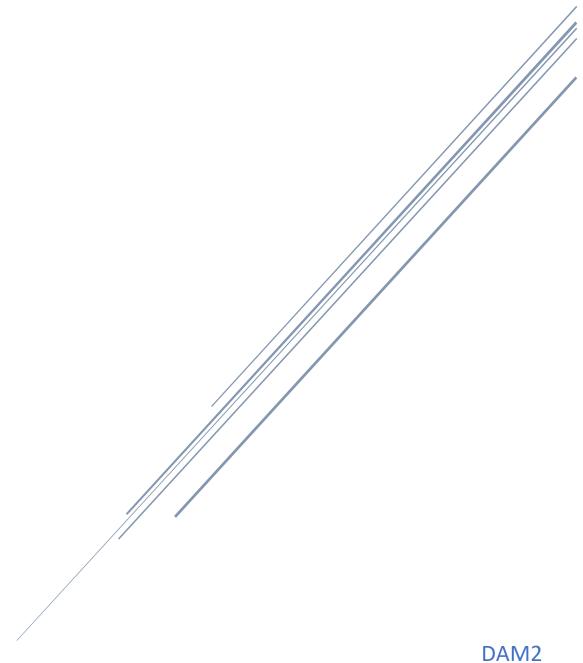
PROYECTO API PYTHON

PSP



DAM2
David Vinagrero Encina

Índice

Introducción	
Objetos	
Método GET	
Método POST	4
Método DELETE	4
Método PUT	5
Armas	6
Método GET	6
Método POST	
Método DELETE	
Método PUT	8
Jefes	9
Método GET	g
Método DELETE	10
Método PUT	

Introducción

La estructura general de las respuestas de los métodos GET es de un JSON, los valores que devuelva dependerán de la clase ha la que se pregunte. Este es un ejemplo de una respuesta de un GET.

```
[
    "BREVE": "Arma para novatos",
    "CALIDAD": "A",
    "DANIO": 30,
    "DESCRIPCION": "Su \u00fanica desventaja es el largo tiempo entre disparos. Hay quien dice que dispara m\u000e1s
    "ID": 8,
    "IMAGEN": "https://static.wikia.nocookie.net/enterthegungeon_gamepedia/images/8/85/A.W.P..png",
    "NOMBRE": "A.W.P"
}
```

Existen seis rutas para los métodos GET, dos por cada clase, que devuelven todo el contenido que hay en la base de datos. Para conseguir todos los datos la ruta que hay que especificar es:

Obtener todos los objetos

http://127.0.0.1:5000/get-all-objects

Obtener todas las armas

http://127.0.0.1:5000/get-all-guns

Obtener todos los jefes

http://127.0.0.1:5000/get-all-jefes

Estos tres métodos siguien esta estructura:

La api posee un front en la raiz que explica todos los métodos y como insertar valores para obtener distintos resultados. Este front también cuenta con una breve explicación de los métodos.

Objetos

A un objeto se le pueden hacer cuatro métodos, get, put, post y delete. Uno de los métodos (GET) ya lo he mencionado antes el otro método <u>GET</u> es más complejo.

Método GET

Este método permite obtener un objeto en base a **la calidad, el tipo o el nombre.** Estos parámetros pueden ponerse juntos o separados excepto el nombre que debe ir solo. Si por algún casual se pone el nombre y otro parámetro solo mostrará los objetos filtrando por el primer parámetro que se puso. Estos son algunos ejemplos que están en el front:

Obtener objetos por la calidad:

http://127.0.0.1:5000/get-objects?calidad=A

Obtener objetos por varias calidades:

http://127.0.0.1:5000/get-objects?calidad=A,S

Obtener objetos por el tipo:

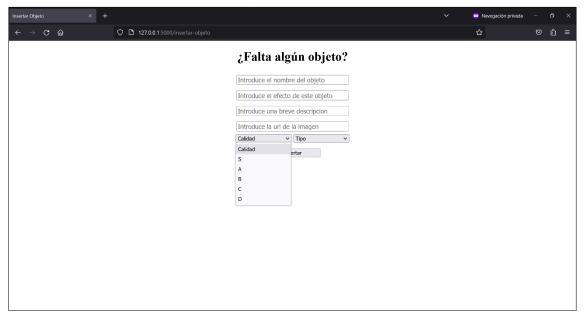
http://127.0.0.1:5000/get-objects?tipo=Activo

Obtener objetos por el tipo y la calidad:

http://127.0.0.1:5000/get-objects?tipo=Pasivo&calidad=S

Método POST

Para poder usar el método <u>POST</u> hay que ir al endpoint **/insertar-objeto**, este nos llevará a un sencillo formulario que rellenar, este formulario valida los datos introducidos para evitar errores. Este es el aspecto del formulario para un objeto:



En la imagen se puede comprobar que hay dos campos donde no se puede introducir valores que no estén permitidos. Además, el formulario deberá completarse por completo o no se podrá insertar el nuevo objeto.

Método DELETE

Para poder usar el método <u>DELETE</u> hay que usar este endpoint /eliminar-objeto, en este caso los parámetros se introducirán por url. Para poder eliminar un objeto sólo es necesario el nombre de este. Si el nombre introducido no existe dará un error de no encontrado. Este es un ejemplo de una url (habría que ejecutarlo con un método delete):

Eliminar un **objeto**:

http://127.0.0.1:5000/eliminar-objeto?nombre=Ejemplo

Método PUT

Por último, para poder usar el método <u>PUT</u> hay que usar el endpoint /modificar-objeto, al igual que en el caso anterior los parámetros se introducirán por la url. En este caso se pueden introducir todos los parámetros que estén como columnas en la base de datos siendo estos: **Breve, Nombre, Efecto, Imagen, Calidad, Tipo**.

Para poder modificar un objeto es **obligatorio poner el nombre del objeto** a modificar, pero no es necesario introducir todos los valores para no perder ningún valor. En el método put se buscan todos los valores antiguos que tienen los campos que <u>no se han introducido</u> y los usa en la consulta a la base de datos, de manera que **no se perderá ningún valor** independientemente de los valores que se introduzcan por la url.

Esta es la consulta a la base de datos:

```
query = "UPDATE `objetos` SET `NOMBRE` = '{}', `EFECTO` = '{}', `BREVE` = '{}', `IMAGEN` = '{}'," \

" `CALIDAD` = '{}', `TIPO` = '{}' WHERE `objetos`.`NOMBRE` = '{}'; ".format(nuevo_nombre, efecto, breve, imagen, calidad, tipo, nombre_input)
```

Y estos son algunos ejemplos de url:

Modificar el nombre actual:

http://127.0.0.1:5000/modificar-objeto?nombre=Corchete&nuevo_nombre=Parentesis

Modificar la calidad y el efecto:

http://127.0.0.1:5000/modificar-objeto?nombre=Reloj supercaliente&calidad=D&efecto=Quema mucho

Todos los ejemplos impuestos también están en el front.

Armas

A un arma se le pueden hacer cuatro métodos, get, put, post y delete (igual que a un objeto). Uno de los métodos (GET) ya lo he mencionado antes el otro método <u>GET</u> es más complejo.

Método GET

Este método permite obtener un arma en base a la **calidad**, el **daño** y el **nombre**. Estos métodos al igual que en el método <u>GET</u> del objeto. en la misma url o separados (el orden también da igual), mientras que el <u>nombre debe ir siempre solo</u>. Si se introduce el nombre y otro parámetro después ignorará este último, si se pone el nombre después de otro parámetro el nombre no se buscará.

Estos son algunos ejemplos:

Buscar por daño (buscará todas las armas cuyo daño sea igual o mayor al introducido):

http://127.0.0.1:5000/get-guns?dano=80

Buscar por varias calidades:

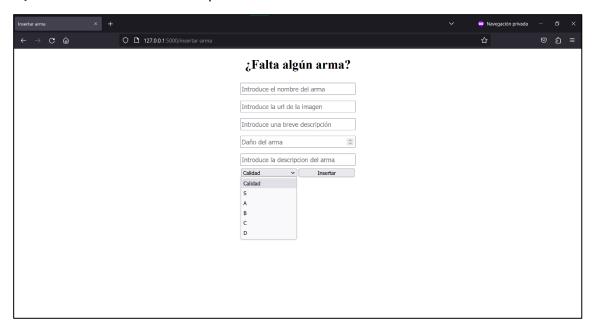
http://127.0.0.1:5000/get-guns?calidad=A,S

Buscar por daño y calidad:

http://127.0.0.1:5000/get-guns?dano=80&calidad=A

Método POST

Para el método <u>POST</u> he optado por usar un formulario y al igual que con los objetos este formulario está preparado para introducir solo datos compatibles y no dejar ninguno en blanco. Al formulario se accede con un método GET con el endpoint: /insertar-arma. Este es el aspecto del formulario:



Método DELETE

Para poder usar el método <u>DELETE</u> hay que usar este endpoint **/eliminar-arma**, en este caso los parámetros se introducirán por url. Para poder eliminar un arma sólo es necesario el nombre de esta. Si el nombre introducido no existe dará un error de no encontrado. Este es un ejemplo de una url (habría que ejecutarlo con un método delete):

Eliminar un arma:

http://127.0.0.1:5000/eliminar-arma?nombre=Ejemplo2

Método PUT

Por último, para poder usar el método <u>PUT</u> hay que usar el endpoint /modificar-arma, al igual que en el caso anterior los parámetros se introducirán por la url. En este caso se pueden introducir todos los parámetros que estén como columnas en la base de datos siendo estos: **Breve, Nombre, Descripción, Imagen, Calidad y Daño**.

Para poder modificar un arma es **obligatorio poner el nombre del arma** a modificar, pero no es necesario introducir todos los valores para no perder ningún valor. En el método put se buscan todos los valores antiguos que tienen los campos que <u>no se han introducido</u> y los usa en la consulta a la base de datos, de manera que **no se perderá ningún valor** independientemente de los valores que se introduzcan por la url.

Esta es la consulta a la base de datos:

Hay ejemplos en el front.

Jefes

Método GET

Este método permite obtener un jefe en base a el **piso** y el **nombre**. Este método solo admite parámetros de uno en uno, esto quiere decir que solo se puede buscar por nombre o piso, pero no los dos a la vez. Si se introduce el nombre y el otro parámetro después ignorará este último, si se pone el nombre después del otro no se buscará.

Estos son algunos ejemplos:

Buscar por nombre:

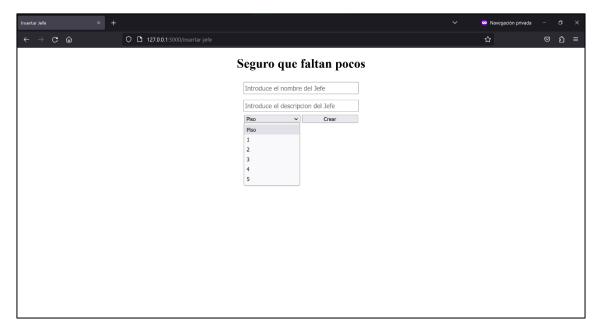
http://127.0.0.1:5000/get-jefes?nombre=Armaconda

Buscar por piso:

http://127.0.0.1:5000/get-jefes?piso=5

Método POST

Este método al igual que en todos los casos anteriores utiliza un pequeño formulario que valida todos los datos, al igual que no deja poner ninguno en blanco. El endpoint para poder acceder al formulario es /insertar-jefe. Este es el aspecto de el formulario:



Método DELETE

Para poder usar el método DELETE hay que usar este endpoint **/eliminar-jefe**, en este caso los parámetros se introducirán por url. Para poder eliminar un jefe sólo es necesario el nombre de este. Si el nombre introducido no existe dará un error de no encontrado. Este es un ejemplo de una url (habría que ejecutarlo con un método delete):

Eliminar un jefe:

http://127.0.0.1:5000/eliminar-jefe?nombre=Armaconda

Método PUT

Por último, para poder usar el método PUT hay que usar el endpoint /modificar-jefe, al igual que en el caso anterior los parámetros se introducirán por la url. En este caso se pueden introducir todos los parámetros que estén como columnas en la base de datos siendo estos: Nombre, Piso y Descripción.

Para poder modificar un jefe es obligatorio poner el nombre del jefe a modificar, pero no es necesario introducir todos los valores para no perder ningún valor. En el método put se buscan todos los valores antiguos que tienen los campos que no se han introducido y los usa en la consulta a la base de datos, de manera que no se perderá ningún valor independientemente de los valores que se introduzcan por la url.

Esta es la consulta a la base de datos:

Hay ejemplos en el front.