


Open Opened 2 days ago by  [Rubén Montero](#)

GET /api/v1/dashboards

Resumen

- Hablaremos de OpenAPI, un formato de fichero para especificaciones de API REST
- Presentaremos [la especificación de DashboardAPI](#)
- Implementaremos, sin guía, el primer *endpoint*

Descripción

Hasta ahora hemos dicho *alegremente* **cómo** se debían llamar los *endpoints* o **cómo** eran los JSON que se transferían.

Esas cuestiones conforman la **especificación de un API REST**.

Dicha **especificación**... ¡Es muy importante!

- Los **clientes HTTP** deberán mandar las peticiones adecuadas que nosotros esperamos
- Por eso, **cliente** y **servidor** deben estar de acuerdo
- Dicho **acuerdo** es casi como un **contrato**...

...y debe estar **formalmente especificado**.

[OpenAPI](#)

Se trata de una manera de **escribir cómo** debe ser un API REST.

No es algo que debamos estudiar. Tan sólo una especificación que va evolucionando y cuando necesitemos saber *cómo* se hace algo, consultaremos la [documentación oficial](#)

La mejor forma de entenderlo es con un ejemplo.

DashboardAPI

Nuestro siguiente proyecto está **especificado** en un fichero OpenAPI (`.yaml`).

Puedes visitarlo:

- [aquí](#)

GitLab se encarga de *renderizarlo* de una forma muy visual.

¿Observas que hay **3** GET y **2** POST?

La idea...

...es una versión ampliada de `WallAPI` .

Tendremos *Dashboards* que contienen *Preguntas*, y a su vez, las *Preguntas* contienen *Respuestas*.

¡ `DashboardAPI` !

Para comenzar

Se pide que uses `django-admin` para **crear** un nuevo proyecto dentro de `apis/` .

- **Se llamará** `DashboardAPI`
- Contendrá una *app*. llamada `dashboard04app`

```
- apis/  
  |  
  |-- DashboardAPI/  
    |  |  
    |  |-- DashboardAPI/  
    |  |  |  
    |  |  |-- __init__.py
```

```

| | | -- asgi.py
| | | -- settings.py
| | | -- urls.py
| | | -- wsgi.py
| |
| | -- dashboard04app/
| | |
| | | -- migrations/
| | | -- __init__.py
| | | -- admin.py
| | | -- apps.py
| | | -- endpoints.py
| | | -- models.py
| | | -- tests.py
| |
| | -- db.sqlite3
| | -- manage.py
|
|-- SimpleAPI/
|
|-- WallAPI/

```

En `models.py`

...deberás tener **3** modelos con **estos atributos**:

- **Dashboard**
 - `title` : CharField
 - `content` : CharField
- **Question**
 - `dashboard` : ForeignKey
 - `title` : CharField
 - `content` : CharField
 - `publication_date` : DateTimeField
- **Answer**
 - `question` : ForeignKey
 - `content` : CharField
 - `publication_date` : DateTimeField

Estos **nombres** son necesarios para que las bases de datos de *test* carguen adecuadamente y los **tests** funcionen.

Recuerda, también, usar:

```
python manage.py makemigrations
python manage.py migrate
```

...para *propagar los cambios* a la base de datos.

La tarea

Se pide que, tras los pasos anteriores, **implementes** el primer *endpoint*:

```
GET /api/v1/dashboards
```

Por último

Comprueba que tu código pasa el *test* asociado a la tarea.

Haz **commit** y **push** para subir los cambios al repositorio.



Rubén Montero @ruben.montero changed milestone to [%Sprint 4](#) 2 days ago

