

FACULTAD DE INGENIERIA EN ELECTRICIDAD Y COMPUTACION
DESARROLLO DE APLICACIONES WEB
I TÉRMINO 2019
TALLER 4

Nombre:	Danny Leonel De La A Yagual
Paralelo:	1
Recurso	

NOTA: TODAS SUS CAPTURAS DEBEN INCLUIR LA HORA DEL SISTEMA

- **Código del servidor productor del RESTful API. Aquí debe incluir la(s) captura(s) de los controladores que respondan a los verbos HTTP: GET, POST, PUT y DELETE.**

post

The screenshot shows a web browser window with the URL `localhost:8000/apilibras/`. The page title is "Libro List" and it's part of the "Django REST framework". The response to a GET request is shown in a light blue box:

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

[
  {
    "url": "http://localhost:8000/apilibras/1/",
    "titulo": "El amanecer de los muertos",
    "anio": 1999,
    "precio": 30.89
  }
]
```

Below the response, there is a form to add a new book. The form has three input fields: "Titulo" (containing "El señor de los anillos"), "Anio" (containing "2000"), and "Precio" (containing "20.89"). There is a "POST" button at the bottom right of the form. The browser's taskbar at the bottom shows the date and time as 10:59 on 14/8/2019.

get

Módulos del curso: DES: x 2019-1T/reporte.docx: x Django: no such table: d: x proyecto-daw/proy_2p... x Libro List - Django REST x Home - Django REST fra x Administrador de Django: x + -

localhost:8000/apilibras/

Django REST framework danry

Libro List

OPTIONS GET

GET /apilibras/

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

```
{
  "url": "http://localhost:8000/apilibras/1/",
  "titulo": "El amanecer de los muertos",
  "anio": 1999,
  "precio": 30.89
},
{
  "url": "http://localhost:8000/apilibras/2/",
  "titulo": "El señor de los anillos",
  "anio": 2000,
  "precio": 20.89
}
}
```

Raw data HTML form

Media type: application/json

Content: {

reporte.docx Mostrar todo

11:00 14/8/2019

Delete y get

Módulos del curso: DES: x 2019-1T/reporte.docx: x Django: no such table: d: x proyecto-daw/proy_2p... x Libro Instance - Django x Home - Django REST fra x Administrador de Django: x + -

localhost:8000/apilibras/1/

Django REST framework danry

Libro Instance

DELETE OPTIONS GET

GET /apilibras/1/

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

```
{
  "url": "http://localhost:8000/apilibras/1/",
  "titulo": "El amanecer de los muertos",
  "anio": 1999,
  "precio": 30.89
}
```

Raw data HTML form

Media type: application/json

Content: {
 "url": "http://localhost:8000/apilibras/1/",
 "titulo": "El amanecer de los muertos",
 "anio": 1999,
 "precio": 30.89
}

reporte.docx Mostrar todo

11:01 14/8/2019

```
C:\Windows\system32\cmd.exe - python manage.py runserver

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.

C:\Users\ClitControl\Documents\taller4_DAW\mysite>python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
Django version 2.2.4, using settings 'mysite.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
[14/Aug/2019 10:55:36] "GET /admin/ HTTP/1.1" 302 0
[14/Aug/2019 10:55:36] "GET /admin/login/?next=/admin/ HTTP/1.1" 200 1819
[14/Aug/2019 10:55:41] "POST /admin/login/?next=/admin/ HTTP/1.1" 302 0
[14/Aug/2019 10:55:41] "GET /admin/ HTTP/1.1" 200 3042
[14/Aug/2019 10:55:41] "GET /static/admin/css/dashboard.css HTTP/1.1" 304 0
[14/Aug/2019 10:55:41] "GET /static/admin/img/icon-changelink.svg HTTP/1.1" 304 0
[14/Aug/2019 10:55:41] "GET /static/admin/img/icon-addlink.svg HTTP/1.1" 304 0
[14/Aug/2019 10:55:41] "GET /static/admin/fonts/Roboto-Bold-webFont.woff HTTP/1.1" 200 86184
Not Found: /
[14/Aug/2019 10:55:46] "GET / HTTP/1.1" 404 2133
[14/Aug/2019 10:55:46] "GET /admin/ HTTP/1.1" 200 3042
[14/Aug/2019 10:55:50] "GET /admin/login/?next=/admin/ HTTP/1.1" 302 0
[14/Aug/2019 10:55:50] "GET /admin/ HTTP/1.1" 200 3042
[14/Aug/2019 10:55:54] "GET /api HTTP/1.1" 200 5709
[14/Aug/2019 10:56:00] "GET /apilibros/ HTTP/1.1" 200 9215
[14/Aug/2019 10:57:10] "GET /apilibros/ HTTP/1.1" 200 9215
[14/Aug/2019 10:57:12] "GET /apilibros/ HTTP/1.1" 200 9215
[14/Aug/2019 10:57:13] "OPTIONS /apilibros/ HTTP/1.1" 200 10669
[14/Aug/2019 10:57:47] "POST /apilibros/ HTTP/1.1" 201 9649
[14/Aug/2019 10:57:51] "GET /apilibros/ HTTP/1.1" 200 9497
[14/Aug/2019 10:57:57] "GET /apilibros/1/ HTTP/1.1" 200 10895
[14/Aug/2019 10:58:54] "PUT /apilibros/1/ HTTP/1.1" 200 10901
[14/Aug/2019 10:59:01] "GET /apilibros/1/?format=api HTTP/1.1" 200 11022
[14/Aug/2019 10:59:06] "GET /apilibros/1/ HTTP/1.1" 200 10901
[14/Aug/2019 10:59:12] "GET /apilibros HTTP/1.1" 301 0
[14/Aug/2019 10:59:12] "GET /apilibros/ HTTP/1.1" 200 9499
[14/Aug/2019 10:59:43] "POST /apilibros/ HTTP/1.1" 201 9649
[14/Aug/2019 10:59:48] "GET /apilibros/2/ HTTP/1.1" 200 10895
[14/Aug/2019 10:59:52] "OPTIONS /apilibros/2/ HTTP/1.1" 200 11971
[14/Aug/2019 10:59:59] "GET /apilibros/2/ HTTP/1.1" 200 10895
[14/Aug/2019 11:00:08] "GET /apilibros/ HTTP/1.1" 200 9781
[14/Aug/2019 11:01:01] "GET /apilibros/1/ HTTP/1.1" 200 10901
```

taller4_DAW [C:\Users\ClitControl\Documents\taller4_DAW] - ..\mysite\api\permissions.py [taller4_DAW] - PyCharm

```
from rest_framework import permissions
from rest_framework.permissions import SAFE_METHODS

class IsAdminUserOrReadOnly(permissions.IsAdminUser):
    def has_permission(self, request, view):
        is_admin = super().has_permission(request, view)
        IsAdminUserOrReadOnly(request, view)
        self.has_permission(request, view)
        # Pythond: is_admin = super().has_permission(request, view)
        return request.method in SAFE_METHODS or is_admin

class IsInvolvedOnTrackingRequest(permissions.BasePermission):
    """
    Custom permission to only allow people involved in a TrackingRequest to view it.
    """
    def has_object_permission(self, request, view, obj):
        return request.user in (obj.source, obj.target)
```

Cannot Run Git
File not found: git.exe
Download Configure...

Cannot Run Git: File not found: git.exe // Download Configure... (45 minutes ago)

taller4_DAW [C:\Users\ChControl\Documents\taller4_DAW] - ...mysite\api\models.py [taller4_DAW] - PyCharm

```
1 from django.db import models
2
3 # Create your models here.
4 from django.contrib.auth.models import User
5 from rest_framework import routers, serializers, viewsets
6
7 # Serializers define the API representation.
8 class UserSerializer(serializers.ModelSerializer):
9     class Meta:
10         model = User
11         fields = ('url', 'username', 'email', 'is_staff')
12
13
14
15
16 class Libro(models.Model):
17     titulo = models.CharField(max_length=100)
18     anio = models.PositiveIntegerField()
19     precio = models.FloatField()
20
21     def __str__(self):
22         return f'Libro {self.titulo}'
23
24     def to_dict(self):
25         return [self.titulo, self.anio, self.precio]
```

Cannot Run Git
File not found: git.exe
[Download](#) [Configure...](#)

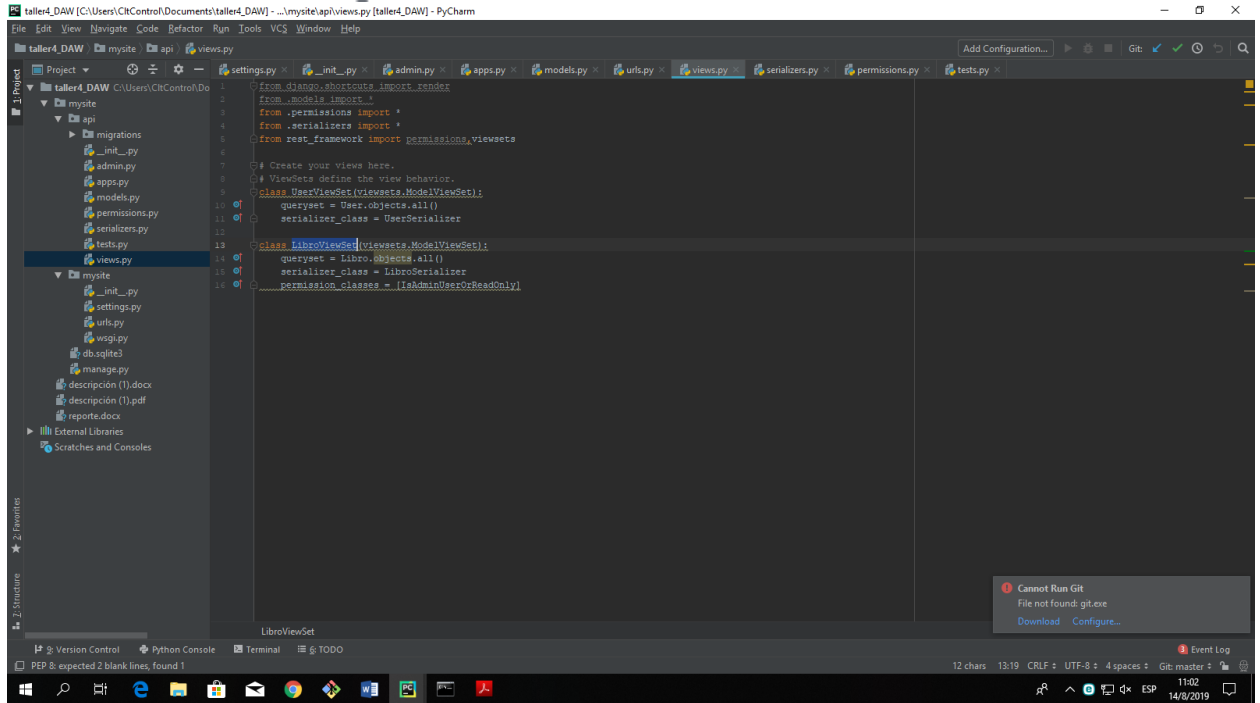
13:1 CRLF : UTF-8 : 4 spaces : Git: master : 11:01 14/8/2019

taller4_DAW [C:\Users\ChControl\Documents\taller4_DAW] - ...mysite\api\serializers.py [taller4_DAW] - PyCharm

```
1 from .models import *
2 from rest_framework import serializers
3
4 class LibroSerializer(serializers.ModelSerializer):
5     class Meta:
6         model = Libro
7         fields = '__all__'
```

Cannot Run Git
File not found: git.exe
[Download](#) [Configure...](#)

7:27 CRLF : UTF-8 : 4 spaces : Git: master : 11:02 14/8/2019

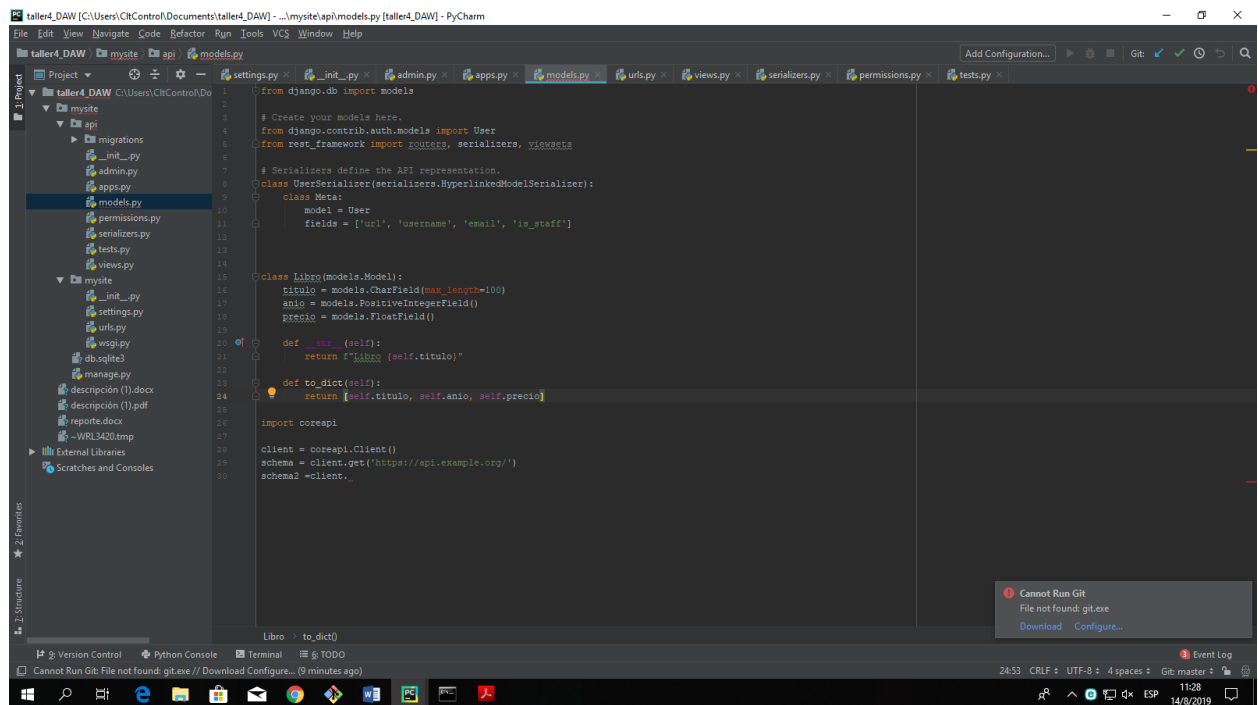


```

1 from django.shortcuts import render
2 from .models import *
3 from .permissions import *
4 from .serializers import *
5 from rest_framework import permissions, viewsets
6
7 # Create your views here.
8 # ViewSets define the view behavior.
9 class LibroViewSet(viewsets.ModelViewSet):
10     queryset = Libro.objects.all()
11     serializer_class = LibroSerializer
12     permission_classes = [IsAdminUserOrReadOnly]

```

- **Código del cliente consumidor del RESTful API. Aquí debe incluir la(s) captura(s) de las llamadas a los métodos: GET, POST, PUT y DELETE.**



```

1 from django.db import models
2
3 # Create your models here.
4 from django.contrib.auth.models import User
5 from rest_framework import routers, serializers, viewsets
6
7 # Serializers define the API representation.
8 class UserSerializer(serializers.HyperlinkedModelSerializer):
9     class Meta:
10         model = User
11         fields = ('url', 'username', 'email', 'is_staff')
12
13
14 class Libro(models.Model):
15     titulo = models.CharField(max_length=100)
16     anio = models.PositiveIntegerField()
17     precio = models.FloatField()
18
19     def __str__(self):
20         return f"Libro {self.titulo}"
21
22     def to_dict(self):
23         return {'titulo': self.titulo, 'anio': self.anio, 'precio': self.precio}
24
25 import coreapi
26
27 client = coreapi.Client()
28 schema = client.get('https://api.example.org/')
29 schema2 = client.get('https://api.example.org/')

```

- Registro de la consola del servidor con la respuesta por cada uno de los métodos.

The screenshot displays a Windows desktop environment. In the foreground, a REST client window is open, showing a GET request to `http://localhost:8000/api/libros/1/` with a response in JSON format. The response contains a single book object with fields: `url`, `titulo`, `anio`, and `precio`. In the background, a terminal window shows the Django server console output, including the server startup process and the receipt of the GET request.

REST Client Response:

```

Media type: application/json
Content: {
  "url": "http://localhost:8000/api/libros/1/",
  "titulo": "El amanecer de los muertos",
  "anio": 1999,
  "precio": 30.89
}

```

Server Console Output:

```

(c) 2018 Microsoft Corporation. Todos los derechos reservados.
C:\Users\CltControl>cd C:\Users\CltControl\Documents\taller4_DAW
C:\Users\CltControl\Documents\taller4_DAW>python manage.py runserver
python: can't open file 'manage.py': [Errno 2] No such file or directory
C:\Users\CltControl\Documents\taller4_DAW>cd mysite
C:\Users\CltControl\Documents\taller4_DAW>python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...
System check identified no issues (0 silenced).
August 14, 2019 - 11:18:09
Django version 2.2.4, using settings 'mysite.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
[14/Aug/2019 11:18:15] "GET /api/libros/1/ HTTP/1.1" 200 10901
[14/Aug/2019 11:18:31] "GET /api/libros/1/ HTTP/1.1" 200 109

```