Topic: 4. Setting API EndPoints

Speaker: Personal | Notebook: API Development using Django Framework

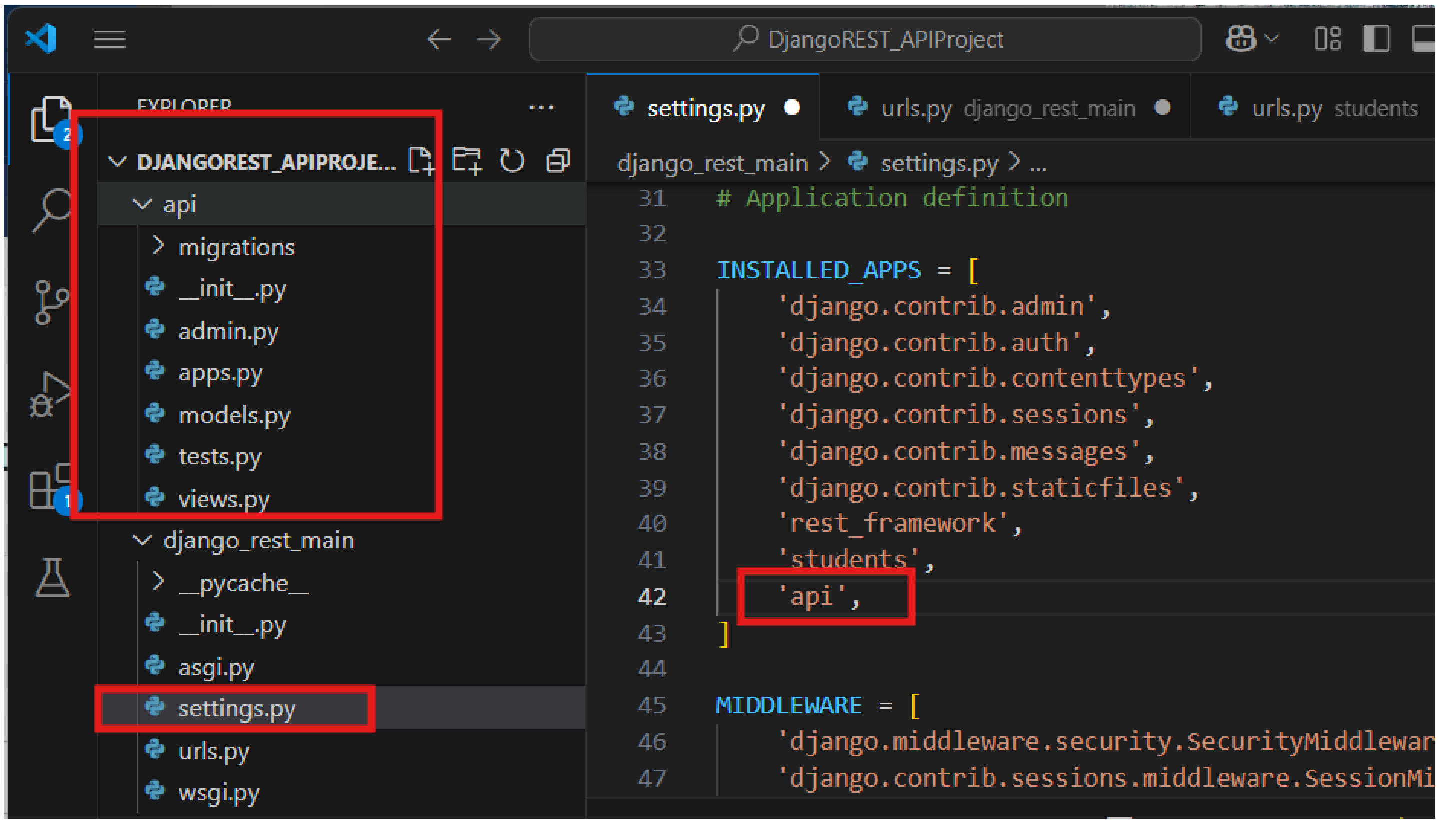
---

For other resources on how to create simple API endpoints using Django, use this reference in Medium.

1. Create an API app in your Django project:

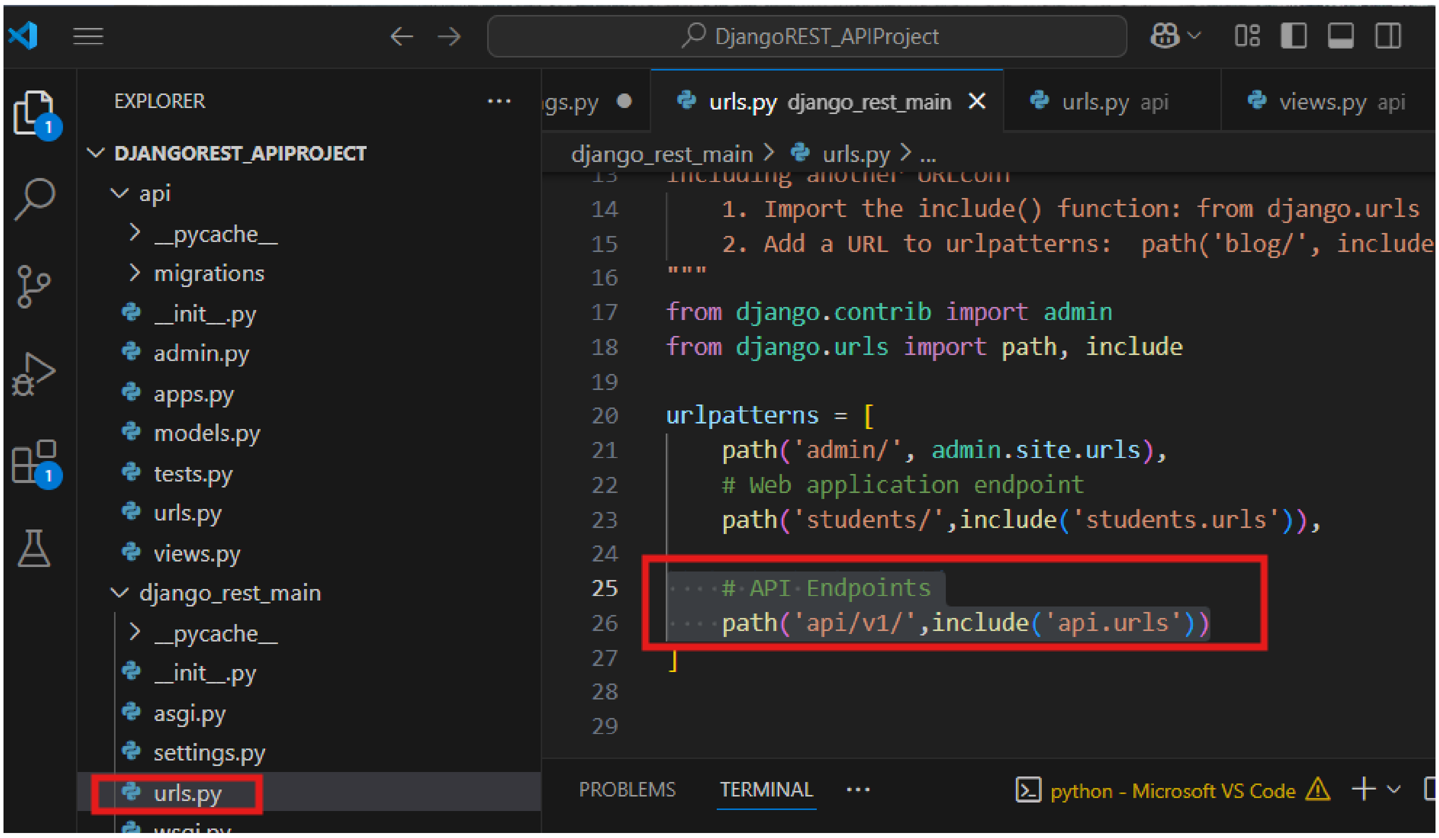
```
$ python manage.py startapp api
```

2. Register this new app in your SETTINGS.PY



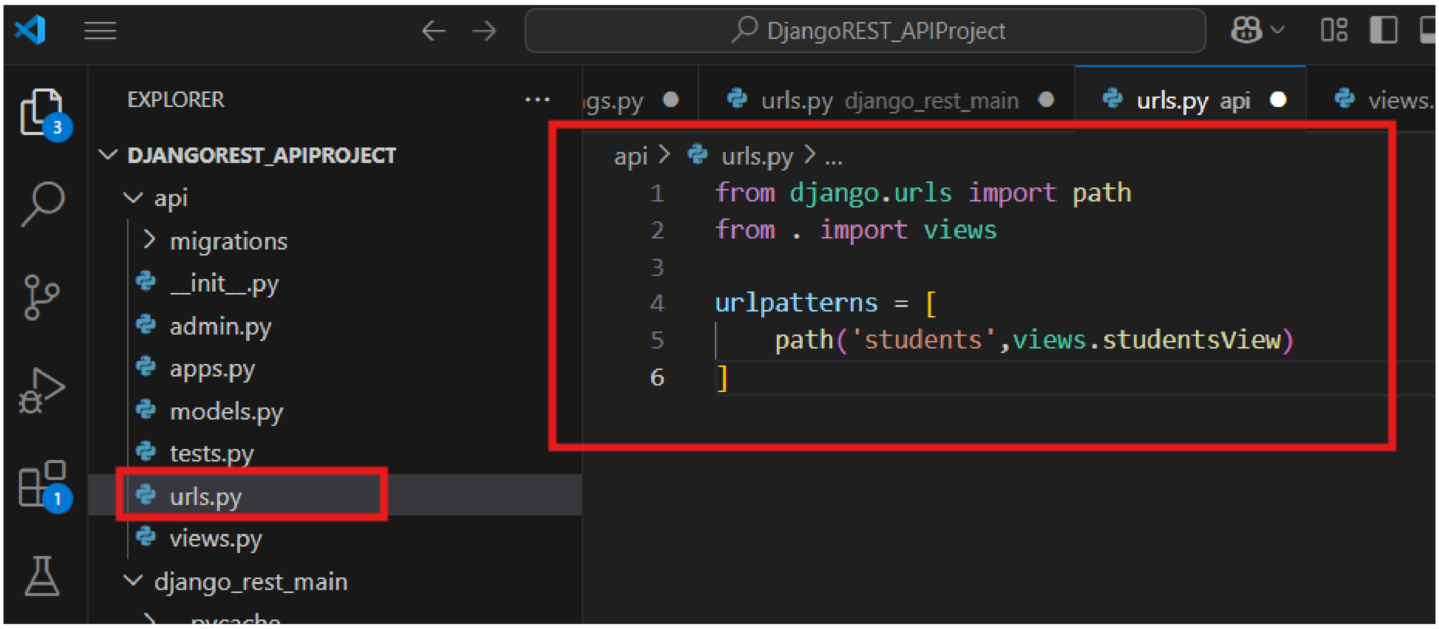
```
# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'rest_framework',
    'students',
    'api',
]

MIDDLEWARE = [
    'django.middleware.security.SecurityMiddlewar
    'django.contrib.sessions.middleware.SessionMi
```

3. Now, update the main project's URLS.PY to include the URLS.PY of the newly created app.



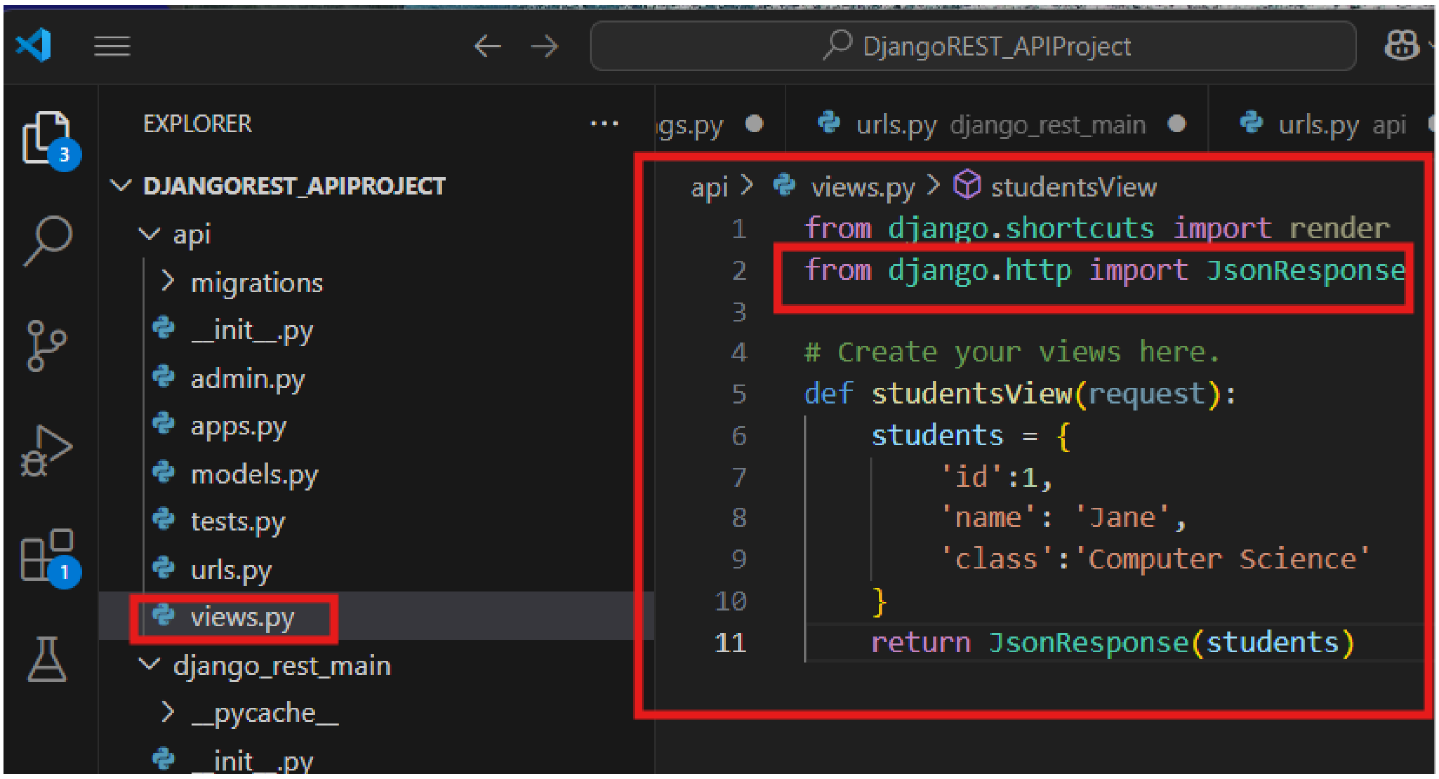
```
    1. Import the include() function: from django.urls
    2. Add a URL to urlpatterns:  path('blog/', include
"""
from django.contrib import admin
from django.urls import path, include

urlpatterns = [
    path('admin/', admin.site.urls),
    # Web application endpoint
    path('students/',include('students.urls')),

    # API Endpoints
    path('api/v1/',include('api.urls'))
]
```

4. We then create a URLS.py in our API app and create the path.



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

urlpatterns = [
    path('students',views.studentsView)
]
```

5. Update the API APP'S VIEWS.PY . Unlike regular views of web-based endpoints, APIs return JSON data.

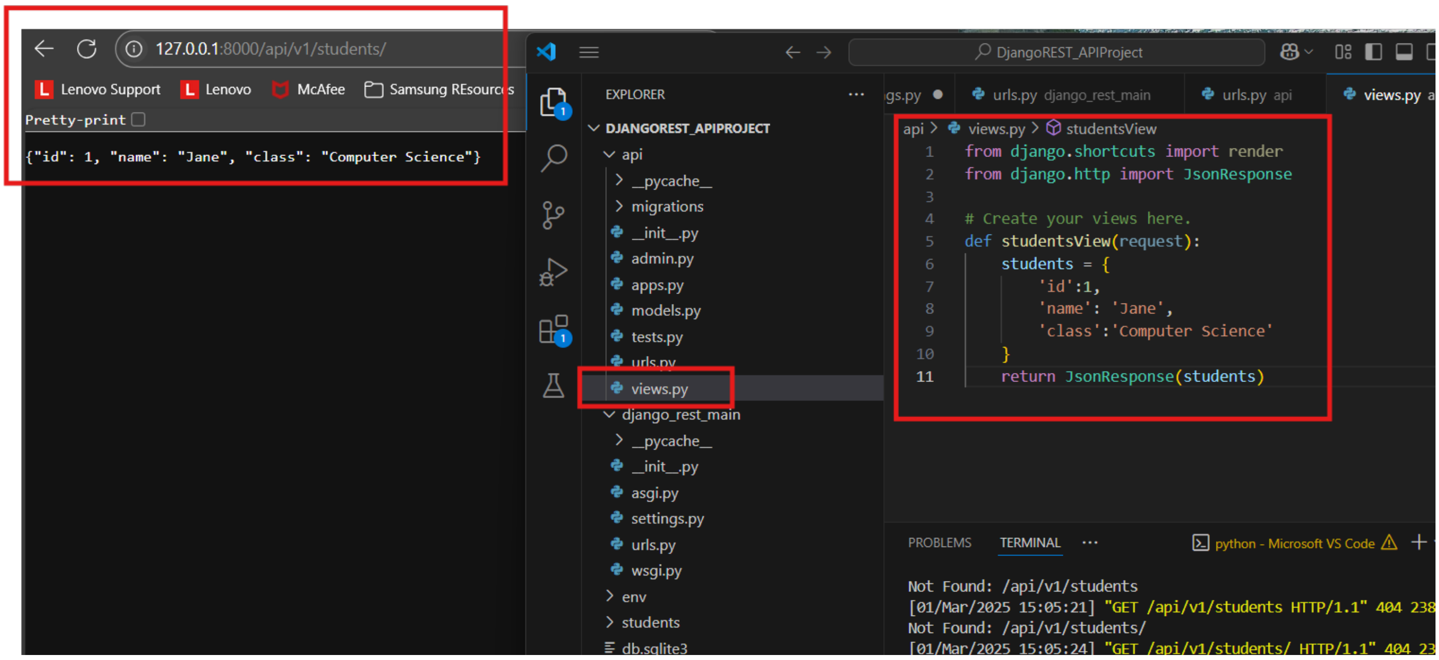


```
from django.shortcuts import render
from django.http import JsonResponse

# Create your views here.
def studentsView(request):
    students = {
        'id':1,
        'name': 'Jane',
        'class':'Computer Science'
    }
    return JsonResponse(students)
```

6. Run the server and add the API endpoint:

```
$ python manage.py runserver
```

In your browser: 127.0.0.1:8000/api/v1/students/



```
{"id": 1, "name": "Jane", "class": "Computer Science"}
```

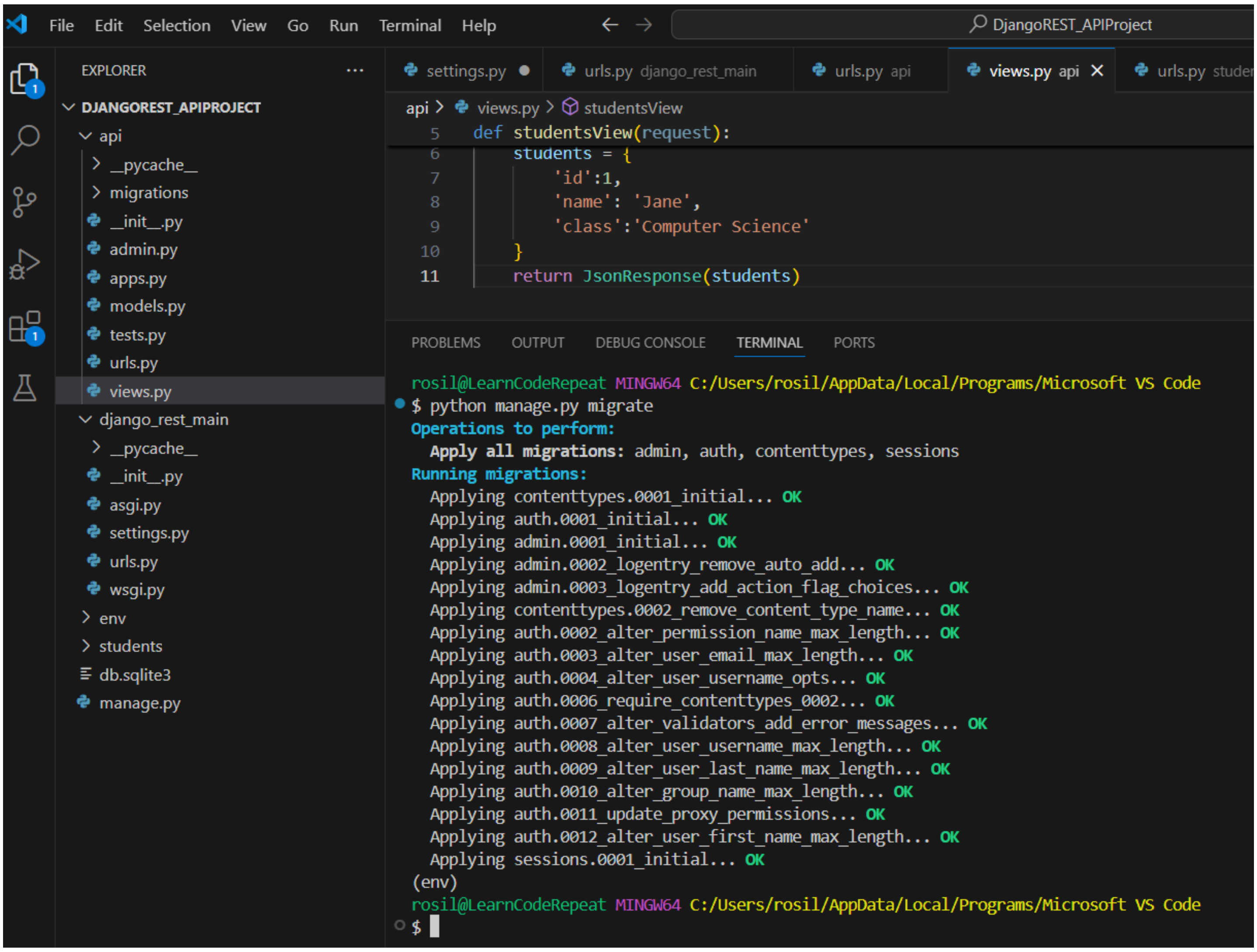
```
from django.shortcuts import render
from django.http import JsonResponse

# Create your views here.
def studentsView(request):
    students = {
        'id':1,
        'name': 'Jane',
        'class':'Computer Science'
    }
    return JsonResponse(students)
```

```
Not Found: /api/v1/students
[01/Mar/2025 15:05:21] "GET /api/v1/students HTTP/1.1" 404 238
Not Found: /api/v1/students/
[01/Mar/2025 15:05:24] "GET /api/v1/students/ HTTP/1.1" 404 23
```

7. Run the migrations command and create superuser.

```
$ python manage.py migrate
```

This will create `user.auths` default table.



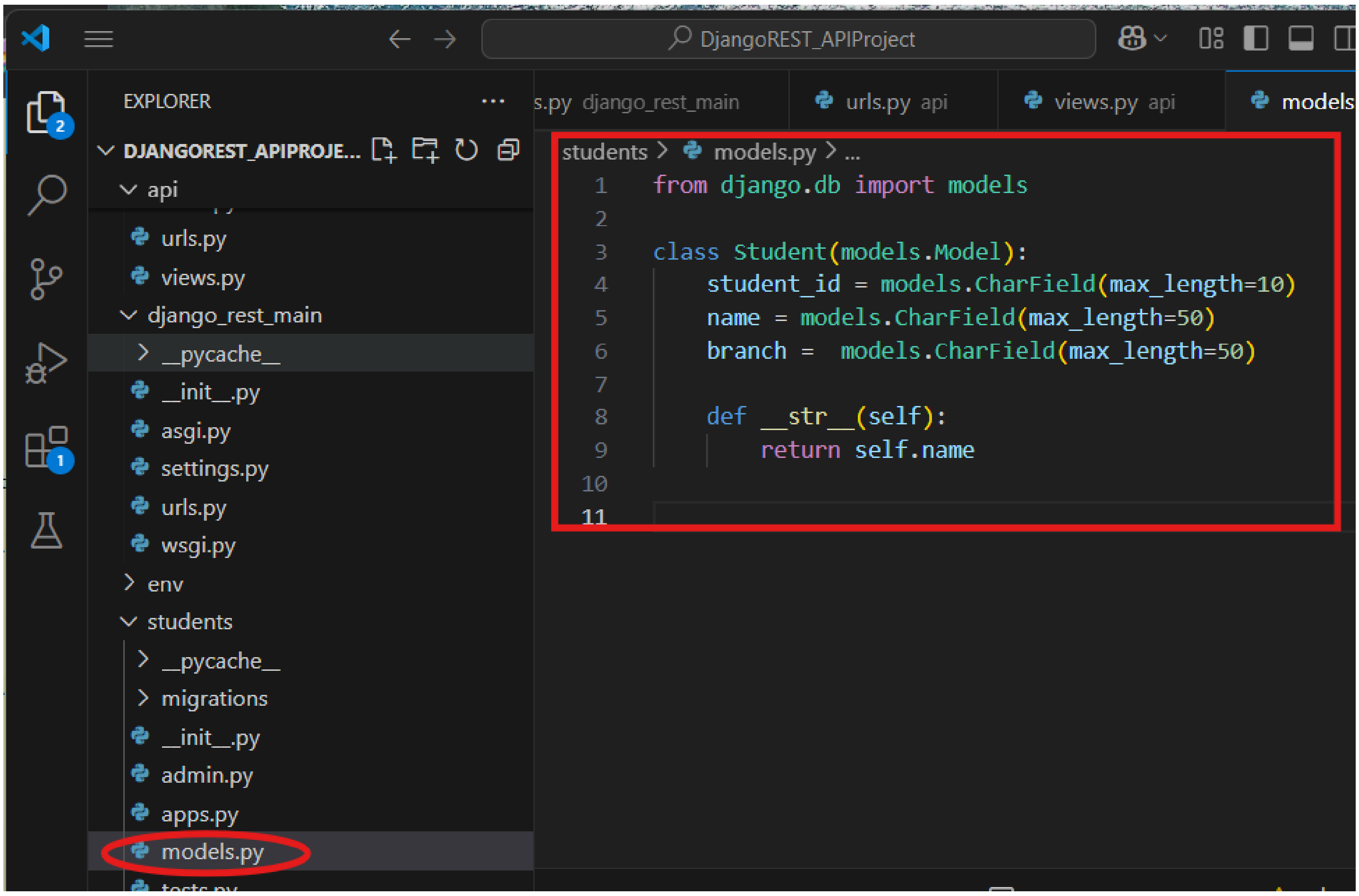
```python
def studentsView(request):
    students = {
        'id':1,
        'name': 'Jane',
        'class':'Computer Science'
    }
    return JsonResponse(students)
```

```
rosil@LearnCodeRepeat MINGW64 C:/Users/rosil/AppData/Local/Programs/Microsoft VS Code
$ python manage.py migrate
Operations to perform:
  Apply all migrations: admin, 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 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.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
(env)
rosil@LearnCodeRepeat MINGW64 C:/Users/rosil/AppData/Local/Programs/Microsoft VS Code
$ 
```

8. Access the admin panel by running the server. Access it using:

```
http://127.0.0.1:8000/admin/
```

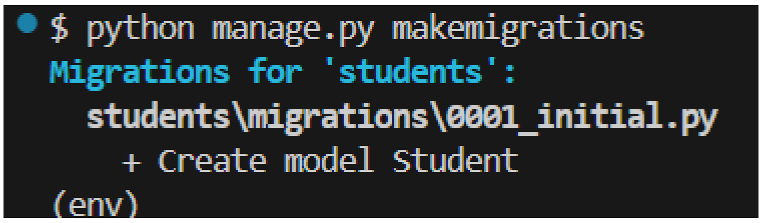
9. Create a student model in the STUDENTS app to create a new table.



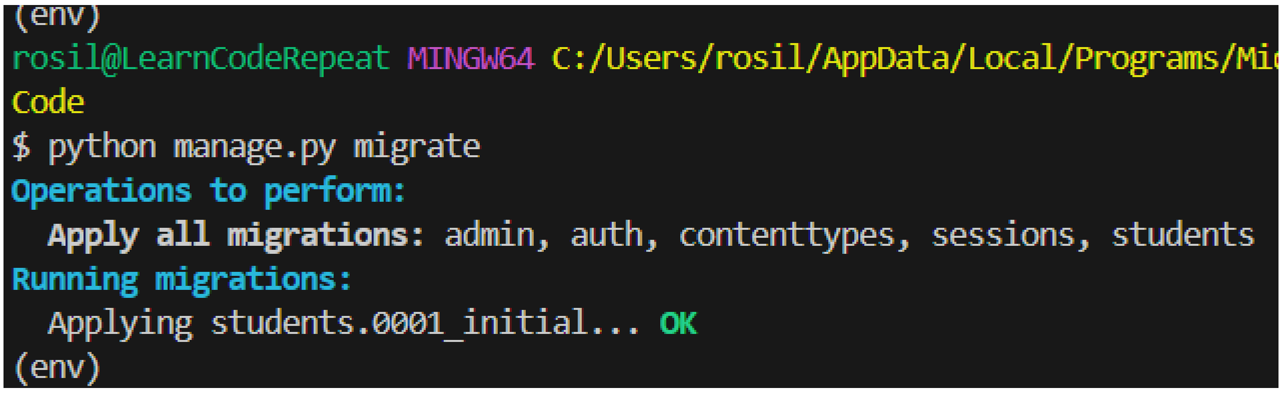
```python
from django.db import models

class Student(models.Model):
    student_id = models.CharField(max_length=10)
    name = models.CharField(max_length=50)
    branch =  models.CharField(max_length=50)

    def __str__(self):
        return self.name

```

10. Run the migrations.

```
$ python manage.py makemigrations

$ python manage.py migrate
```

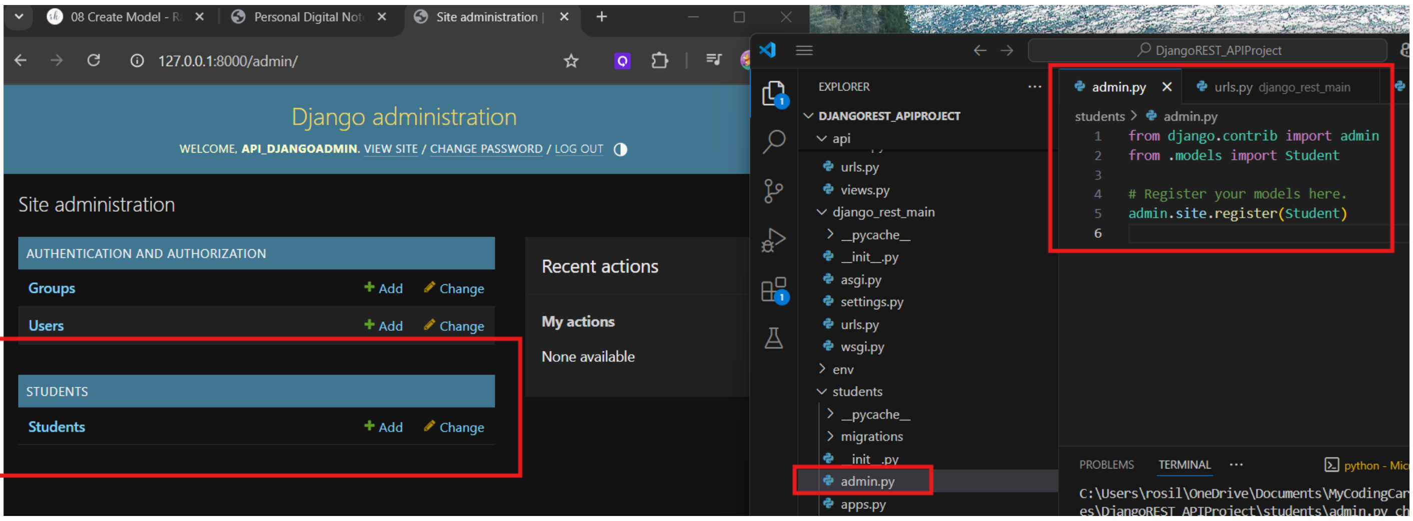


```
$ python manage.py makemigrations
Migrations for 'students':
  students\migrations\0001_initial.py
    + Create model Student
(env)
```



```
(env)
rosil@LearnCodeRepeat MINGW64 C:/Users/rosil/AppData/Local/Programs/Mi
Code
$ python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions, students
Running migrations:
  Applying students.0001_initial... OK
(env)
```

11. To make this table appear on your admin dashboard, register this in the ADMIN.PY of STUDENTS app.


Django administration

WELCOME, API_DJANGOADMIN. VIEW SITE / CHANGE PASSWORD / LOG OUT

Site administration

AUTHENTICATION AND AUTHORIZATION		
Groups	Add	Change
Users	Add	Change

STUDENTS		
Students	Add	Change

Recent actions

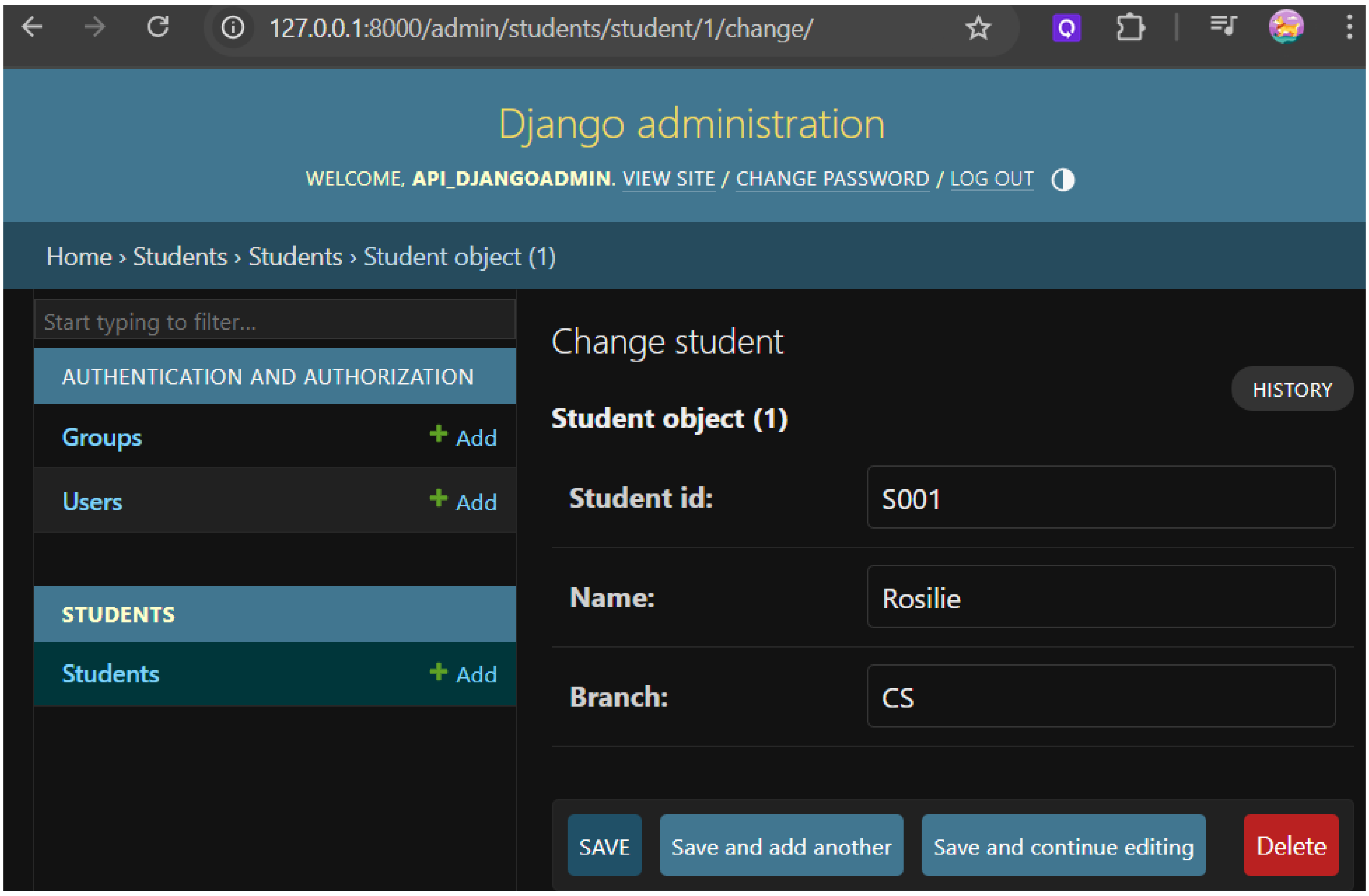
My actions

None available

```python
from django.contrib import admin
from .models import Student

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

```
C:\Users\rosil\OneDrive\Documents\MyCodingCar
es\DjangoREST_APIProject\students\admin.py ch
```

12. Create a sample record in the table.


127.0.0.1:8000/admin/students/student/1/change/

Django administration

WELCOME, API_DJANGOADMIN. VIEW SITE / CHANGE PASSWORD / LOG OUT

Home › Students › Students › Student object (1)

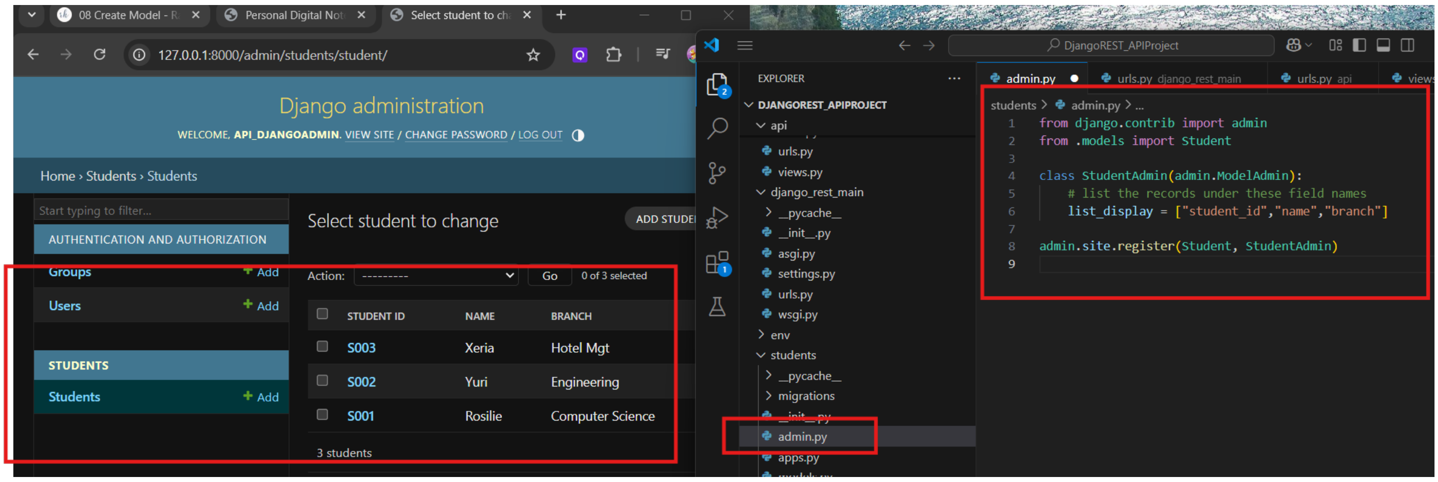
Change student

Student object (1)

Student id:	S001
Name:	Rosilie
Branch:	CS

Start typing to filter...
AUTHENTICATION AND AUTHORIZATION
Groups + Add
Users + Add
STUDENTS
Students + Add
HISTORY
SAVE
Save and add another
Save and continue editing
Delete

13. To display the model student records using the admin dashboard:


127.0.0.1:8000/admin/students/student/

Django administration

WELCOME, API_DJANGOADMIN. VIEW SITE / CHANGE PASSWORD / LOG OUT

Home › Students › Students

Select student to change

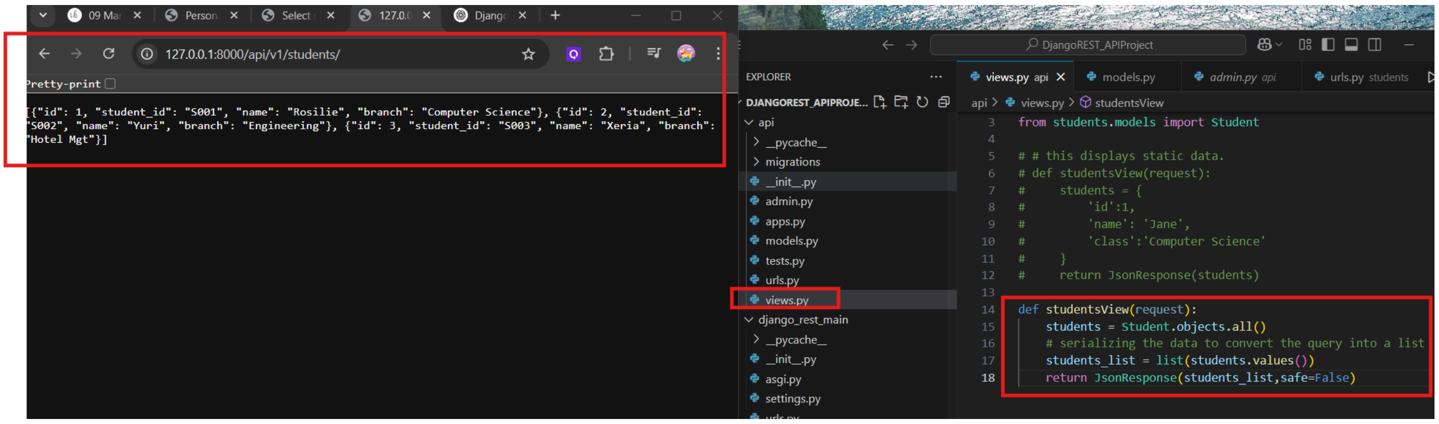
Action: --------- Go 0 of 3 selected

STUDENT ID	NAME	BRANCH
S003	Xeria	Hotel Mgt
S002	Yuri	Engineering
S001	Rosilie	Computer Science

3 students

```python
from django.contrib import admin
from .models import Student

class StudentAdmin(admin.ModelAdmin):
    # list the records under these field names
    list_display = ["student_id","name","branch"]

admin.site.register(Student, StudentAdmin)
```

14. To display the dynamic data from the database. We need to **SERIALIZE** to convert the returned query set into a list.


127.0.0.1:8000/api/v1/students/

```
[{"id": 1, "student_id": "S001", "name": "Rosilie", "branch": "Computer Science"}, {"id": 2, "student_id":
"S002", "name": "Yuri", "branch": "Engineering"}, {"id": 3, "student_id": "S003", "name": "Xeria", "branch":
"Hotel Mgt"}]
```

```python
from students.models import Student

# # this displays static data.
# def studentsView(request):
#     students = {
#         'id':1,
#         'name': 'Jane',
#         'class':'Computer Science'
#     }
#     return JsonResponse(students)

def studentsView(request):
    students = Student.objects.all()
    # serializing the data to convert the query into a list
    students_list = list(students.values())
    return JsonResponse(students_list,safe=False)
```

15. SERIALIZERS are like TRANSLATORS that convert certain data i.e QUERYSET from your database into other types of data like JSON data that can be used on HTML. While DESERIALIZERS will reverse the translation i.e from JSON file into Query set (database records).