Topic: 6. Single Record Fetching/Updating using PK (PUT/DELETE)

Speaker: | Notebook: API Development using Django Framework

1. To fetch a single record using a primary, we update the URLS.PY and add a new path:
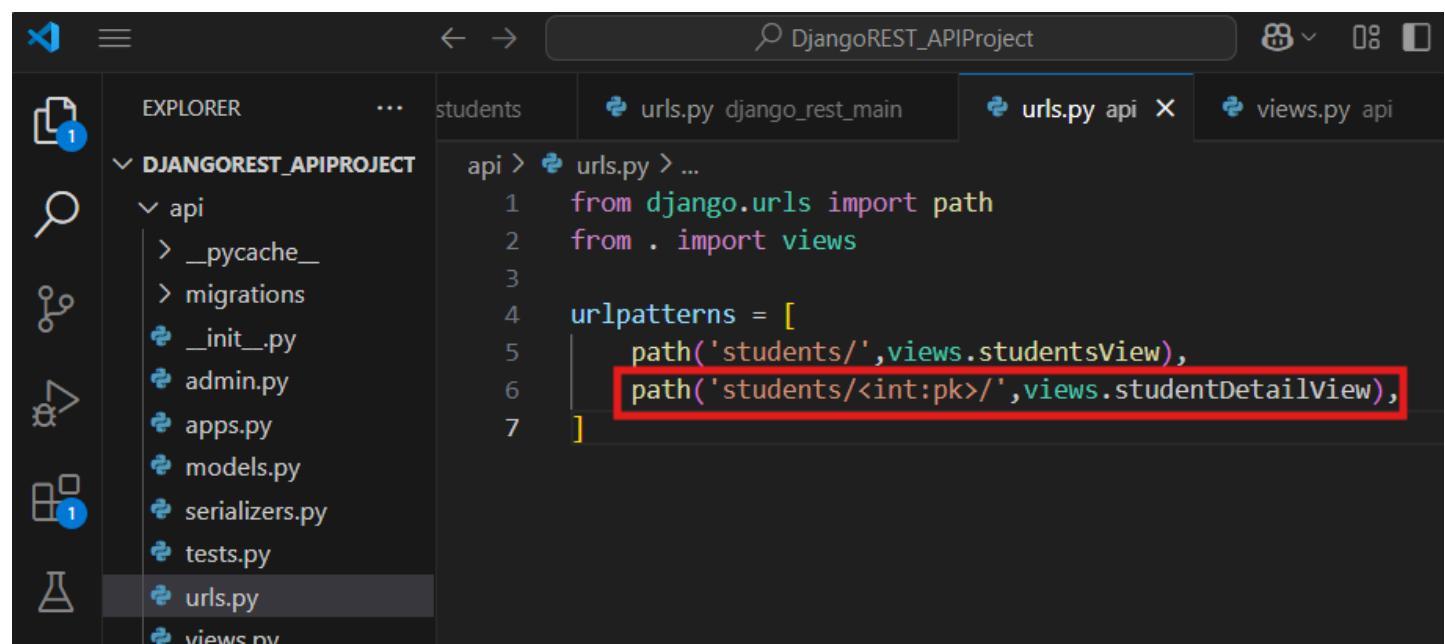
2. Update the VIEWS.PY and add the path plus the primary to search:
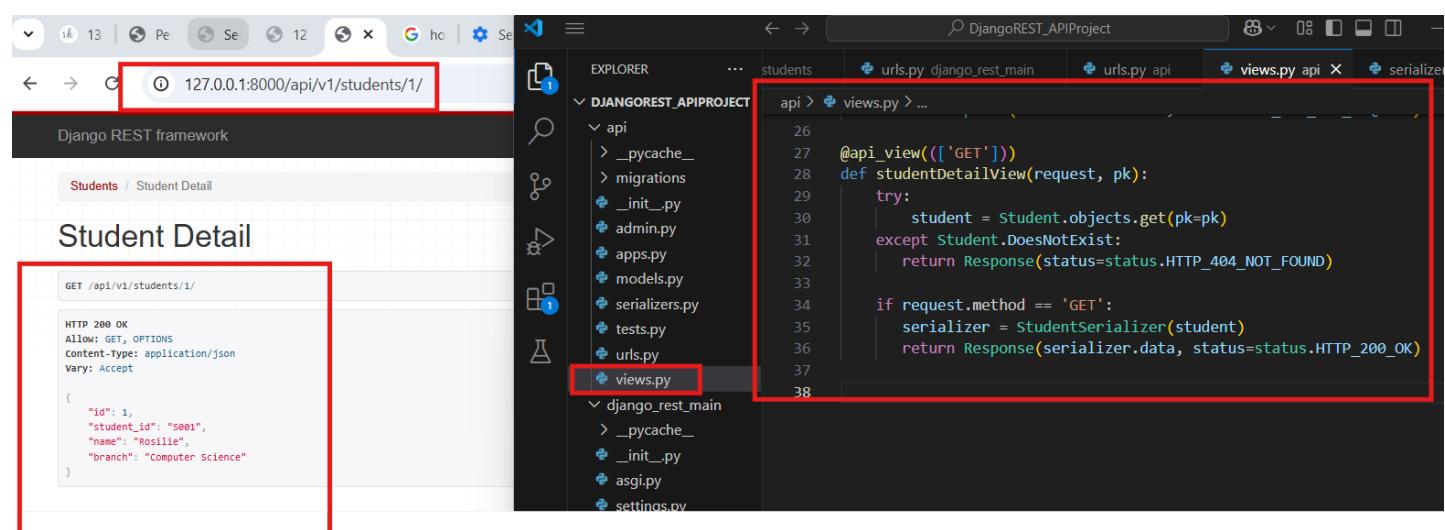
3. Using the POSTMAN, add the part and the primary key to retrieve that specific record. Click SEND:

In case, you search for the record that doesn't exist, the 404 error message should display:

4. In POSTMAN (we changed the theme), if you have a non-existing record:

5. To update a specific record, we update the VIEWS.PY. We use the PUT method.
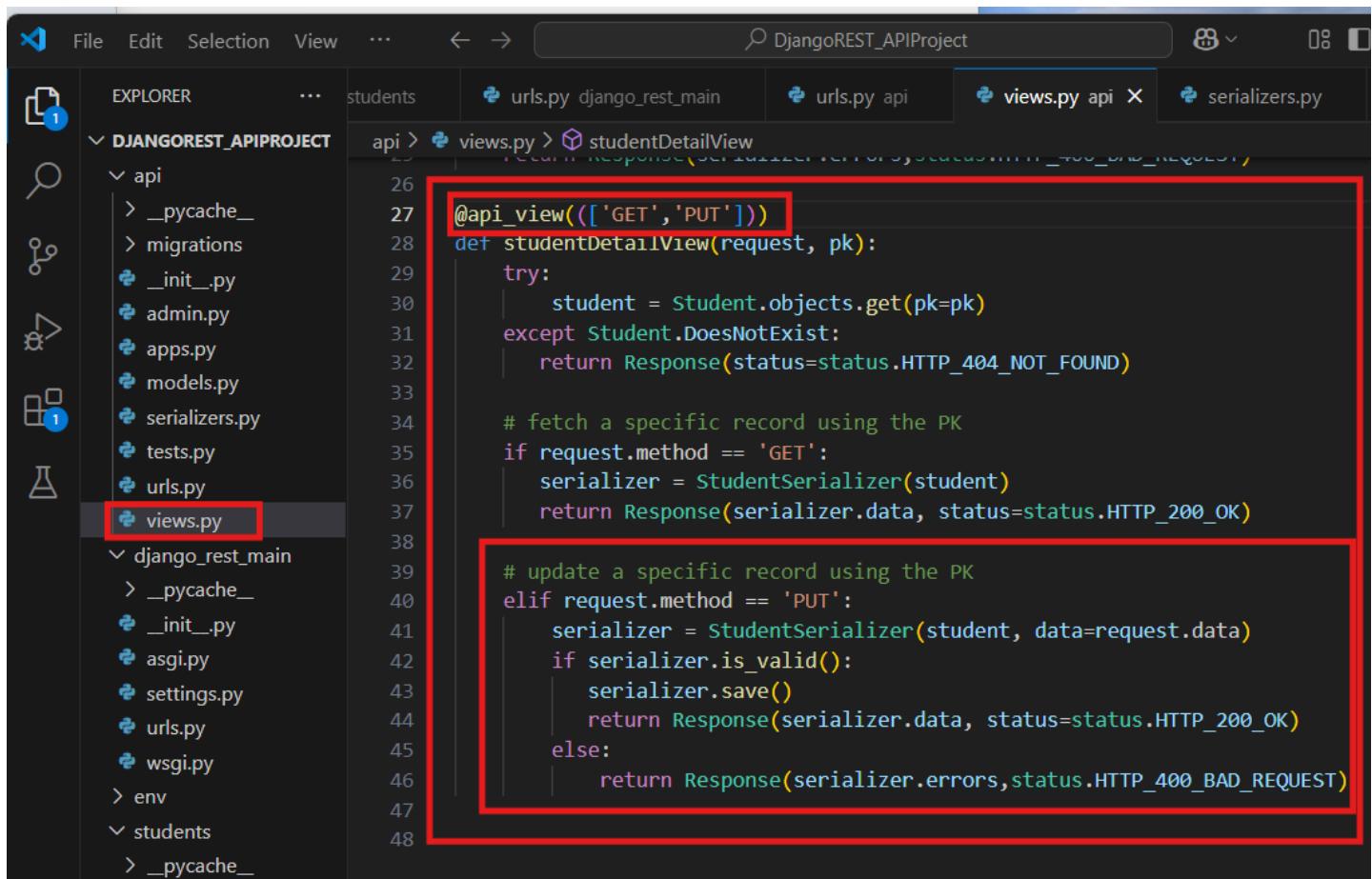
EXPLORER ··· | students | urls.py django_rest_main | urls.py api | views.py api X | serializers.py

∨ DJANGOREST_APIPROJECT

api > views.py > studentDetailView

∨ api
  > __pycache__
  > migrations
  __init__.py
  admin.py
  apps.py
  models.py
  serializers.py
  tests.py
  urls.py
  views.py
∨ django_rest_main
  > __pycache__
  __init__.py
  asgi.py
  settings.py
  urls.py
  wsgi.py
> env
∨ students
  > __pycache__

```python
26
27  @api_view((['GET','PUT']))
28  def studentDetailView(request, pk):
29      try:
30          student = Student.objects.get(pk=pk)
31      except Student.DoesNotExist:
32          return Response(status=status.HTTP_404_NOT_FOUND)
33
34      # fetch a specific record using the PK
35      if request.method == 'GET':
36          serializer = StudentSerializer(student)
37          return Response(serializer.data, status=status.HTTP_200_OK)
38
39      # update a specific record using the PK
40      elif request.method == 'PUT':
41          serializer = StudentSerializer(student, data=request.data)
42          if serializer.is_valid():
43              serializer.save()
44              return Response(serializer.data, status=status.HTTP_200_OK)
45          else:
46              return Response(serializer.errors,status.HTTP_400_BAD_REQUEST)
47
48
```

Updating the record:
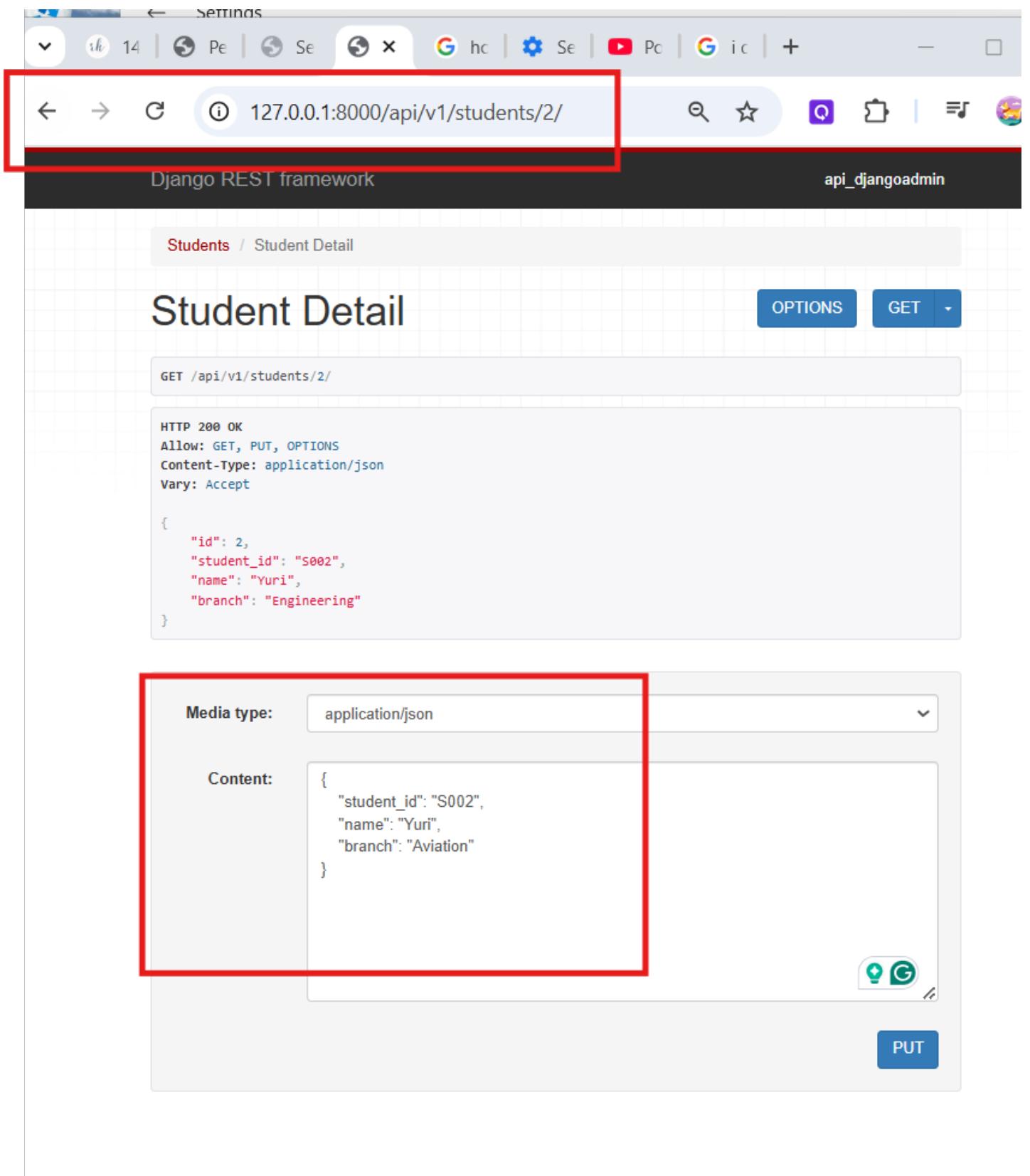
14 | Pe | Se | × | hc | Se | Pc | i c | +

127.0.0.1:8000/api/v1/students/2/

Django REST framework   api_djangoadmin

Students / Student Detail

Student Detail

OPTIONS   GET

GET /api/v1/students/2/

```
HTTP 200 OK
Allow: GET, PUT, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "id": 2,
    "student_id": "S002",
    "name": "Yuri",
    "branch": "Engineering"
}
```

Media type: application/json

Content:
```
{
    "student_id": "S002",
    "name": "Yuri",
    "branch": "Aviation"
}
```

PUT

6. Using the POSTMAN , use the PUT method with the primary record number:

7. To delete a record, use the DELETE method. Update the VIEWS.PY:


```python
25          return Response(serializer.errors,status.HTTP_400_BAD_REQUEST)
26
27  @api_view(('['GET','PUT','DELETE']))
28  def studentDetailView(request, pk):
29      try:
30          student = Student.objects.get(pk=pk)
31      except Student.DoesNotExist:
32          return Response(status=status.HTTP_404_NOT_FOUND)
33
34      # fetch a specific record using the PK
35      if request.method == 'GET':
36          serializer = StudentSerializer(student)
37          return Response(serializer.data, status=status.HTTP_200_OK)
38
39      # update a specific record using the PK
40      elif request.method == 'PUT':
41          serializer = StudentSerializer(student, data=request.data)
42          if serializer.is_valid():
43              serializer.save()
44              return Response(serializer.data, status=status.HTTP_200_OK)
45          else:
46              return Response(serializer.errors,status.HTTP_400_BAD_REQUEST)
47
48      # delete a specific record using the PK
49      elif request.method == 'DELETE':
50          student.delete()
51          return Response(status=status.HTTP_204_NO_CONTENT)
```
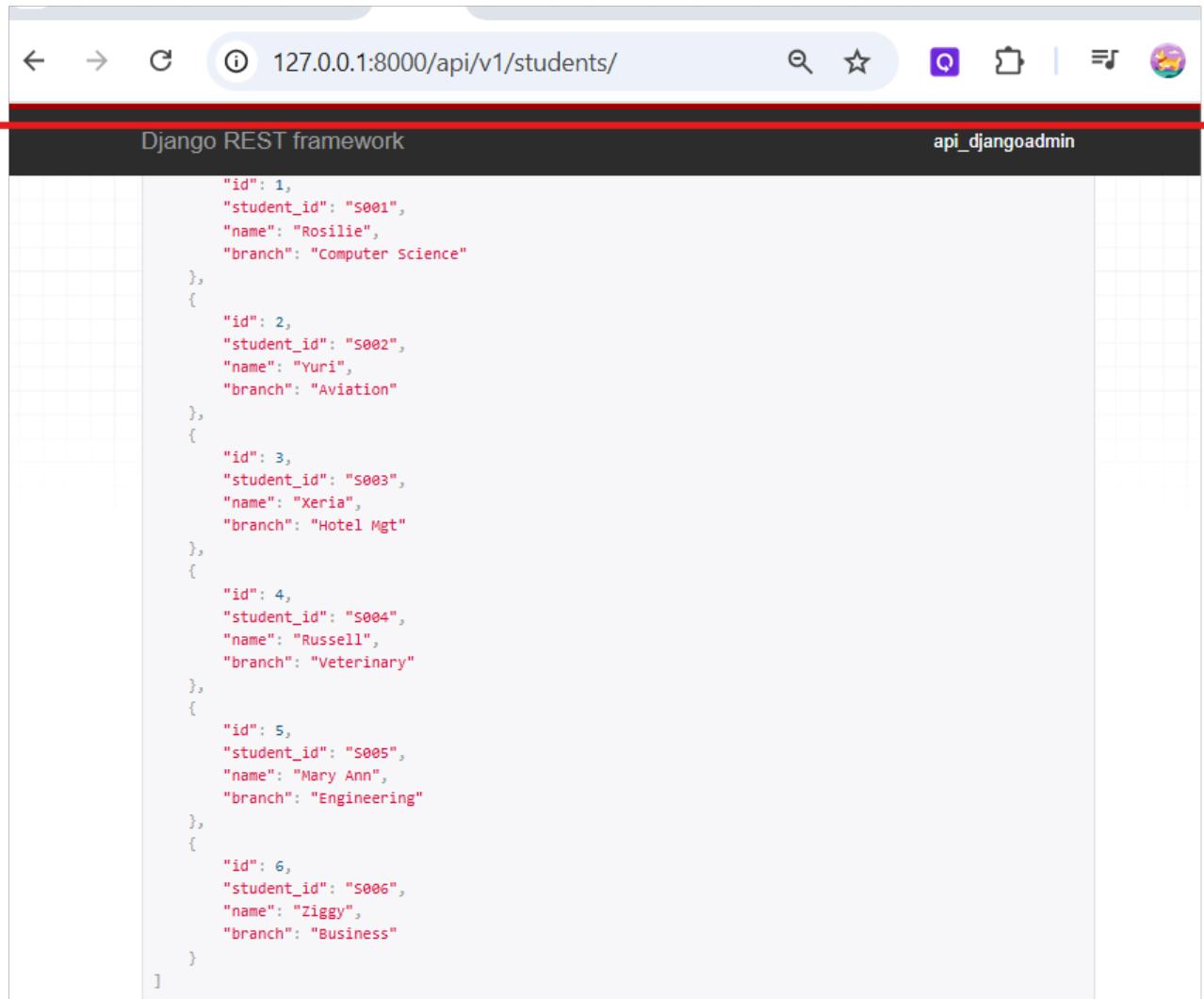
```
        "id": 1,
        "student_id": "S001",
        "name": "Rosilie",
        "branch": "Computer Science"
    },
    {
        "id": 2,
        "student_id": "S002",
        "name": "Yuri",
        "branch": "Aviation"
    },
    {
        "id": 3,
        "student_id": "S003",
        "name": "Xeria",
        "branch": "Hotel Mgt"
    },
    {
        "id": 4,
        "student_id": "S004",
        "name": "Russell",
        "branch": "Veterinary"
    },
    {
        "id": 5,
        "student_id": "S005",
        "name": "Mary Ann",
        "branch": "Engineering"
    },
    {
        "id": 6,
        "student_id": "S006",
        "name": "Ziggy",
        "branch": "Business"
    }
]
```
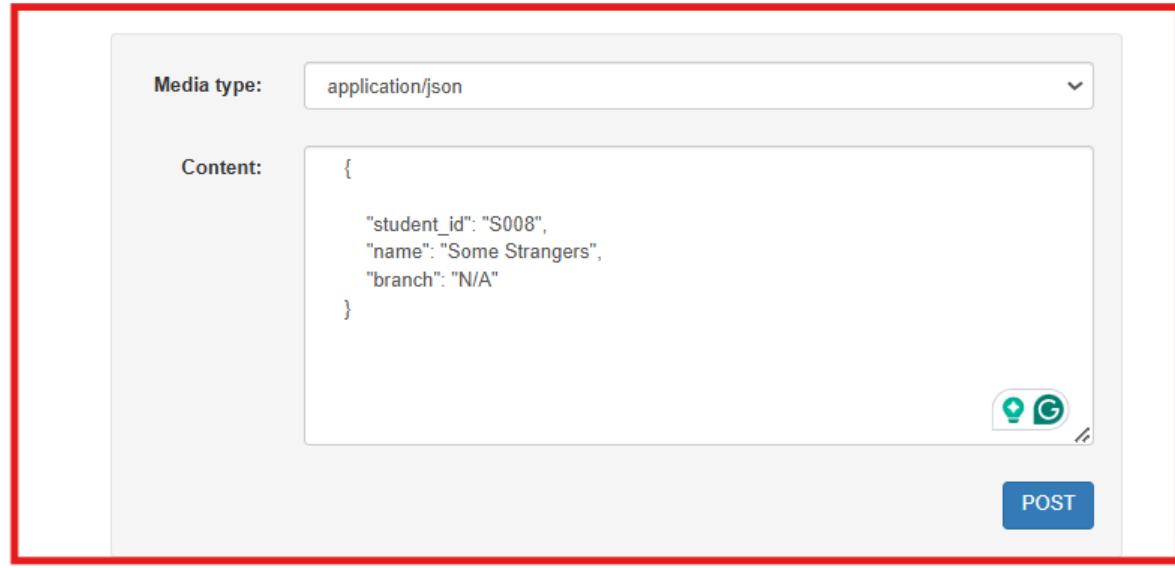
Media type: application/json

Content:
```
{

    "student_id": "S008",
    "name": "Some Strangers",
    "branch": "N/A"
}
```

POST

When you reload your page, you will see the DELETE button:

← → ↻ ⓘ 127.0.0.1:8000/api/v1/students/7/

Django REST framework   api_djangoadmin
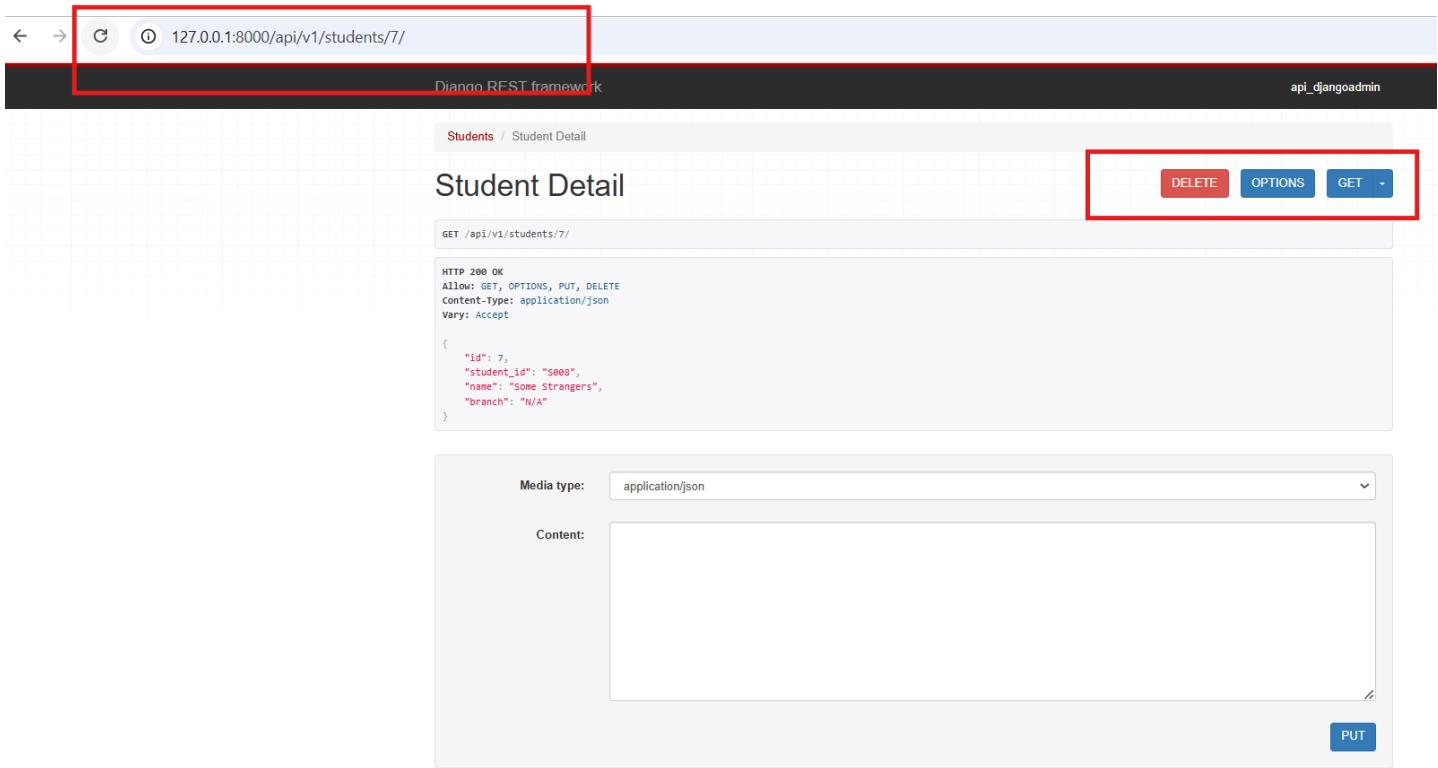
Students / Student Detail

Student Detail   DELETE   OPTIONS   GET ▾

GET /api/v1/students/7/

```
HTTP 200 OK
Allow: GET, OPTIONS, PUT, DELETE
Content-Type: application/json
Vary: Accept

{
    "id": 7,
    "student_id": "S008",
    "name": "Some Strangers",
    "branch": "N/A"
}
```

Media type: application/json   ▾

Content:

PUT

8. Now, reload your page and locate that deleted record, it should return HTTP error message:

→ ⓘ 127.0.0.1:8000/api/v1/students/7/

Django REST framework   api_djangoadmin

Students / Student Detail
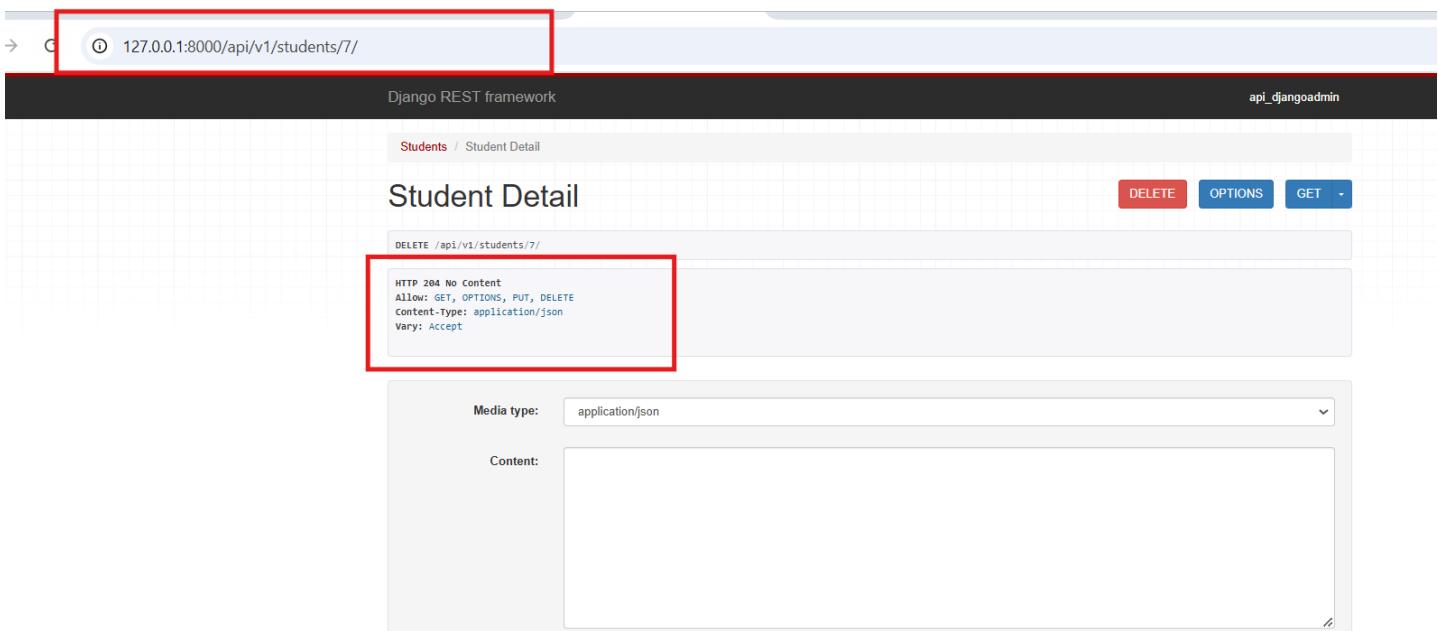
## Student Detail   DELETE   OPTIONS   GET ▾

DELETE /api/v1/students/7/

```
HTTP 204 No Content
Allow: GET, OPTIONS, PUT, DELETE
Content-Type: application/json
Vary: Accept
```

Media type: application/json   ▾

Content:

9. Using the POSTMAN, use the delete the method. Assume you have added several records. Delete these unnecessary records.

When you use the DELETE method and click SEND, the output should be.

10.