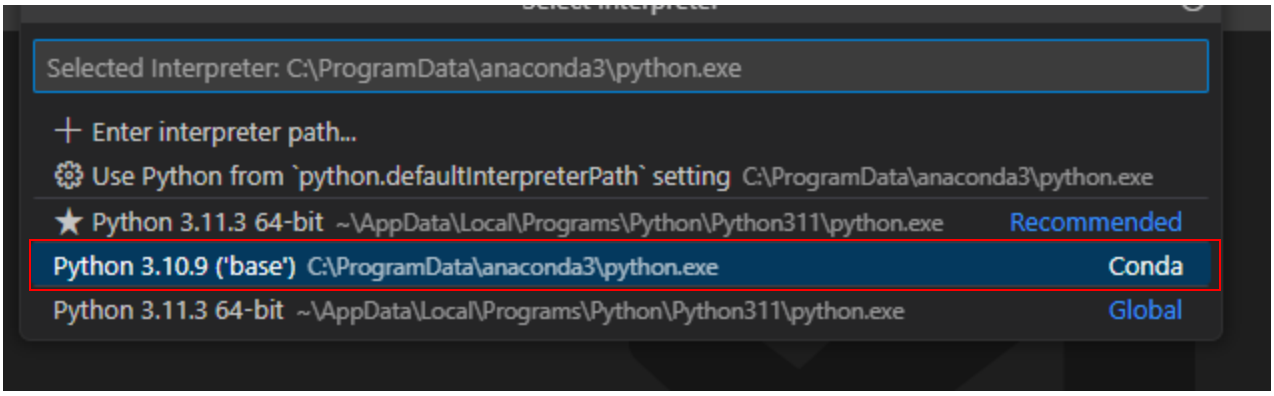
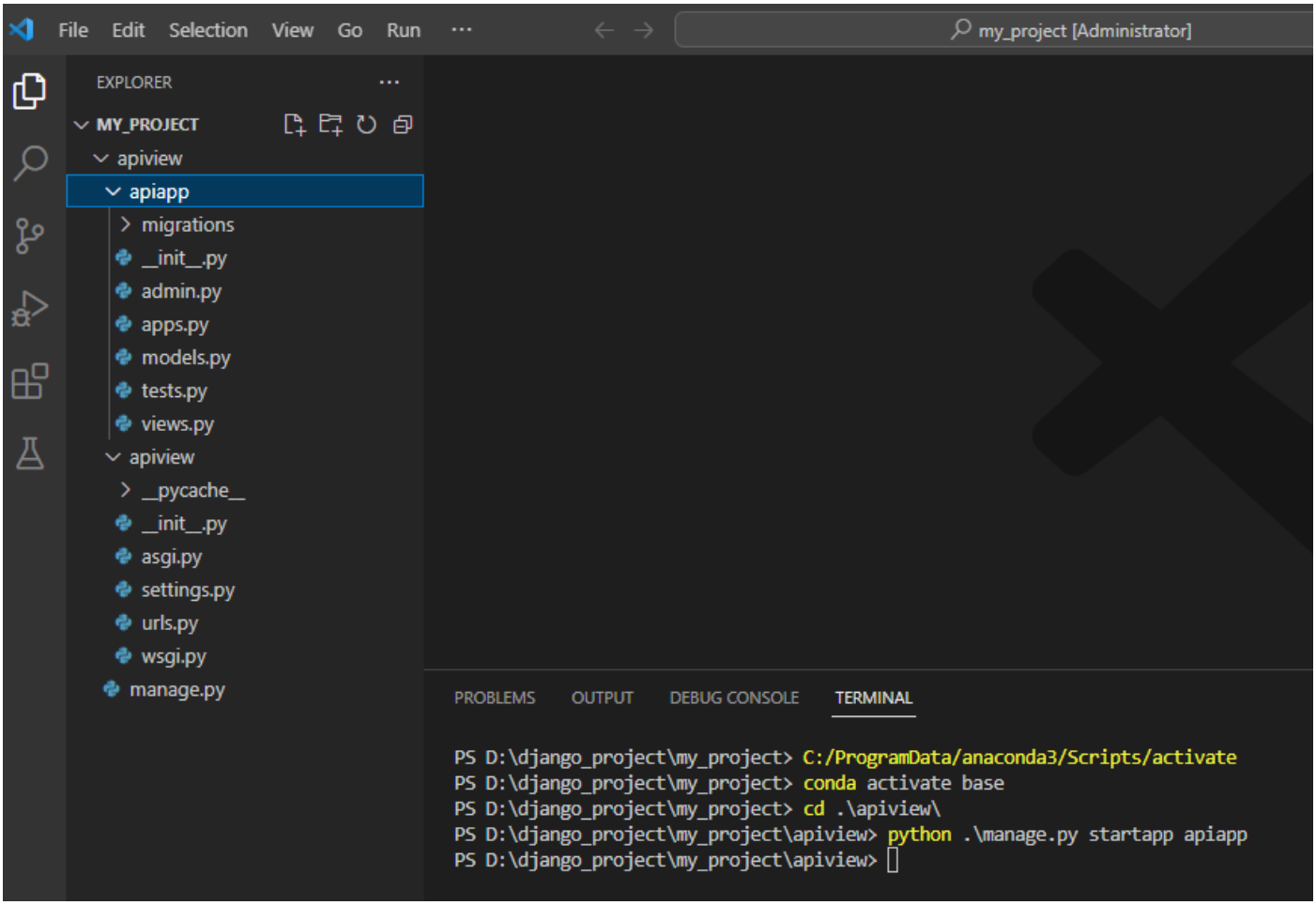
Work: CRUD Functionality with REST API VIEW

1. Create a folder named 'MY_PROJECT'. Open VS code and open the MY_PROJECT folder. Go to View -> command palette -> Select Python Interpreter -> Select *base* virtual environment



2. Open new terminal and type: *django-admin startproject apiview* to create a project named apiview.
3. Type: *cd apiview* and then *python manage.py startapp apiapp to* create apiapp.



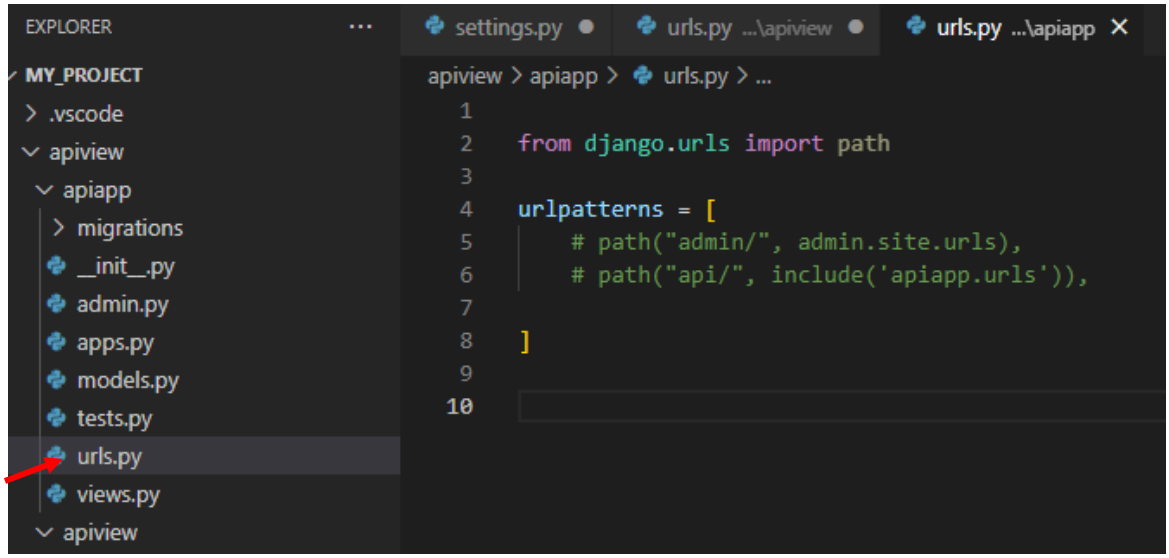
4. To USE REST framework install rest_framework in our app. Type in the terminal: `pip install djangorestframework`
5. Put the apps in the setting.py file

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

6. Configure the apiview/urls.py:

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

urlpatterns = [
    path("admin/", admin.site.urls),
    path("api/", include('apiapp.urls')),

]
```

7. Create *urls.py* in the ***app*** and configure:

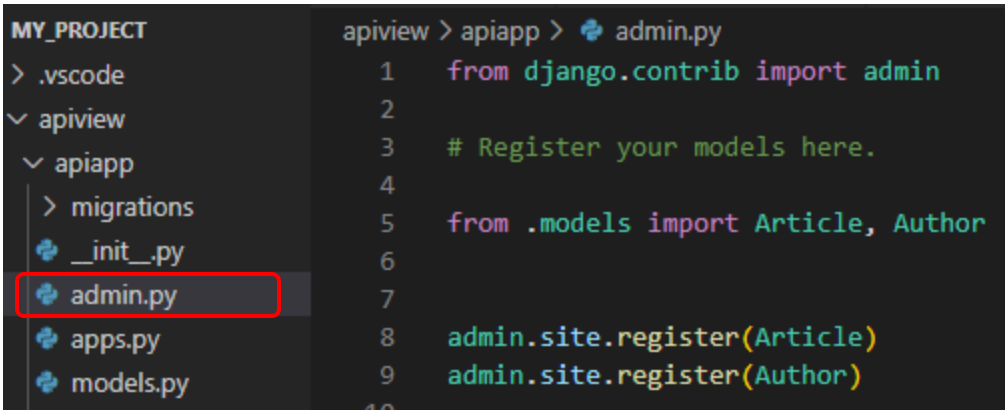


8. Start by creating the following models in the *models.py*

```python
from django.db import models


class Author(models.Model):
    name = models.CharField(max_length=255)
    email = models.EmailField()

    def __str__(self):
        return self.name

# Create your models here.

class Article(models.Model):
    title = models.CharField(max_length=120)
    description = models.TextField()
    body = models.TextField()
    author = models.ForeignKey('Author', related_name='articles', on_delete=models.CASCADE)

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

9. Create a *serializers.py* file in the app directory that contains JSON converter. Our serializers.py should look like this:

10. let's register our models in order for them to appear on the admin page. To do this, we open the *admin.py* file and register the models as shown below :



11. Now we are ready to migrate our project. Create super admin as well

```
PS D:\django_project\my_project\apiview> python .\manage.py makemigrations
Migrations for 'apiapp':
  apiapp\migrations\0001_initial.py
    - Create model Author
    - Create model Article
PS D:\django_project\my_project\apiview> python .\manage.py migrate
Operations to perform:
  Apply all migrations: admin, apiapp, auth, contenttypes, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying apiapp.0001_initial... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying auth.0010_alter_group_name_max_length... OK
  Applying auth.0011_update_proxy_permissions... OK
  Applying auth.0012_alter_user_first_name_max_length... OK
  Applying sessions.0001_initial... OK
PS D:\django_project\my_project\apiview>
```

```
PS D:\django_project\my_project\apiview> python .\manage.py createsuperuser
Username (leave blank to use 'cse'): admin
Email address:
Password:
Password (again):
This password is too short. It must contain at least 8 characters.
This password is too common.
This password is entirely numeric.
Bypass password validation and create user anyway? [y/N]: y
Superuser created successfully.
PS D:\django_project\my_project\apiview>
```

12. Now type: *python manage.py runserver* . on the browser type: http://127.0.0.1:8000/admin/

Enter admin id and password to enter into the admin panel. Add some author and article information.

13. Now write API View in the *views.py* file. This should look like this:

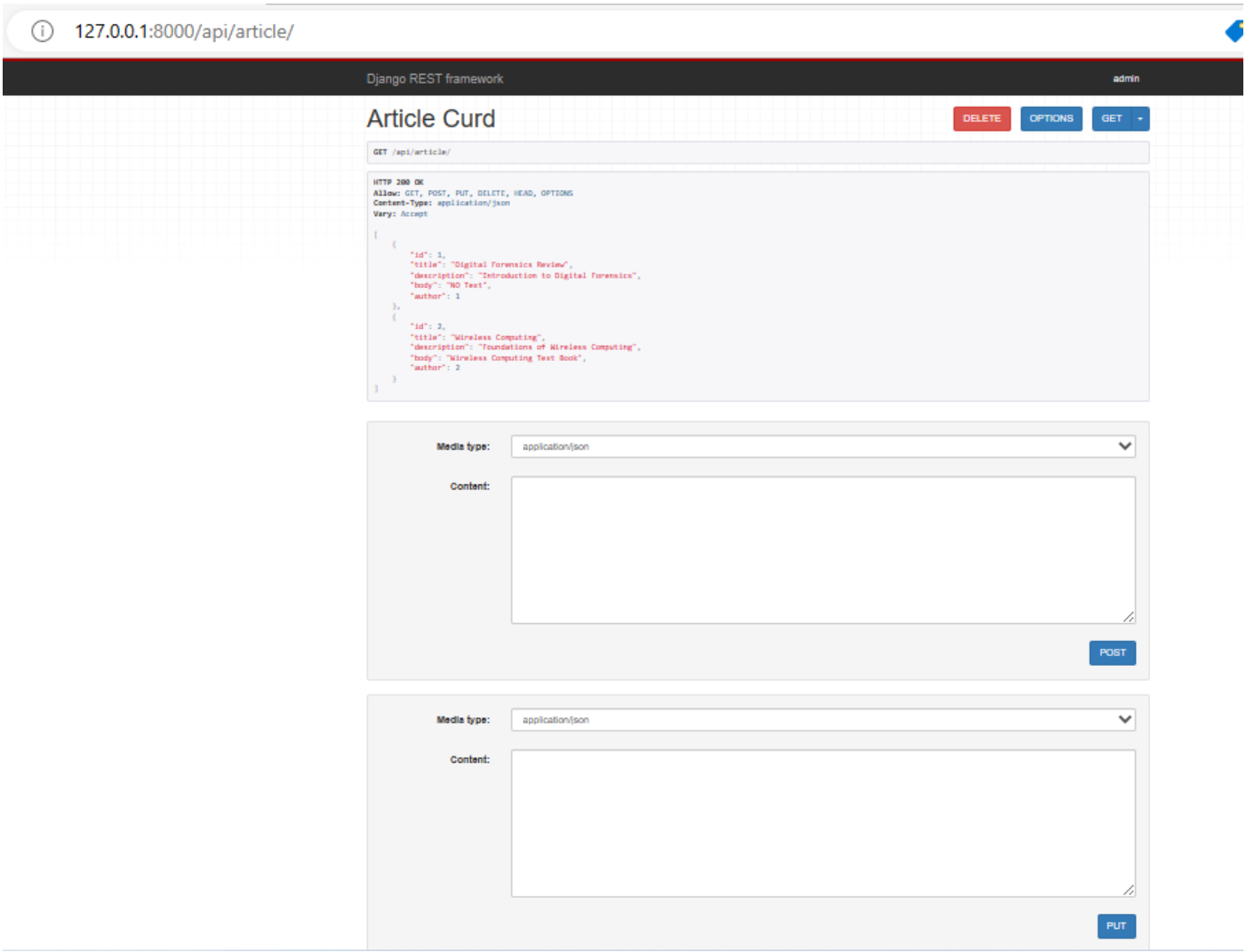
```
from django.shortcuts import render

# Create your views here.
from rest_framework.views import APIView
from rest_framework import status, permissions
from rest_framework.response import Response
from django.shortcuts import get_object_or_404
```

```python
from .serializers import AuthorSerializer, ArticleSerializer
from .models import Article, Author


class ArticleCurd(APIView):
    permission_classes = [permissions.AllowAny]

    def get(self, request, pk=None, format=None):
        if pk:
            article = get_object_or_404(Article, id=pk)
            article_serializers = ArticleSerializer(article)
            return Response(article_serializers.data, status=status.HTTP_200_OK)
        else:
            articles_qs = Article.objects.all()
            article_serializers = ArticleSerializer(articles_qs, many=True)
            return Response(article_serializers.data, status=status.HTTP_200_OK)

    def post(self, request):
        article_serializers = ArticleSerializer(data=request.data)
        article_serializers.is_valid(raise_exception=True)
        article_serializers.save()
        return Response(article_serializers.data, status=status.HTTP_201_CREATED)

    def put(self, request, pk=None, format=None):
        article = get_object_or_404(Article, id=pk)
        article_serializers = ArticleSerializer(
            instance=Article, data=request.data)
        article_serializers.is_valid(raise_exception=True)
        article_serializers.save()
        return Response(article_serializers.data, status=status.HTTP_200_OK)

    def delete(self, request, pk=None, format=None):
        article = get_object_or_404(Article, id=pk)
        article.delete()
        return Response({'msg': 'done'}, status=status.HTTP_204_NO_CONTENT)
```

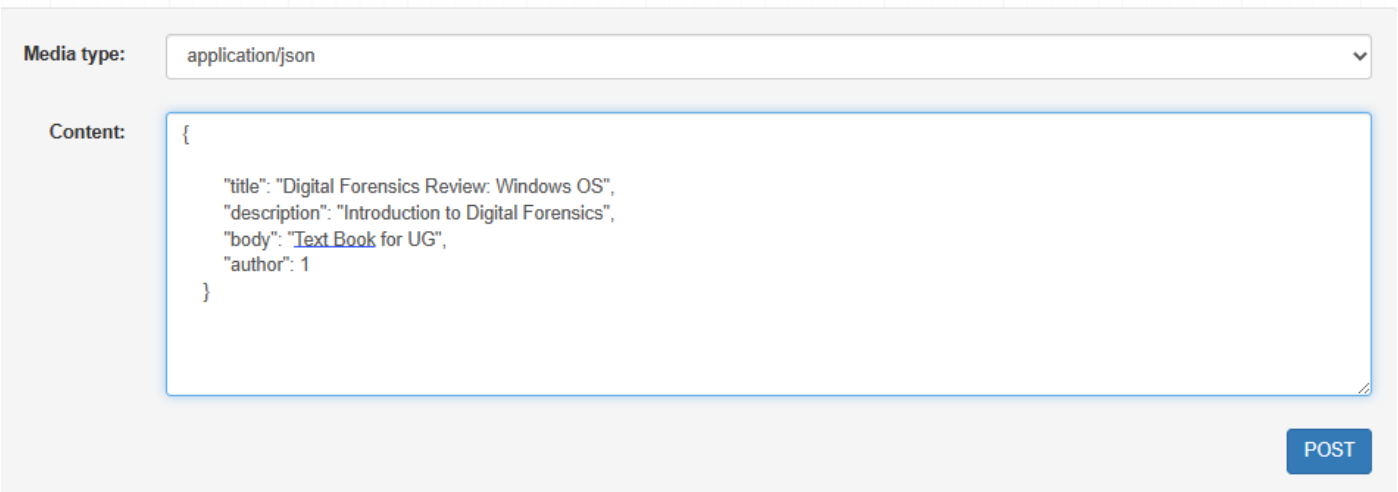
14. Update urls.py file as follows:

```python
from django.urls import path

from .views import ArticleCurd

urlpatterns = [
    path('article/', ArticleCurd.as_view(), name="article_list_or_create"),
    path('article/<int:pk>/', ArticleCurd.as_view(), name="article_get_or_update"),

]
```

15. Type on the browser: http://127.0.0.1:8000/api/article/ to show the list of articles.



16. To add new article add data in json format:



17. **Similarly add crud application for Author model. In this case, modify urls.py and views.py**

GOOD LUCK