

CHALLENGE BACKEND – MercadoLibre

Para coordinar acciones de respuesta ante fraudes, es útil tener disponible información contextual del lugar de origen detectado en el momento de comprar, buscar y pagar. Para ello se decide crear una herramienta que dada una IP obtenga información asociada.

El ejercicio consiste en construir una API REST que permita:

1. **Dada una dirección IP, encontrar el país al que pertenece y mostrar:**
 - a. El nombre y código ISO del país.
 - b. Moneda local y su cotización actual en dólares o euros.

Solución: Se construye el servicio **country-info** en la API que, como lo indica el enunciado, recibe la IP para consultar la información del país.

API.

GET ▼ http://localhost:8080/api/v1/country-info

Body del request.

```
1 {  
2   "ip": "195.43.88.26"  
3 }
```

Resultado.

```
1 {  
2   "code": "DE",  
3   "countryName": "Germany",  
4   "localCurrency": "EUR",  
5   "currentValue": 0.959375  
6 }
```

NOTA: En la respuesta del primer servicio se obtuvo la información correspondiente al nombre del país, el código ISO, su moneda local y su cotización actual respecto al dólar (USD), que fue la moneda seleccionada para el ejercicio, como se requiere en los puntos **a** y **b**.

2. Ban/Blacklist de una IP: marcar la IP en una lista negra no permitiéndole consultar la información del punto 1.

Solución: Se construye el servicio **blacklist** en la API que recibe la IP para marcarla y persistirla en una lista negra impidiendo consultar la información del país del punto 1.

API.

POST	▼	http://localhost:8080/api/v1/blacklist
------	---	--

Body del request.

```
1 {  
2   "ip": "195.43.88.26"  
3 }
```

Resultado.

```
1 {  
2   "ip": "195.43.88.26",  
3   "message": "Successfully registered IP."  
4 }
```

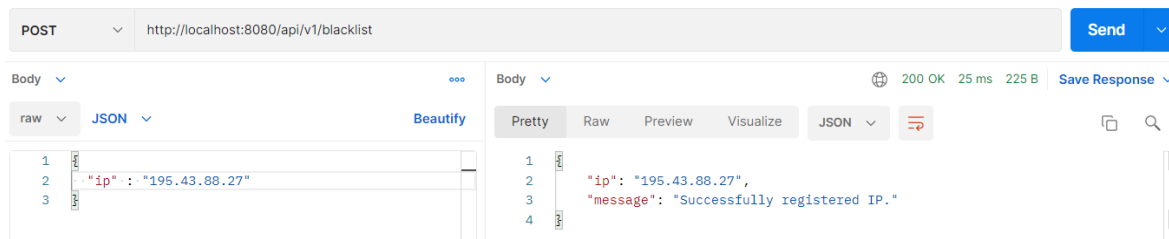
NOTA: En la respuesta del servicio se obtuvo el mensaje de satisfacción indicando que se ha guardado correctamente la IP en la lista negra. Así que, si se desea consultar nuevamente la IP 195.43.88.26 en el primer servicio **country-info**, ya no se podría obtener la información del país (**IP not available.**), así como se observa en la siguiente imagen:

```
1 {  
2   "code": "403",  
3   "message": "IP not available."  
4 }
```

ESPECIFICACIÓN FUNCIONAL, TÉCNICA Y DETALLADA

Se crea un servicio (API REST) con Spring Boot y Gradle, que permite consultar la información de un país dada una IP como parámetro, y también persistir en una base de datos el listado de IP's impidiendo consultar la información de dicho país.

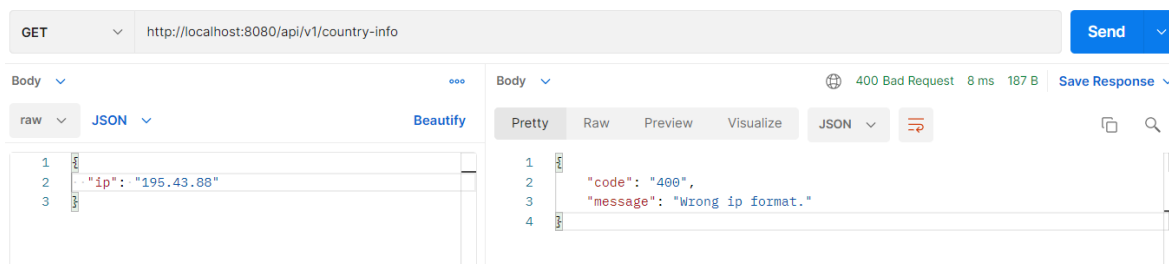
El primer EndPoint es <http://localhost:8080/api/v1/country-info> y por convención es utilizado el verbo GET, para obtener la información del país. (Para ilustrar mejor el funcionamiento de la API se utilizará la herramienta Postman).



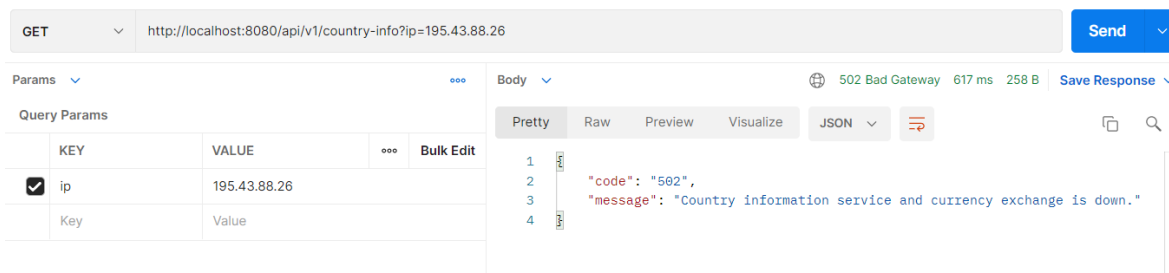
El servicio ha retornado la información solicitada con satisfacción, con código HTTP 200 de respuesta.

Validaciones para tener en cuenta.

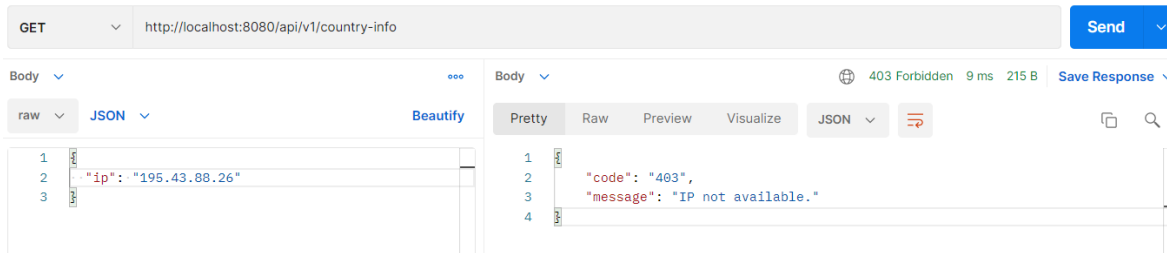
Si se ingresa una cadena que no corresponde a una IP válida, el servicio responderá un Bad Request (400) con mensaje **“Wrong ip format.”**, así como se observa en la imagen:



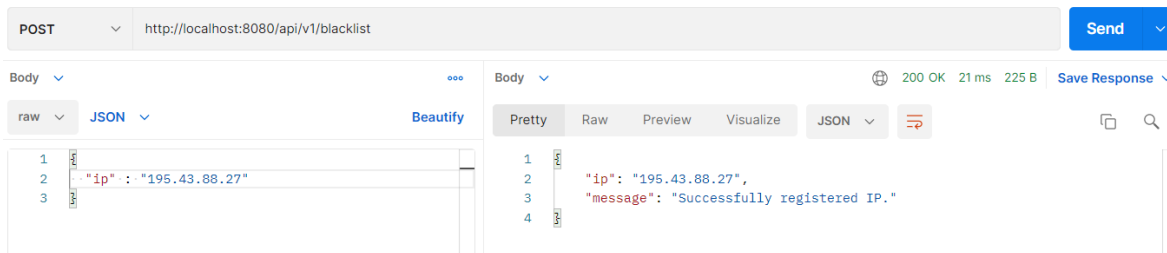
Si se intenta consultar la información del país y las API's externas no están disponibles, el servicio responderá un Bad Gateway (502), así como se observa en la siguiente imagen:



Si se intenta consultar una IP que ya está registrada en la lista negra, el servicio responderá un Forbidden (403), con el mensaje de IP no disponible **“IP not available.”**, así como se observa en la imagen:



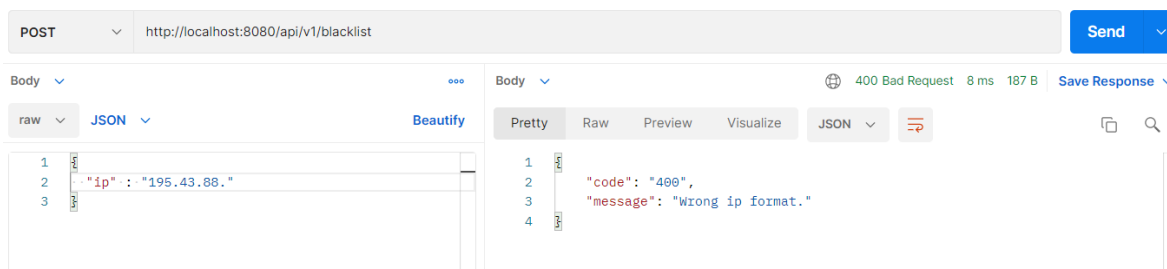
El segundo EndPoint es <http://localhost:8080/api/v1/blacklist> con sus métodos de GET y POST; el primer método retorna el listado de las IP's registradas en la lista negra, mientras que el segundo registra en el listado la IP.



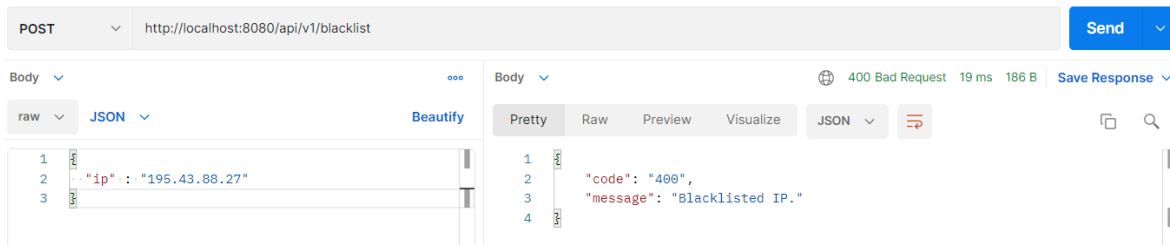
El servicio ha retornado el mensaje que indica que la IP fue registrada satisfactoriamente **“Successfully registered IP.”**, con código HTTP 200 de respuesta.

Validaciones para tener en cuenta.

Si se ingresa una cadena que no corresponde a una IP válida, el servicio responderá un Bad Request (400) con mensaje **“Wrong ip format.”**, así como se observa en la imagen:

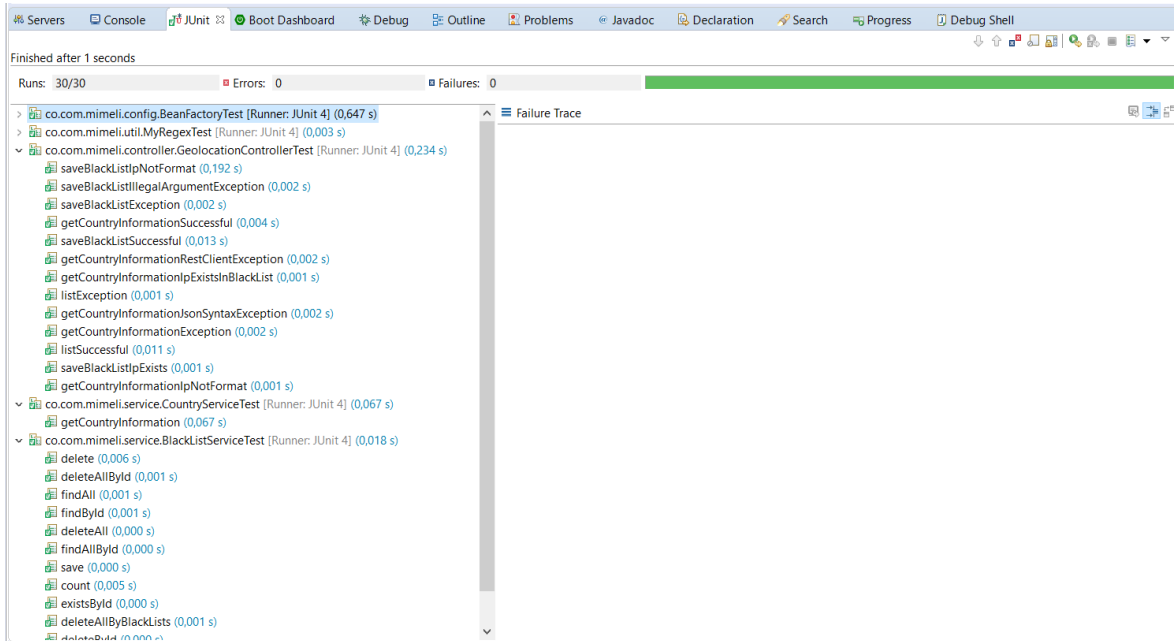


Si se intenta consultar una IP que ya está registrada en la lista negra, el servicio responderá un Bad Request (400), con el mensaje de IP ya registrada en lista negra **“Blacklisted IP.”**, así como se observa en la imagen:



PRUEBAS UNITARIAS (JUnit – Mockito)

Se ejecutan las pruebas unitarias con **JUnit**, con un total de 30 ejecuciones satisfactorias.



De igual manera se ejecuta el **Coverage** en todo el proyecto para validar la cobertura de las pruebas unitarias, alcanzando más de 98%. Así como se observa en la siguiente imagen:

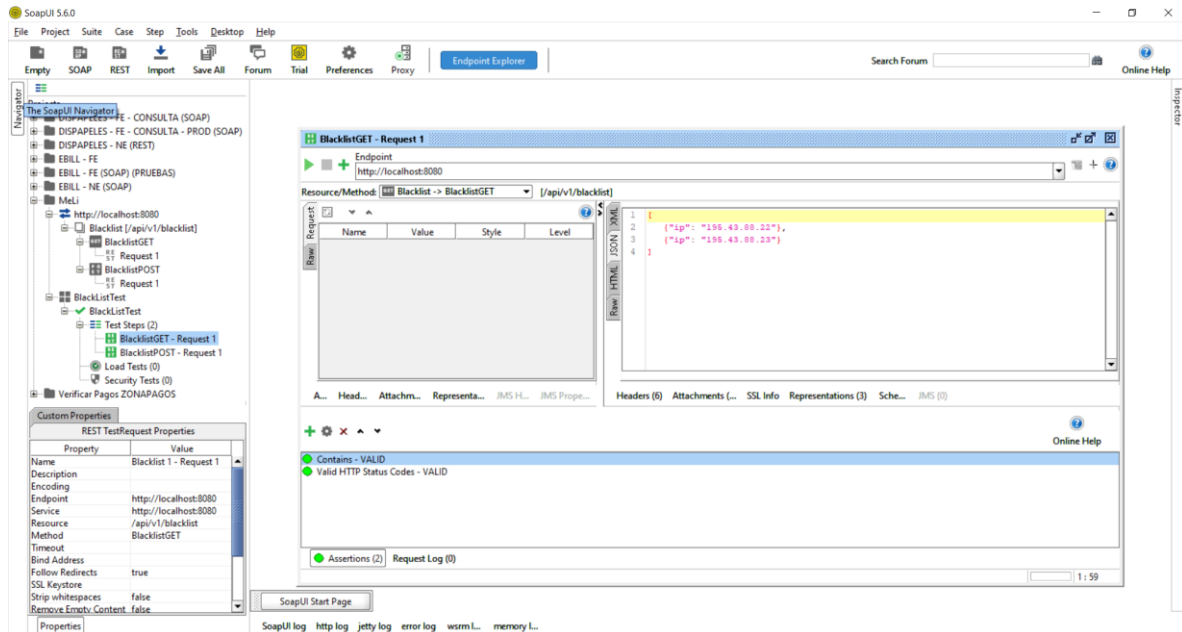
The screenshot shows the Coverage tool interface. It displays a table with columns: Element, Coverage, Covered Instru..., and Missed Instru... The table lists various project elements and their coverage percentages.

Element	Coverage	Covered Instru...	Missed Instru...
mimeli	98,7 %	1.873	
src/main/java	96,2 %	501	
co.com.mimeli	0,0 %	0	
co.com.mimeli.model.request	75,0 %	18	
co.com.mimeli.model.entity	75,0 %	9	
co.com.mimeli.util	92,7 %	38	
co.com.mimeli.config	100,0 %	11	
co.com.mimeli.controller	100,0 %	176	
co.com.mimeli.model	100,0 %	15	
co.com.mimeli.model.response	100,0 %	61	
co.com.mimeli.service	100,0 %	173	
src/test/java	99,7 %	1.372	
co.com.mimeli	0,0 %	0	
co.com.mimeli.config	100,0 %	21	
co.com.mimeli.controller	100,0 %	916	
co.com.mimeli.service	100,0 %	404	
co.com.mimeli.util	100,0 %	31	

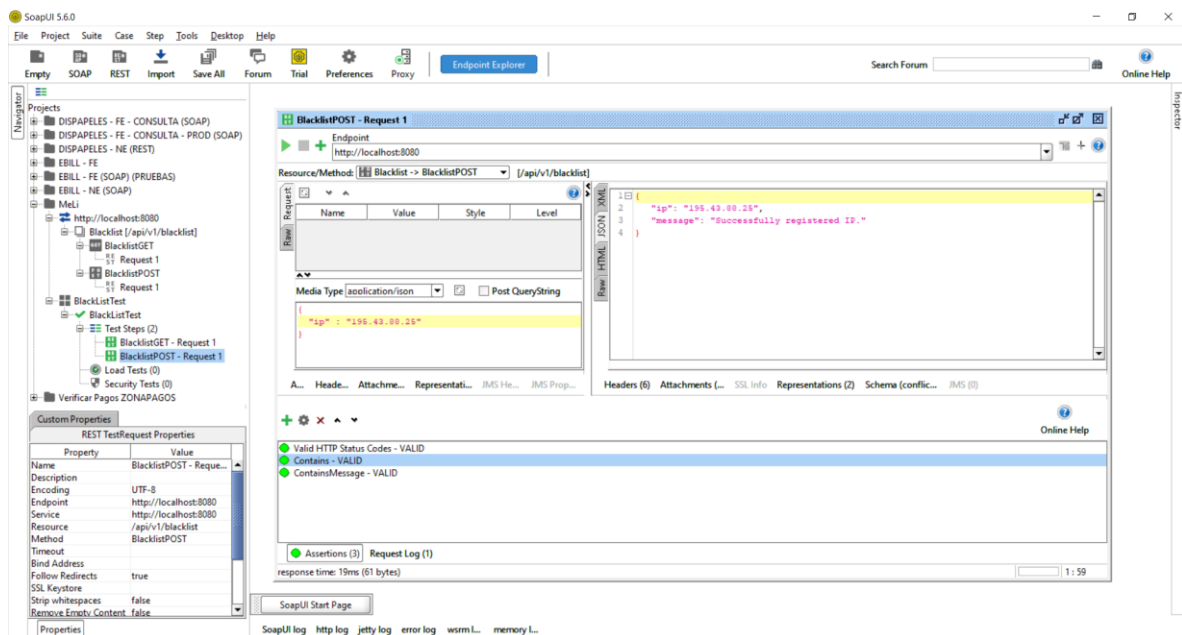
PRUEBAS FUNCIONALES (SoapUI)

Las pruebas funcionales o pruebas de tipo caja negra, basadas en la ejecución y revisión se realizarán en la herramienta SoapUI.

<http://localhost:8080/api/v1/blacklist> (GET)



<http://localhost:8080/api/v1/blacklist> (POST)



NOTA: Se observan en ambas pruebas las respectivas **“Assertions”**.

<http://localhost:8080/api/v1/country-info> (GET)

CountryInfoGET - Request 1

Endpoint
http://localhost:8080

Resource/Method: CountryInfo -> CountryInfo... [/api/v1/country-info?ip=104.72.224.3]

Name	Value	Style	Level
ip	104.72.224.3	QUERY	RESOURCE

```
{
  "code": "AR",
  "countryName": "Argentina",
  "localCurrency": "ARS",
  "currentValue": 117.863301
}
```

A... Head... Attachm... Representa... JMS H... JMS Prope...

Headers (6) Attachments (...) SSL Info Representations (1) Sche... JMS (0)

Online Help

- Valid HTTP Status Codes - VALID
- Contains - VALID
- ContainsCountryName - VALID
- ContainsLocalCurrency - VALID
- ContainsCurrentValue - VALID

Assertions (5) Request Log (0)

1 : 59

CONFIGURACIÓN DOCKER

En la raíz del proyecto **mimeli** se encuentran dos archivos, un **Dockerfile** y un **docker-compose.yml**, que contienen la configuración para construir la aplicación incluyendo la base de datos y el despliegue y/o ejecución de ambas aplicaciones.

Una vez estando en ahí en la raíz del proyecto se ejecuta el siguiente comando: **docker-compose up**

```
Windows PowerShell
PS E:\Mis documentos\MeLi\mimeli> docker-compose up -d
```

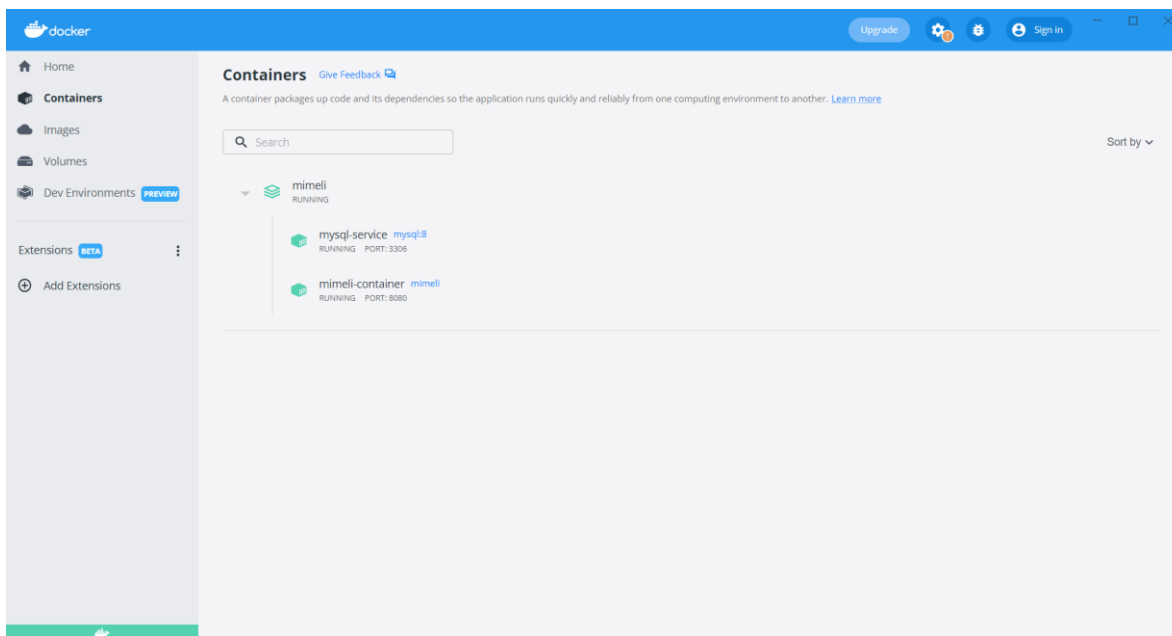
```
Windows PowerShell
PS E:\Mis documentos\MeLi\mimeli> docker-compose up -d
Creating network "mimeli_mimeli-network" with driver "bridge"
Pulling mysql-service (mysql:8)...
8: Pulling from library/mysql
c32ce6654453: Pull complete
415d08ee031a: Pull complete
7a38fec2542f: Pull complete
352881ee8fe9: Pull complete
b8e20da291b6: Pull complete
66c2a8cc1999: Extracting [=====] 5.243MB/14.06MB
d3a3a8e49878: Download complete
e33a48832bec: Download complete
410b942b8b28: Downloading [=====] 15.51MB/109.1MB
d5323c9dd265: Download complete
3212737f31c0: Download complete
d0032d4b0dc5: Download complete
```

```
Use 'docker scan' to run Snyk tests against images to
WARNING: Image for service mimeli-service was built by
'docker-compose build' or 'docker-compose up --build'
Creating mysql-service ... done
Creating mimeli-container ... done
PS E:\Mis documentos\MeLi\mimeli>
```

Una vez se haya ejecutado el comando se consultan las imágenes creadas.

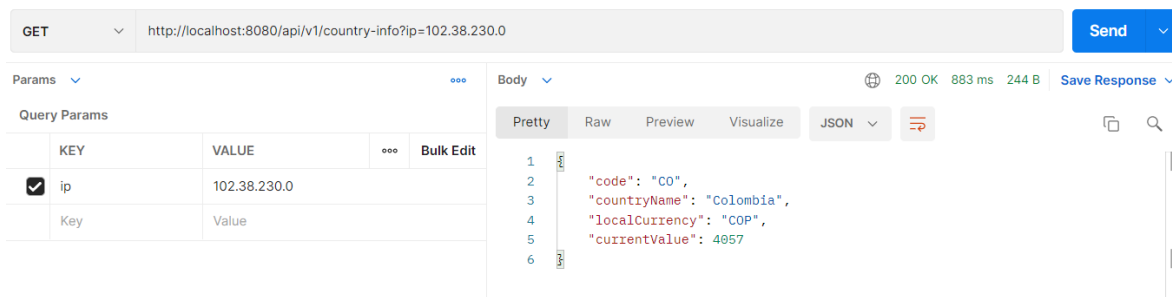
```
PS E:\Mis documentos\MeLi\mimeli> docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
mimeli        latest    ecaac54ee267   4 minutes ago  194MB
mysql         8         76152be68449   5 days ago    524MB
PS E:\Mis documentos\MeLi\mimeli>
```

Vista de contenedores desde Docker Desktop.

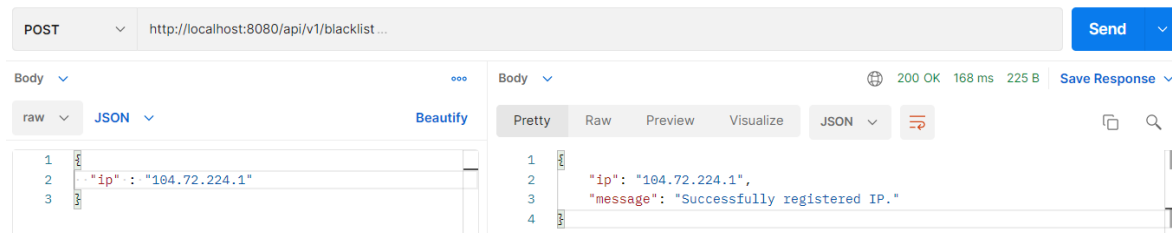


EJECUCIÓN DE LOS SERVICIOS YA INSTALADOS EN DOCKER DESDE LA HERRAMIENTA POSTMAN.





Servicio **country-info**.



Servicio **blacklist**.



ARCHIVO JSON CON EL COLLECTION DEL POSTMAN

 HELP.md	4/05/2022 9:11 p. m.	Archivo MD	2 KB
 hs_err_pid12140	4/05/2022 7:52 p. m.	Documento de tex...	91 KB
 MeLi.postman_collection.json	9/05/2022 9:29 a. m.	Archivo JSON	2 KB
 settings.gradle	7/05/2022 6:07 p. m.	Archivo GRADLE	1 KB

API'S EXTERNAS UTILIZADAS PARA LA ALIMENTACIÓN DE INFORMACIÓN

1. <https://ipapi.com/>
2. <https://www.currencyconverterapi.com/>

TECNOLOGÍAS Y HERRAMIENTAS UTILIZADAS

Java 11

Spring Boot v2.6.7

SpringToolSuite v4

JUnit v4

MySQL v8.0.29

Docker v20.10.14

Gradle v7.4.2

SoapUI v5.6.0

Postman v8.12.1