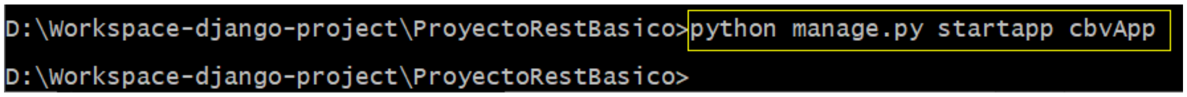
Laboratorio API REST

Nombre: Teresa Jesús Tapia Soto

Utilizar el Proyecto ProyectoRestBasico

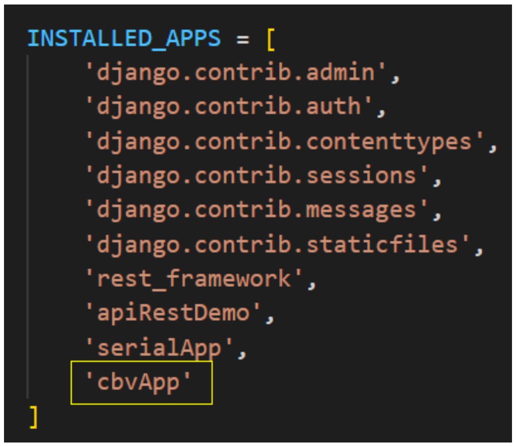
INSTRUCCIONES DEL PASO A PASO: **Ejemplo 4: Vistas basadas en clases….**

1. Abrir CMD
2. Crear aplicación cbvApp -→ (cbv hace referencia a clases basadas en vistas)



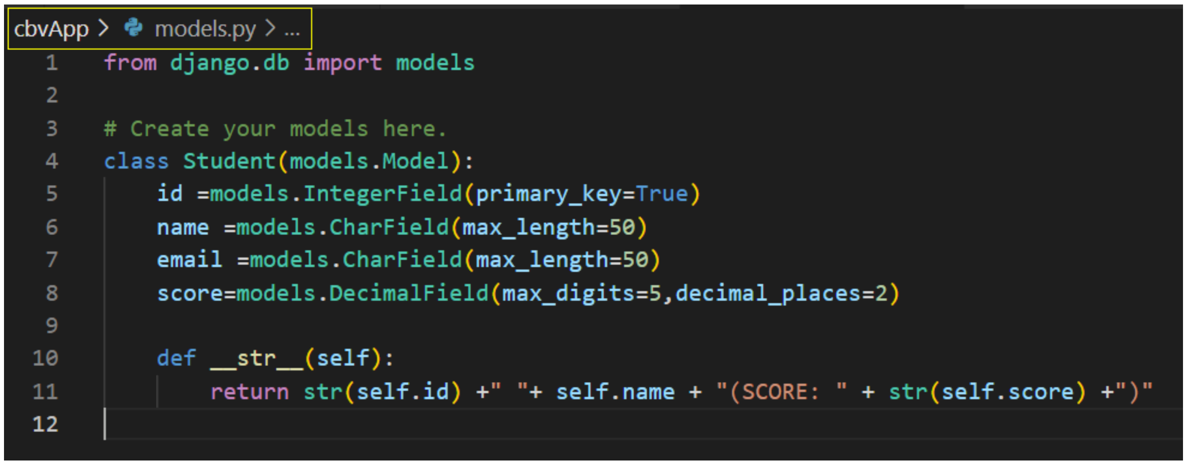
```
D:\Workspace-django-project\ProyectoRestBasico>python manage.py startapp cbvApp

D:\Workspace-django-project\ProyectoRestBasico>
```

3. Configurar **settings.py** agregando la aplicación cbvApp



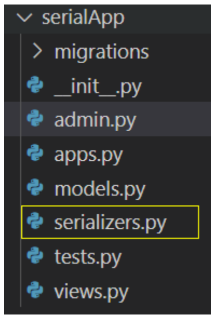
```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'rest_framework',
    'apiRestDemo',
    'serialApp',
    'cbvApp'
]
```

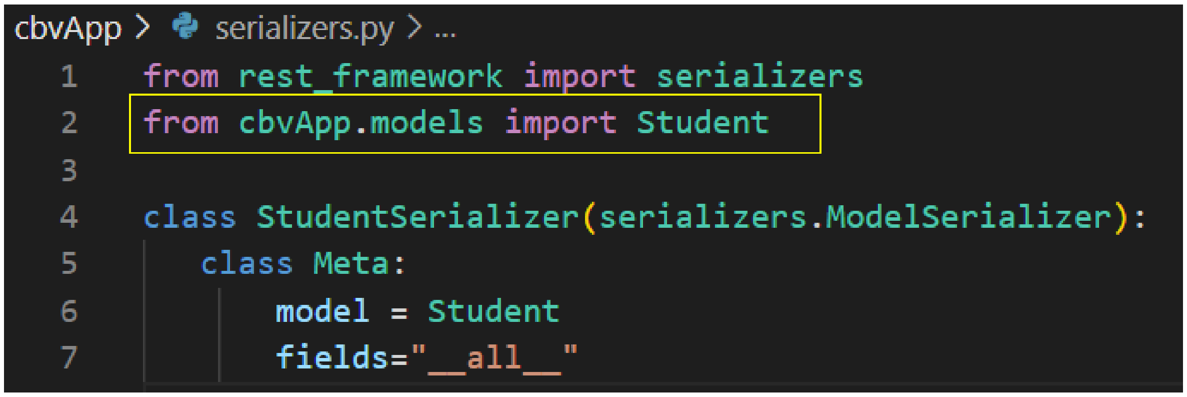
4. Creamos el modelo (en model.py) en la aplicación cbvApp



```
from django.db import models

# Create your models here.
class Student(models.Model):
    id =models.IntegerField(primary_key=True)
    name =models.CharField(max_length=50)
    email =models.CharField(max_length=50)
    score=models.DecimalField(max_digits=5,decimal_places=2)

    def __str__(self):
        return str(self.id) +" "+ self.name + "(SCORE: " + str(self.score) +")"
```

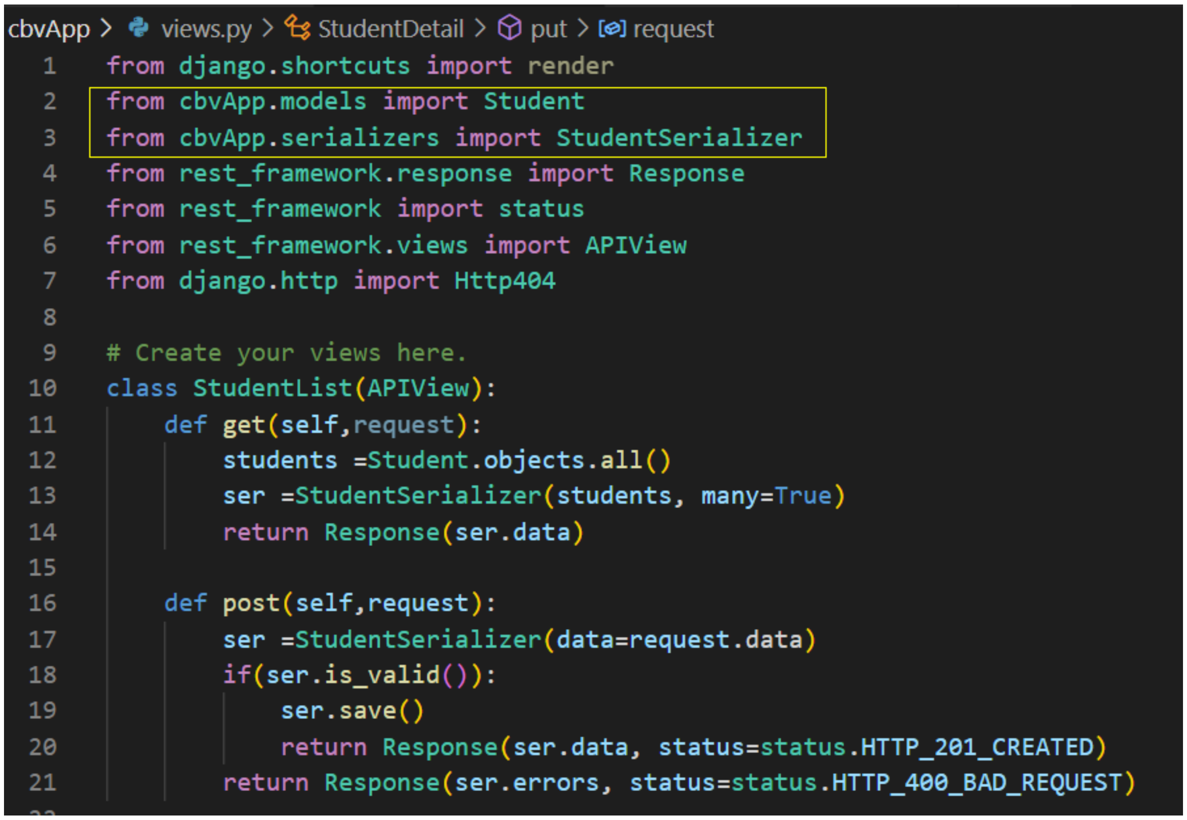
5. Creamos el archivo serializers (serializers.py) en la aplicación cbvApp





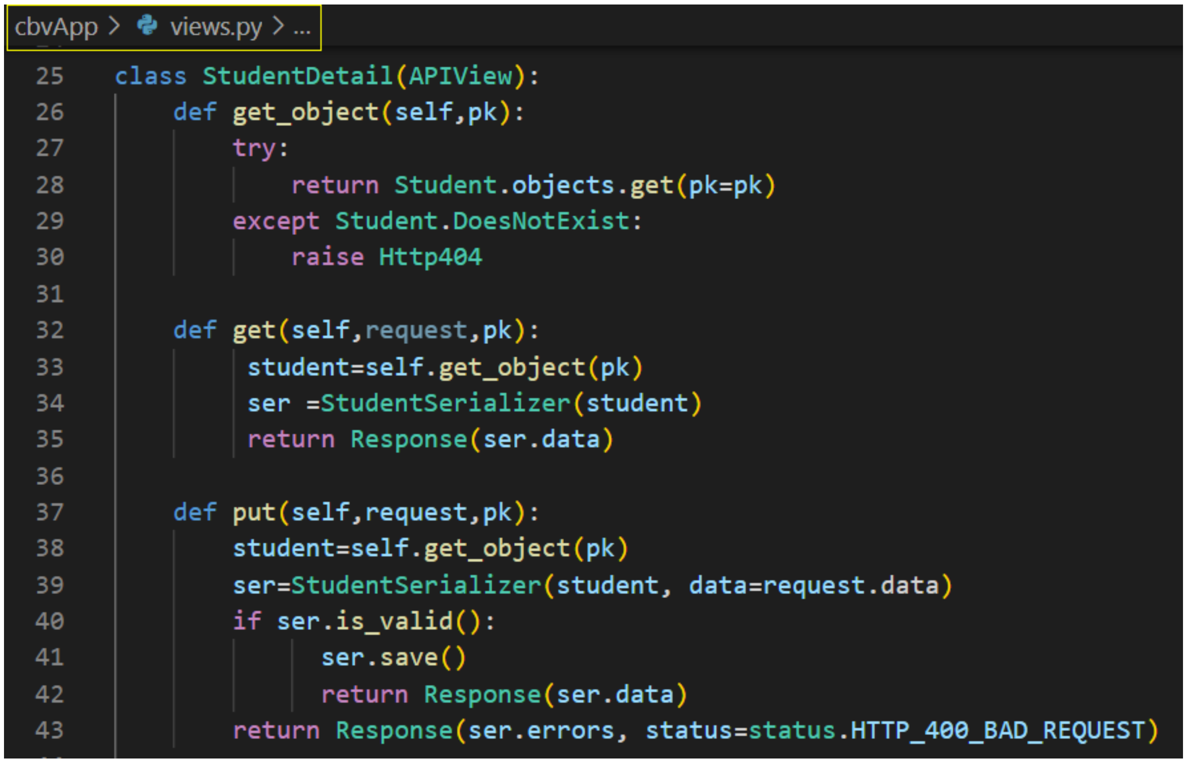
```python
from rest_framework import serializers
from cbvApp.models import Student

class StudentSerializer(serializers.ModelSerializer):
    class Meta:
        model = Student
        fields="__all__"
```

6. Creamos la vista class StudentList para implementar los métodos GET y POST

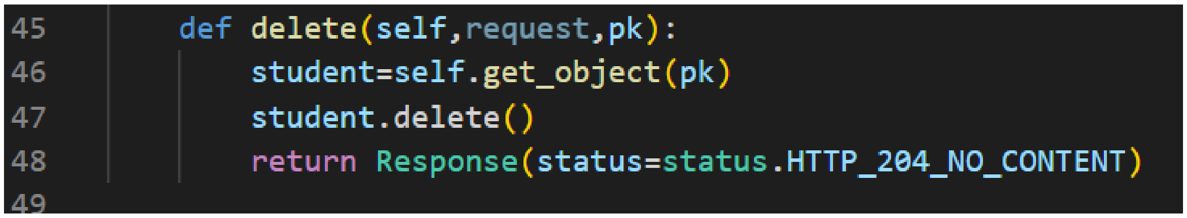


```python
from django.shortcuts import render
from cbvApp.models import Student
from cbvApp.serializers import StudentSerializer
from rest_framework.response import Response
from rest_framework import status
from rest_framework.views import APIView
from django.http import Http404

# Create your views here.
class StudentList(APIView):
    def get(self,request):
        students =Student.objects.all()
        ser =StudentSerializer(students, many=True)
        return Response(ser.data)

    def post(self,request):
        ser =StudentSerializer(data=request.data)
        if(ser.is_valid()):
            ser.save()
            return Response(ser.data, status=status.HTTP_201_CREATED)
        return Response(ser.errors, status=status.HTTP_400_BAD_REQUEST)
```

7. Creamos la vista class StudentDetail para implementar los métodos GET, PUT y DELETE

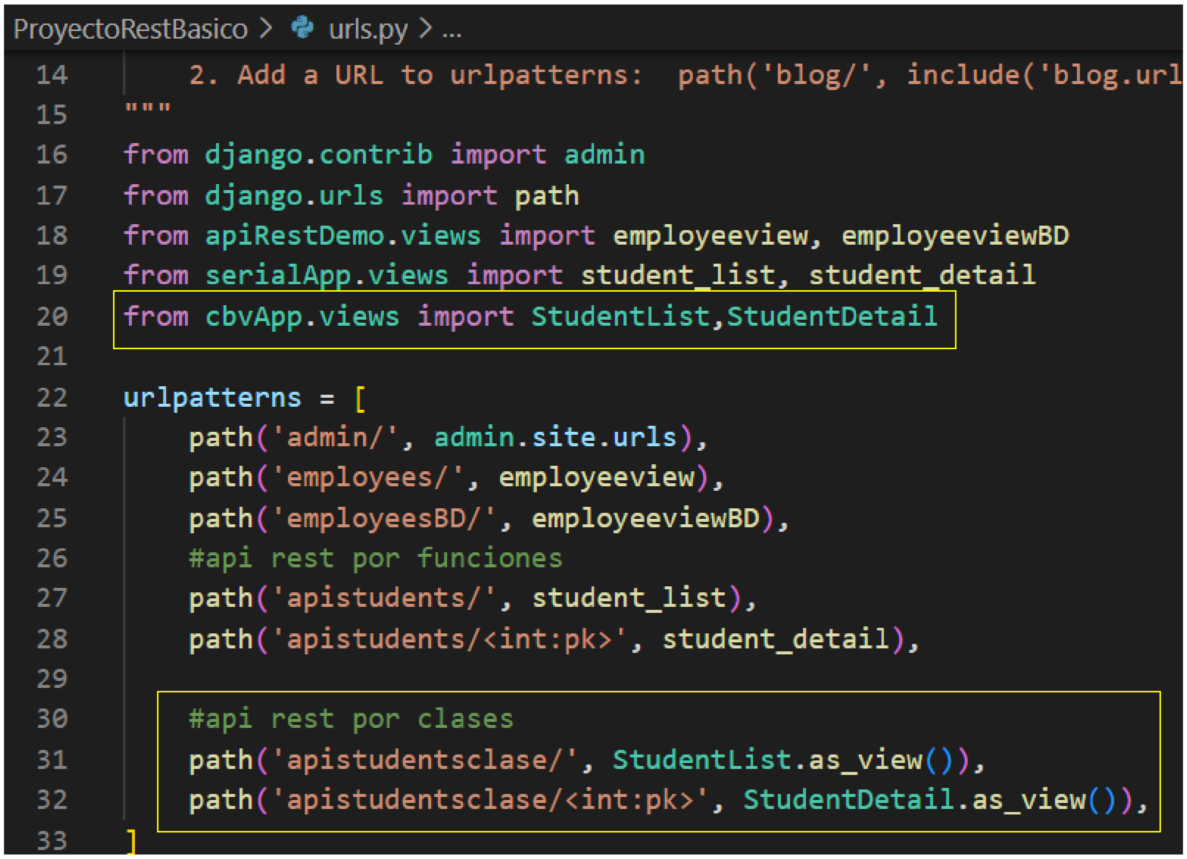


```python
class StudentDetail(APIView):
    def get_object(self,pk):
        try:
            return Student.objects.get(pk=pk)
        except Student.DoesNotExist:
            raise Http404

    def get(self,request,pk):
         student=self.get_object(pk)
         ser =StudentSerializer(student)
         return Response(ser.data)

    def put(self,request,pk):
        student=self.get_object(pk)
        ser=StudentSerializer(student, data=request.data)
        if ser.is_valid():
              ser.save()
              return Response(ser.data)
        return Response(ser.errors, status=status.HTTP_400_BAD_REQUEST)
```



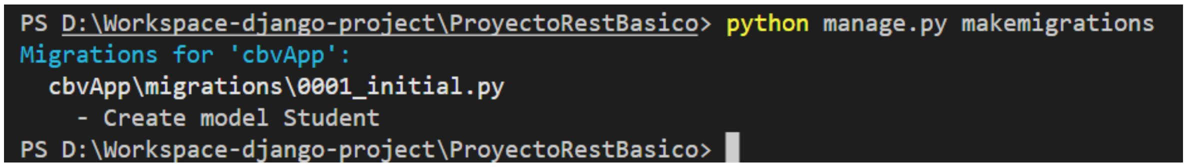
```python
    def delete(self,request,pk):
        student=self.get_object(pk)
        student.delete()
        return Response(status=status.HTTP_204_NO_CONTENT)
```

8. Configurar las rutas en el archivo urls.py



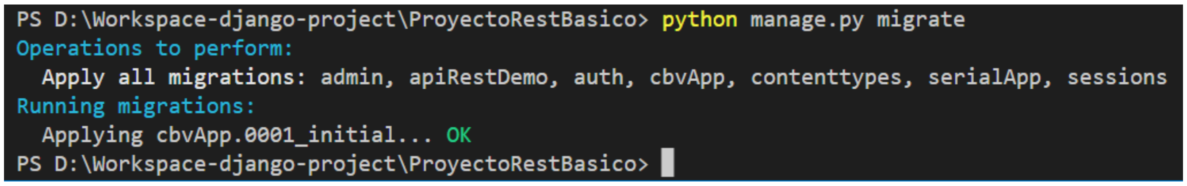
```
ProyectoRestBasico > urls.py > ...
    2. Add a URL to urlpatterns:  path('blog/', include('blog.url
"""
from django.contrib import admin
from django.urls import path
from apiRestDemo.views import employeeview, employeeviewBD
from serialApp.views import student_list, student_detail
from cbvApp.views import StudentList,StudentDetail

urlpatterns = [
    path('admin/', admin.site.urls),
    path('employees/', employeeview),
    path('employeesBD/', employeeviewBD),
    #api rest por funciones
    path('apistudents/', student_list),
    path('apistudents/<int:pk>', student_detail),

    #api rest por clases
    path('apistudentsclase/', StudentList.as_view()),
    path('apistudentsclase/<int:pk>', StudentDetail.as_view()),
]
```

9. Realizar las migraciones ejecutando la siguiente instrucción:
python manage.py makemigrations



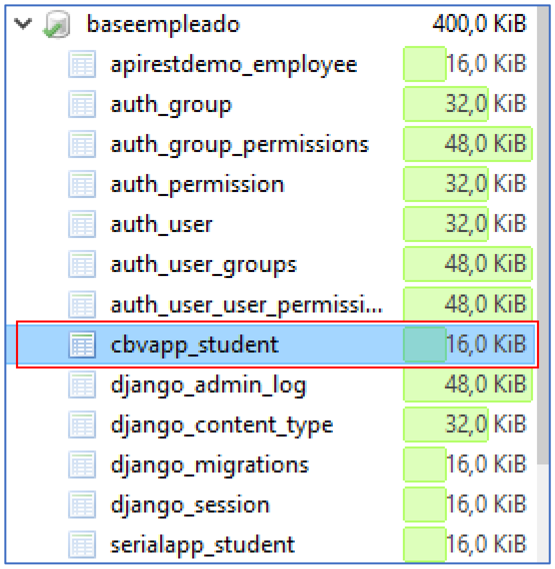
```
PS D:\Workspace-django-project\ProyectoRestBasico> python manage.py makemigrations
Migrations for 'cbvApp':
  cbvApp\migrations\0001_initial.py
    - Create model Student
PS D:\Workspace-django-project\ProyectoRestBasico>
```

10. Ejecutar las migraciones ejecutando la siguiente instrucción:
python manage.py migrate

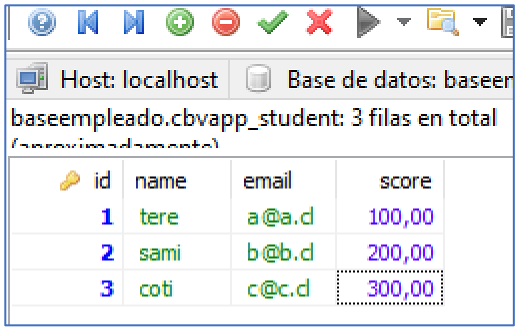


```
PS D:\Workspace-django-project\ProyectoRestBasico> python manage.py migrate
Operations to perform:
  Apply all migrations: admin, apiRestDemo, auth, cbvApp, contenttypes, serialApp, sessions
Running migrations:
  Applying cbvApp.0001_initial... OK
PS D:\Workspace-django-project\ProyectoRestBasico>
```

11. Ingresar datos a la base de datos…manualmente



baseempleado	400,0 KiB
apirestdemo_employee	16,0 KiB
auth_group	32,0 KiB
auth_group_permissions	48,0 KiB
auth_permission	32,0 KiB
auth_user	32,0 KiB
auth_user_groups	48,0 KiB
auth_user_user_permissi...	48,0 KiB
cbvapp_student	16,0 KiB
django_admin_log	48,0 KiB
django_content_type	32,0 KiB
django_migrations	16,0 KiB
django_session	16,0 KiB
serialapp_student	16,0 KiB

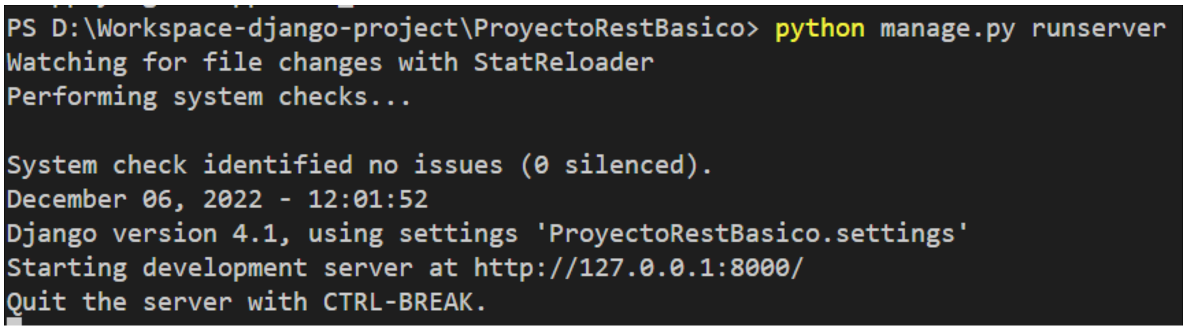


Host: localhost   Base de datos: baseer

baseempleado.cbvapp_student: 3 filas en total

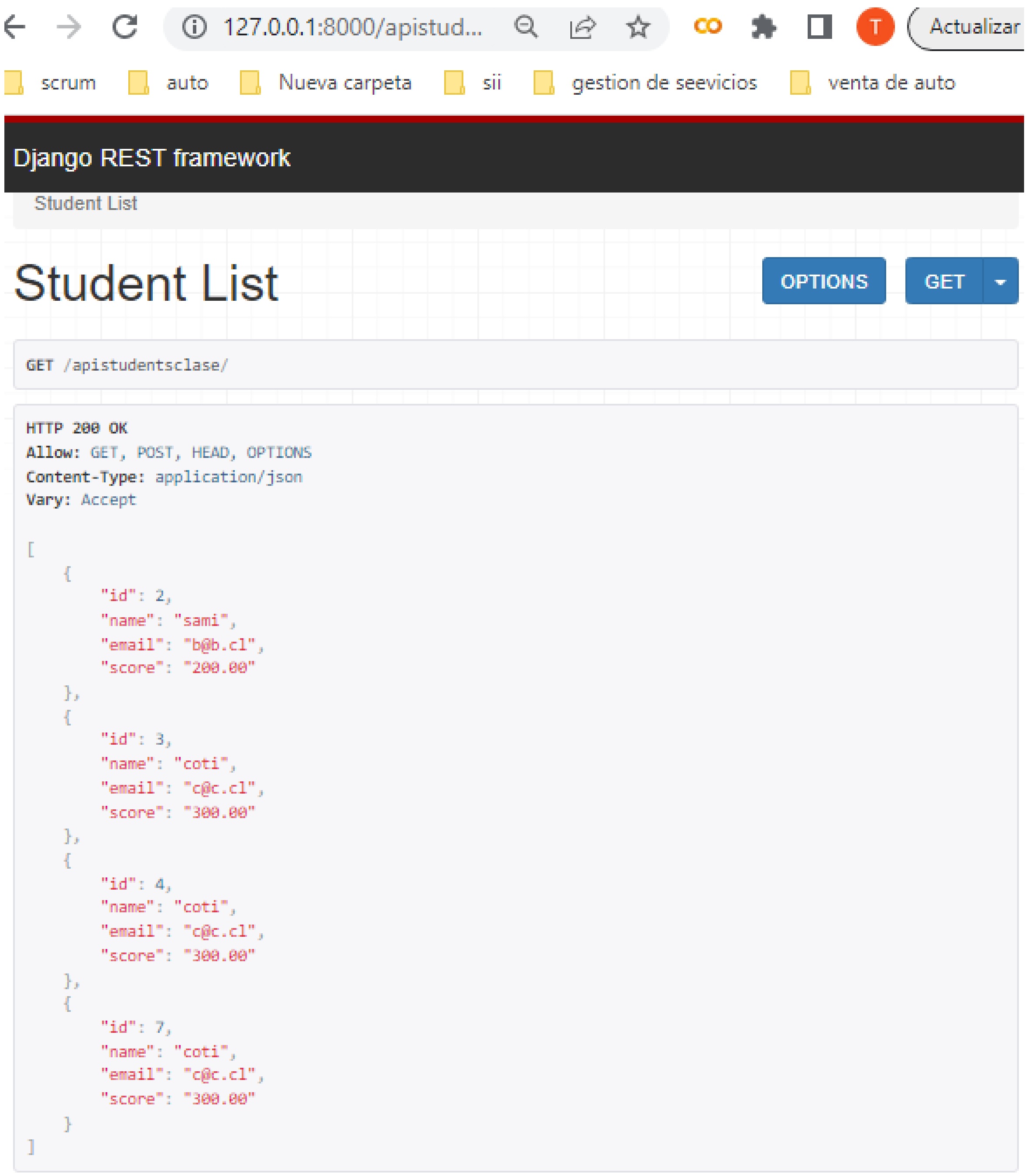
id	name	email	score
1	tere	a@a.cl	100,00
2	sami	b@b.cl	200,00
3	coti	c@c.cl	300,00

12. Subir el servidor



```
PS D:\Workspace-django-project\ProyectoRestBasico> python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
December 06, 2022 - 12:01:52
Django version 4.1, using settings 'ProyectoRestBasico.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

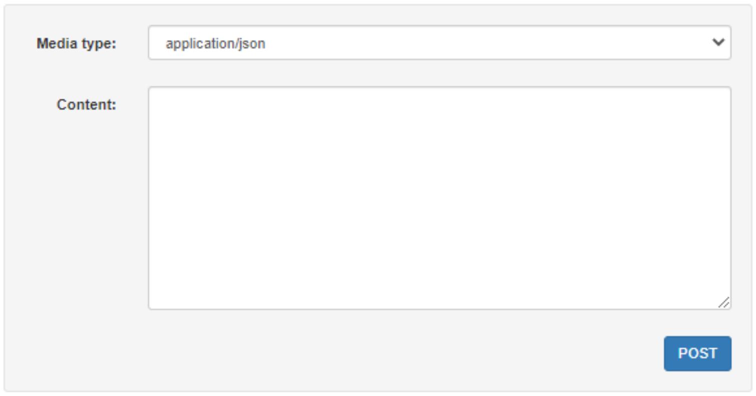
13. Acceder a la vista creada. Probar obtener el listado completo y agregar un registro con POST
http://127.0.0.1:8000/apistudentsclase/



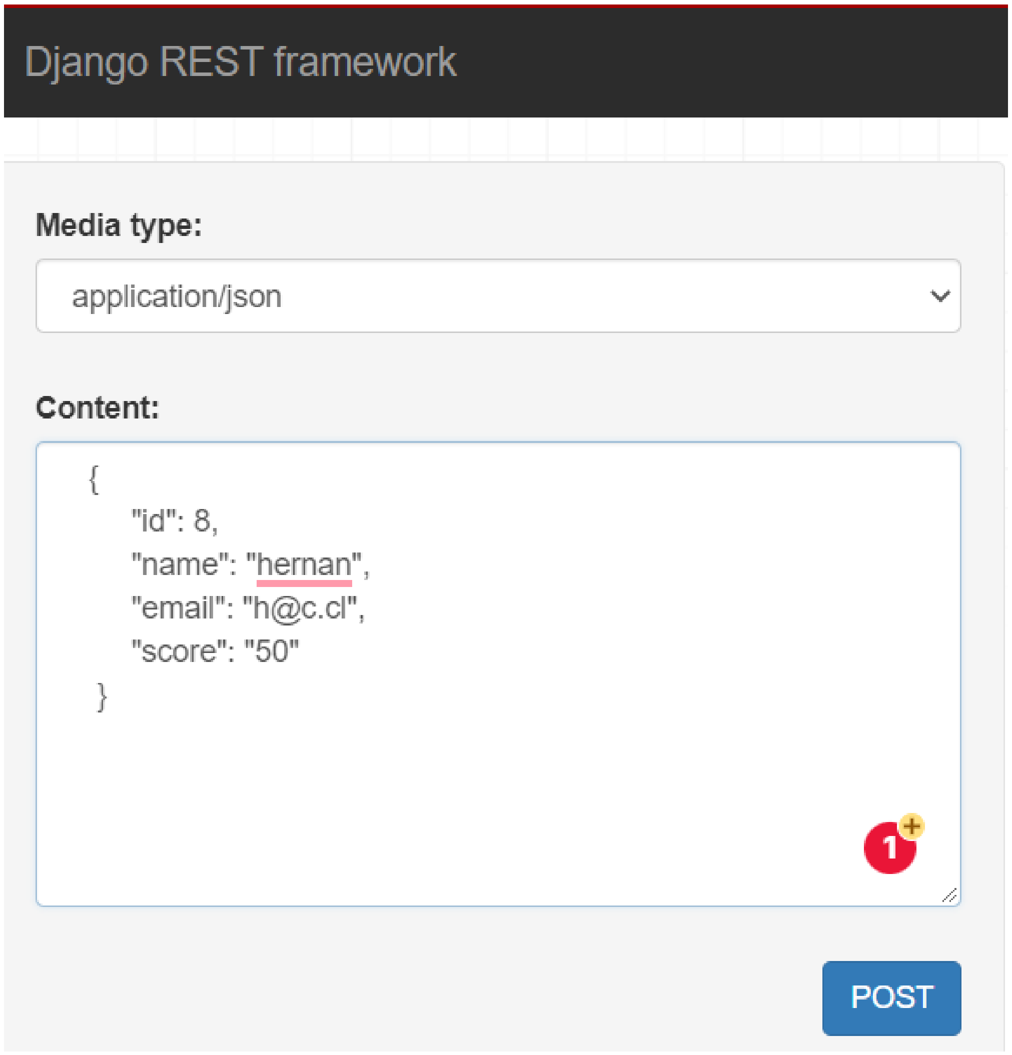
Student List

```
GET /apistudentsclase/
```

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

[
    {
        "id": 2,
        "name": "sami",
        "email": "b@b.cl",
        "score": "200.00"
    },
    {
        "id": 3,
        "name": "coti",
        "email": "c@c.cl",
        "score": "300.00"
    },
    {
        "id": 4,
        "name": "coti",
        "email": "c@c.cl",
        "score": "300.00"
    },
    {
        "id": 7,
        "name": "coti",
        "email": "c@c.cl",
        "score": "300.00"
    }
]
```

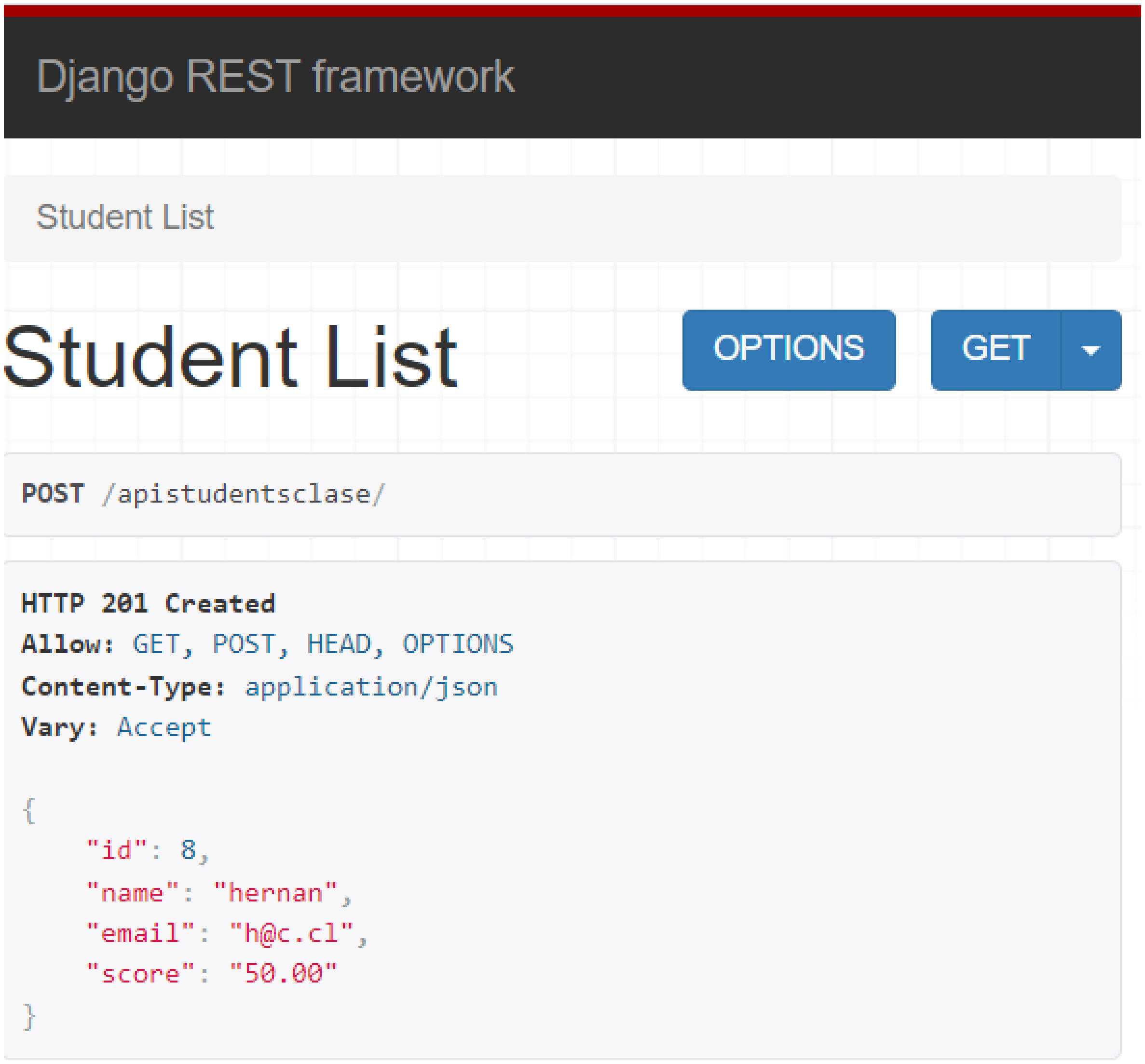


Agregar un registro … ingresar la siguiente data



```
{
    "id": 8,
    "name": "hernan",
    "email": "h@c.cl",
    "score": "50"
}
```

El resultado es …

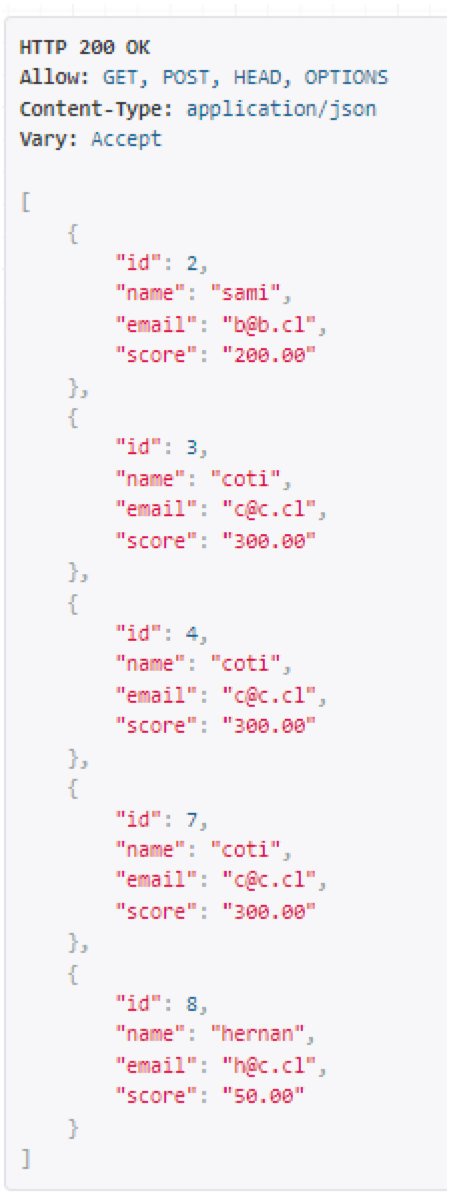


Student List

```
POST /apistudentsclase/
```

```
HTTP 201 Created
Allow: GET, POST, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "id": 8,
    "name": "hernan",
    "email": "h@c.cl",
    "score": "50.00"
}
```

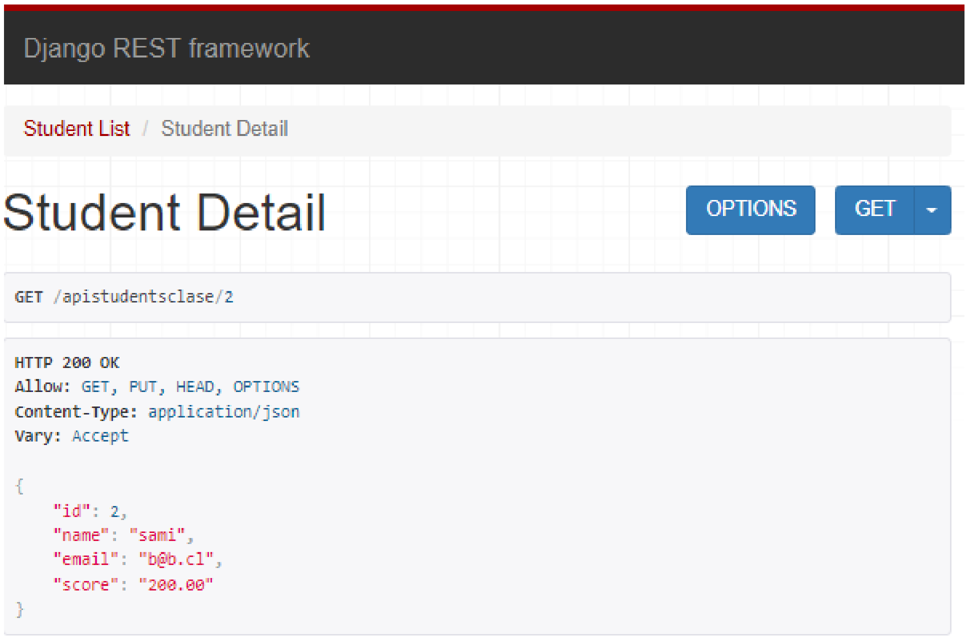
Presionando GET se tiene todo el listado



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

[
    {
        "id": 2,
        "name": "sami",
        "email": "b@b.cl",
        "score": "200.00"
    },
    {
        "id": 3,
        "name": "coti",
        "email": "c@c.cl",
        "score": "300.00"
    },
    {
        "id": 4,
        "name": "coti",
        "email": "c@c.cl",
        "score": "300.00"
    },
    {
        "id": 7,
        "name": "coti",
        "email": "c@c.cl",
        "score": "300.00"
    },
    {
        "id": 8,
        "name": "hernan",
        "email": "h@c.cl",
        "score": "50.00"
    }
]
```

14. Acceder a la vista creada. Probar obtener el registro 2

Para registro existente
http://127.0.0.1:8000/apistudentsclase/2

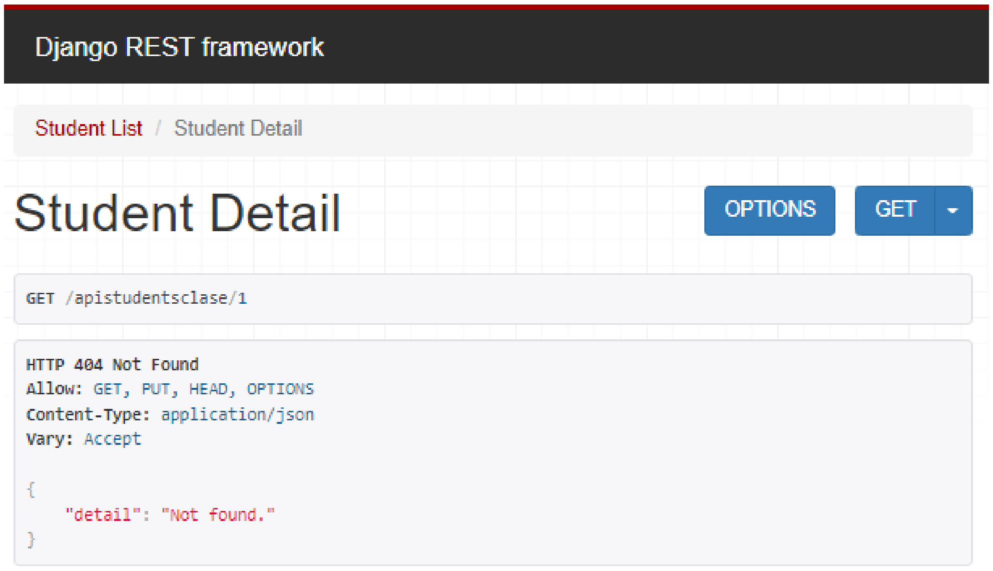


```
GET /apistudentsclase/2
```

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

{
    "id": 2,
    "name": "sami",
    "email": "b@b.cl",
    "score": "200.00"
}
```

Para registro no existente se tiene….

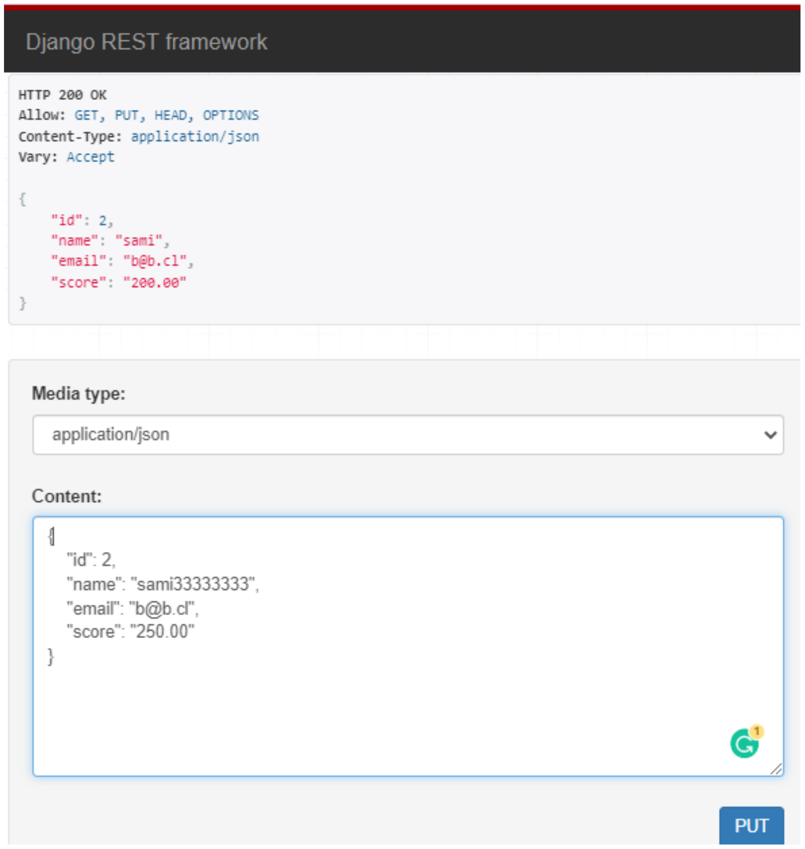
http://127.0.0.1:8000/apistudentsclase/1



```
GET /apistudentsclase/1
```

```
HTTP 404 Not Found
Allow: GET, PUT, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "detail": "Not found."
}
```

Para actualizar un registro y presionando PUT
http://127.0.0.1:8000/apistudentsclase/2



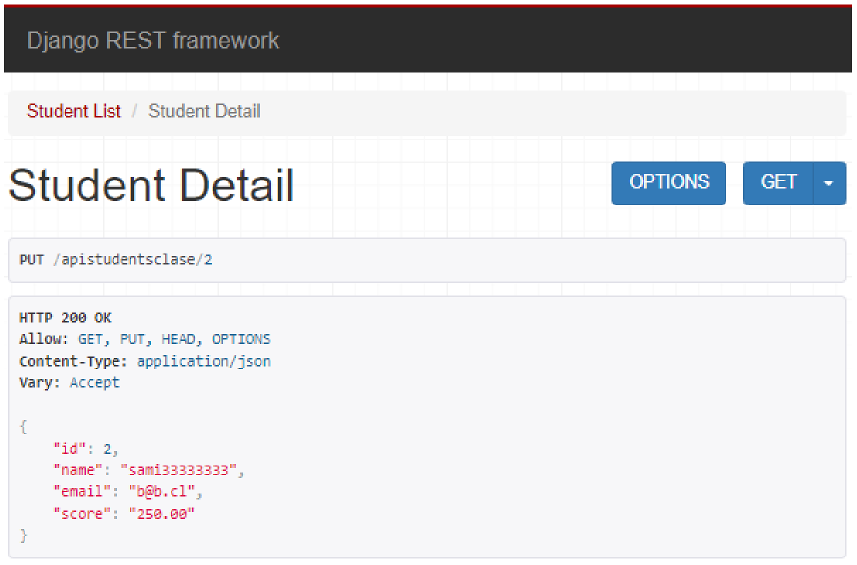
```
HTTP 200 OK
Allow: GET, PUT, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "id": 2,
    "name": "sami",
    "email": "b@b.cl",
    "score": "200.00"
}
```

Media type: application/json

Content:

```
{
    "id": 2,
    "name": "sami33333333",
    "email": "b@b.cl",
    "score": "250.00"
}
```

El resultado es:

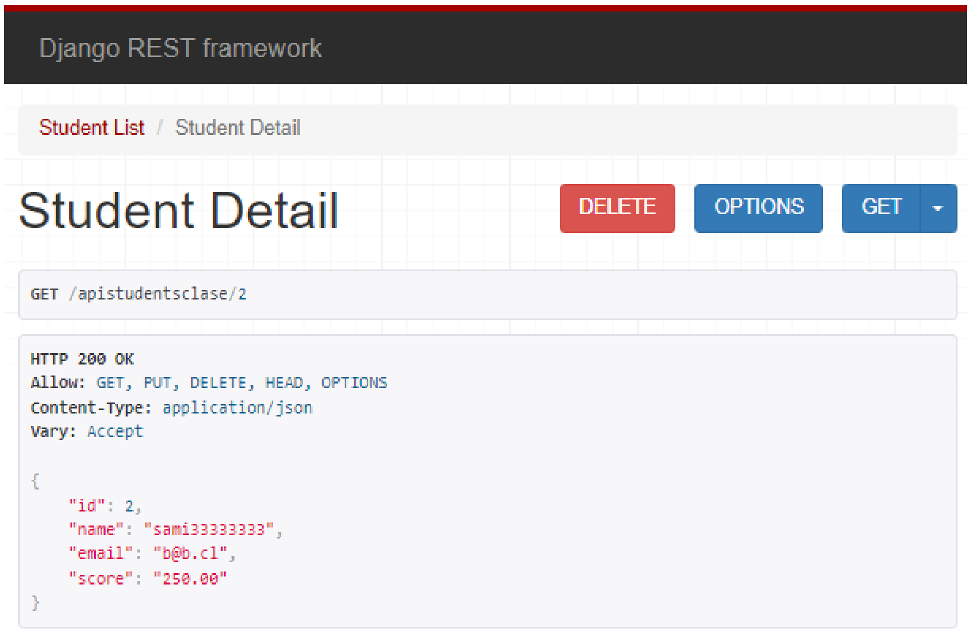


Student Detail

```
PUT /apistudentsclase/2
```

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

{
    "id": 2,
    "name": "sami33333333",
    "email": "b@b.cl",
    "score": "250.00"
}
```

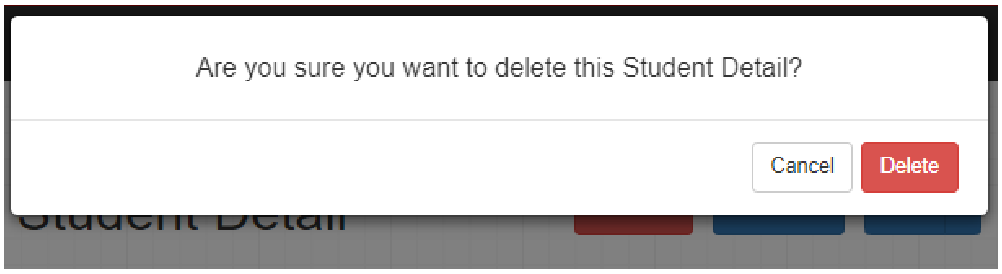
Presionando el botón delete..
http://127.0.0.1:8000/apistudentsclase/2



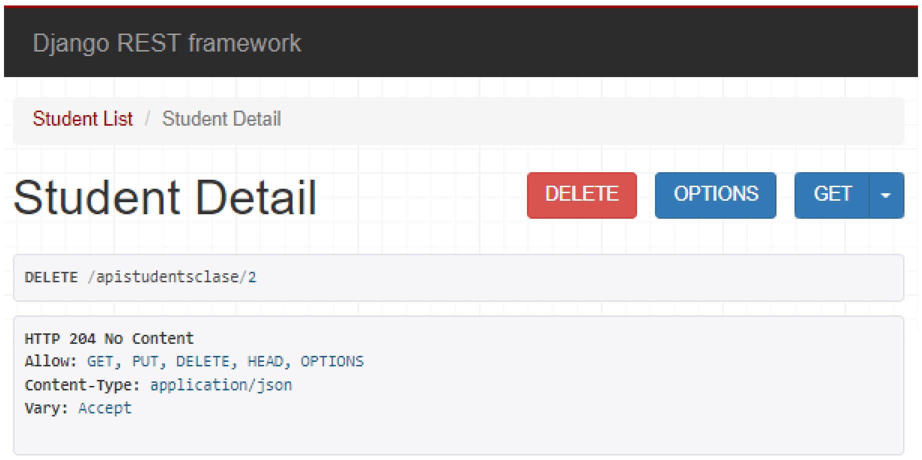
Student Detail

```
GET /apistudentsclase/2
```

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

{
    "id": 2,
    "name": "sami33333333",
    "email": "b@b.cl",
    "score": "250.00"
}
```


Are you sure you want to delete this Student Detail?

El resultado es …

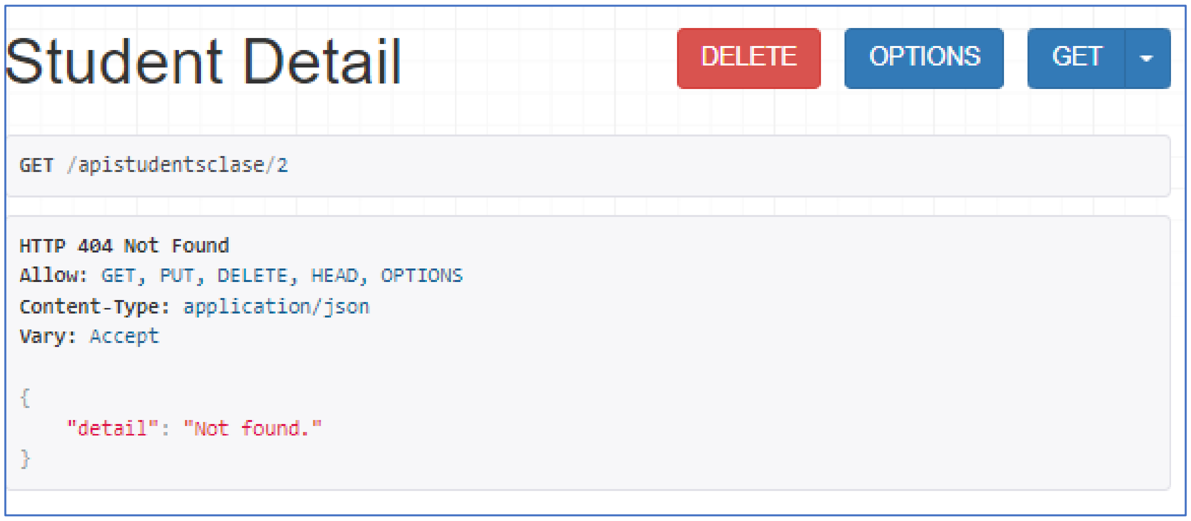


Student Detail

```
DELETE /apistudentsclase/2
```

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

Consultando por el registro 2 se tiene:



```
GET /apistudentsclase/2
```

```
HTTP 404 Not Found
Allow: GET, PUT, DELETE, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "detail": "Not found."
}
```