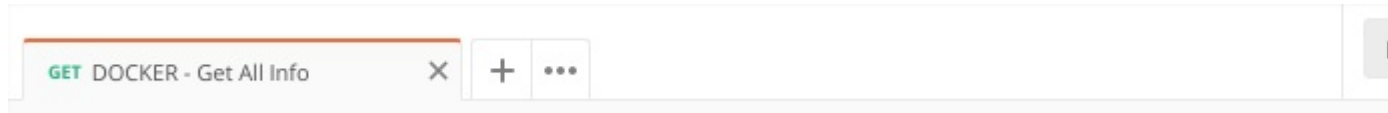


Lanzar las dos peticiones

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



Posteriormente deberemos rellenar 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 aparecera una ventana emergente, la rellenaremos como se muestra a continuación:

SAVE REQUEST

×

Requests in Postman are saved in collections (a group of requests).
[Learn more about creating collections](#)

Request name

DOCKER - Get One

Request description (Optional)

Adding a description makes your docs better

Descriptions support Markdown

Select a collection or folder to save to:

Q Search for a collection or folder

◀ Openwebinar Docker + Create Folder

GET DOCKER - Get All Info

Cancel Save to Openwebinar Docker

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 abra la petición que nos permite recuperar todos las entradas que hay en la BBDD, pulsemos sobre *Headers* es donde estarán las cabeceras necesarias para nuestra API:

The screenshot displays the Postman application interface. On the left, the 'Collections' tab is active, showing a collection named 'Openwebinar Docker' with 2 requests. The first request, 'DOCKER - Get All Info', is selected. The main panel on the right shows the configuration for this request. The method is 'GET' and the URL is 'http://localhost:7001/app-rest-api/avengers/g'. The 'Headers' tab is selected, showing four headers: 'Accept', 'Accept-Encoding', 'Accept-Language', and 'X-Requested-With', all of which are checked. The 'KEY' column is visible on the left of the header table.

KEY	Value
<input checked="" type="checkbox"/>	Accept
<input checked="" type="checkbox"/>	Accept-Encoding
<input checked="" type="checkbox"/>	Accept-Language
<input checked="" type="checkbox"/>	X-Requested-With
	Key

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

Body
Cookies
Headers (3)
Test Results

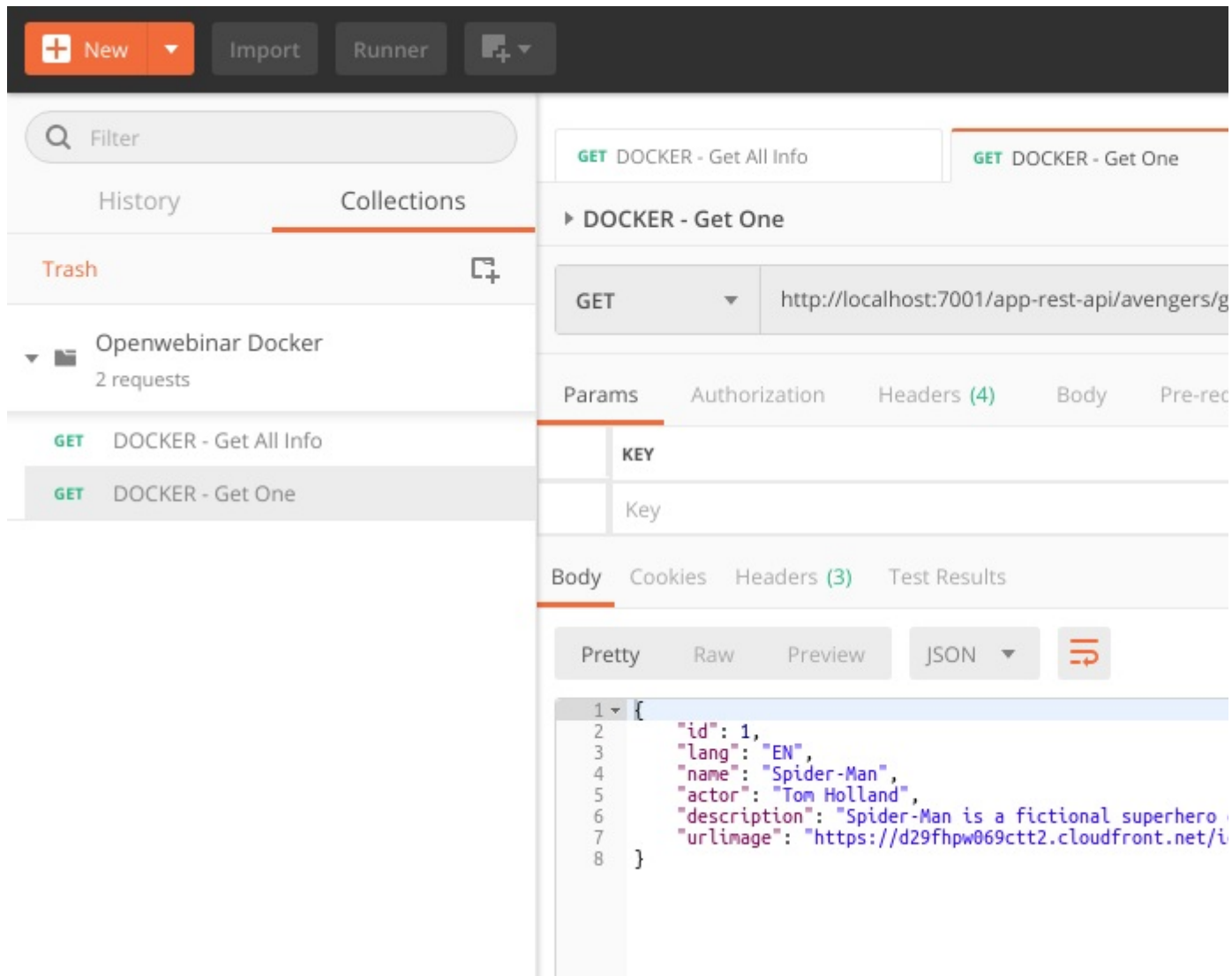
Pretty
Raw
Preview
JSON

```

1  [
2    {
3      "id": 0,
4      "lang": "EN",
5      "name": "Iron Man",
6      "actor": "Robert Downey Jr.",
7      "description": "Iron Man (Anthony Edward \"Tony\" Stark) is a fictional superhero appearing in American comic books published by Marvel Comics, designed by artists Don Heck and Jack Kirby...",
8      "urlimage": "https://d29fhpw069ctt2.cloudfront.net/icon/image/59598/preview.svg"
9    },
10   {
11     "id": 1,
12     "lang": "EN",
13     "name": "Spider-Man",
14     "actor": "Tom Holland",
15     "description": "Spider-Man is a fictional superhero created by writer-editor Stan Lee and writer-artist Steve Ditko, appearing in American comic books published by Marvel Comics.",
16     "urlimage": "https://d29fhpw069ctt2.cloudfront.net/icon/image/59595/preview.svg"
17   },
18   {
19     "id": 2,
20     "lang": "EN",
21     "name": "American Captain",
22     "actor": "Chris Evans",
23     "description": "Captain America (Steve Rogers) is a fictional superhero appearing in American comic books published by Marvel Comics, the character is a precursor to Marvel's current line of comic book superheroes (cover dated March 1941) from Timely Comics, a predecessor of Marvel Comics...",
24     "urlimage": "https://d29fhpw069ctt2.cloudfront.net/icon/image/59598/preview.svg"
25   },
26   {
27     "id": 3,
28     "lang": "EN",
29     "name": "Black Widow",
30     "actor": "Scarlett Johansson",
31     "description": "Natalia Alianovna Romanova (alias: Natasha Romanoff), colloquial: Black Widow is a fictional character appearing in American comic books published by Marvel Comics.",
32     "urlimage": "https://d29fhpw069ctt2.cloudfront.net/icon/image/59601/preview.svg"
33   },
34   {
35     "id": 4,
36     "lang": "EN",
37     "name": "Thor",
38     "actor": "Chris Hemsworth",

```

Si hacemos lo mismo con la otra petición obtendremos:



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.