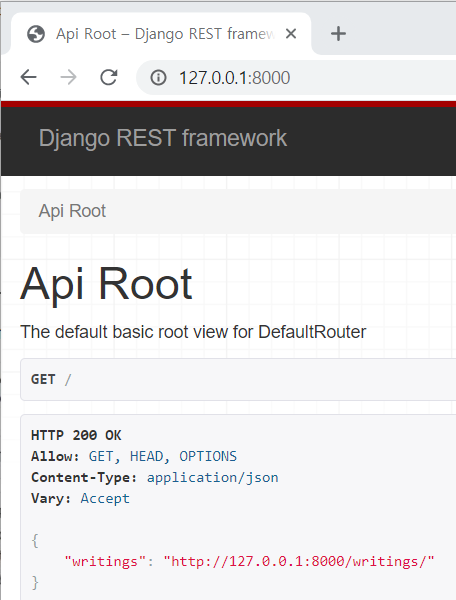
]

라우터 관련된 정규표현식이랑 urlpatterns의 새로운 표현법같은데 마찬가지로 ㄹㅇㅋㅋ나 치는 수 밖에 없다.

router.register(r'writings', views.WritingViewset)

근데 여기 이 부분에서 r’writings’했기 때문에 127.0.0.1:8000/writings/가 됨. 주소 바꾸기 가능.

#127.0.0.1:8000/

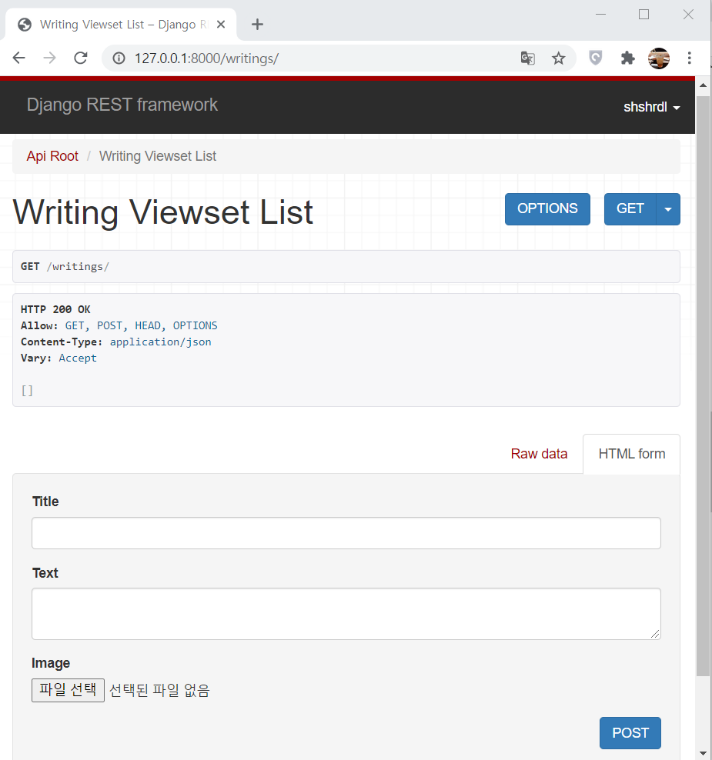


이런 모습이다.

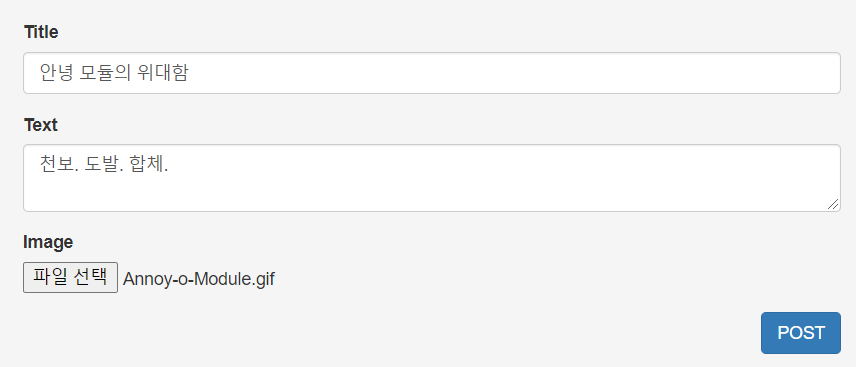
"writings": "<http://127.0.0.1:8000/writings/>"

클릭하면

#127.0.0.1:8000/writings/



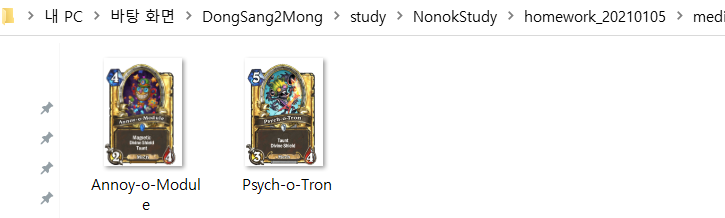
이래 나온다.



이래 쓰고 파일 선택해서 POST하면



요로케 잘 추가가 되고 있다.



사진도 잘 받아오는 모습이다.

@@@계정 관리 추가하기@@@

!졸려서 스크린샷 뜨기 귀찮음!  
그냥 py manage.py createsuperuser 할 수 있긴 한데 그거랑 좀 다른건가보다. setting 편집 생략함.

#project/urls.py -> urlpatterns

    url(r'^rest-auth/', include('rest\_auth.urls')),

    url(r'^rest-auth/registration/', include('rest\_auth.registration.urls')),

이후 migrate

127.0.0.1:8000/rest-auth/registration/에서 계정 등록할 수 있고  
127.0.0.1:8000/rest-auth/login/에서 로그인 할 수 있다.

!!이 url(r'^rest-auth/registration/', 부분도 /rest-auth/resgistration/ 바꿔서 주소 바꿀 수 있는듯. 실험해봤는데   
위의 r’^rest-auth/’ -> r’’  
아래의 r’^rest-auth/registration/’ -> r’^registration/’ 하니까  
127.0.0.1:8000/registration/  
127.0.0.1:8000/login/으로 바뀜