API CODE

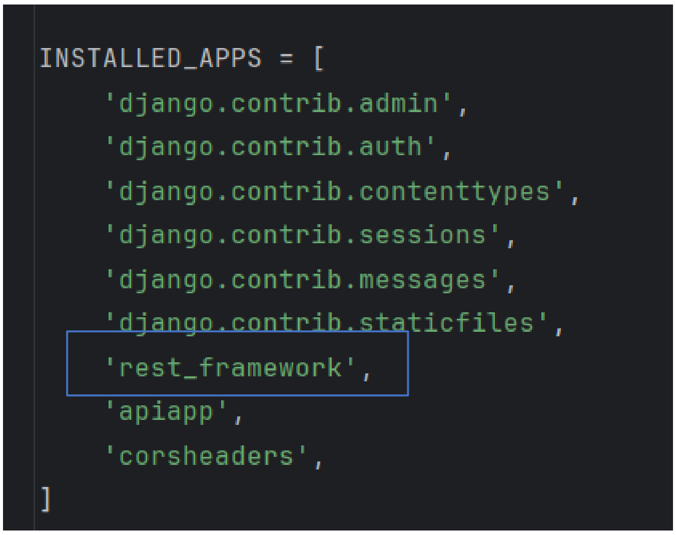
1 pip install djangorestframework

2 django-admin startproject projectname

3 cd project

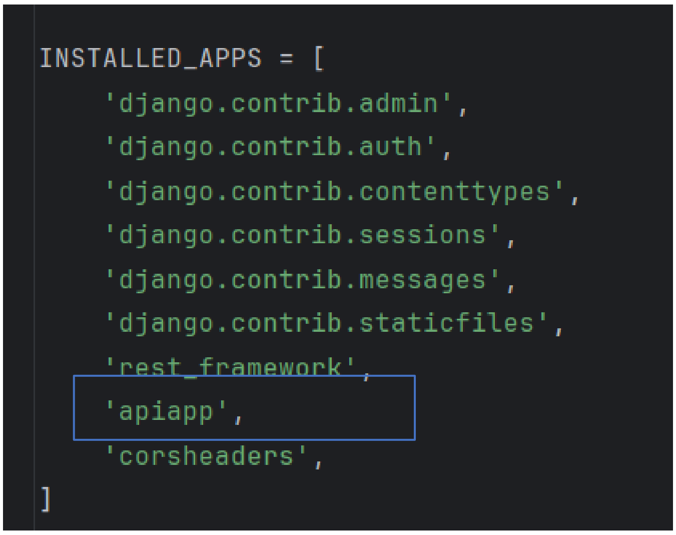
4 python manage.py startapp app

5 add rest_framework in settings.py



```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'rest_framework',
    'apiapp',
    'corsheaders',
]
```

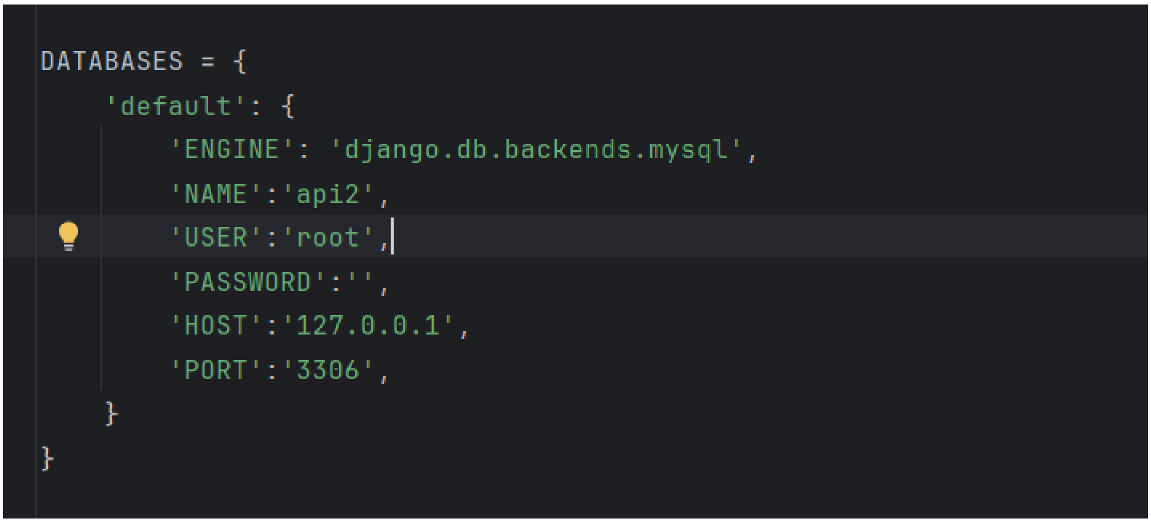
6 add app name (configure)



```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'rest_framework',
    'apiapp',
    'corsheaders',
]
```

7) pip install pymysql(Connect Django project to MySQL)

8) go to settings.py change database

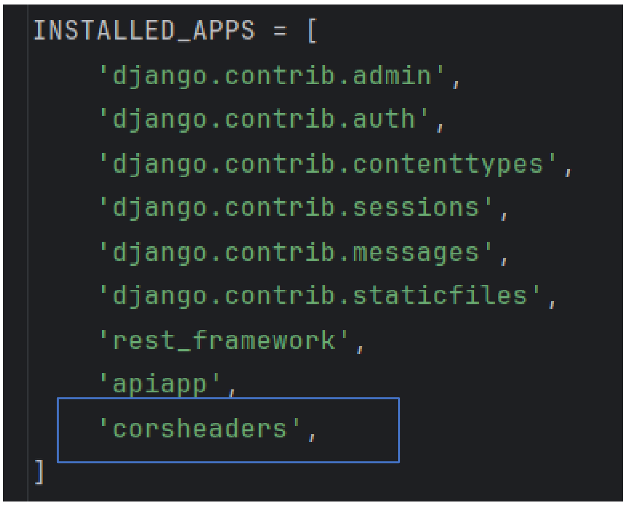


```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME':'api2',
        'USER':'root',
        'PASSWORD':'',
        'HOST':'127.0.0.1',
        'PORT':'3306',
    }
}
```

9) pip install django-cors-headers

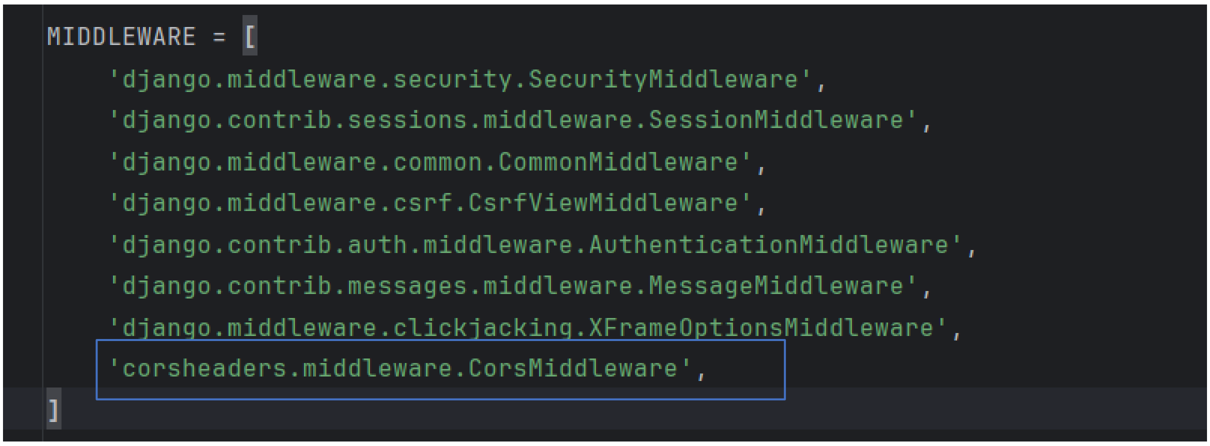
Cross-origin resource sharing, more often known as CORS, is a method that enables users to communicate with resources that are hosted on multiple domains.

10) settings.py



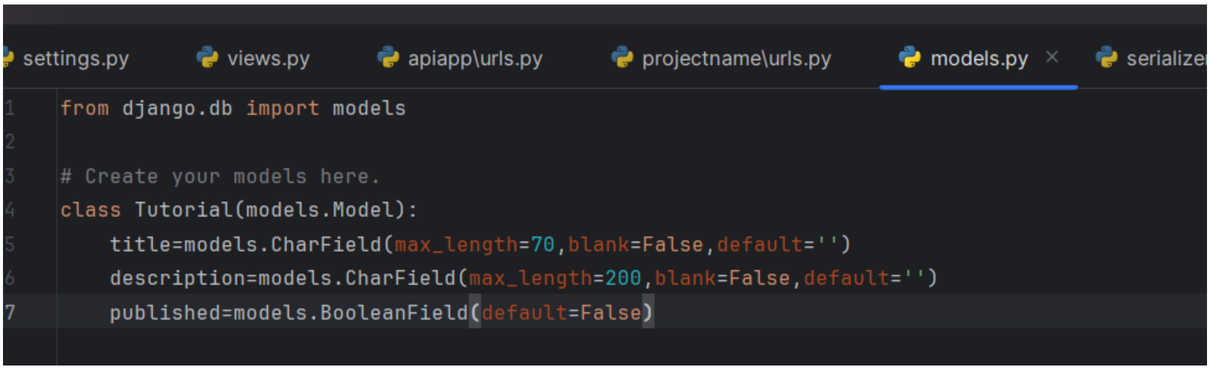
```python
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'rest_framework',
    'apiapp',
    'corsheaders',
]
```

11)settings.py



```python
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',
    'corsheaders.middleware.CorsMiddleware',
]
```

12)models.py

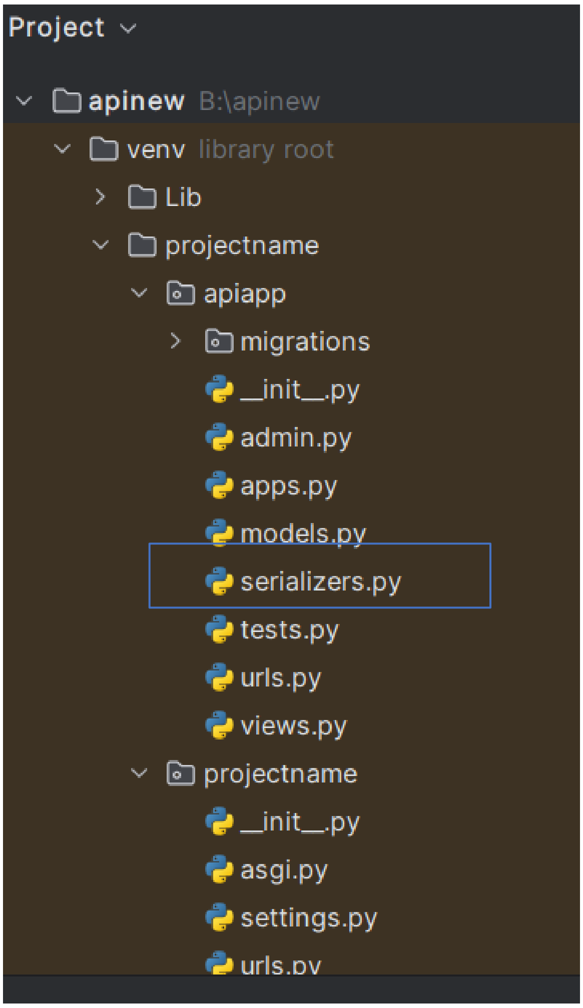


```python
from django.db import models

# Create your models here.
class Tutorial(models.Model):
    title=models.CharField(max_length=70,blank=False,default='')
    description=models.CharField(max_length=200,blank=False,default='')
    published=models.BooleanField(default=False)
```

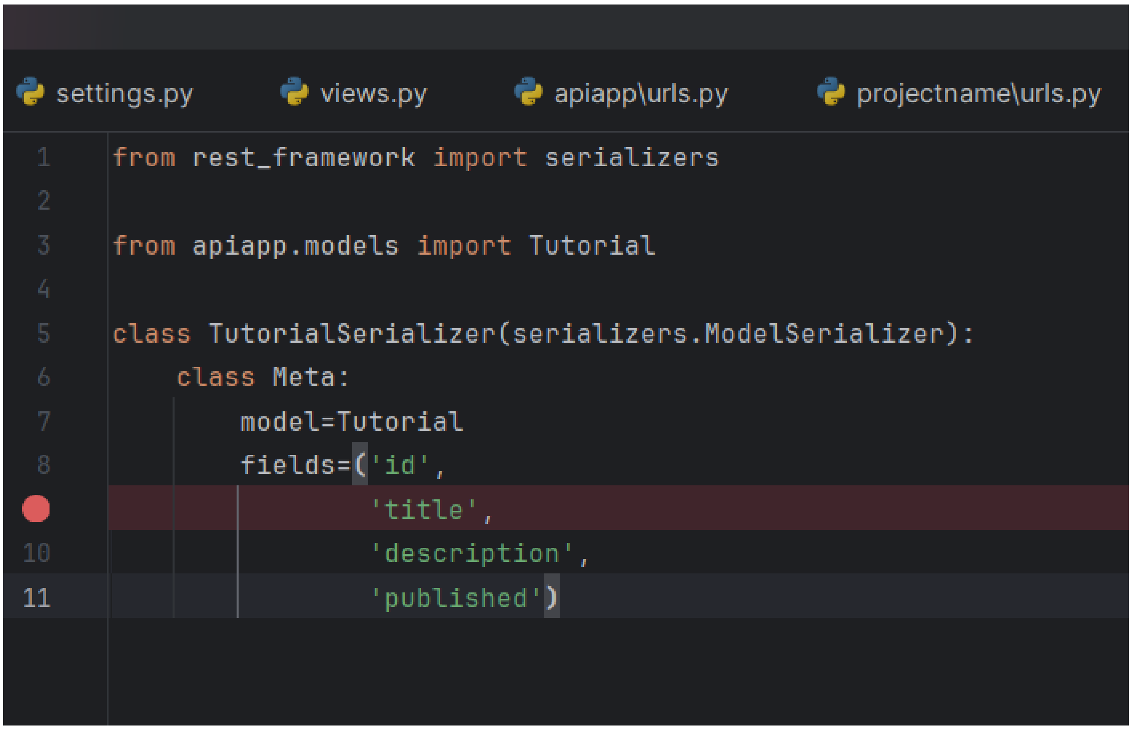
13) python manage.py makemigrations

14) python manage.py migrate

15)create serializers.py python file in app



16)serializers.py

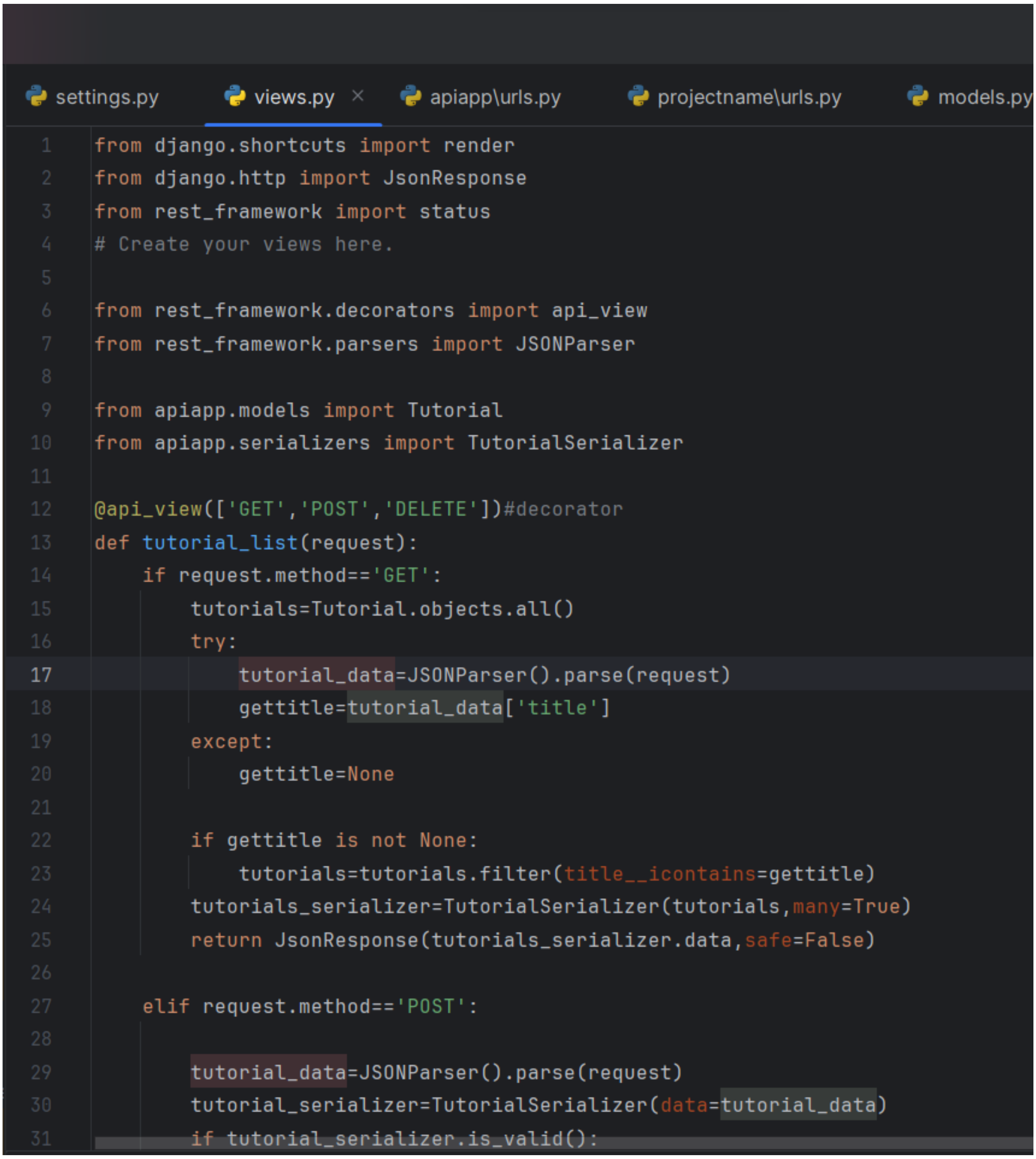


```python
from rest_framework import serializers

from apiapp.models import Tutorial

class TutorialSerializer(serializers.ModelSerializer):
    class Meta:
        model=Tutorial
        fields=('id',
                'title',
                'description',
                'published')
```

17)create urls.py in app and paste details from project urls.py file.

Also connect app urls.py to project urls.py

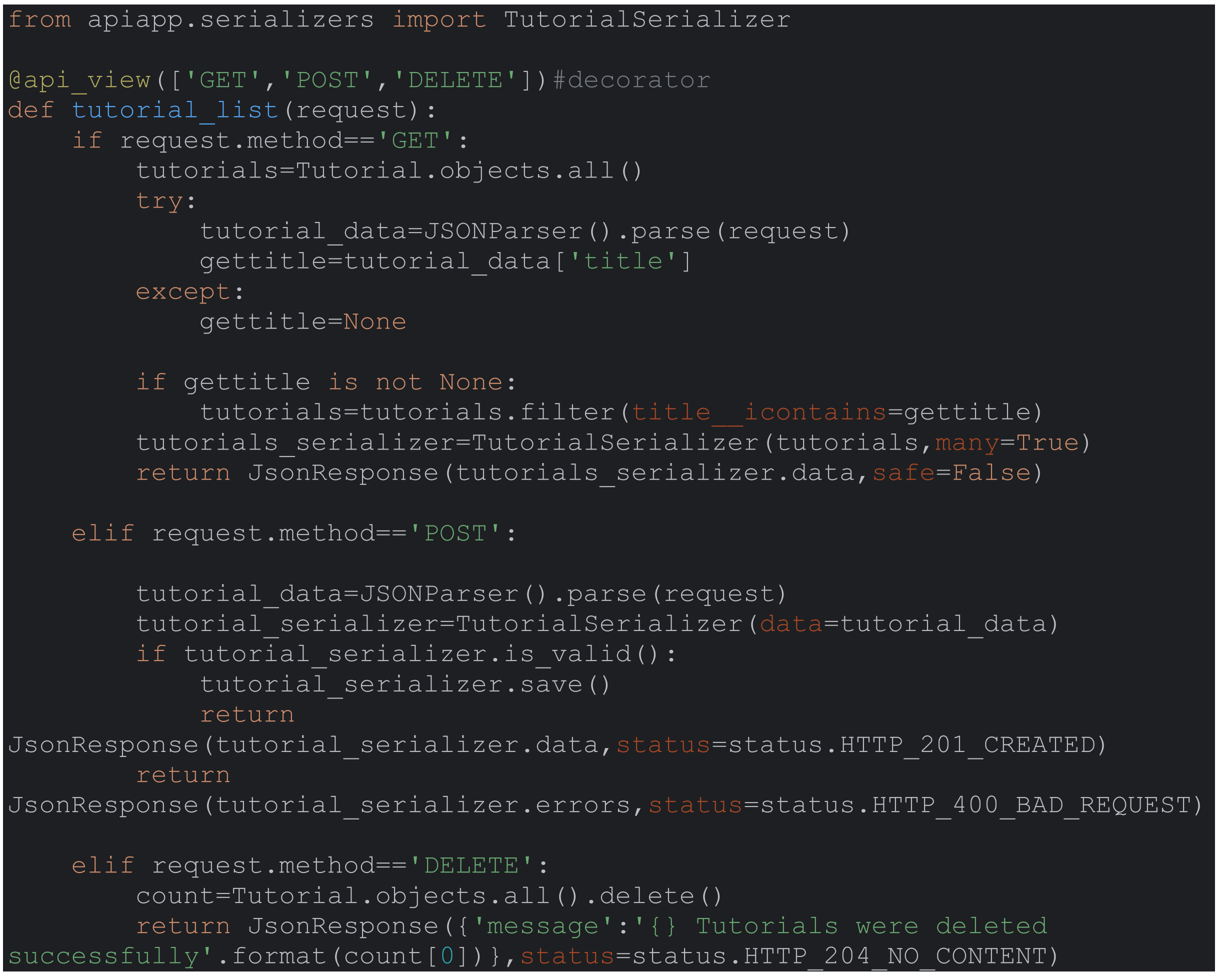
18)views.py



```
from django.shortcuts import render
from django.http import JsonResponse
from rest_framework import status
# Create your views here.

from rest_framework.decorators import api_view
from rest_framework.parsers import JSONParser

from apiapp.models import Tutorial
from apiapp.serializers import TutorialSerializer

@api_view(['GET','POST','DELETE'])#decorator
def tutorial_list(request):
    if request.method=='GET':
        tutorials=Tutorial.objects.all()
        try:
            tutorial_data=JSONParser().parse(request)
            gettitle=tutorial_data['title']
        except:
            gettitle=None

        if gettitle is not None:
            tutorials=tutorials.filter(title__icontains=gettitle)
        tutorials_serializer=TutorialSerializer(tutorials,many=True)
        return JsonResponse(tutorials_serializer.data,safe=False)

    elif request.method=='POST':

        tutorial_data=JSONParser().parse(request)
        tutorial_serializer=TutorialSerializer(data=tutorial_data)
        if tutorial_serializer.is_valid():
```

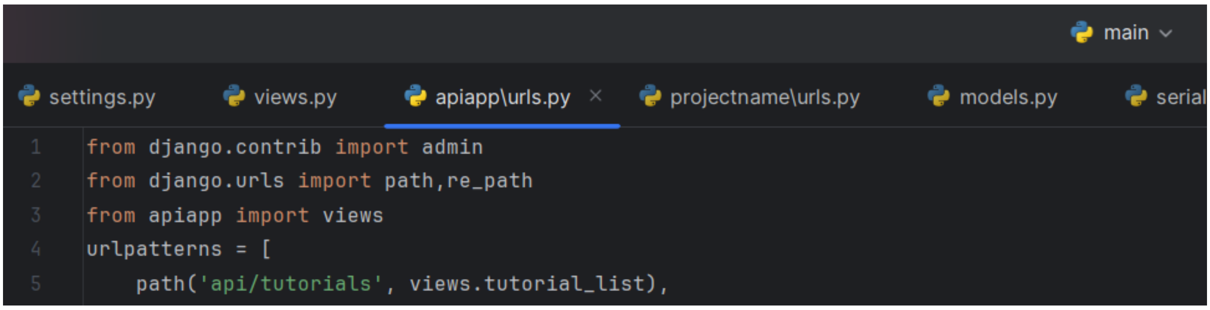
Full code

```
from django.shortcuts import render
from django.http import JsonResponse
from rest_framework import status
# Create your views here.

from rest_framework.decorators import api_view
from rest_framework.parsers import JSONParser

from apiapp.models import Tutorial
```



```
from apiapp.serializers import TutorialSerializer

@api_view(['GET','POST','DELETE'])#decorator
def tutorial_list(request):
    if request.method=='GET':
        tutorials=Tutorial.objects.all()
        try:
            tutorial_data=JSONParser().parse(request)
            gettitle=tutorial_data['title']
        except:
            gettitle=None

        if gettitle is not None:
            tutorials=tutorials.filter(title__icontains=gettitle)
        tutorials_serializer=TutorialSerializer(tutorials,many=True)
        return JsonResponse(tutorials_serializer.data,safe=False)

    elif request.method=='POST':

        tutorial_data=JSONParser().parse(request)
        tutorial_serializer=TutorialSerializer(data=tutorial_data)
        if tutorial_serializer.is_valid():
            tutorial_serializer.save()
            return
JsonResponse(tutorial_serializer.data,status=status.HTTP_201_CREATED)
        return
JsonResponse(tutorial_serializer.errors,status=status.HTTP_400_BAD_REQUEST)

    elif request.method=='DELETE':
        count=Tutorial.objects.all().delete()
        return JsonResponse({'message':'{} Tutorials were deleted
successfully'.format(count[0])},status=status.HTTP_204_NO_CONTENT)
```

19) urls.py



```
from django.contrib import admin
from django.urls import path,re_path
from apiapp import views
urlpatterns = [
    path('api/tutorials', views.tutorial_list),
```

20)python manage.py runserver

21)test api using postman