

Laboratorio 8

Autor: Chura Monroy Daniel Wilston

July 25, 2023

1 Competencias del curso

- General: Diseña responsablemente aplicaciones web, sus componentes o procesos para satisfacer necesidades dentro de restricciones realistas: económicas, medioambientales, sociales, políticas, éticas, de salud, de seguridad, manufacturación y sostenibilidad.
- Específica: C.p. Aplica de forma flexible técnicas, métodos, principios, normas, estándares y herramientas del desarrollo web necesarias para la construcción de aplicaciones web e implementación de estos sistemas en una organización.

2 Marco Teórico

2.1 Django Rest Framework

- Django REST framework es un conjunto de herramientas potente y flexible para crear API web.
- La API navegable por la Web es una gran ganancia de usabilidad para sus desarrolladores.
- Políticas de autenticación que incluyen paquetes para OAuth1 y OAuth2
- Serialización que admite fuentes de datos ORM y no ORM.
- Personalizable hasta el final: solo use las vistas regulares basadas en funciones si no necesita las funciones más potentes .
- Amplia documentación y gran apoyo de la comunidad .
- Utilizado y confiado por empresas reconocidas internacionalmente, como Mozilla, Red Hat, Heroku y Eventbrite.

3 Soluciones y pruebas

```
(env) PS D:\workSpaceUniversidad\pw2\labs\lab8> python .\manage.py createsuperuser
Username (leave blank to use 'daniel'): admin
Email address: dchuramc@unsa.edu.pe
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.
(env) PS D:\workSpaceUniversidad\pw2\labs\lab8> []
```

Django REST framework admin

Api Root / Task List

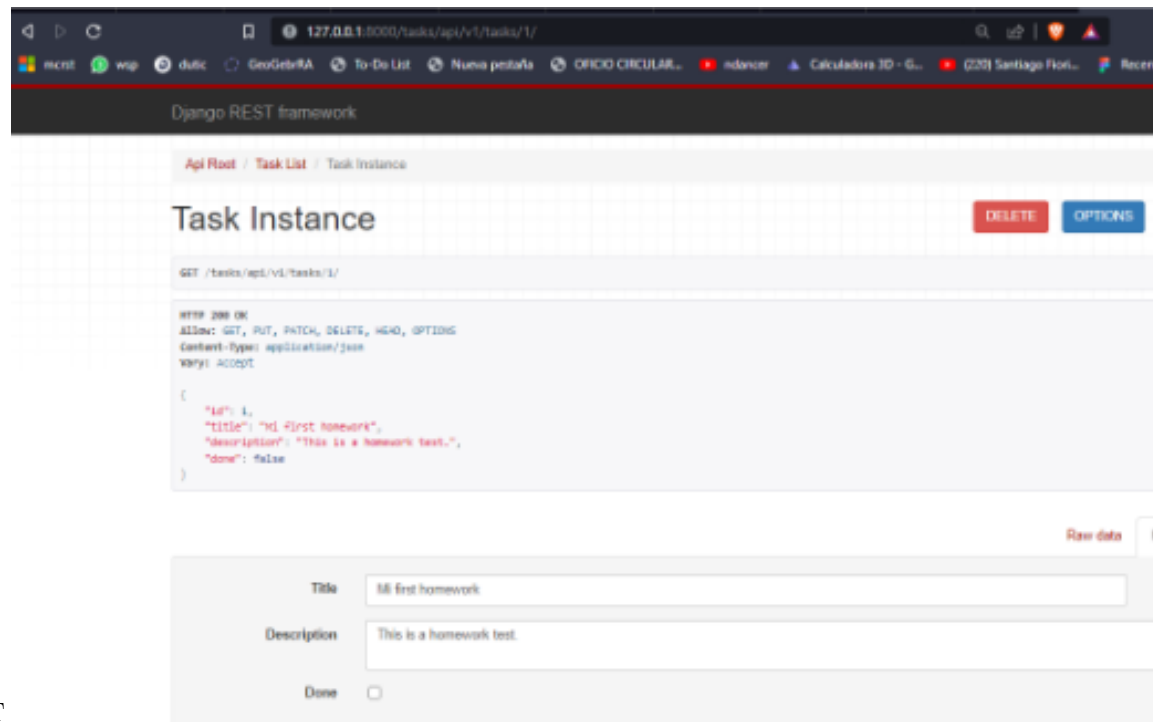
Task List OPTIONS GET

GET /tasks/api/v1/tasks/

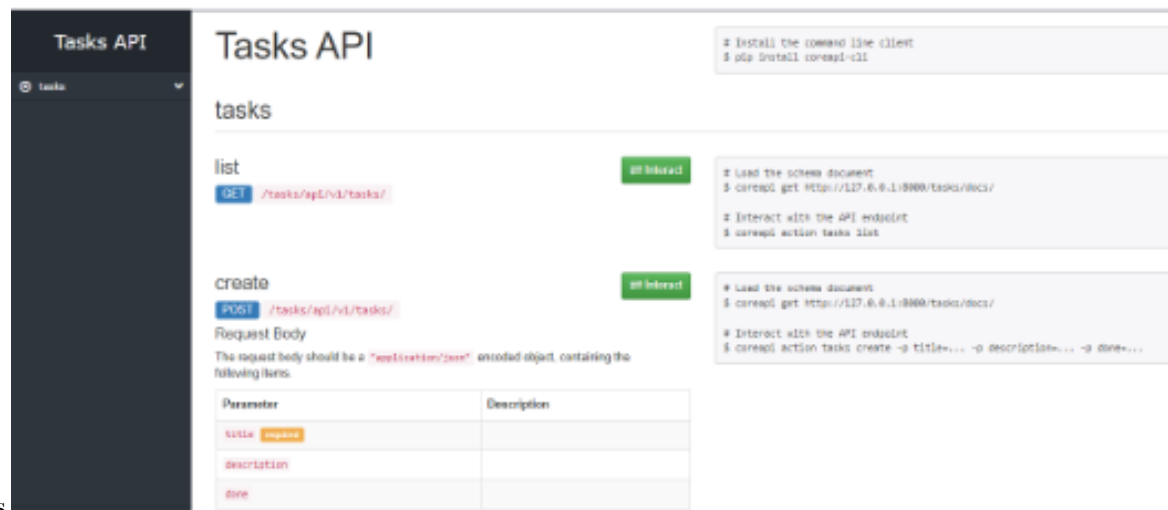
HTTP 200 OK
Allow: GET, POST, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

```
[
  {
    "id": 1,
    "title": "Py first homework",
    "description": "This is a homework test.",
    "done": false
  },
  {
    "id": 2,
    "title": "Py second homework",
    "description": "this is a test",
    "done": false
  }
]
```

Iniciando proyecto



Implementando REST



Verificando funciones

read

GET /tasks/api/v1/tasks/{id}/

Path Parameters

The following parameters should be included in the URL path.

Parameter	Description
id <small>required</small>	A unique integer value identifying this task.

Interact

```
# Load the schema document
$ curl http://127.0.0.1:8000/tasks/docs/

# Interact with the API endpoint
$ curl http://127.0.0.1:8000/tasks/{id}/
```

update

PUT /tasks/api/v1/tasks/{id}/

Path Parameters

The following parameters should be included in the URL path.

Parameter	Description
id <small>required</small>	A unique integer value identifying this task.

Interact

```
# Load the schema document
$ curl http://127.0.0.1:8000/tasks/docs/

# Interact with the API endpoint
$ curl http://127.0.0.1:8000/tasks/{id}/
```

Request Body

The request body should be a "application/json" encoded object, containing the following items.

Parameter	Description
-----------	-------------

Empleando API

delete

DELETE /tasks/api/v1/tasks/{id}/

Path Parameters

The following parameters should be included in the URL path.

Parameter	Description
id <small>required</small>	A unique integer value identifying this task.

Interact

```
# Load the schema document
$ curl http://127.0.0.1:8000/tasks/docs/

# Interact with the API endpoint
$ curl http://127.0.0.1:8000/tasks/{id}/
```

Funcion read y update

list

Data Raw

GET /tasks/api/v1/tasks/

200

```
[
  {
    "id": 1,
    "title": "Mi first homework",
    "description": "This is a homework test.",
    "done": false
  },
  {
    "id": 2,
    "title": "My second homework",
    "description": "this is a test",
    "done": false
  },
  {
    "id": 3,
    "title": "tarea 4",
    "description": "this is a pendent work",
    "done": true
  }
]
```

Close

Send Request

Funcion delete

create Data Raw

Title *

new task

Description

this is a new task

☒ Done

POST `/tasks/api/v1/tasks/` 201

```
{
  "id": 4,
  "title": "new task",
  "description": "this is a new task",
  "done": true
}
```

Close Send Request

Ejemplo de list (get)

read Data Raw

ID *

1

A unique integer value identifying this task.

GET `/tasks/api/v1/tasks/1/` 200

```
{
  "id": 1,
  "title": "Mi first homework",
  "description": "This is a homework test.",
  "done": false
}
```

Close Send Request

Ejemplo de post

partial_update Data Raw

ID *

1

A unique integer value identifying this task.

Title

Description

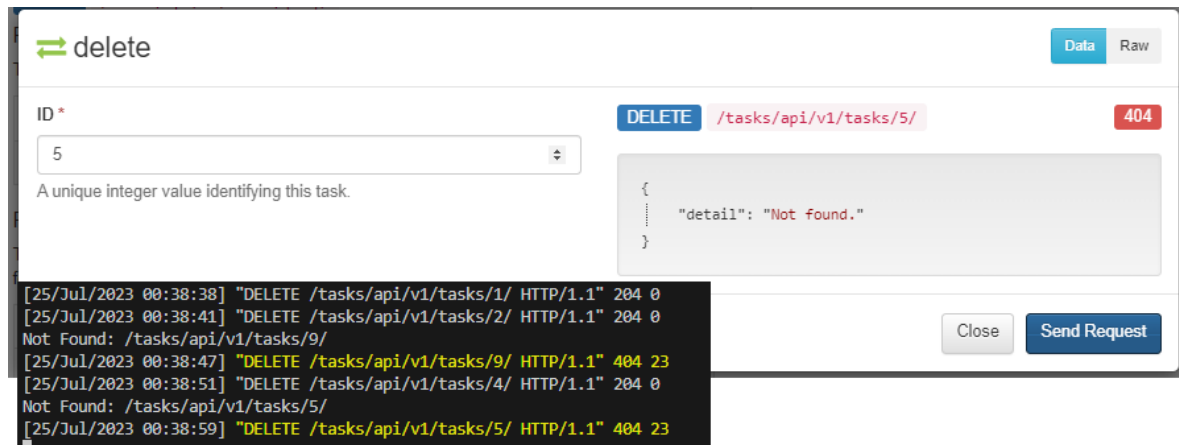
☐ Done

PATCH `/tasks/api/v1/tasks/1/` 200

```
{
  "id": 1,
  "title": "Mi first homework",
  "description": "This is a homework test.",
  "done": false
}
```

Close Send Request

Ejemplo de get



Ejemplo de patch
Ejemplo de delete

References

- <https://www.django-rest-framework.org/coreapi/>
- <https://github.com/Dan1elMon/l8>
- <https://www.django-rest-framework.org/tutorial/quickstart/>
- <https://www.youtube.com/watch?v=38XWpyEK8IY>