pycharm

wsl에서 장고 하는법

pip install virtualenv
cd workspace/
pip install django
django-admin
django-admin startproject mysite ->안전하게 하려고 가상환경
ls | grep mysite ->폴더확인

virtualenv -> python module, 현재설치된 파이썬 버전 따라감
python -m virtualenv venv // venv라는 폴더생성
가상환경 사용하기 로그인할때는 venv폴더밑에 source ./venv/bin/activate 실행하자!
# python3 -m virtualenv venv
# source ./v
# source ./venv/bin/activate //이제 venv로 들어가진다
which python - 내가 어느 파이썬을 실행하겠다 /home/jotaesik/workspace/venv/bin/python, 시스템상의 파이썬이 아닌 독립적인파이썬
pip list | grep djang   //장고버전알아보자 안뜰시
pip install django 장고설치하기
django-admin startproject mysite

conda - anaconda, miniconda를 설치해야 사용 가능 , 32bit 환경세팅가능, 다양한 파이썬 버전 세팅가능

file-open 들어가서 /home/jotaesik/workspace/mysite
cd ..
source ./venv/bin/activate
ls -al - 버전확인

서버가동
python manage.py runserver //런서버라는 매개변수
실시간 웹에보여준다
sqlite3 내장db이다

setting.py에서 두줄바꾸기
LANGUAGE_CODE = 'ko'
TIME_ZONE = 'Asia/Seoul'

python manage.py migrate

#https ://sqlitebrowser.org/dl/ 들어가서 - DB Browser for SQLite - .zip (no installer) for 64-bit Windows 설치.
그리고 exe 실행.

/home/jotaesik/workspace/mysite/db.sqlite3 lock걸려있으므로 아무곳이나 복사한후
db browser.exe에서 실행시키기

ORM-object relational mapping

sqlalchemy-sql을 몰라도 관계로 연결시켜주므로 바로 db에 넣을수있다.

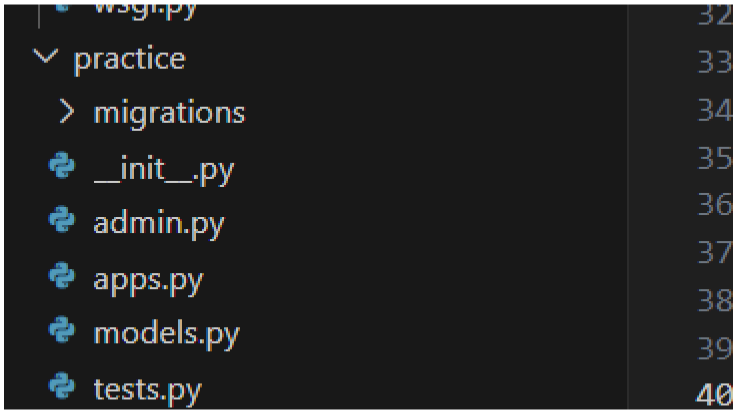
웹서버는 아파치는 정적인거

wsgi서버는 동적인거
요새 nginx를 사용한다
브니콘 , 톰캣(wsgi와 웹서버 둘다 역할한다)

oracle weblogic server- wasi서버

venv terminal에서 쳐보기

python manage.py startapp practice



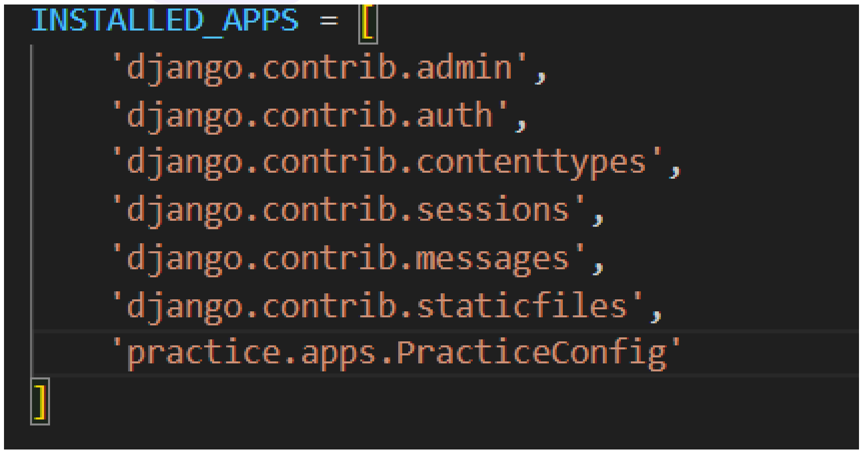
def TellHello(request):#실시간변수전달

html="

Hi!!!!

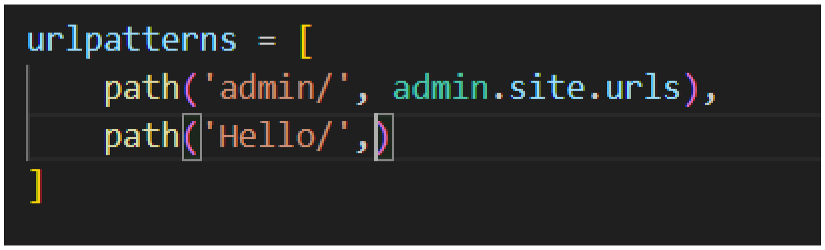
"

return HttpResponse(html) #문자열을 html로 만들어주는 함수



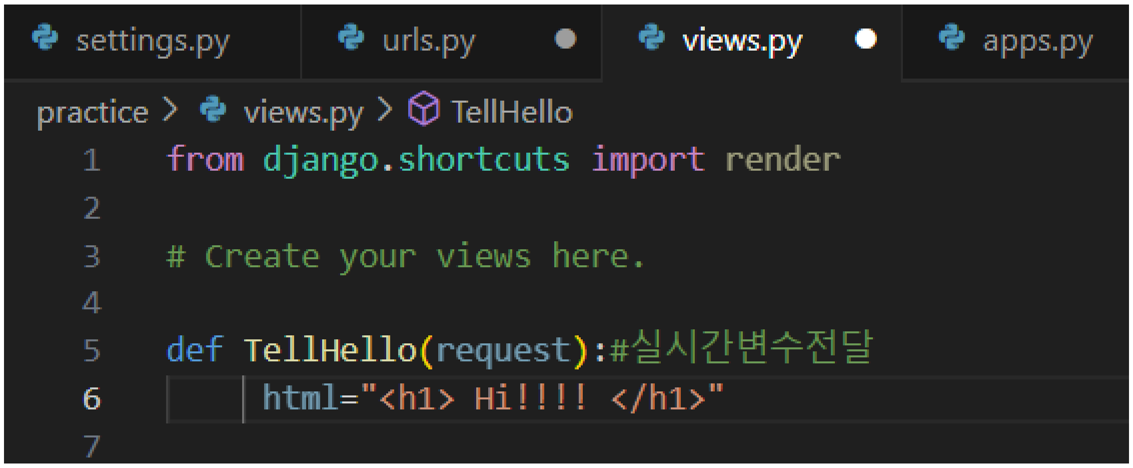
```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'practice.apps.PracticeConfig'
]
```

settings.py들어가서
앱등록하기



```
urlpatterns = [
    path('admin/', admin.site.urls),
    path('Hello/',)
]
```

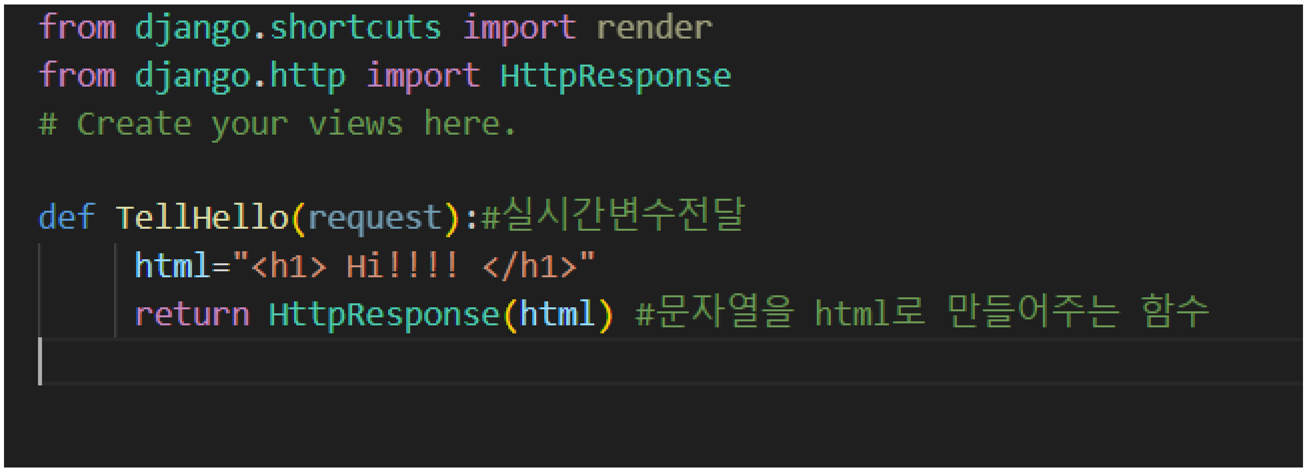
urls.py들어가서
practice->views.py에 로직을 적는다



settings.py urls.py views.py apps.py
practice > views.py > TellHello

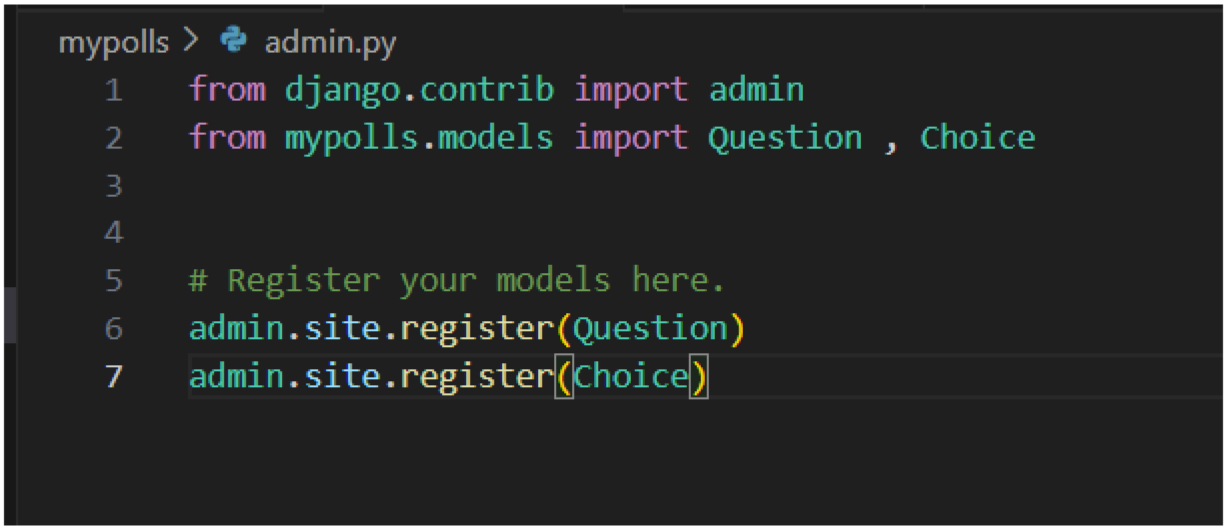
```
from django.shortcuts import render

# Create your views here.

def TellHello(request):#실시간변수전달
    html="<h1> Hi!!!! </h1>"

```

부트스트랩은 반응형웹,웹크기조절



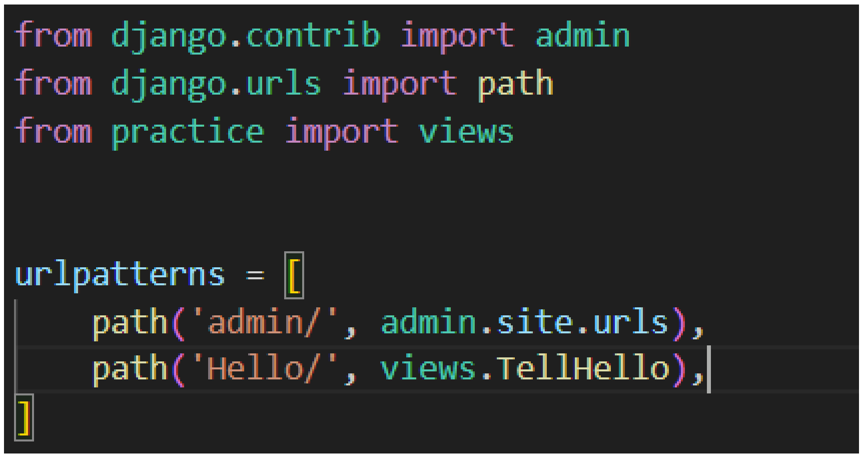
```
from django.shortcuts import render
from django.http import HttpResponse
# Create your views here.

def TellHello(request):#실시간변수전달
    html="<h1> Hi!!!! </h1>"
    return HttpResponse(html) #문자열을 html로 만들어주는 함수

```

mypools의 admin.py



```python
from django.contrib import admin
from mypolls.models import Question , Choice


# Register your models here.
admin.site.register(Question)
admin.site.register(Choice)
```

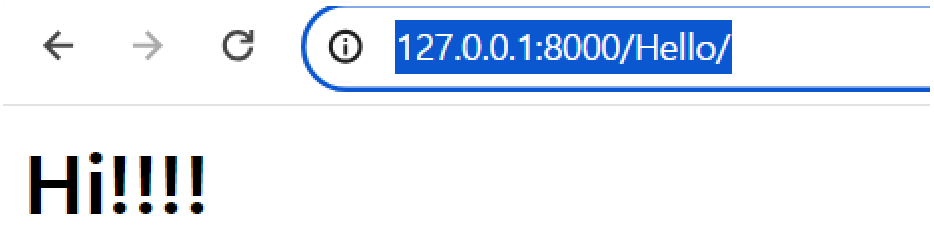
urls.py



```python
from django.contrib import admin
from django.urls import path
from practice import views


urlpatterns = [
    path('admin/', admin.site.urls),
    path('Hello/', views.TellHello),
]
```

python manage.py runserver 실행하기

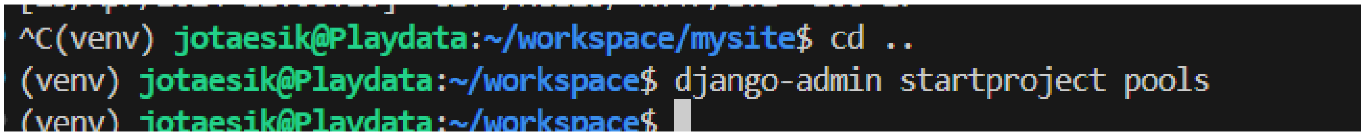
http://127.0.0.1:8000/Hello/



Hi!!!!

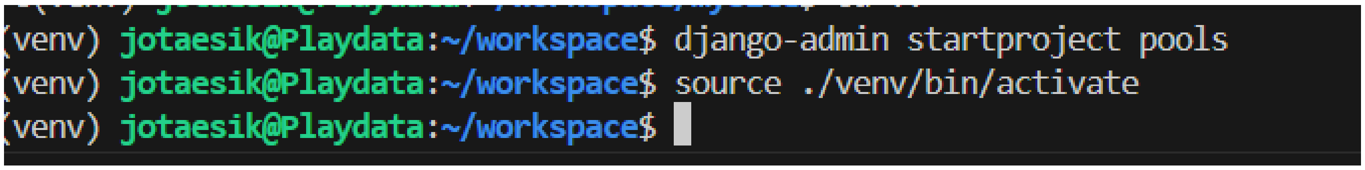
127.0.0.1 rootbackip localhost를 가르킨다. 본인자신을 의미.
port 8000

urls.py에 물어봐서 웹을 열어준다 mvc 패턴



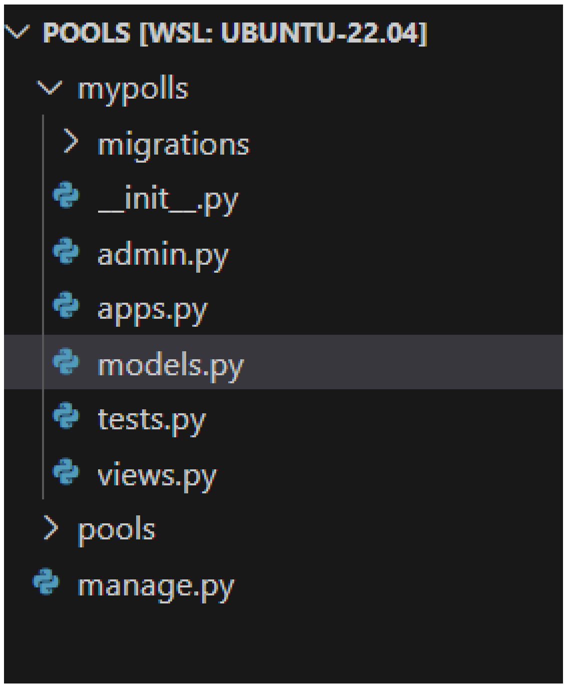
```
^C(venv) jotaesik@Playdata:~/workspace/mysite$ cd ..
(venv) jotaesik@Playdata:~/workspace$ django-admin startproject pools
(venv) jotaesik@Playdata:~/workspace$
```

상위폴더올라가서 pools라는 폴더하나더 만들기

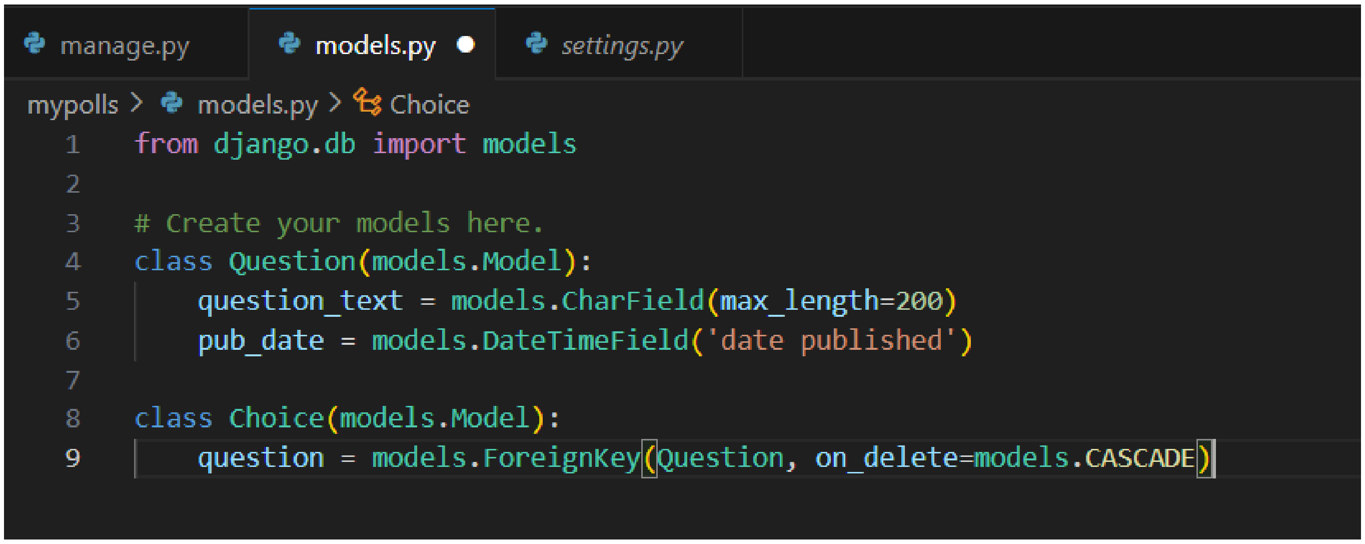


```
(venv) jotaesik@Playdata:~/workspace$ django-admin startproject pools
(venv) jotaesik@Playdata:~/workspace$ source ./venv/bin/activate
(venv) jotaesik@Playdata:~/workspace$
```

새로운 WSL 열어서

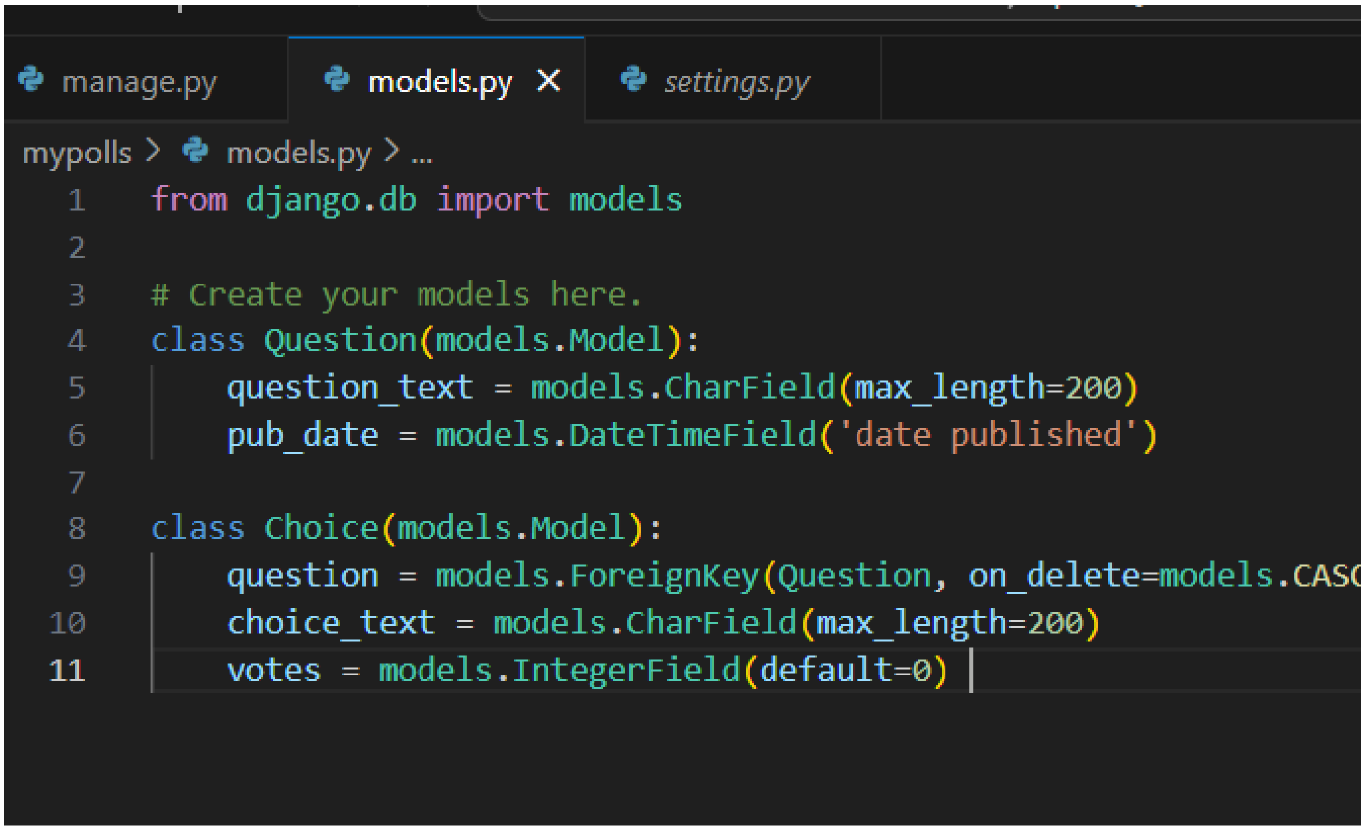


```
POOLS [WSL: UBUNTU-22.04]
  mypolls
    migrations
    __init__.py
    admin.py
    apps.py
    models.py
    tests.py
    views.py
  pools
  manage.py
```



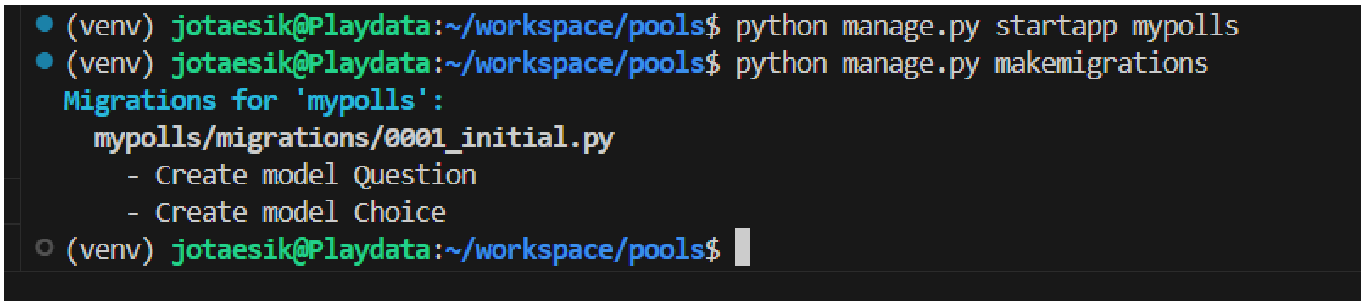
```python
from django.db import models

# Create your models here.
class Question(models.Model):
    question_text = models.CharField(max_length=200)
    pub_date = models.DateTimeField('date published')

class Choice(models.Model):
    question = models.ForeignKey(Question, on_delete=models.CASCADE)
```

models.py에 작성하기

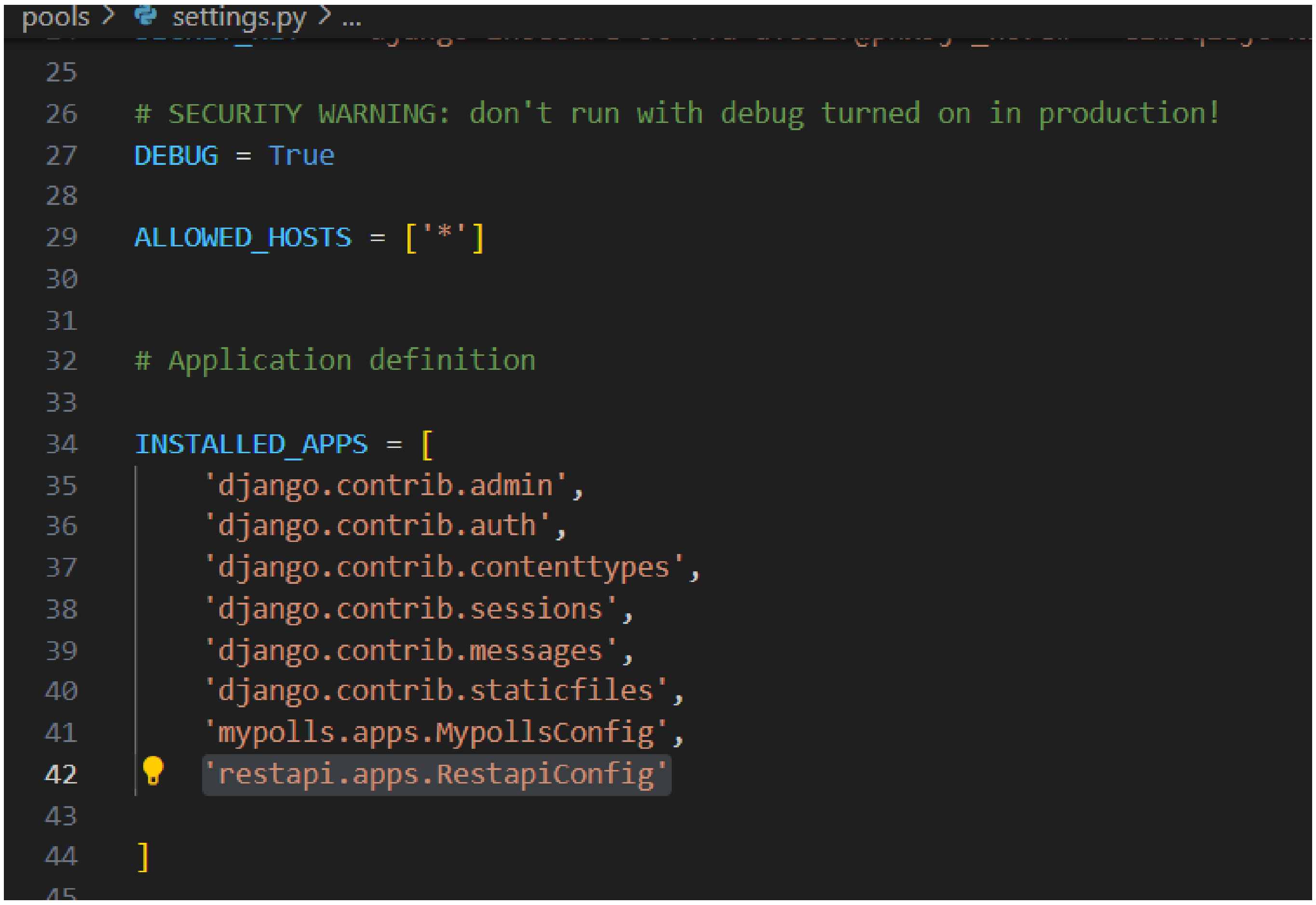


```python
from django.db import models

# Create your models here.
class Question(models.Model):
    question_text = models.CharField(max_length=200)
    pub_date = models.DateTimeField('date published')

class Choice(models.Model):
    question = models.ForeignKey(Question, on_delete=models.CASC
    choice_text = models.CharField(max_length=200)
    votes = models.IntegerField(default=0)
```

추가로작성하기



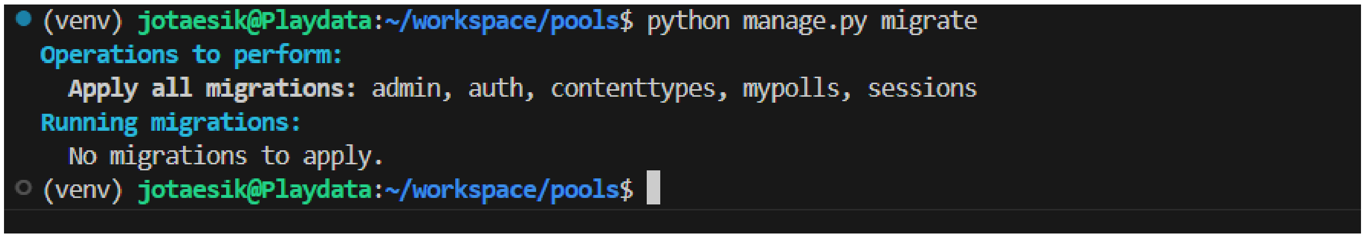
```
(venv) jotaesik@Playdata:~/workspace/pools$ python manage.py startapp mypolls
(venv) jotaesik@Playdata:~/workspace/pools$ python manage.py makemigrations
Migrations for 'mypolls':
  mypolls/migrations/0001_initial.py
    - Create model Question
    - Create model Choice
(venv) jotaesik@Playdata:~/workspace/pools$
```



pools > settings.py > ...

```python

# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True

ALLOWED_HOSTS = ['*']


# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'mypolls.apps.MypollsConfig',
    'restapi.apps.RestapiConfig'

]
```

'mypolls.apps.MypollsConfig', 추가하기
'restapi.apps.RestapiConfig' 추가하기
웹이름_클래스명
python manage.py sqlmigrate mypolls 0001 //sql보여준다
python manage.py migrate //commit 실행까지시켜준다



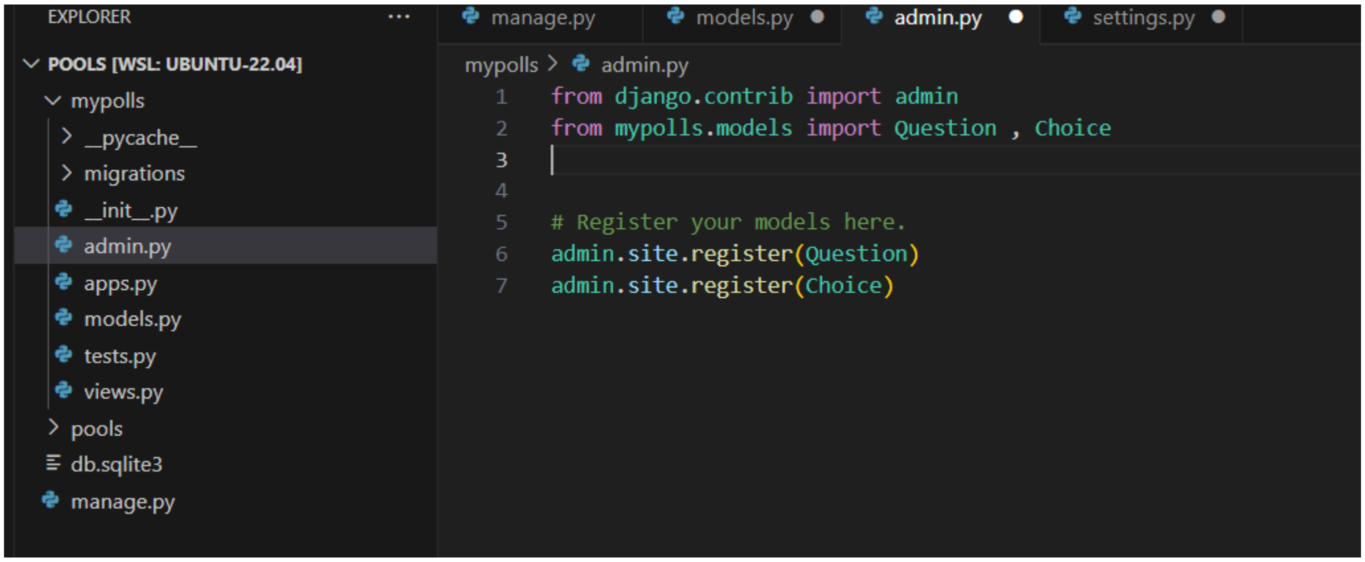
```
● (venv) jotaesik@Playdata:~/workspace/pools$ python manage.py migrate
  Operations to perform:
    Apply all migrations: admin, auth, contenttypes, mypolls, sessions
  Running migrations:
    No migrations to apply.
○ (venv) jotaesik@Playdata:~/workspace/pools$ 
```

sqlite에서 db.sqlite3 열기

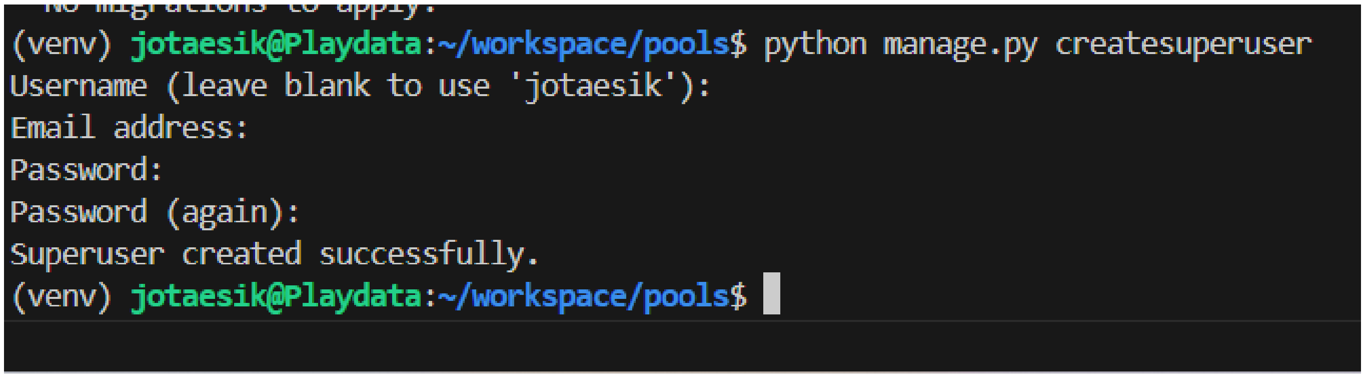
테이블(T): django_migrations

db구조 볼수있다.

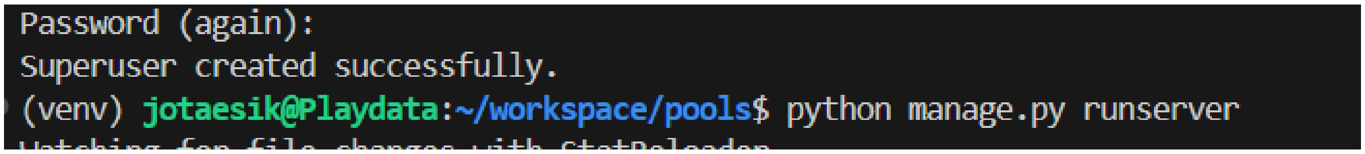
파이썬 셀러리와 래빗mq 비동기작업
장고 flask



```python
from django.contrib import admin
from mypolls.models import Question , Choice


# Register your models here.
admin.site.register(Question)
admin.site.register(Choice)
```

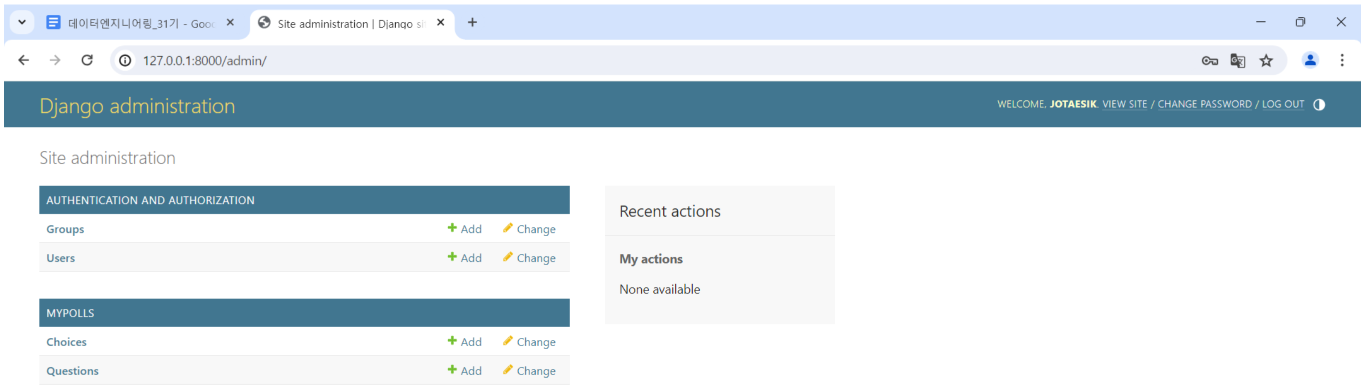


```
No migrations to apply.
(venv) jotaesik@Playdata:~/workspace/pools$ python manage.py createsuperuser
Username (leave blank to use 'jotaesik'):
Email address:
Password:
Password (again):
Superuser created successfully.
(venv) jotaesik@Playdata:~/workspace/pools$
```



```
Password (again):
Superuser created successfully.
(venv) jotaesik@Playdata:~/workspace/pools$ python manage.py runserver
Watching for file changes with StatReloader
```

http://127.0.0.1:8000/admin



Django administration

WELCOME, **JOTAESIK**. VIEW SITE / CHANGE PASSWORD / LOG OUT

Site administration

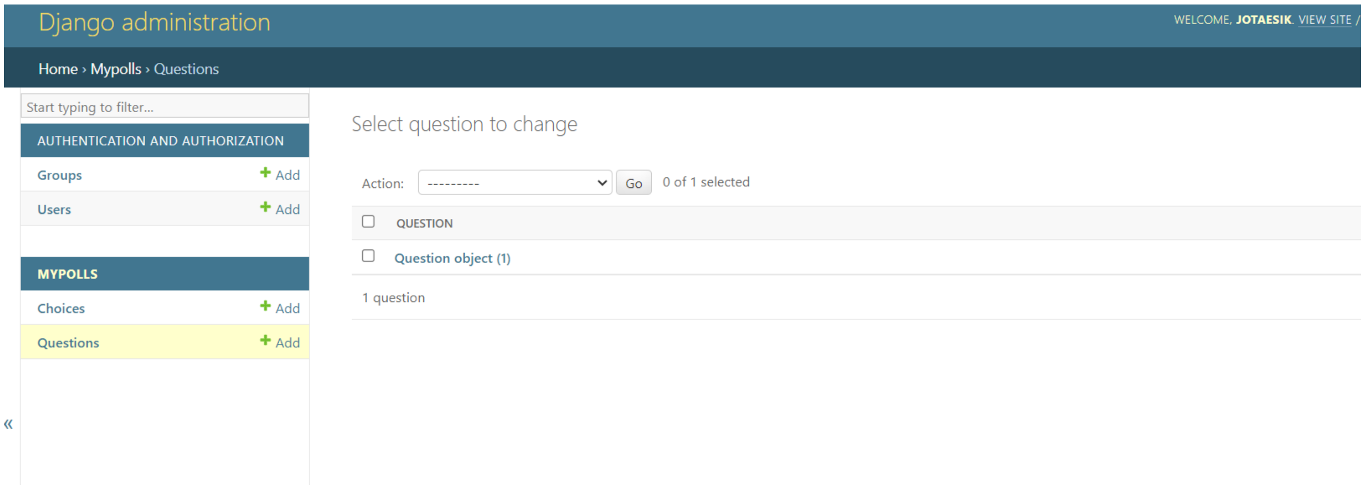
AUTHENTICATION AND AUTHORIZATION		
Groups	+ Add	Change
Users	+ Add	Change

MYPOLLS		
Choices	+ Add	Change
Questions	+ Add	Change

Recent actions

My actions

None available



Django administration

WELCOME, JOTAESIK. VIEW SITE /

Home › Mypolls › Questions

Start typing to filter...

AUTHENTICATION AND AUTHORIZATION

Groups + Add

Users + Add

MYPOLLS

Choices + Add

Questions + Add

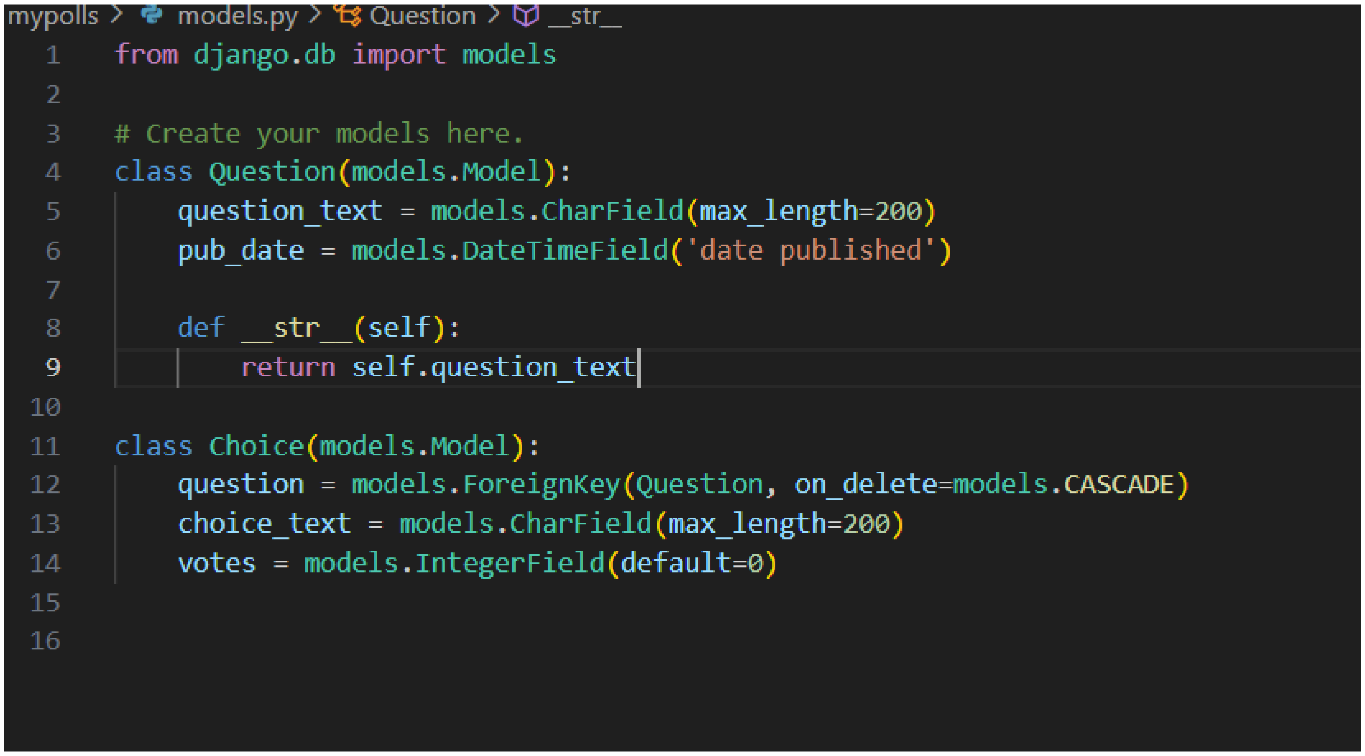
Select question to change

Action: --------- Go 0 of 1 selected

	QUESTION
☐	Question object (1)

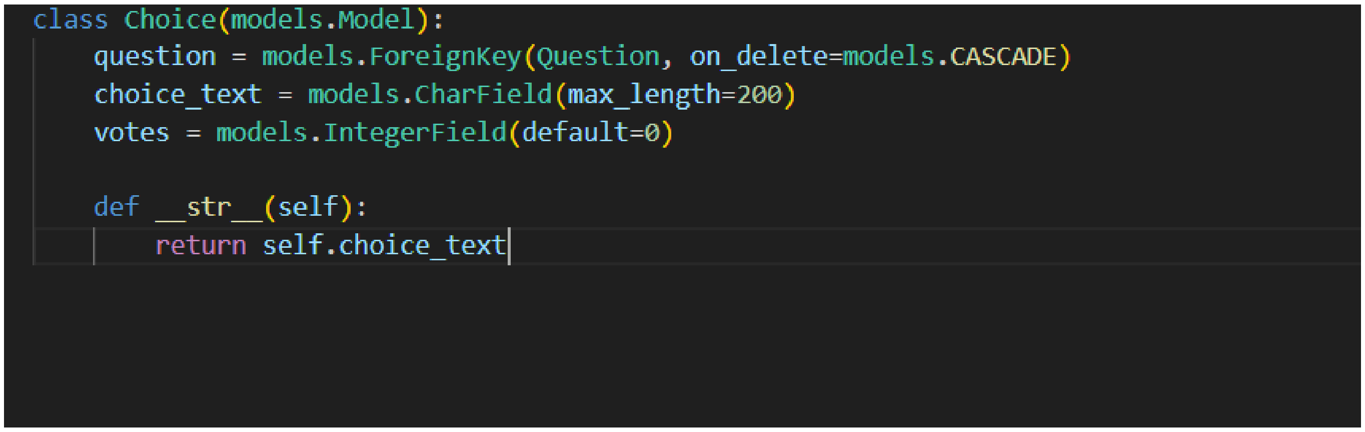
1 question

뭐가먼지모르겠으니 models.py수정

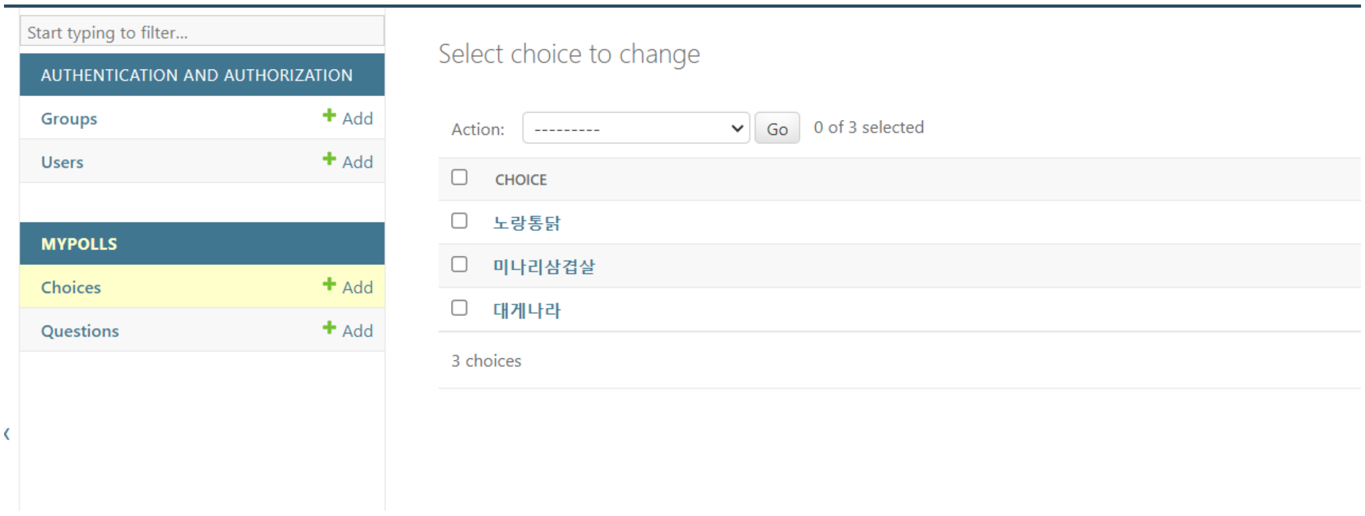


mypolls > models.py > Question > __str__

```python
from django.db import models

# Create your models here.
class Question(models.Model):
    question_text = models.CharField(max_length=200)
    pub_date = models.DateTimeField('date published')

    def __str__(self):
        return self.question_text

class Choice(models.Model):
    question = models.ForeignKey(Question, on_delete=models.CASCADE)
    choice_text = models.CharField(max_length=200)
    votes = models.IntegerField(default=0)


```



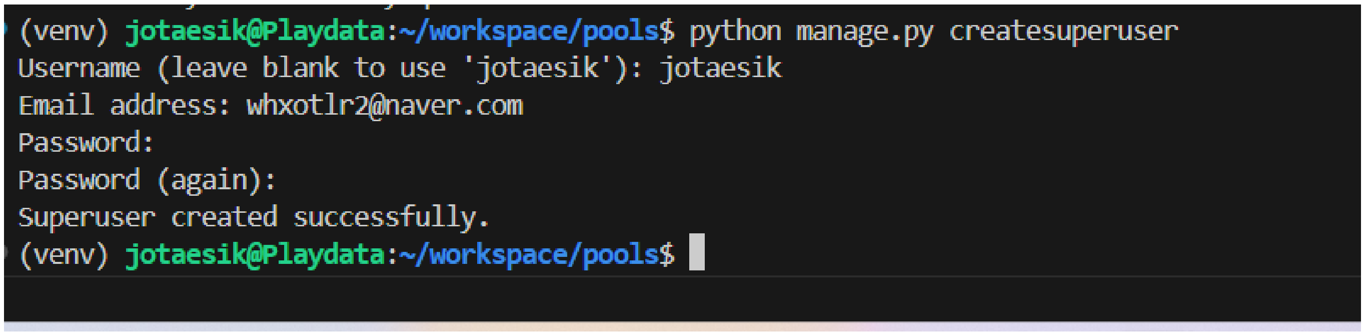
```python
class Choice(models.Model):
    question = models.ForeignKey(Question, on_delete=models.CASCADE)
    choice_text = models.CharField(max_length=200)
    votes = models.IntegerField(default=0)

    def __str__(self):
        return self.choice_text
```



pools->settings.py에 DATABASES에 default선언

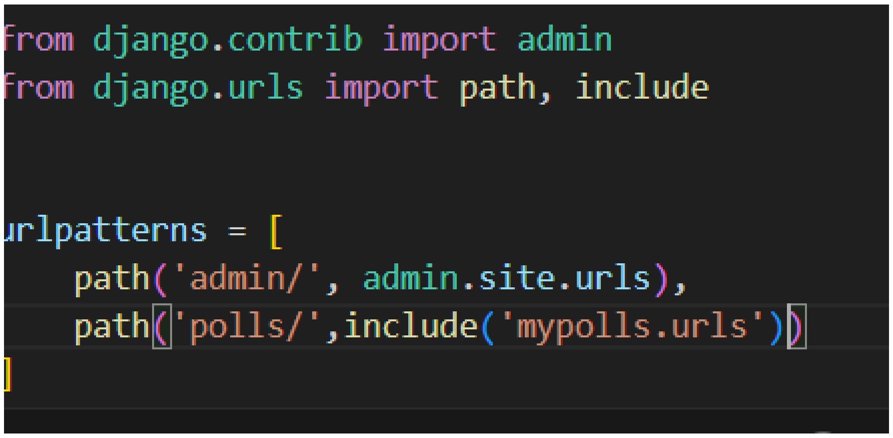
'default' : {
'ENGINE' : 'django.db.backends.mysql',
'NAME' : 'name',
'USER' : 'user',
'PASSWORD' : 'password',
'HOST' : "host",
'PORT' : port
}

pip install mysqlclient



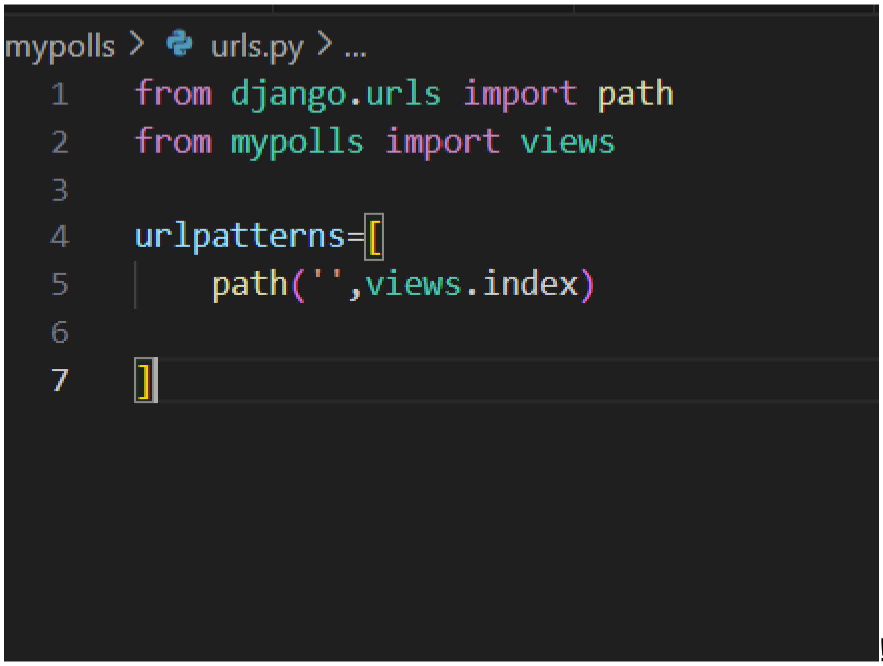
```
(venv) jotaesik@Playdata:~/workspace/pools$ python manage.py createsuperuser
Username (leave blank to use 'jotaesik'): jotaesik
Email address: whxotlr2@naver.com
Password:
Password (again):
Superuser created successfully.
(venv) jotaesik@Playdata:~/workspace/pools$
```

pools의 urls.py



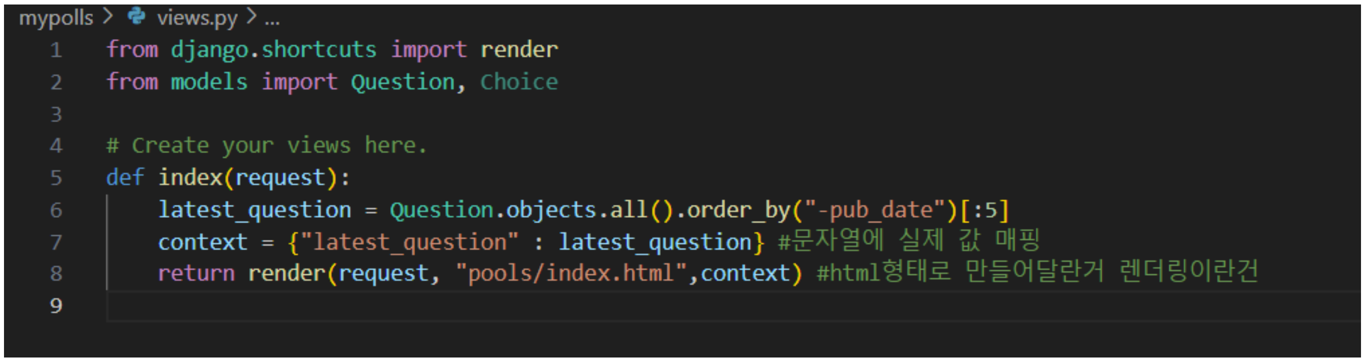
```
from django.contrib import admin
from django.urls import path, include


urlpatterns = [
    path('admin/', admin.site.urls),
    path('polls/',include('mypolls.urls'))
]
```

더 한번에 url을 넣을수있지만 그러면 유지보수가 안되므로 polls라는 url이 있으므로 거기서 관여하라는ㅁ말임
mypolls에서 urls.py파일생성
views는 로직이 적혀있는것이다.



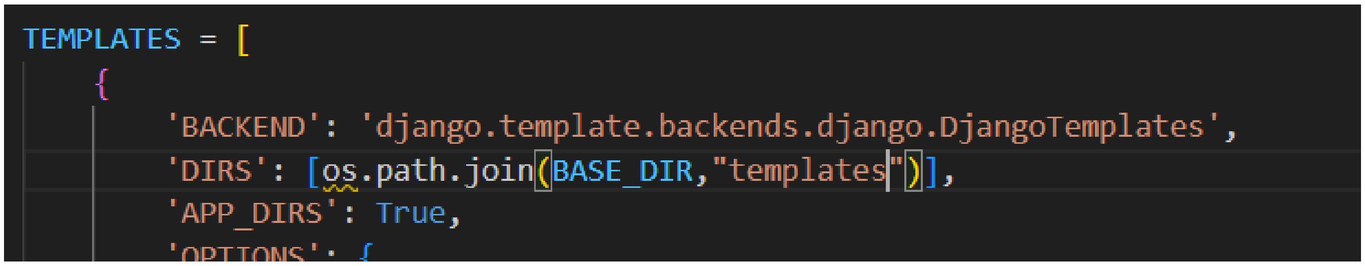
```
mypolls > urls.py > ...
from django.urls import path
from mypolls import views

urlpatterns=[
    path('',views.index)

]
```

![[
쿼리를 쓸필요가없다. views.py에



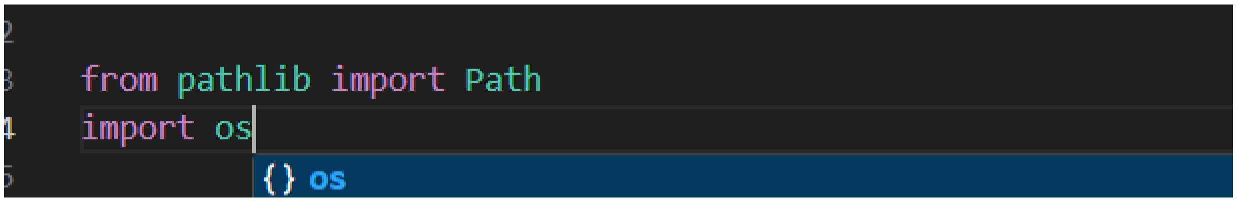
```
mypolls > views.py > ...
from django.shortcuts import render
from models import Question, Choice

# Create your views here.
def index(request):
    latest_question = Question.objects.all().order_by("-pub_date")[:5]
    context = {"latest_question" : latest_question} #문자열에 실제 값 매핑
    return render(request, "pools/index.html",context) #html형태로 만들어달란거 렌더링이란건
```

setting.py에

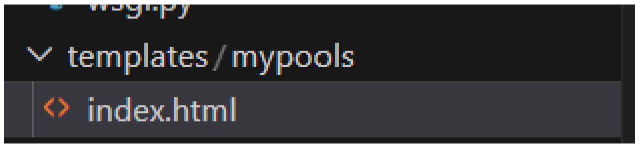


```
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [os.path.join(BASE_DIR,"templates")],
        'APP_DIRS': True,
        'OPTIONS': {
```

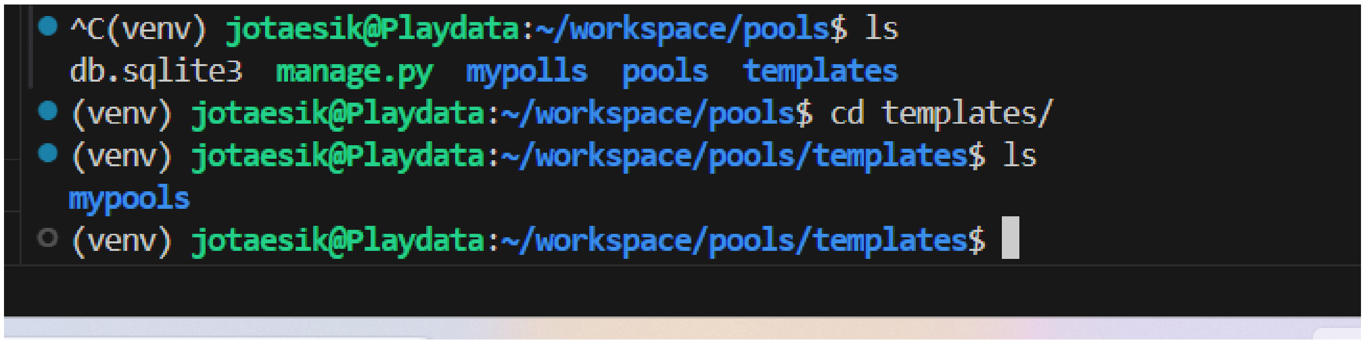
os에 오류나므로

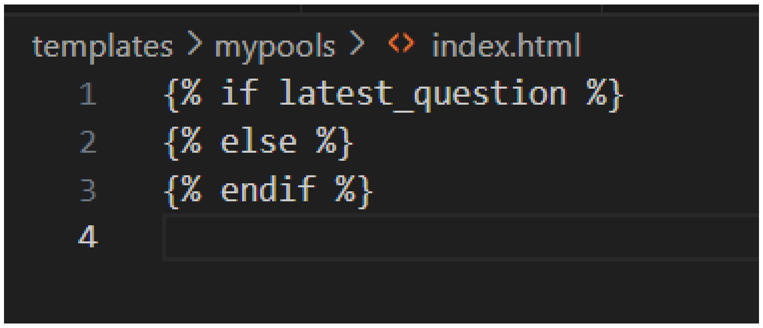


```
from pathlib import Path
import os
```

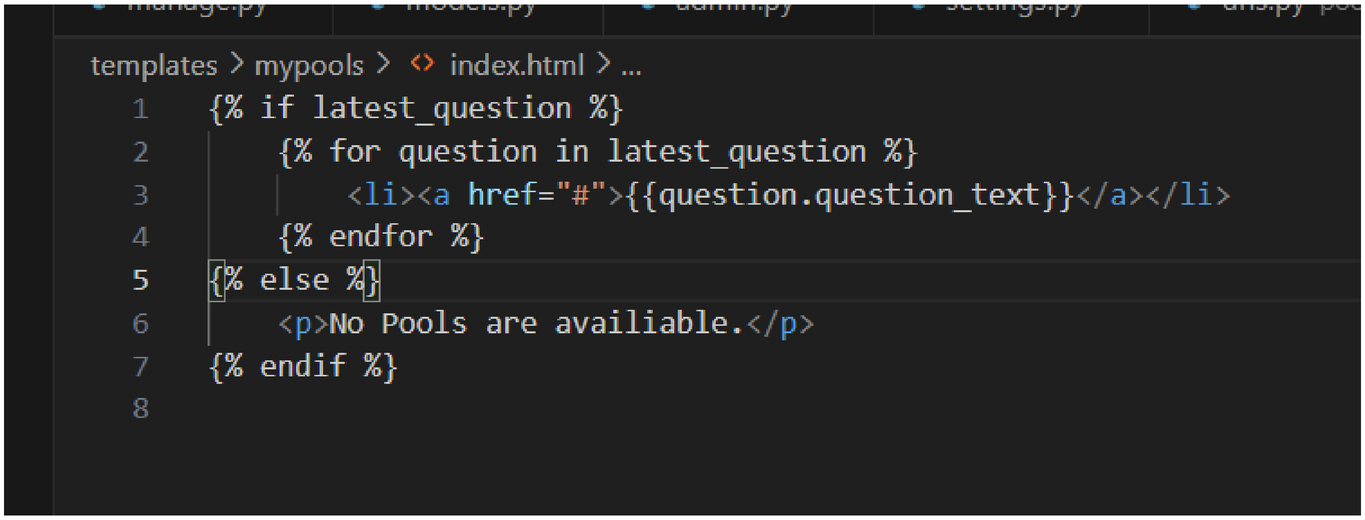

```
templates / mypools
  index.html
```

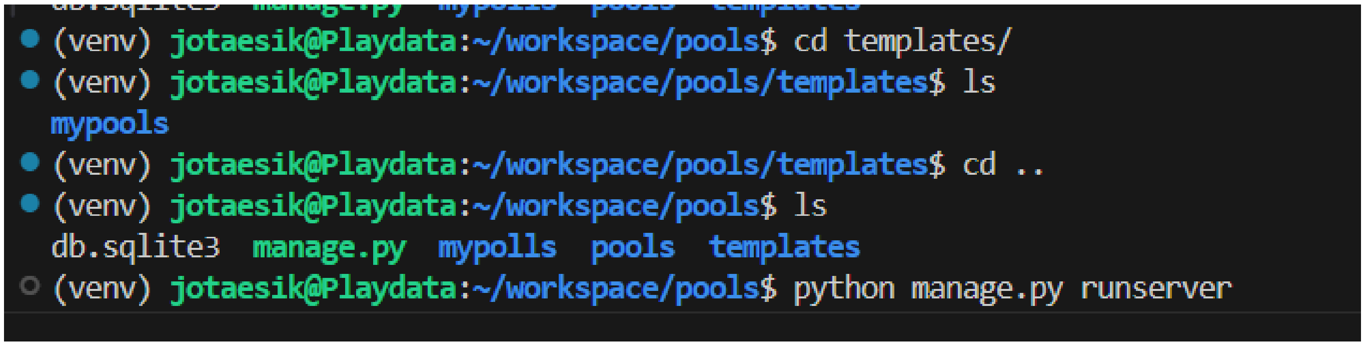
생성한다음


```
^C(venv) jotaesik@Playdata:~/workspace/pools$ ls
db.sqlite3  manage.py  mypolls  pools  templates
(venv) jotaesik@Playdata:~/workspace/pools$ cd templates/
(venv) jotaesik@Playdata:~/workspace/pools/templates$ ls
mypools
(venv) jotaesik@Playdata:~/workspace/pools/templates$
```

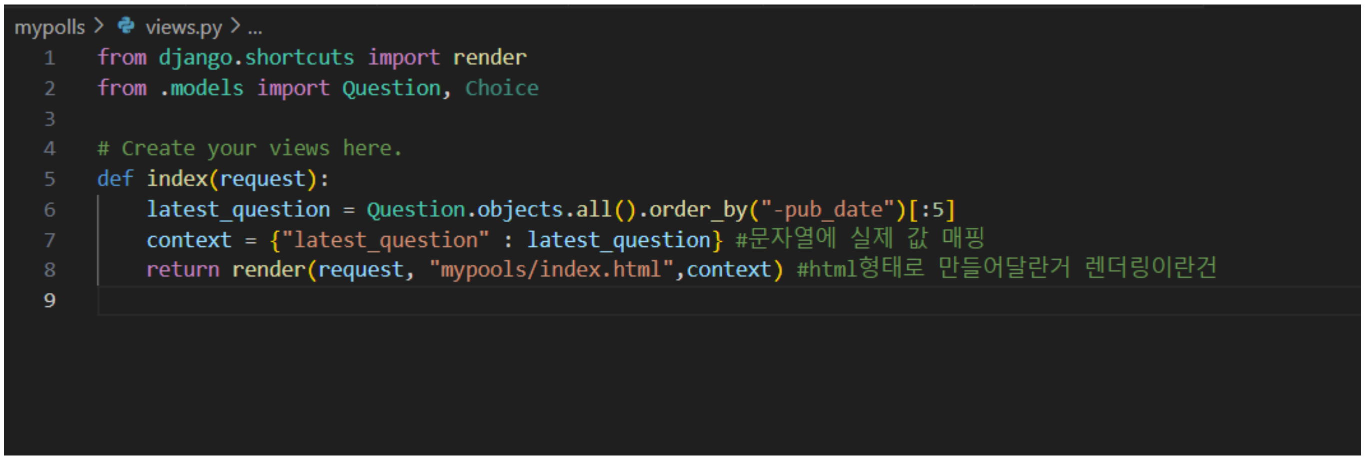

templates > mypools > index.html
```
{% if latest_question %}
{% else %}
{% endif %}

```

탬플릿


templates > mypools > index.html > ...
```
{% if latest_question %}
    {% for question in latest_question %}
        <li><a href="#">{{question.question_text}}</a></li>
    {% endfor %}
{% else %}
    <p>No Pools are availiable.</p>
{% endif %}

```

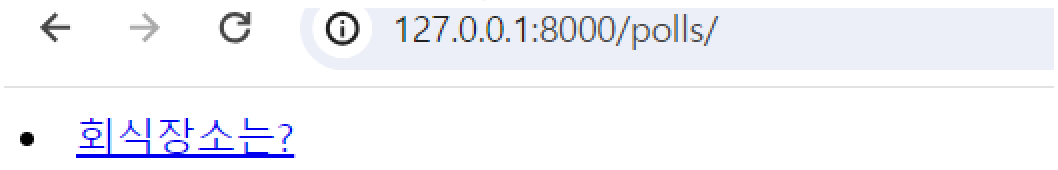

```
(venv) jotaesik@Playdata:~/workspace/pools$ cd templates/
(venv) jotaesik@Playdata:~/workspace/pools/templates$ ls
mypools
(venv) jotaesik@Playdata:~/workspace/pools/templates$ cd ..
(venv) jotaesik@Playdata:~/workspace/pools$ ls
db.sqlite3  manage.py  mypolls  pools  templates
(venv) jotaesik@Playdata:~/workspace/pools$ python manage.py runserver
```

manage.py가있어야한다



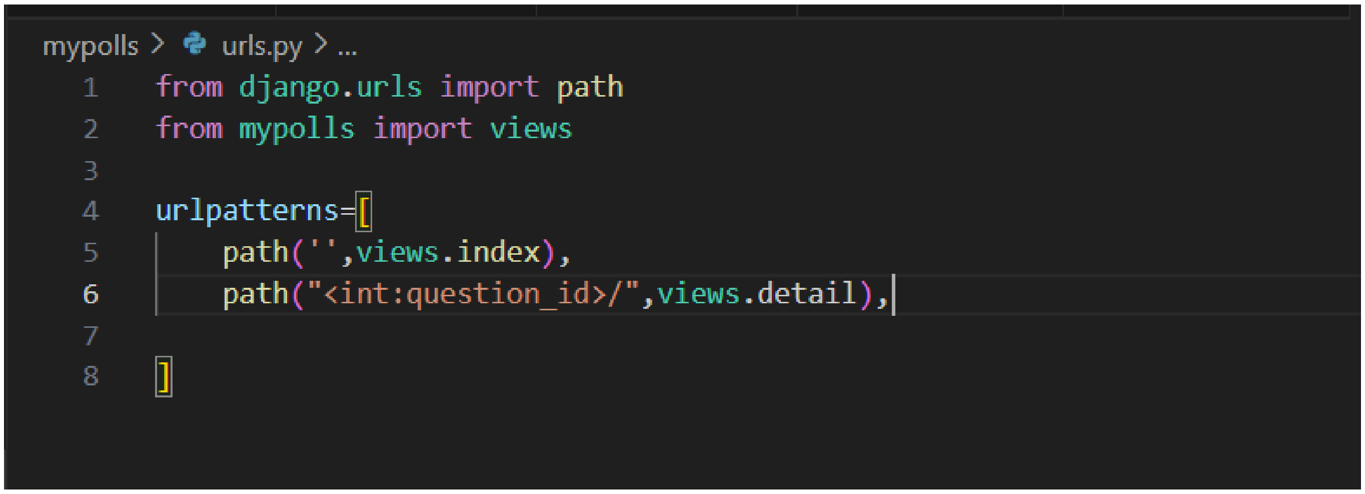
```
mypolls > views.py > ...
from django.shortcuts import render
from .models import Question, Choice

# Create your views here.
def index(request):
    latest_question = Question.objects.all().order_by("-pub_date")[:5]
    context = {"latest_question" : latest_question} #문자열에 실제 값 매핑
    return render(request, "mypools/index.html",context) #html형태로 만들어달란거 렌더링이란건

```

.models 현재폴더의 .models을 import



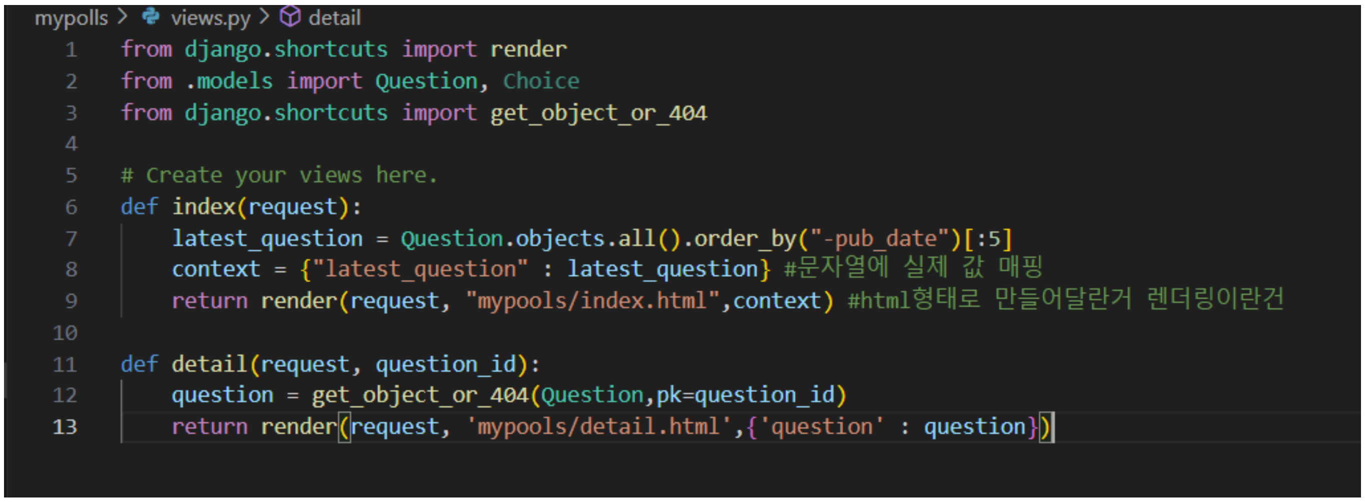
```
127.0.0.1:8000/polls/
```

- 회식장소는?



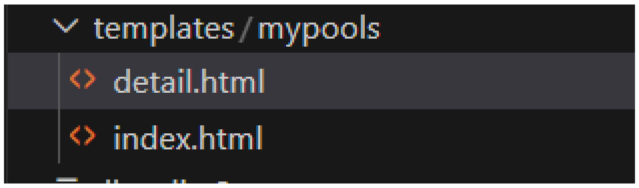
```
mypolls > urls.py > ...
from django.urls import path
from mypolls import views

urlpatterns=[
    path('',views.index),
    path("<int:question_id>/",views.detail),

]
```

views.detail이 없으므로 def detail 선언

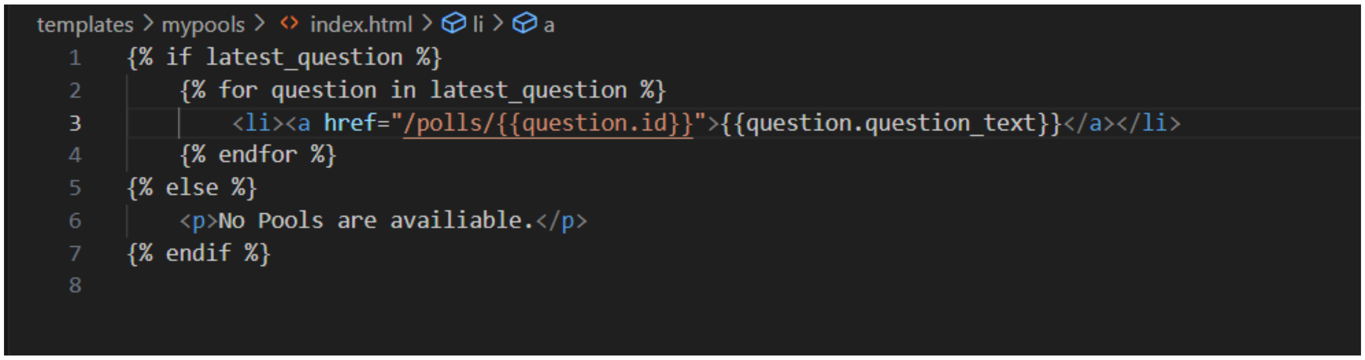


```
mypolls > views.py > detail
from django.shortcuts import render
from .models import Question, Choice
from django.shortcuts import get_object_or_404

# Create your views here.
def index(request):
    latest_question = Question.objects.all().order_by("-pub_date")[:5]
    context = {"latest_question" : latest_question} #문자열에 실제 값 매핑
    return render(request, "mypools/index.html",context) #html형태로 만들어달란거 렌더링이란건

def detail(request, question_id):
    question = get_object_or_404(Question,pk=question_id)
    return render(request, 'mypools/detail.html',{'question' : question})
```

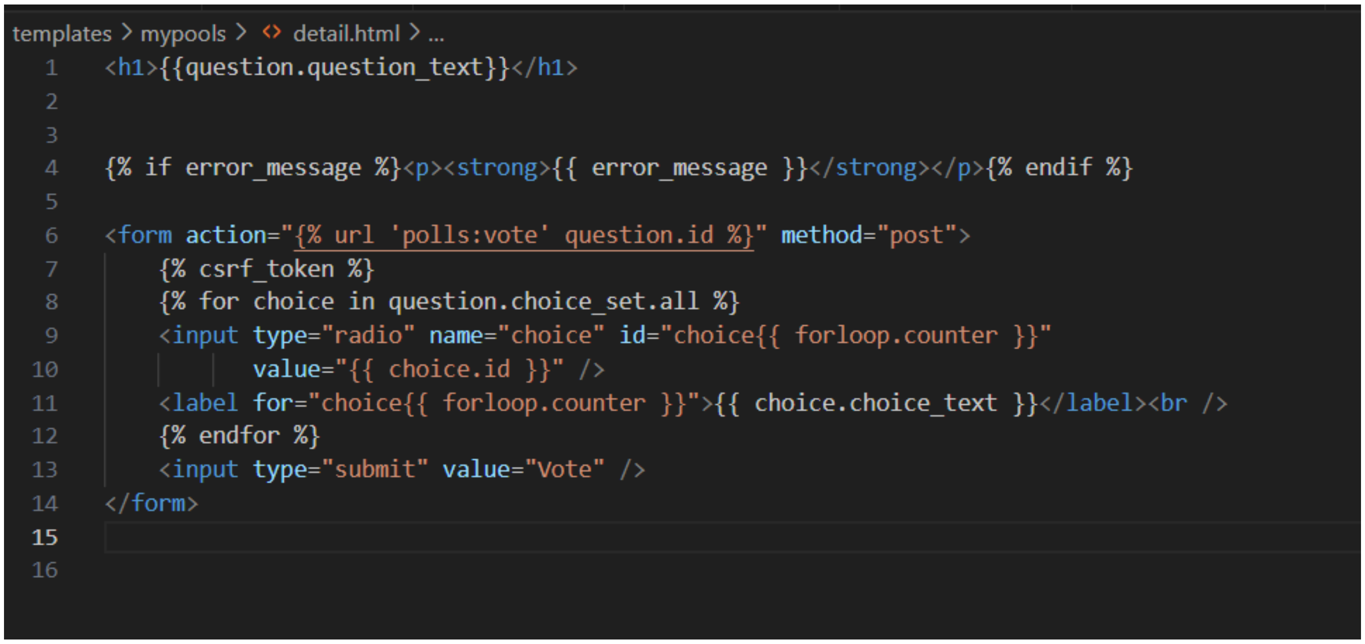
m은 데이터 t는 템플릿
조회했는데 없으면 404 있으면 data return


templates / mypools
detail.html
index.html

detail.html이 없으므로 생성

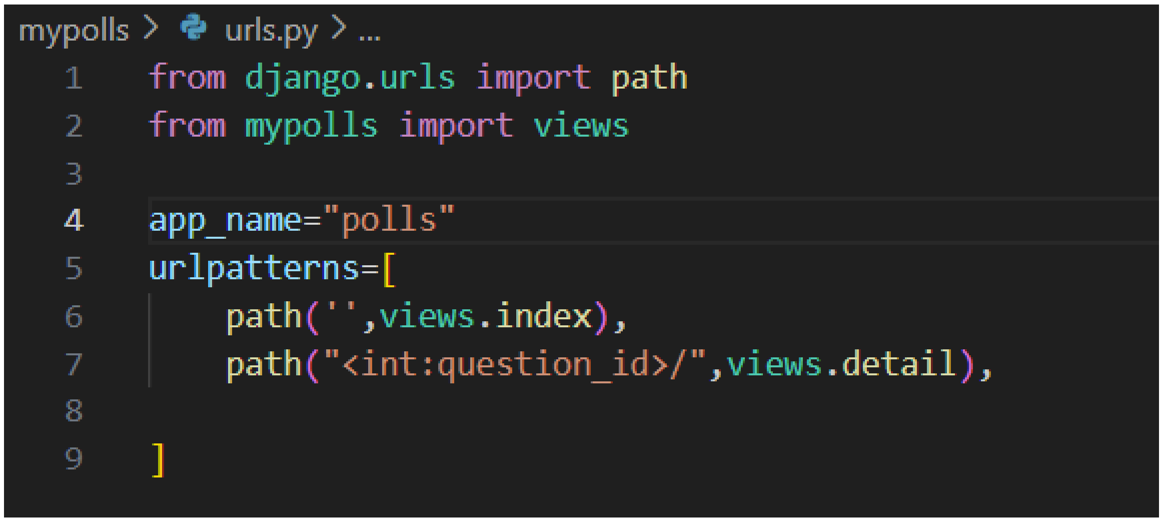

templates > mypools > index.html > li > a

```
{% if latest_question %}
    {% for question in latest_question %}
        <li><a href="/polls/{{question.id}}">{{question.question_text}}</a></li>
    {% endfor %}
{% else %}
    <p>No Pools are availiable.</p>
{% endif %}

```

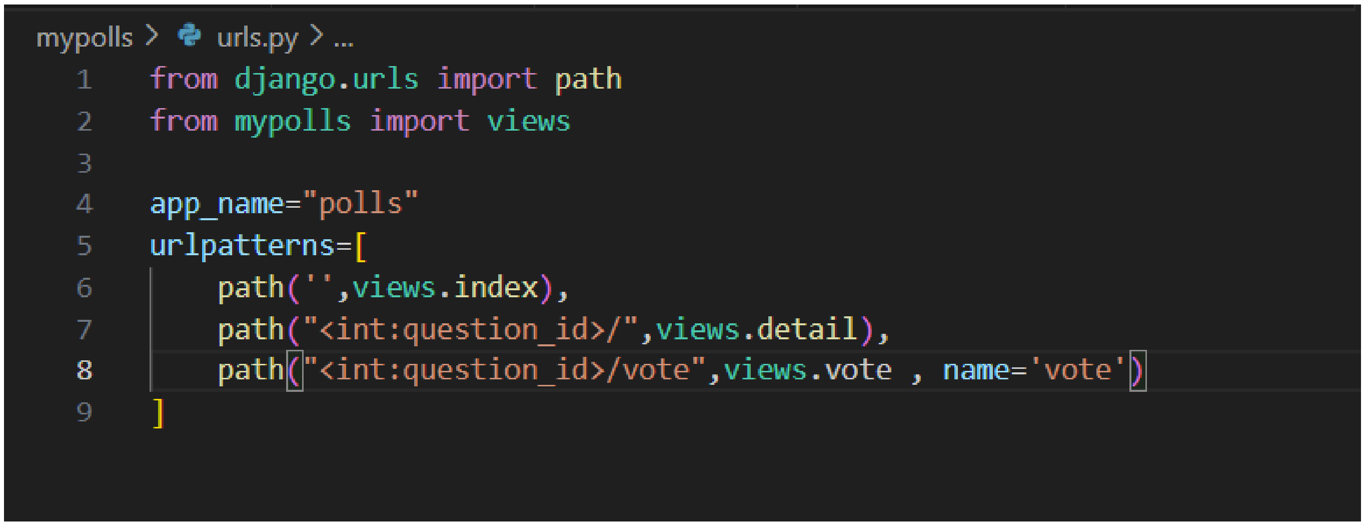

templates > mypools > detail.html > ...

```
<h1>{{question.question_text}}</h1>


{% if error_message %}<p><strong>{{ error_message }}</strong></p>{% endif %}

<form action="{% url 'polls:vote' question.id %}" method="post">
    {% csrf_token %}
    {% for choice in question.choice_set.all %}
    <input type="radio" name="choice" id="choice{{ forloop.counter }}"
            value="{{ choice.id }}" />
    <label for="choice{{ forloop.counter }}">{{ choice.choice_text }}</label><br />
    {% endfor %}
    <input type="submit" value="Vote" />
</form>

```

forloop.counter 전체건수보여준다 submit누르는순간 choice_id가 pools:vote로 포스트된다 vote를 누루는순간 url에 name이 vote인 애가 있을거야. 값은 question.id전달할게

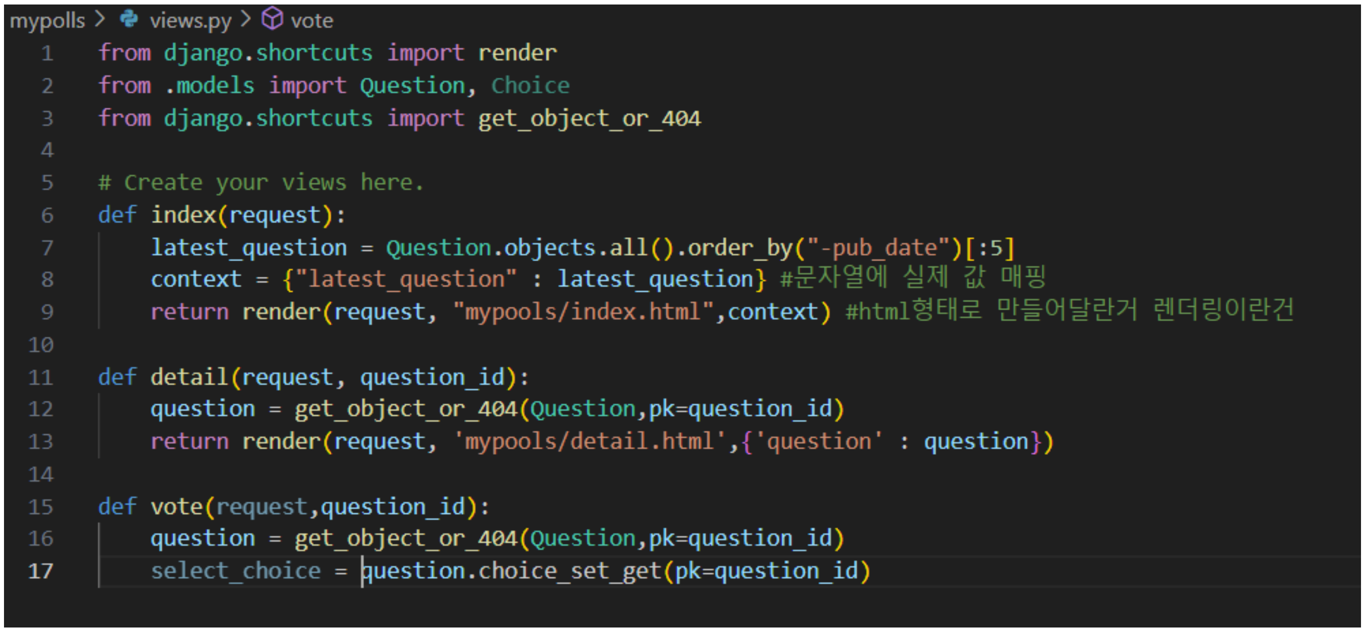

mypolls > urls.py > ...

```
from django.urls import path
from mypolls import views

app_name="polls"
urlpatterns=[
    path('',views.index),
    path("<int:question_id>/",views.detail),

]
```


mypolls > urls.py > ...

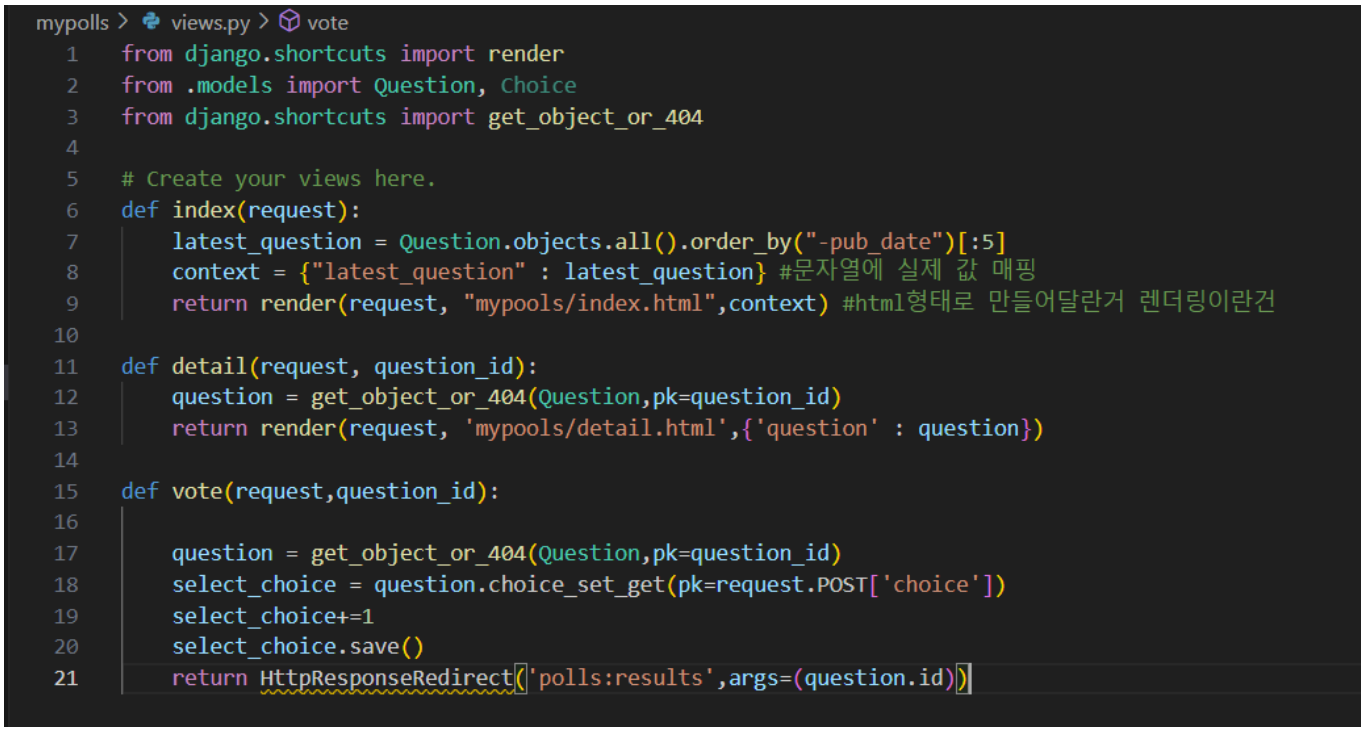
```python
from django.urls import path
from mypolls import views

app_name="polls"
urlpatterns=[
    path('',views.index),
    path("<int:question_id>/",views.detail),
    path("<int:question_id>/vote",views.vote , name='vote')
]
```

이제는 views.vote를 만들ㄹ어야지
이미 누군가가 먼저 투표를 했다면 그 value를 가져와야한


mypolls > views.py > vote

```python
from django.shortcuts import render
from .models import Question, Choice
from django.shortcuts import get_object_or_404

# Create your views here.
def index(request):
    latest_question = Question.objects.all().order_by("-pub_date")[:5]
    context = {"latest_question" : latest_question} #문자열에 실제 값 매핑
    return render(request, "mypools/index.html",context) #html형태로 만들어달란거 렌더링이란건

def detail(request, question_id):
    question = get_object_or_404(Question,pk=question_id)
    return render(request, 'mypools/detail.html',{'question' : question})

def vote(request,question_id):
    question = get_object_or_404(Question,pk=question_id)
    select_choice = question.choice_set_get(pk=question_id)
```

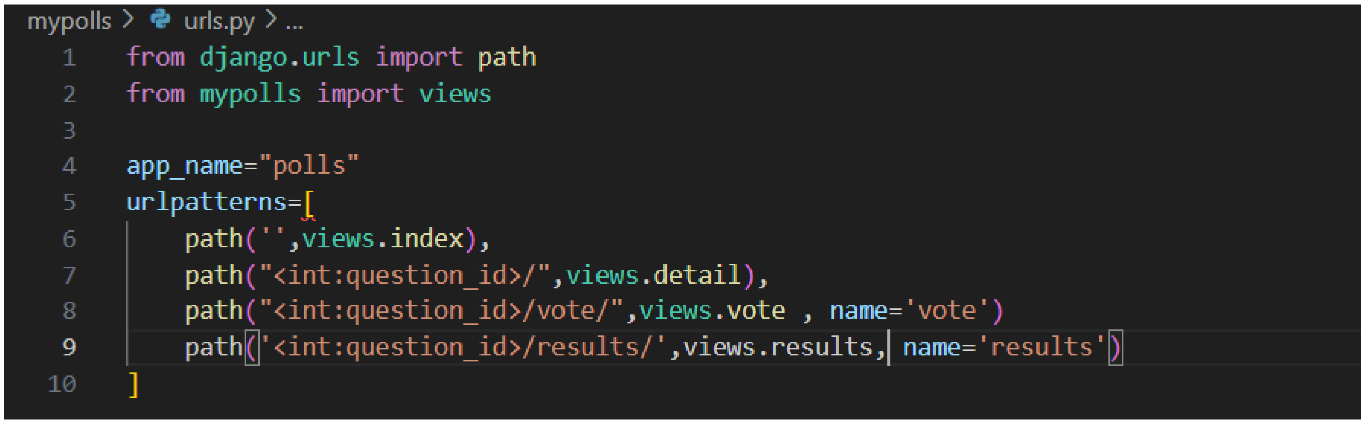
post방식으로 값을 가져오는데 왜래키의 값들을 모두가져와서 체크를해야지
choice란 이름으로 값을 전달받을게



```
mypolls > views.py > vote
from django.shortcuts import render
from .models import Question, Choice
from django.shortcuts import get_object_or_404

# Create your views here.
def index(request):
    latest_question = Question.objects.all().order_by("-pub_date")[:5]
    context = {"latest_question" : latest_question} #문자열에 실제 값 매핑
    return render(request, "mypools/index.html",context) #html형태로 만들어달란거 렌더링이란건

def detail(request, question_id):
    question = get_object_or_404(Question,pk=question_id)
    return render(request, 'mypools/detail.html',{'question' : question})

def vote(request,question_id):

    question = get_object_or_404(Question,pk=question_id)
    select_choice = question.choice_set_get(pk=request.POST['choice'])
    select_choice+=1
    select_choice.save()
    return HttpResponseRedirect('polls:results',args=(question.id))
```

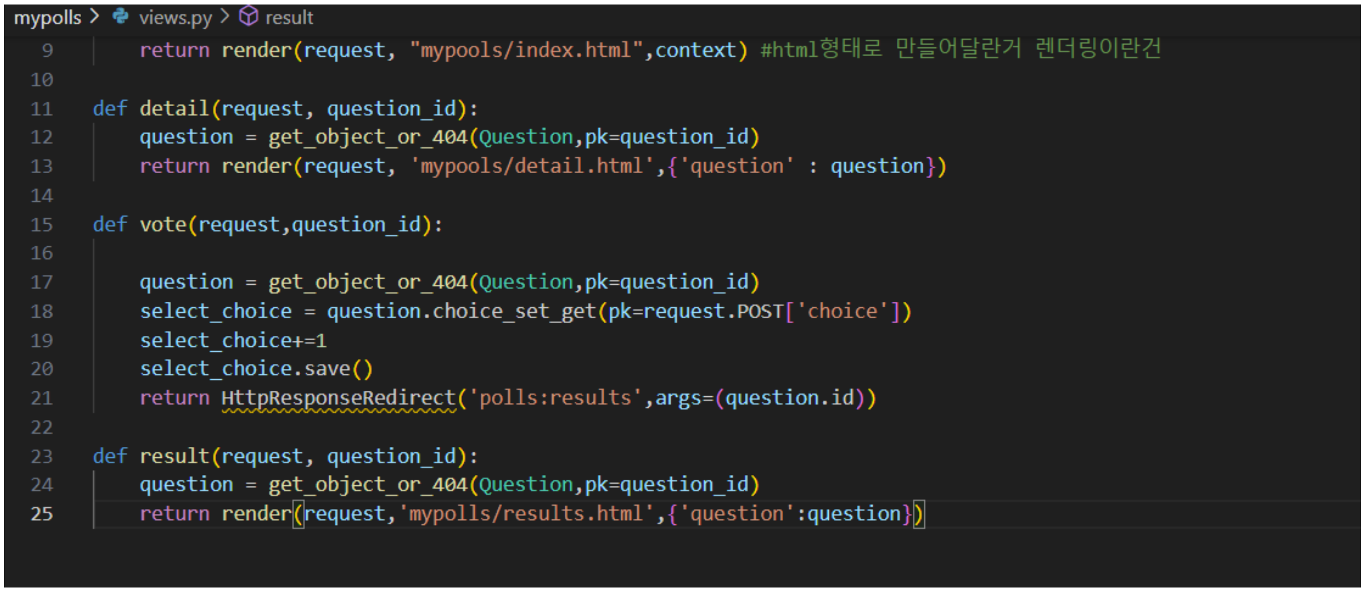
값을 불러오는데 없으면 404뜨고 있으면 화면에 choice값 리스트를 들고와야지, 사용자가 누른 값만, post방식으로 question_choice_set 식당들마다 다 key값이 있기이ㅔ key값이 들어온다. 노랑통닭 데이터가 들어오고1증가한다음 save 그리고 이 결과를 polls:results로 뿌린다.

template가서 result.html만들기

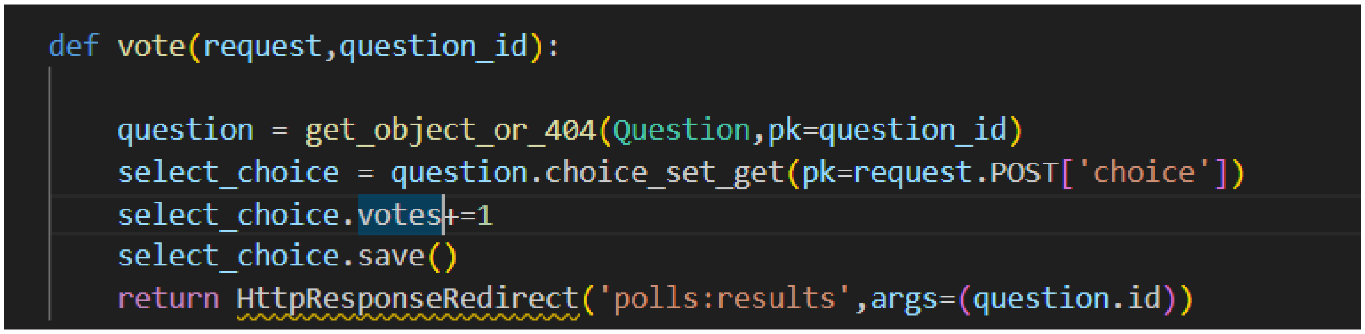


```
mypolls > urls.py > ...
from django.urls import path
from mypolls import views

app_name="polls"
urlpatterns=[
    path('',views.index),
    path("<int:question_id>/",views.detail),
    path("<int:question_id>/vote/",views.vote , name='vote')
    path('<int:question_id>/results/',views.results, name='results')
]
```

그 전에 url에 선언하기



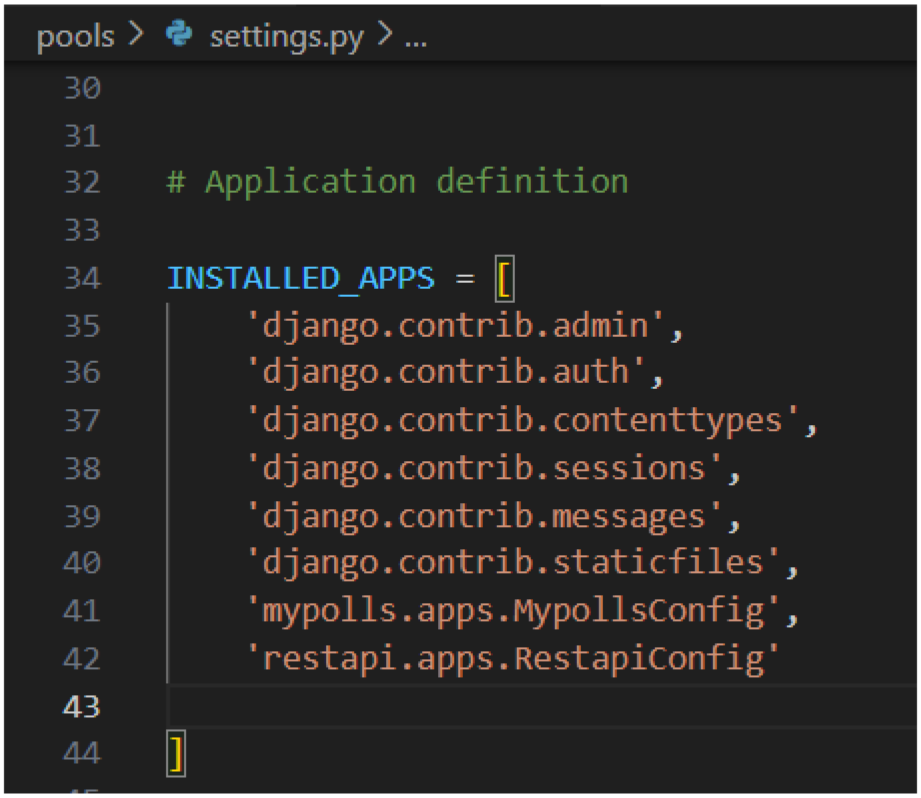
```python
    return render(request, "mypools/index.html",context) #html형태로 만들어달란거 렌더링이란건

def detail(request, question_id):
    question = get_object_or_404(Question,pk=question_id)
    return render(request, 'mypools/detail.html',{'question' : question})

def vote(request,question_id):

    question = get_object_or_404(Question,pk=question_id)
    select_choice = question.choice_set_get(pk=request.POST['choice'])
    select_choice+=1
    select_choice.save()
    return HttpResponseRedirect('polls:results',args=(question.id))

def result(request, question_id):
    question = get_object_or_404(Question,pk=question_id)
    return render(request,'mypolls/results.html',{'question':question})
```



```python
def vote(request,question_id):

    question = get_object_or_404(Question,pk=question_id)
    select_choice = question.choice_set_get(pk=request.POST['choice'])
    select_choice.votes+=1
    select_choice.save()
    return HttpResponseRedirect('polls:results',args=(question.id))
```

vote칼럼을 필요로하므로

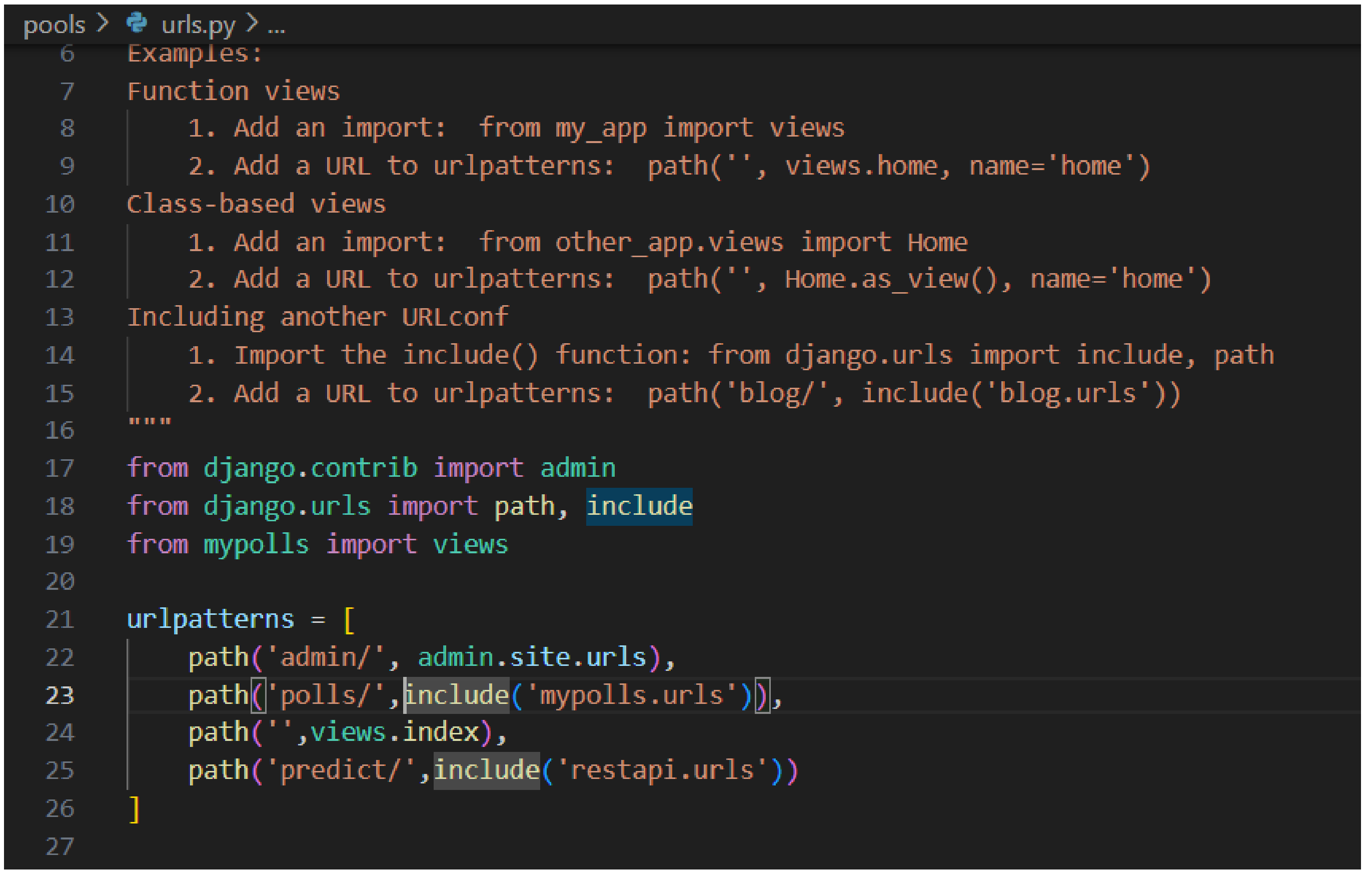
장고는 모듈이 다 정해져있지만 다 넣어줘야하고 느리고 flask는 일일이 다 넣어줘야하는대신 빨드ㅏ



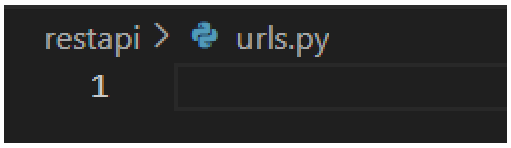
```python

# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'mypolls.apps.MypollsConfig',
    'restapi.apps.RestapiConfig'

]
```

추가하기 restapi

```
jotaesik@Playdata:~/workspace/pools$ cd ..
jotaesik@Playdata:~/workspace$ source ./venv/bin/activate
(venv) jotaesik@Playdata:~/workspace$ cd pools
(venv) jotaesik@Playdata:~/workspace/pools$ python manage.py startapp restapi
(venv) jotaesik@Playdata:~/workspace/pools$ pip install djangorestframework
```

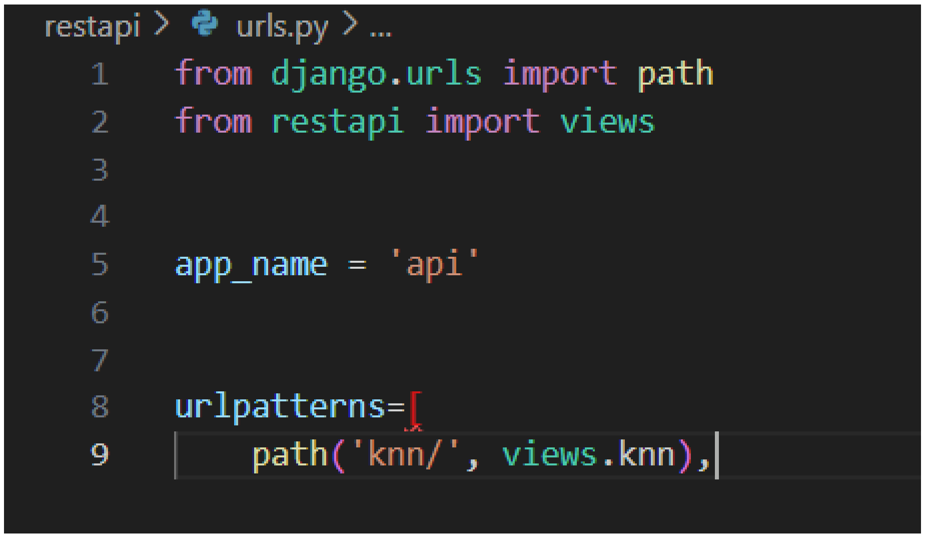


```
pools > urls.py > ...
Examples:
Function views
    1. Add an import:  from my_app import views
    2. Add a URL to urlpatterns:  path('', views.home, name='home')
Class-based views
    1. Add an import:  from other_app.views import Home
    2. Add a URL to urlpatterns:  path('', Home.as_view(), name='home')
Including another URLconf
    1. Import the include() function: from django.urls import include, path
    2. Add a URL to urlpatterns:  path('blog/', include('blog.urls'))
"""
from django.contrib import admin
from django.urls import path, include
from mypolls import views

urlpatterns = [
    path('admin/', admin.site.urls),
    path('polls/',include('mypolls.urls')),
    path('',views.index),
    path('predict/',include('restapi.urls'))
]
```



```
restapi > urls.py
```

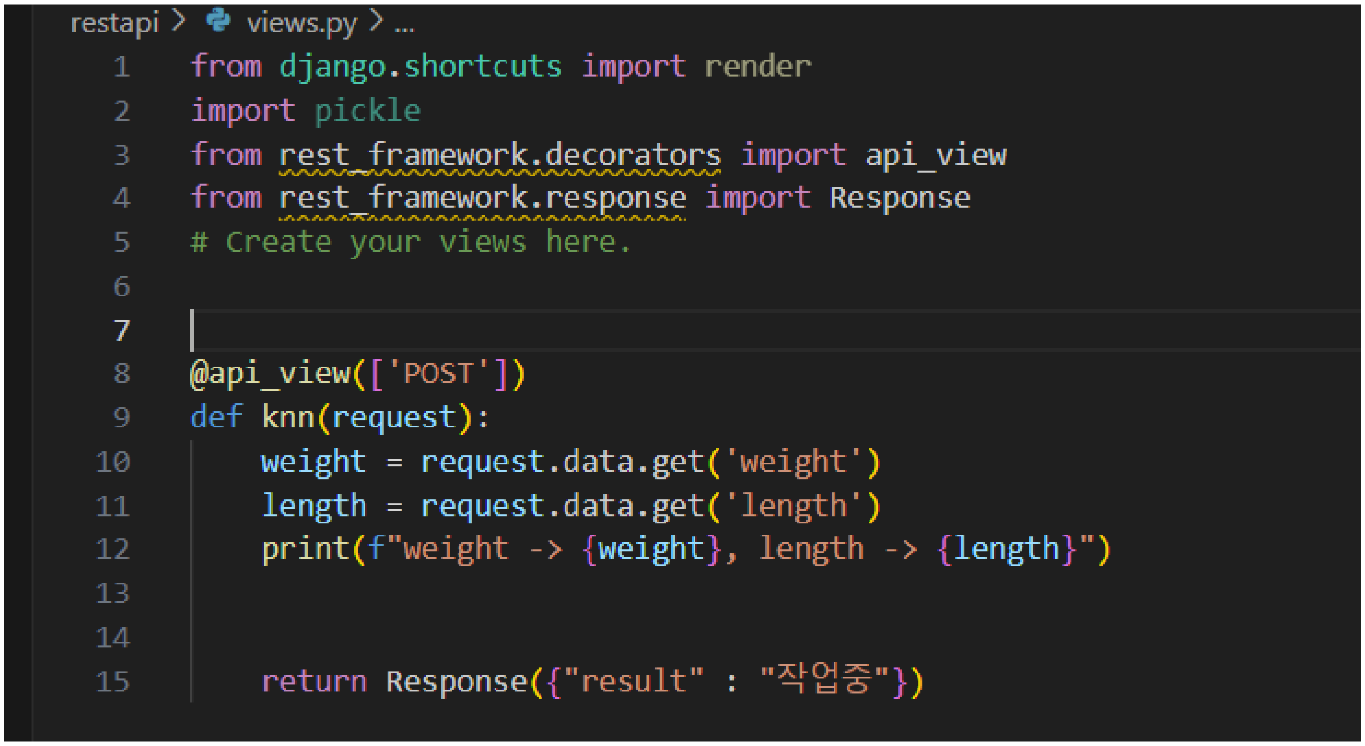
생성하기 urls.py



```
restapi > urls.py > ...
from django.urls import path
from restapi import views


app_name = 'api'


urlpatterns=[
    path('knn/', views.knn),
```

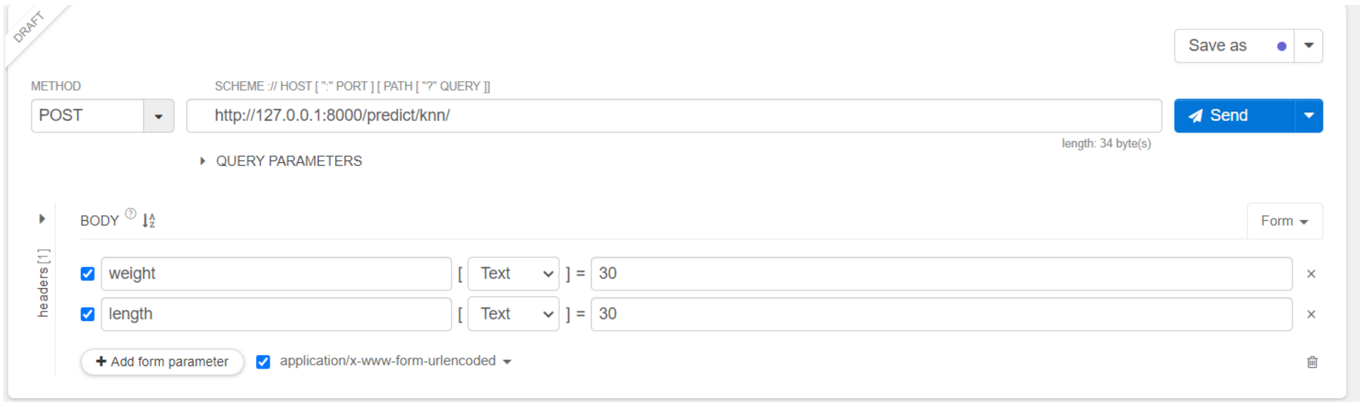
프로그램끼리 통신하는건 api
restfulapi=>restapi라고 부른다

Restful API 만들기
사용되는 웹request를 포스트방식으로 만들어준다.
'weight' : 30,
'length' : 150 이렇게 전달하면 돔인지 빙어인지 return 해준다



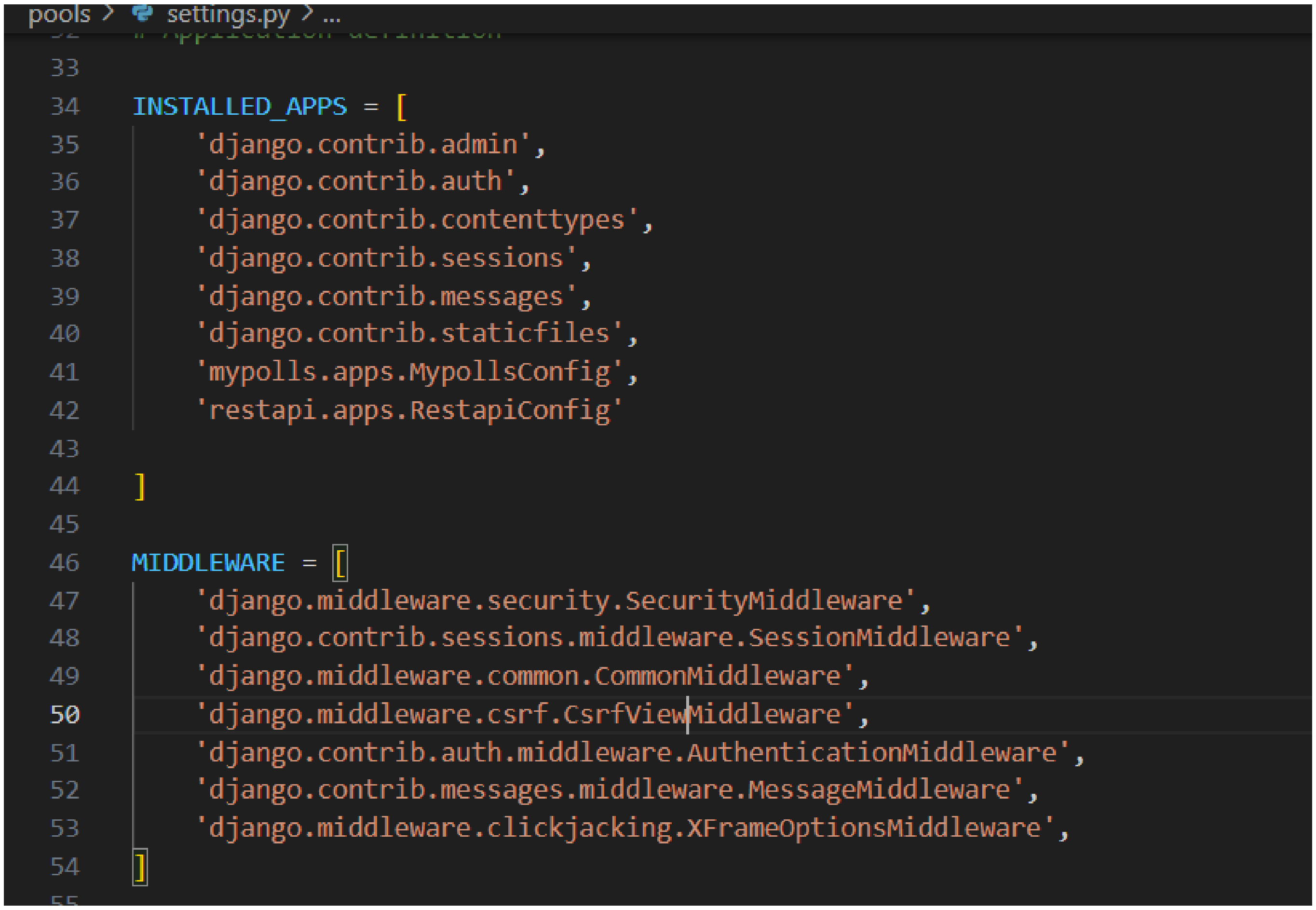
```
restapi > views.py > ...
from django.shortcuts import render
import pickle
from rest_framework.decorators import api_view
from rest_framework.response import Response
# Create your views here.


@api_view(['POST'])
def knn(request):
    weight = request.data.get('weight')
    length = request.data.get('length')
    print(f"weight -> {weight}, length -> {length}")


    return Response({"result" : "작업중"})
```

https://chromewebstore.google.com/detail/talend-api-tester-free-ed/aejoelaoggembcahagimdiliamlcdmfm
다운받기 이거랑 postman



```
METHOD  POST
SCHEME :// HOST [ ":" PORT ] [ PATH [ "?" QUERY ]]
http://127.0.0.1:8000/predict/knn/
length: 34 byte(s)
BODY  Form
☑ weight [ Text ] = 30
☑ length [ Text ] = 30
☑ application/x-www-form-urlencoded
```

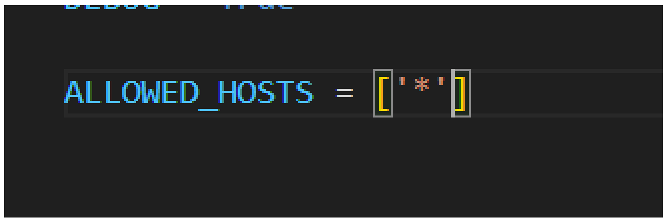
주소넣고 (venv) jotaesik@Playdata:~/workspace/pools$ python manage.py runserver 주소붙여넣구 HEADER는 지우고 BODY를 FORM으로 해서 SEND해보기
403에러뜬다



pools > settings.py > ...

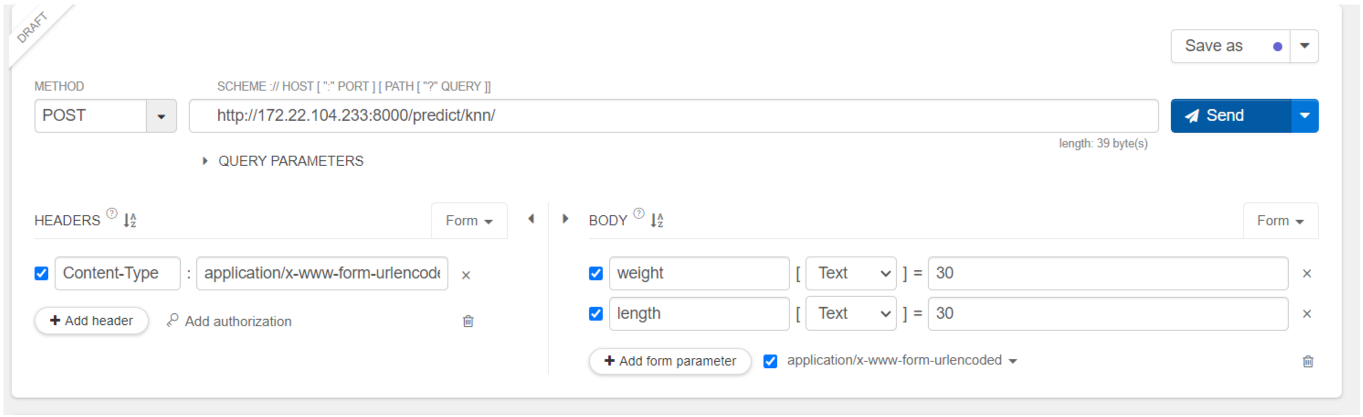
```python
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'mypolls.apps.MypollsConfig',
    'restapi.apps.RestapiConfig'

]

MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
]
```

CSRF의 에러가 일어났다.



```python
ALLOWED_HOSTS = ['*']
```

POOLS SETTINGS.PY에



METHOD: POST

SCHEME :// HOST [":" PORT] [PATH ["?" QUERY]]: http://172.22.104.233:8000/predict/knn/

length: 39 byte(s)

HEADERS — Form

✓	Content-Type	application/x-www-form-urlencode

BODY — Form

✓	weight	Text	30
✓	length	Text	30

✓ application/x-www-form-urlencoded

```
jotaesik@Playdata:~/workspace/pools$ ip a
jotaesik@Playdata:~/workspace/pools$ python manage.py runserver ip:8000
```

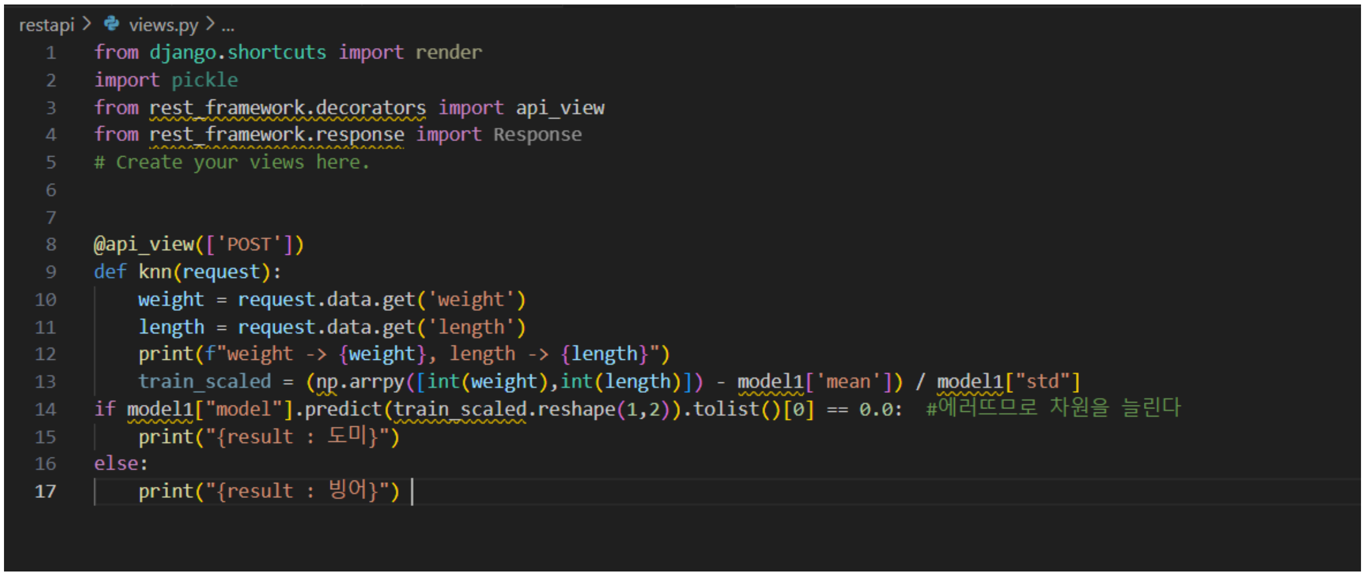
이제 pickle만 넣으면 끝난다

데이터엔지니어링-31기 > 05_web > serving

파일2개있다
2024.04.26_2 파일참

모델서비스 api 서비스
사이킷런에 있던 객체를 피클로 저장한다, 평균ㄴ과 분산을 넣어서 그래야 새로운 애가 정규화로 변환을 할수있따. 데이터를 표준화로 바꾸고

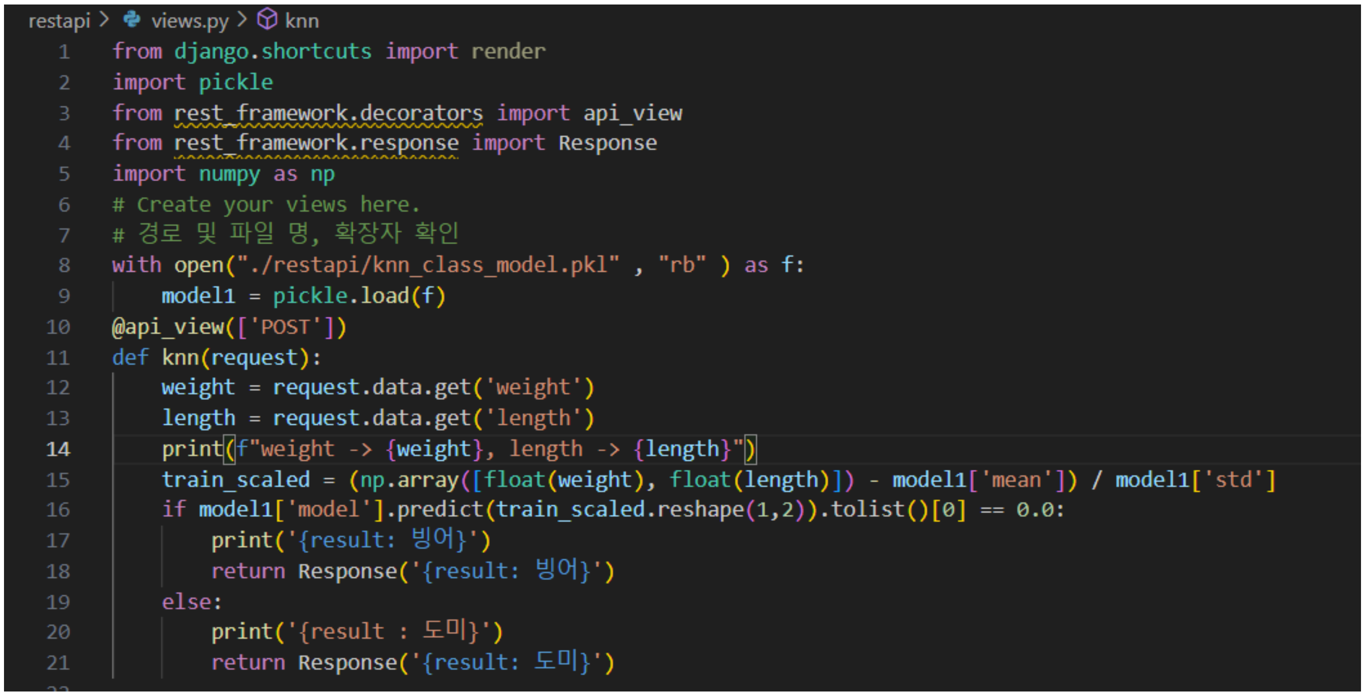
restapi폴더에 pickle 파일2개 옮기기



restapi > views.py > ...

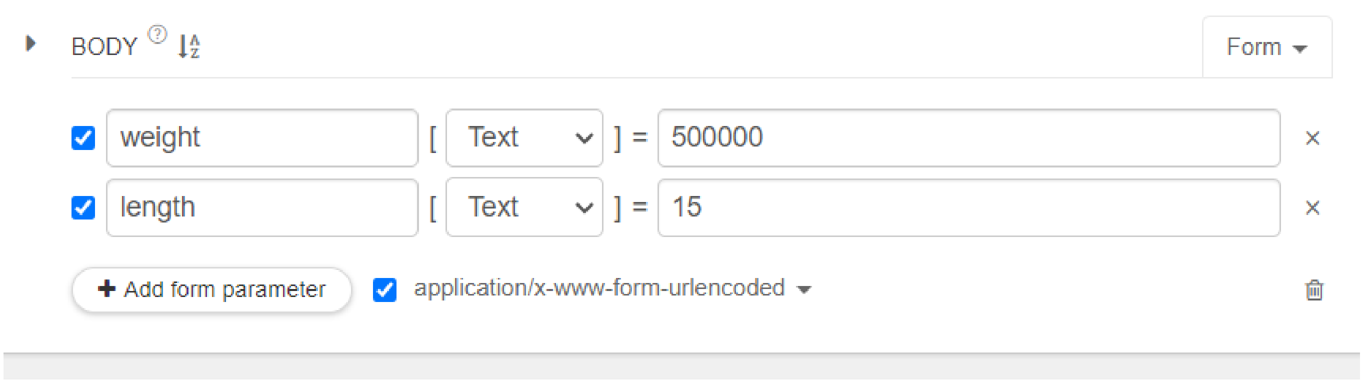
```python
from django.shortcuts import render
import pickle
from rest_framework.decorators import api_view
from rest_framework.response import Response
# Create your views here.


@api_view(['POST'])
def knn(request):
    weight = request.data.get('weight')
    length = request.data.get('length')
    print(f"weight -> {weight}, length -> {length}")
    train_scaled = (np.arrpy([int(weight),int(length)]) - model1['mean']) / model1["std"]
if model1["model"].predict(train_scaled.reshape(1,2)).tolist()[0] == 0.0:  #에러뜨므로 차원을 늘린다
    print("{result : 도미}")
else:
    print("{result : 빙어}")
```

(venv) jotaesik@Playdata:~/workspace/pools$ pip install numpy 설치
(venv) jotaesik@Playdata:~/workspace/pools$ pip install scikit-learn
(venv) jotaesik@Playdata:~/workspace/pools$ python manage.py runserver ip:8000



restapi > views.py > knn

```python
from django.shortcuts import render
import pickle
from rest_framework.decorators import api_view
from rest_framework.response import Response
import numpy as np
# Create your views here.
# 경로 및 파일 명, 확장자 확인
with open("./restapi/knn_class_model.pkl" , "rb" ) as f:
    model1 = pickle.load(f)
@api_view(['POST'])
def knn(request):
    weight = request.data.get('weight')
    length = request.data.get('length')
    print(f"weight -> {weight}, length -> {length}")
    train_scaled = (np.array([float(weight), float(length)]) - model1['mean']) / model1['std']
    if model1['model'].predict(train_scaled.reshape(1,2)).tolist()[0] == 0.0:
        print('{result: 빙어}')
        return Response('{result: 빙어}')
    else:
        print('{result : 도미}')
        return Response('{result: 도미}')
```



BODY

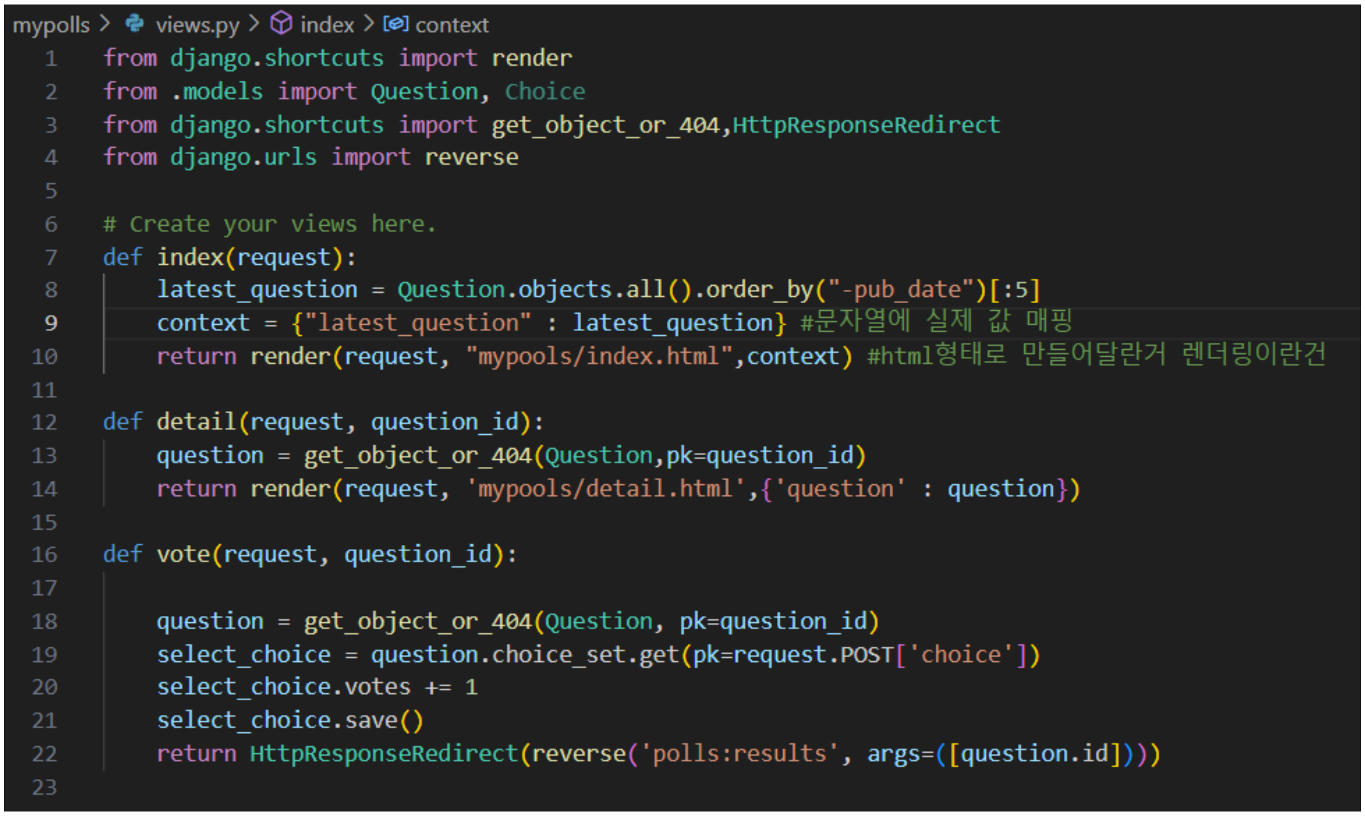
Form

✓	weight	Text	500000
✓	length	Text	15

Add form parameter

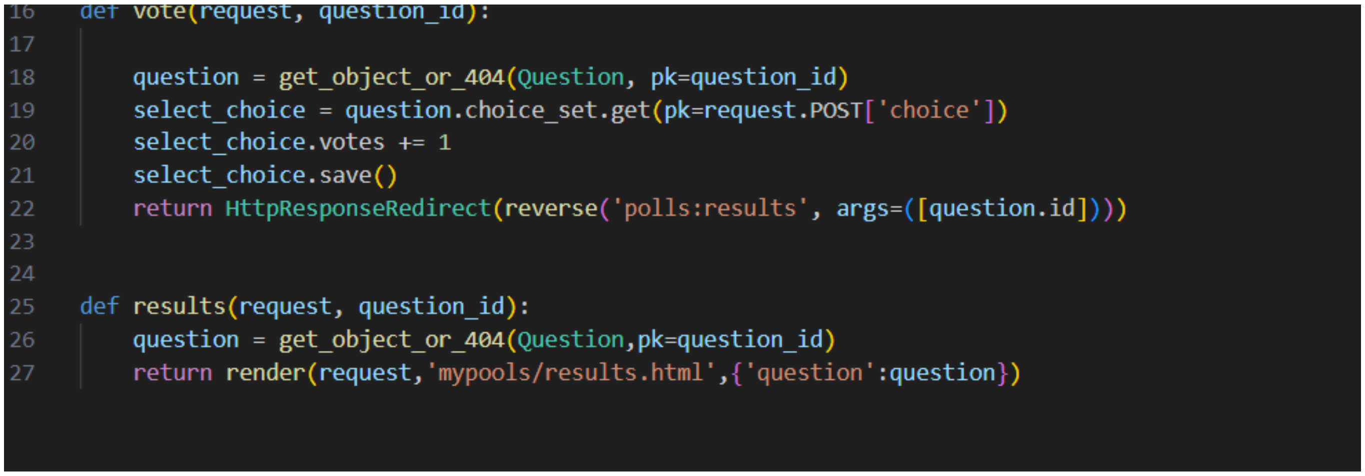
✓ application/x-www-form-urlencoded

mypools의 views.py



mypolls > views.py > index > context

```python
from django.shortcuts import render
from .models import Question, Choice
from django.shortcuts import get_object_or_404,HttpResponseRedirect
from django.urls import reverse

# Create your views here.
def index(request):
    latest_question = Question.objects.all().order_by("-pub_date")[:5]
    context = {"latest_question" : latest_question} #문자열에 실제 값 매핑
    return render(request, "mypools/index.html",context) #html형태로 만들어달란거 렌더링이란건

def detail(request, question_id):
    question = get_object_or_404(Question,pk=question_id)
    return render(request, 'mypools/detail.html',{'question' : question})

def vote(request, question_id):

    question = get_object_or_404(Question, pk=question_id)
    select_choice = question.choice_set.get(pk=request.POST['choice'])
    select_choice.votes += 1
    select_choice.save()
    return HttpResponseRedirect(reverse('polls:results', args=([question.id])))

```



```python
def vote(request, question_id):

    question = get_object_or_404(Question, pk=question_id)
    select_choice = question.choice_set.get(pk=request.POST['choice'])
    select_choice.votes += 1
    select_choice.save()
    return HttpResponseRedirect(reverse('polls:results', args=([question.id])))


def results(request, question_id):
    question = get_object_or_404(Question,pk=question_id)
    return render(request,'mypools/results.html',{'question':question})
```