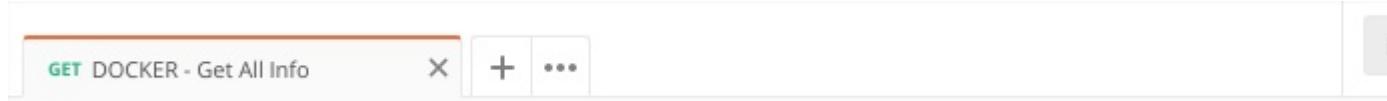


# Lanzar las dos peticiones

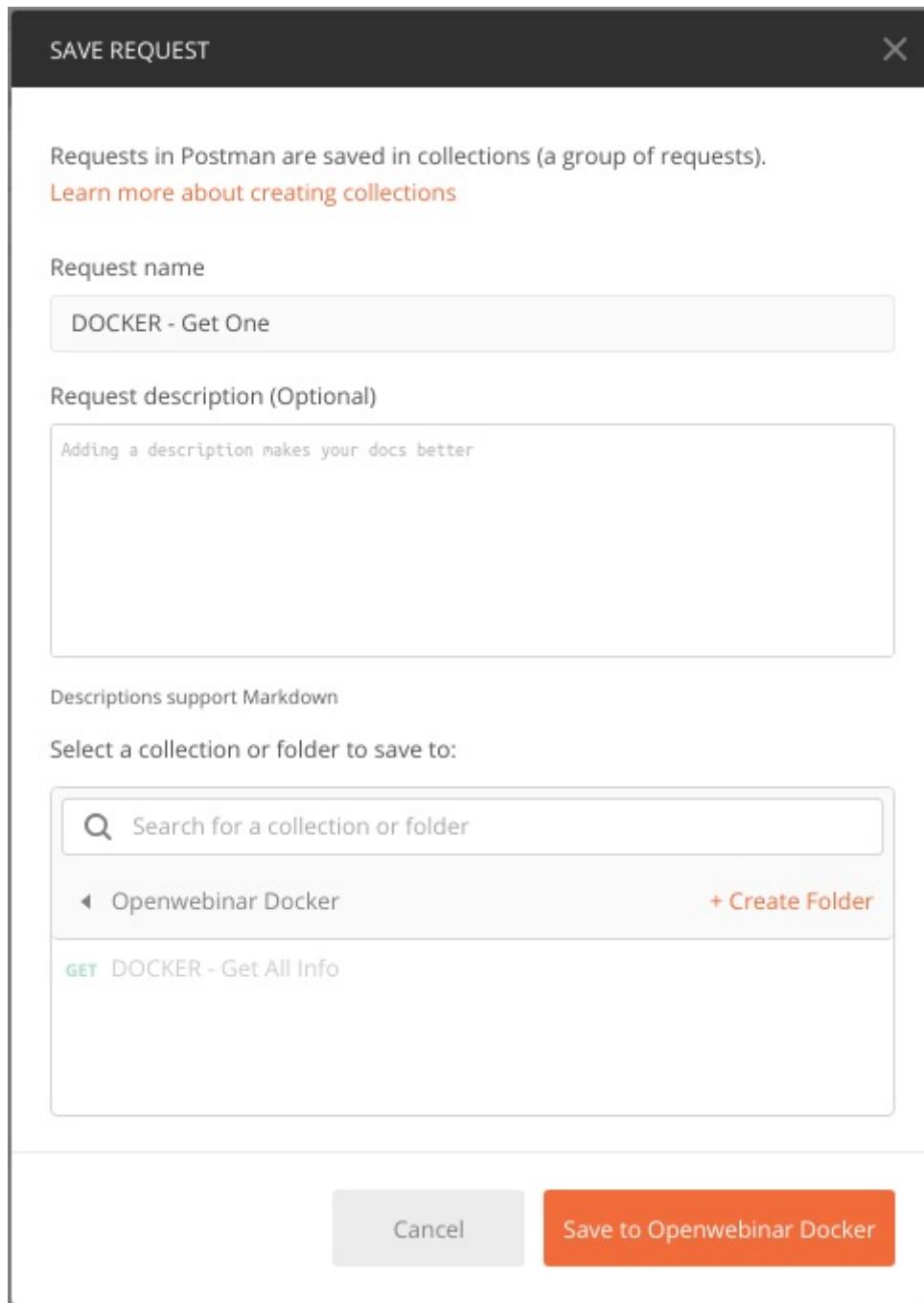
Para añadir una nueva petición deberemos pulsar sobre el **más** que está en la sección de las pestañas:



Posteriormente deberemos llenar la información del endpoint:

- **Tipo de petición:** GET
- **URL:** `http://localhost:7001/app-rest-api/avengers/getOne`
- **Headers:**
  - **Accept:** application/json, text/plain, /
  - **Accept-Encoding:** gzip, deflate, br
  - **Accept-Language:** en-US,en;q=0.9,es-ES;q=0.8,es;q=0.7
  - **X-Requested-With:** XMLHttpRequest

Finalmente guardaremos la petición pulsando **ctrl + s** y aparecerá una ventana emergente, la rellenaremos como se muestra a continuación:



Es decir, en la *Request name* pondremos *DOCKER - Get One* y posteriormente seleccionaremos la carpeta *Openwebinar Docker*. Finalmente haremos clic sobre *Save to Openwebinar Docker*.

A continuación (para validar el funcionamiento de lo ya creado) deberemos hacer clic sobre *Get All info*, se nos abrirá la petición que nos permite recuperar todos las entradas que hay en la BBDD, pulsaremos sobre *Headers* es donde estarán las cabeceras necesarias para nuestra API:

The screenshot shows the Postman application interface. On the left, there's a sidebar with a search bar labeled 'Filter', a 'History' tab, a 'Collections' tab which is currently selected, and a 'Trash' tab. Below the tabs, there's a section for 'Openwebinar Docker' containing '2 requests'. The first request, 'GET DOCKER - Get All Info', is highlighted. On the right, the main panel displays the details for this request. At the top, it shows 'GET DOCKER - Get All Info' with an 'X' button, a '+' button, and a '...' button. Below that, the URL 'http://localhost:7001/app-rest-api/avengers/g' is shown. Underneath the URL, there are tabs for 'Params', 'Authorization', 'Headers (4)', 'Body', and 'Pre-request'. The 'Headers (4)' tab is currently selected, and it lists four headers with checked checkboxes: 'Accept', 'Accept-Encoding', 'Accept-Language', and 'X-Requested-With'. There is also an empty row below them with the label 'Key'.

Acto seguido pulsaremos el botón *Send* para ejecutar la petición y deberemos obtener:

Pretty

Raw

Preview

JSON



```
1 [ [ {  
2   "id": 0,  
3   "lang": "EN",  
4   "name": "Iron Man",  
5   "actor": "Robert Downey Jr.",  
6   "description": "Iron Man (Anthony Edward \"Tony\" Stark) is a fictional superhero appearing in American comic books published by Marvel Comics. He was created by Stan Lee, Jack Kirby, and Bob Layton. Iron Man's alter ego is Tony Stark, a genius-level industrialist, engineer, inventor, and weapons designer.",  
7   "urlimage": "https://d29fhpw069ctt2.cloudfront.net/icon/image/59598/preview.svg"  
8 },  
9   {  
10    "id": 1,  
11    "lang": "EN",  
12    "name": "Spider-Man",  
13    "actor": "Tom Holland",  
14    "description": "Spider-Man is a fictional superhero created by writer-editor Stan Lee and writer-illustrator Steve Ditko. He first appeared in the comic book Amazing Fantasy in 1962.",  
15    "urlimage": "https://d29fhpw069ctt2.cloudfront.net/icon/image/59595/preview.svg"  
16 },  
17   {  
18    "id": 2,  
19    "lang": "EN",  
20    "name": "American Captain",  
21    "actor": "Chris Evans",  
22    "description": "Captain America (Steve Rogers) is a fictional superhero appearing in American comic books published by Marvel Comics. He was created by Stan Lee and Jack Kirby.",  
23    "urlimage": "https://d29fhpw069ctt2.cloudfront.net/icon/image/59598/preview.svg"  
24 },  
25   {  
26    "id": 3,  
27    "lang": "EN",  
28    "name": "Black Widow",  
29    "actor": "Scarlett Johansson",  
30    "description": "Natalia Alianovna Romanova (alias: Natasha Romanoff), colloquial: Black Widow is a fictional superhero appearing in American comic books published by Marvel Comics.",  
31    "urlimage": "https://d29fhpw069ctt2.cloudfront.net/icon/image/59601/preview.svg"  
32 },  
33   {  
34    "id": 4,  
35    "lang": "EN",  
36    "name": "Thor",  
37    "actor": "Chris Hemsworth",  
38  } ]
```

Si hacemos lo mismo con la otra petición obtendremos:

The screenshot shows the Postman application interface. At the top, there are buttons for 'New', 'Import', 'Runner', and a settings icon. Below the header, there's a search bar labeled 'Filter' and tabs for 'History' and 'Collections'. The 'Collections' tab is selected, showing a folder named 'Openwebinar Docker' which contains 2 requests. On the left, under 'Openwebinar Docker', are the requests: 'GET DOCKER - Get All Info' and 'GET DOCKER - Get One'. On the right, the details for 'DOCKER - Get One' are shown. The 'Body' tab is selected, displaying a JSON response:

```
1 [ {  
2   "id": 1,  
3   "lang": "EN",  
4   "name": "Spider-Man",  
5   "actor": "Tom Holland",  
6   "description": "Spider-Man is a fictional superhero",  
7   "urlimage": "https://d29fhpw069ctt2.cloudfront.net/i/  
8 } ]
```

## Resultado final

Si ambas peticiones han ido correctamente la aplicación desplegada en el contenedor de WL que consume la BBDD del contenedor de Oracle DB habrá quedado correctamente configurada. Sino, habrá que revisar como están desplegados los contenedores y como se ha desplegado la aplicación dentro de WL.

Habiendo finalizado esto ya tendríamos un entorno funcional Oracle basado en contenedores. En los siguientes temas vamos a empezar con las primeras prácticas.