

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": 1.054352  
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 }
```

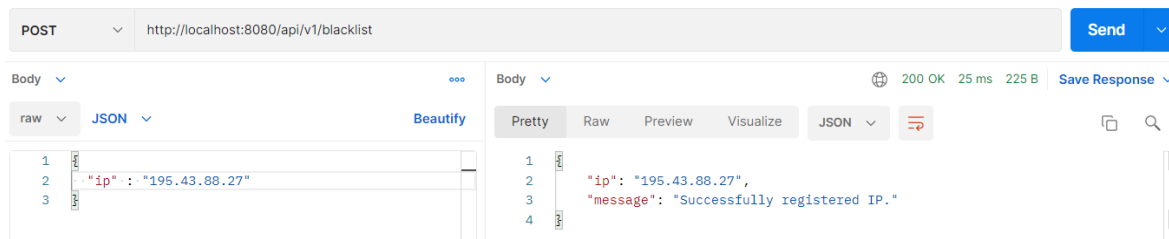
NOTA: En la respuesta del servicio se obtuvo el mensaje de satisfacción indicando que se ha guardado correctamente la IP en la lista negra. Así que, si se desea consultar nuevamente la IP 195.43.88.26 en el primer servicio **country-info**, ya no se podría obtener la información del país (**IP not available.**),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

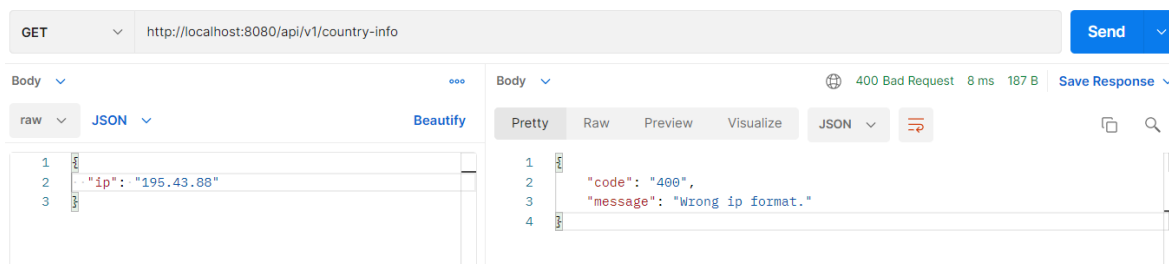
El primer EndPoint es <http://localhost:8080/api/v1/country-info> y por convención es utilizado el verbo GET, para obtener la información del país. (Para ilustrar mejor el funcionamiento de la API se utilizará la herramienta Postman).



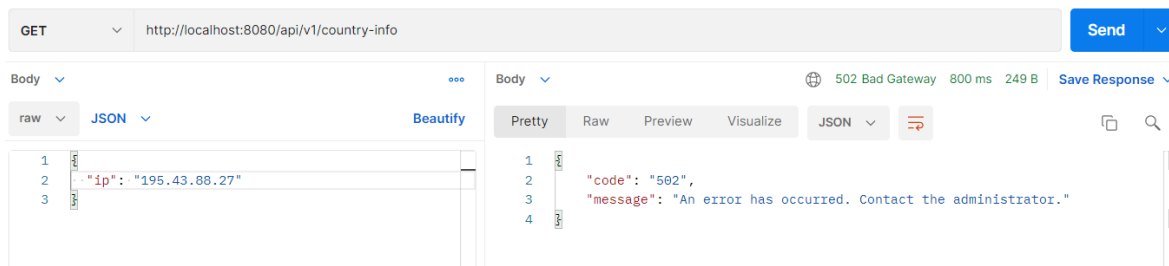
El servicio ha retornado la información solicitada con satisfacción, con código HTTP 200 de respuesta.

Validaciones para tener en cuenta.

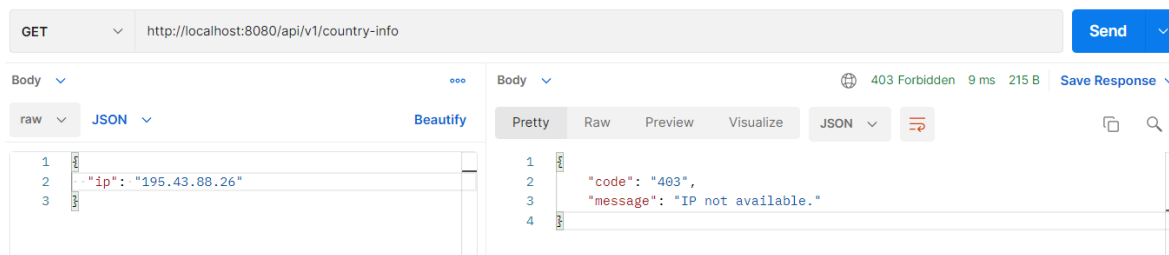
Si se ingresa una cadena que no corresponde a una IP válida, el servicio responderá un Bad Request (400) con mensaje **“Wrong ip format.”**, así como se observa en la imagen:



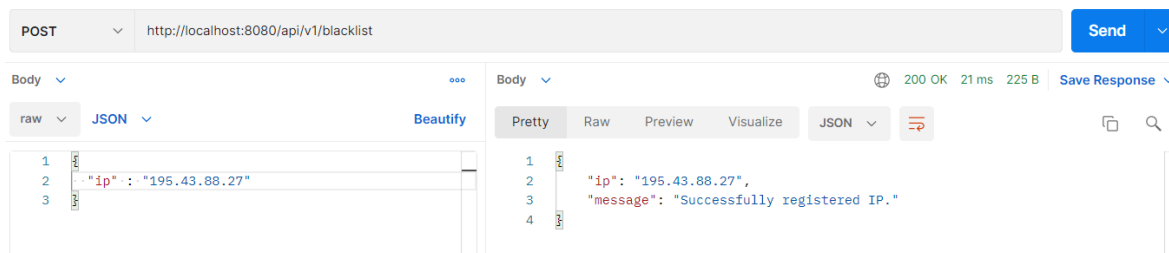
Si se intenta consultar la información del país y las API's externas no están disponibles, el servicio responderá un Bad Gateway (502), así como se observa en la siguiente imagen:



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:



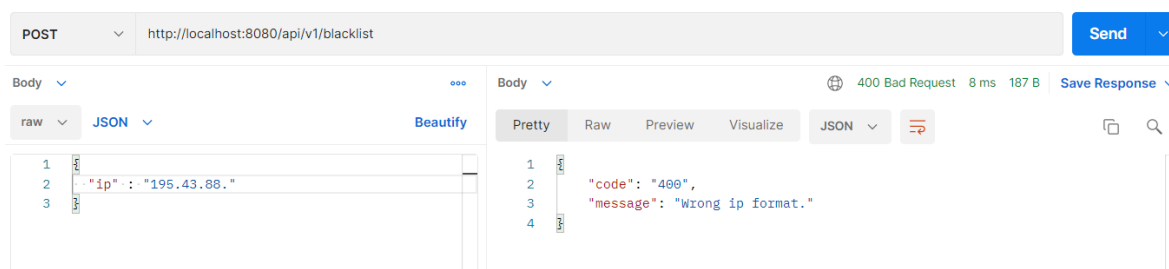
El segundo EndPoint es <http://localhost:8080/api/v1/blacklist> con sus métodos de GET y POST; el primer método retorna el listado de las IP's registradas en la lista negra, mientras que el segundo registra en el listado la IP.



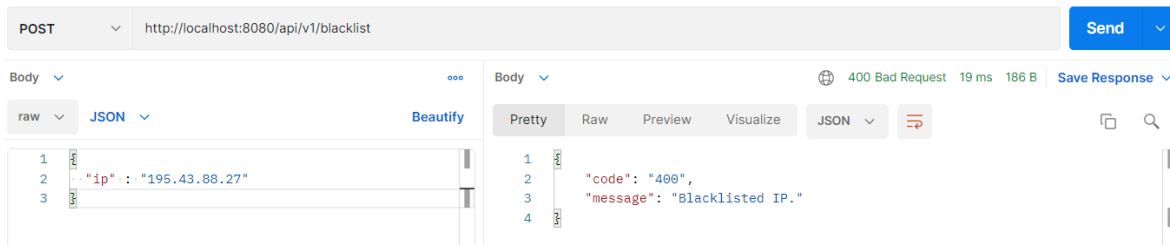
El servicio ha retornado el mensaje que indica que la IP fue registrada satisfactoriamente **“Successfully registered IP.”**, con código HTTP 200 de respuesta.

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:

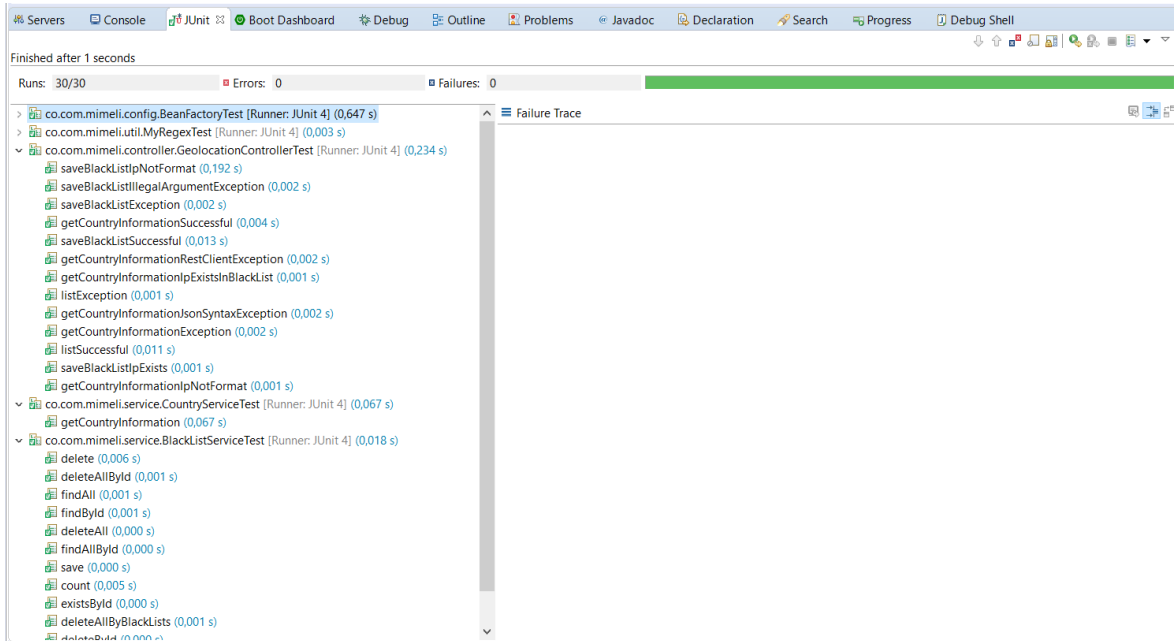


Si se intenta consultar una IP que ya está registrada en la lista negra, el servicio responderá un Bad Request (400), con el mensaje de IP ya registrada en lista negra **“Blacklisted IP.”**, así como se observa en la imagen:



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:

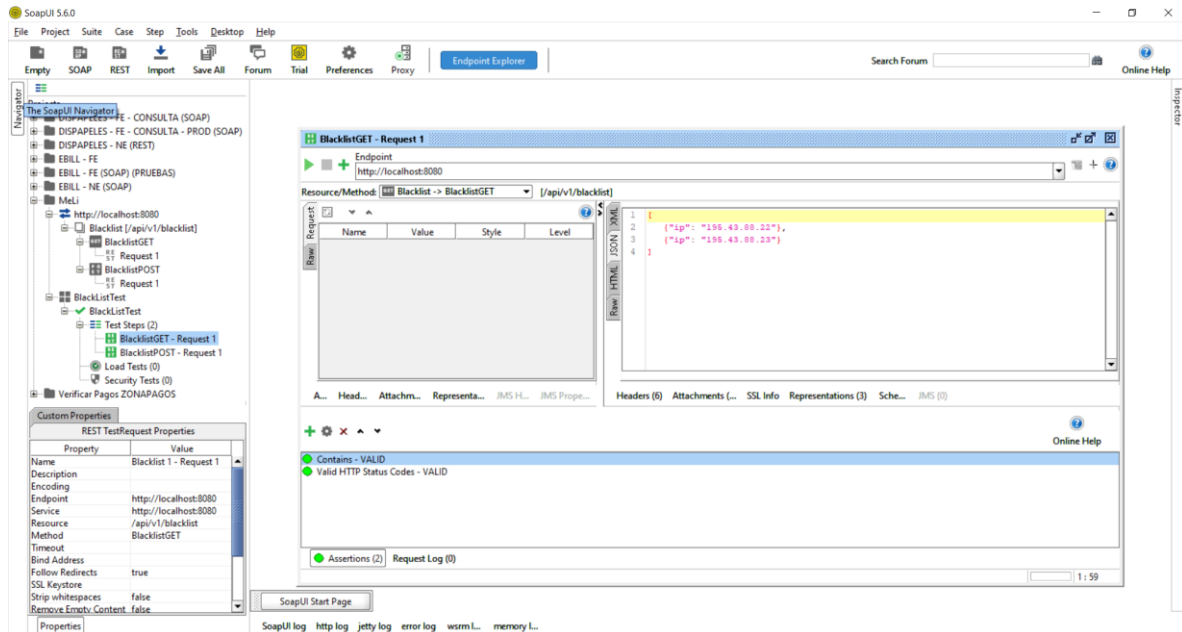
The screenshot shows the Coverage tool interface. It displays a table with columns: Element, Coverage, Covered Instru..., and Missed Instru... The table lists various project elements and their coverage percentages.

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	

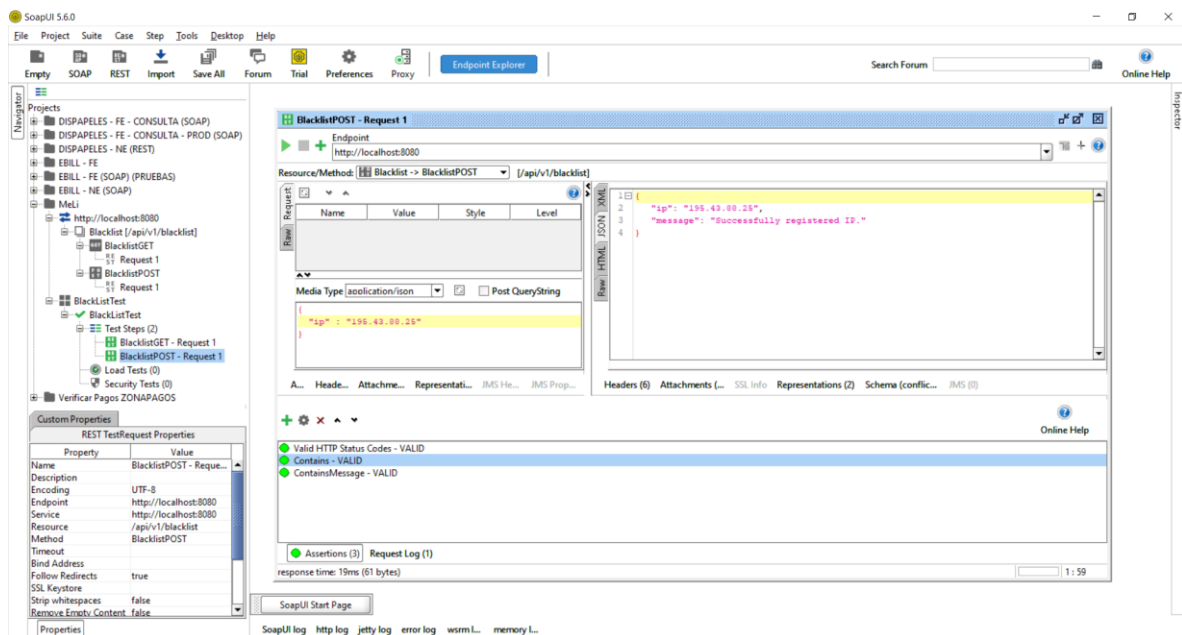
PRUEBAS FUNCIONALES (SoapUI)

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)

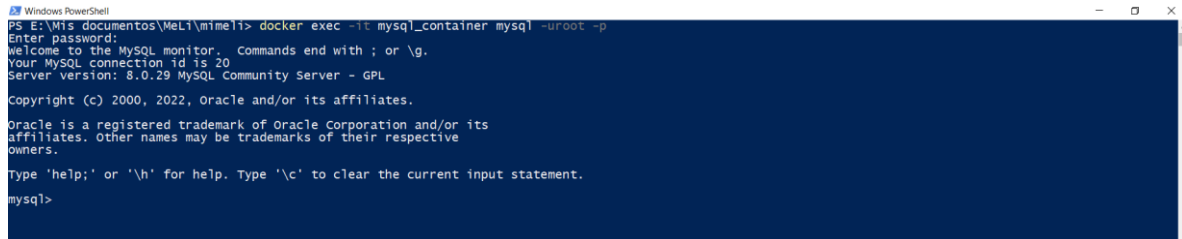


NOTA: Se observan en ambas pruebas las respectivas **“Assertions”**.

CONFIGURACIÓN DOCKER

Para el ejercicio se generarán dos imágenes docker con el fin de desplegar y validar los servicios desarrollados y toda su funcionalidad.

La primera imagen docker es para MySQL, y toda su configuración. Así que se descarga y se crea la imagen para la gestión de la base de datos, y luego validamos su funcionamiento y versión de la instalación, así como se observa en la siguiente imagen:



```
PS E:\Mis documentos\Meli\mimeli> docker exec -it mysql_container mysql -uroot -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 20
Server version: 8.0.29 MySQL Community Server - GPL

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

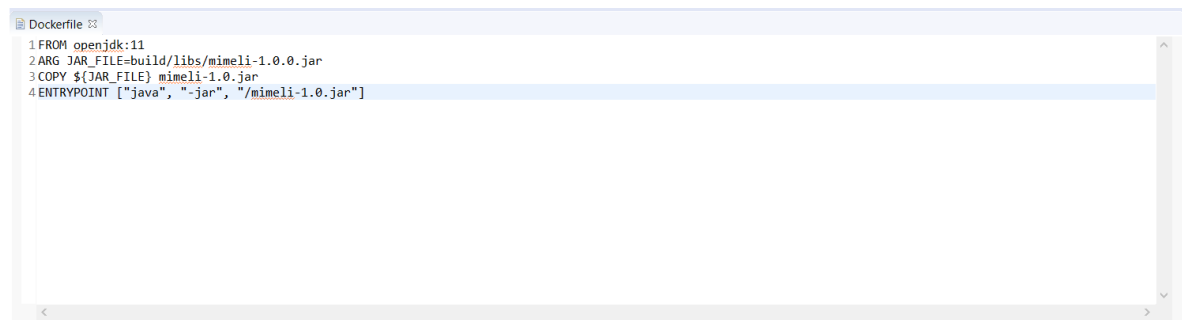
Luego se crea la base de datos **meli** con la tabla **blacklist** para persistir las IP's en lista negra.



```
mysql> SELECT * FROM blacklist;
+----+
| ip |
+----+
| 195.43.88.22 |
| 195.43.88.23 |
| 195.43.88.24 |
| 195.43.88.25 |
| 195.43.88.26 |
| 195.43.88.27 |
| 65.49.22.68 |
+----+
7 rows in set (0.00 sec)

mysql>
```

Ahora, para la segunda imagen que corresponde al servicio de nuestra API, se crea el archivo Dockerfile dentro de nuestro proyecto Spring.



```
Dockerfile
1 FROM openjdk:11
2 ARG JAR_FILE=build/libs/mimeli-1.0.0.jar
3 COPY ${JAR_FILE} mimeli-1.0.jar
4 ENTRYPOINT ["java", "-jar", "/mimeli-1.0.jar"]
```


Y una vez creado el archivo, nos ubicamos dentro del mismo directorio y abrimos la consola para crear la nueva imagen.

```
Windows PowerShell
PS E:\Mis documentos\Meli\mimeli> docker build -t mimeli .

Windows PowerShell
PS E:\Mis documentos\Meli\mimeli> docker build -t mimeli .
[+] Building 2.9s (7/7) FINISHED
=> [internal] load build definition from Dockerfile                                0.4s
=> => transferring dockerfile: 32B                                                0.3s
=> [internal] load .dockerignore                                                  0.3s
=> => transferring context: 2B                                                    0.3s
=> [internal] load metadata for docker.io/library/openjdk:11                    2.3s
=> [internal] load build context                                                  0.1s
=> => transferring context: 100B                                                  0.1s
=> [1/2] FROM docker.io/library/openjdk:11@sha256:f1d7736dda6d8ba321025433f7e7878ba3a0bc0631b0c3fa06da9415a5da3f57 0.0s
=> CACHED [2/2] COPY build/libs/mimeli-1.0.0.jar mimeli-1.0.jar                 0.0s
=> exporting to image                                                            0.0s
=> => exporting layers                                                            0.0s
=> => writing image sha256:ed14525c815fd2fdebe9fa22475766796c2949cice8fa39d311324fa8637f7d6 0.0s
=> => naming to docker.io/library/mimeli                                         0.0s

Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them
PS E:\Mis documentos\Meli\mimeli>
```

Y luego de haber creado la imagen, listamos las que hay creadas hasta el momento.

```
Windows PowerShell
PS E:\Mis documentos\Meli\mimeli> docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
mimeli latest ed14525c815f 34 hours ago 700MB
mysql latest 96d0ae5ed60 11 days ago 524MB
PS E:\Mis documentos\Meli\mimeli>
```

Y por último, se ejecuta, creando el contenedor de nombre **mimeli_container**.

```
Windows PowerShell
PS E:\Mis documentos\Meli\mimeli> docker run --network meli-net -d -p 8080:8080 --name mimeli_container mimeli
2fdb48ebdc4d518283206f85ff4760bdee24ff41e76c235ba5c10d23a021a0c9
PS E:\Mis documentos\Meli\mimeli>

Windows PowerShell
PS E:\Mis documentos\Meli\mimeli> docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
2fdb48ebdc4d mimeli "java -jar /mimeli-1..." 3 minutes ago Up 3 minutes 0.0.0.0:8080->8080/tcp mimeli_container
ea00b23620f7 mysql "docker-entrypoint.s..." 35 hours ago Up 48 minutes 33060/tcp, 0.0.0.0:13306->3306/tcp mysql_container
PS E:\Mis documentos\Meli\mimeli>
```

Ejecución de los servicios ya instalados en Docker desde la herramienta Postman.

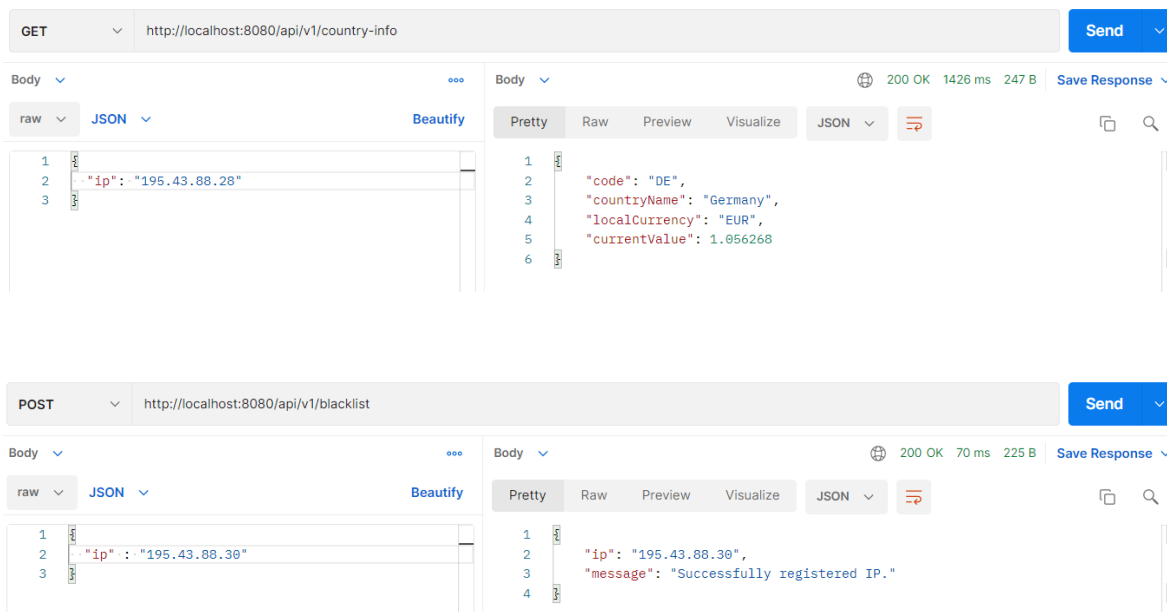
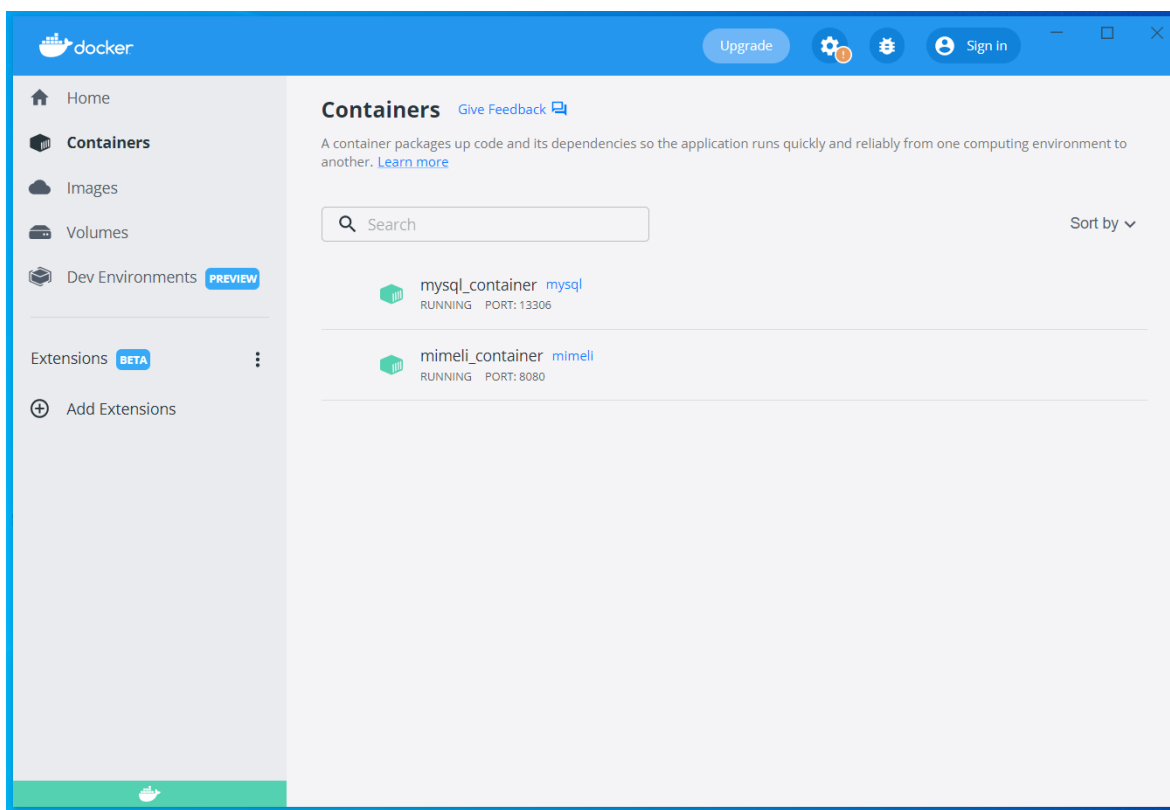


Imagen de Docker con el listado de contenedores creados.



Por último, se exportan las imágenes Docker.

```
Windows PowerShell
PS E:\Mis documentos\MeLi\mimeli> docker ps -a
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS                               NAMES
2fdb48ebdc4d   mimeli     "java -jar /mimeli-1..." 30 minutes ago Up 30 minutes 0.0.0.0:8080->8080/tcp             mimeli_container
ea00b23620f7   mysql     "docker-entrypoint.s..." 36 hours ago  Up About an hour 33060/tcp, 0.0.0.0:13306->3306/tcp mysql_container

PS E:\Mis documentos\MeLi\mimeli> docker save -o mysql.tar mysql
PS E:\Mis documentos\MeLi\mimeli> docker save -o mimeli.tar mimeli
PS E:\Mis documentos\MeLi\mimeli>
```

Nombre	Fecha de modificación	Tipo	Tamaño
.git	7/05/2022 4:47 p. m.	Carpeta de archivos	
.gradle	7/05/2022 6:09 p. m.	Carpeta de archivos	
.settings	4/05/2022 5:42 p. m.	Carpeta de archivos	
bin	4/05/2022 4:20 p. m.	Carpeta de archivos	
build	7/05/2022 10:03 p. m.	Carpeta de archivos	
gradle	4/05/2022 9:11 p. m.	Carpeta de archivos	
src	4/05/2022 9:11 p. m.	Carpeta de archivos	
.classpath	4/05/2022 4:20 p. m.	Archivo CLASSPATH	2 KB
.gitignore	4/05/2022 9:11 p. m.	Documento de tex...	1 KB
.project	4/05/2022 4:20 p. m.	Archivo PROJECT	1 KB
build.gradle	7/05/2022 9:30 p. m.	Archivo GRADLE	2 KB
Dockerfile	7/05/2022 9:32 p. m.	Archivo	1 KB
gradlew	4/05/2022 9:11 p. m.	Archivo	8 KB
gradlew	4/05/2022 9:11 p. m.	Archivo por lotes ...	3 KB
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
mimeli.tar	9/05/2022 8:46 a. m.	Archivo TAR	694.829 KB
mysql.tar	9/05/2022 8:47 a. m.	Archivo TAR	517.054 KB
settings.gradle	7/05/2022 6:07 p. m.	Archivo GRADLE	1 KB

Y también el 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