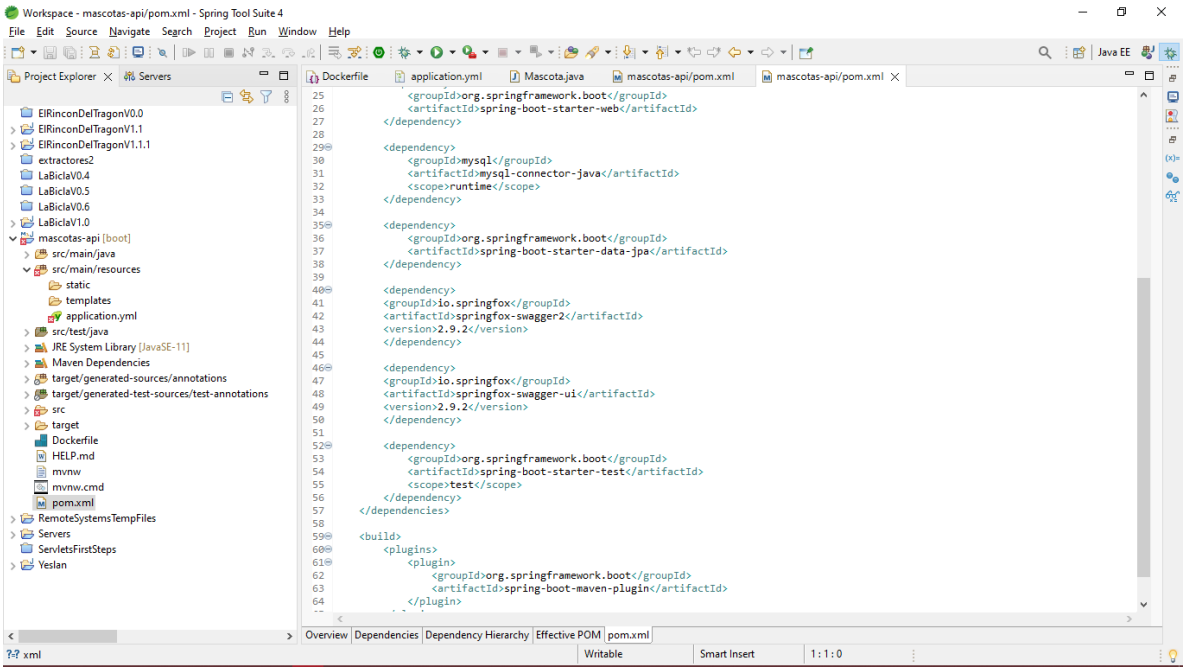
Practica Microservicios

(mascotas-api)

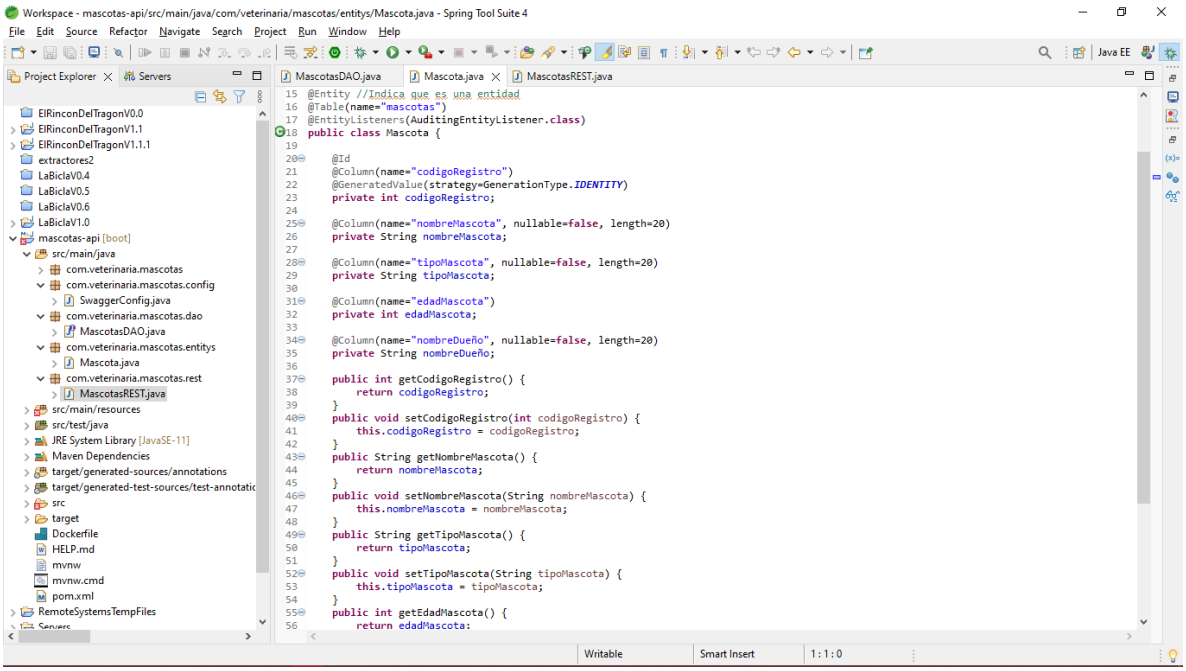
La práctica se comenzó seleccionando las siguientes tecnologías para su desarrollo:

- El lenguaje de Java mediante su jdk11.
- El framework de Spring Boot 2.6.3.
- Hibernate para la persistencia de datos que incluye como dependencia el framework.
- Maven para la generación del JAR.
- MySQL para la creación de la base de datos.
- Docker para la contenerizar los componentes.

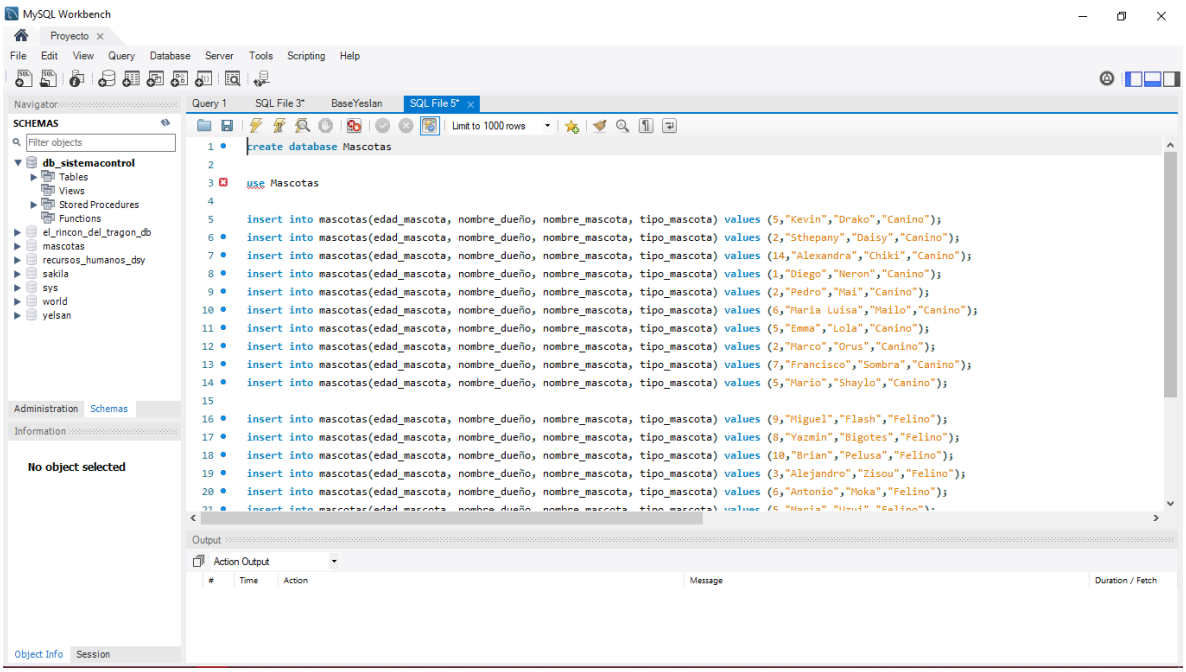
La practica se comienza creando el proyecto en un IDE especializado en la creación de proyectos Spring. Se comienza por la integración de las dependencias en el archivo **pom.xml**



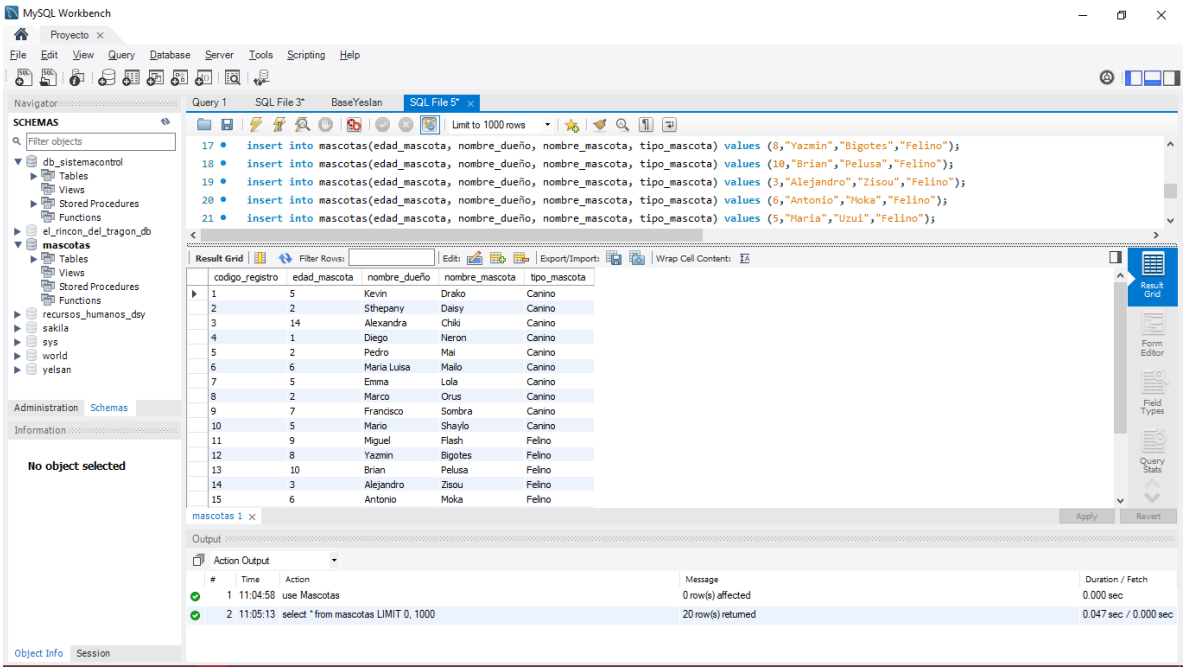
A continuación, se crean las propiedades y los métodos de acceso del objeto que consultaremos y/o devolveremos, utilizando el etiquetado que ofrece Spring, con el que podemos decirle que genere en contenido dentro de una base de datos.



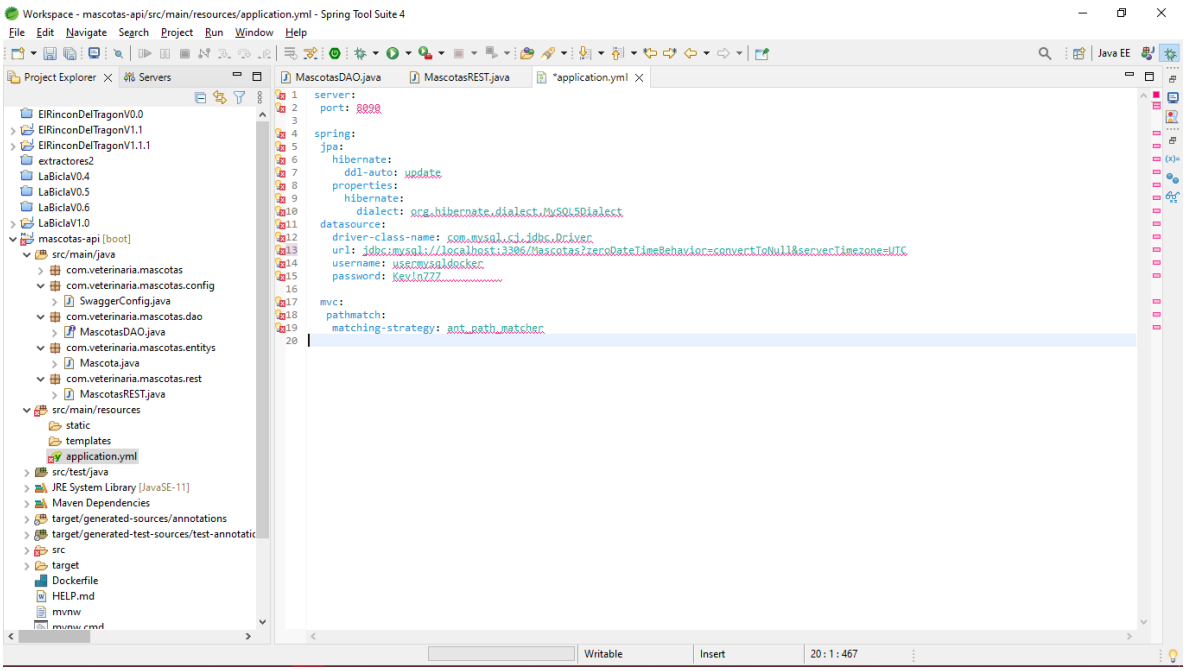
Estas propiedades, mediante su etiquetado indican los campos que contendrá la tabla “mascotas”, por lo que únicamente se deberá de crear la base de datos.



Una vez creada la base de datos y que se hayan generado las tablas al compilar el proyecto, podemos insertar datos para nuestras pruebas con la API.

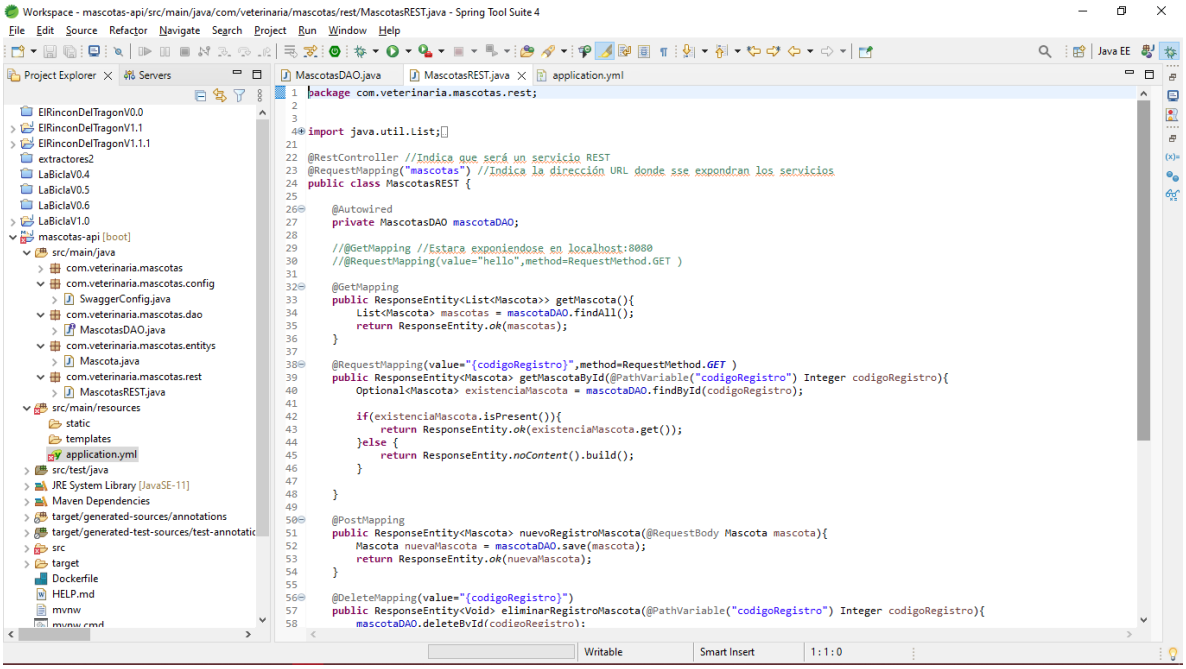


Lo siguiente es establecer la conexión a la base de datos dentro del proyecto mediante el archivo “application”, el cual se utiliza en el formato yml. Aquí se indica el puerto en donde se quiere desplegar la API y la dirección de la base de datos a donde se consumirán los datos.



Ya que se comprueba que hay conexión a la base de datos se comienzan a crear los servicios rest.

Se comienza por el servicio **GET**, dicho servicio se divide en dos, uno con el que obtenemos todos los datos de la base de datos que consultemos y otro que obtiene solo un dato en específico mediante su id.

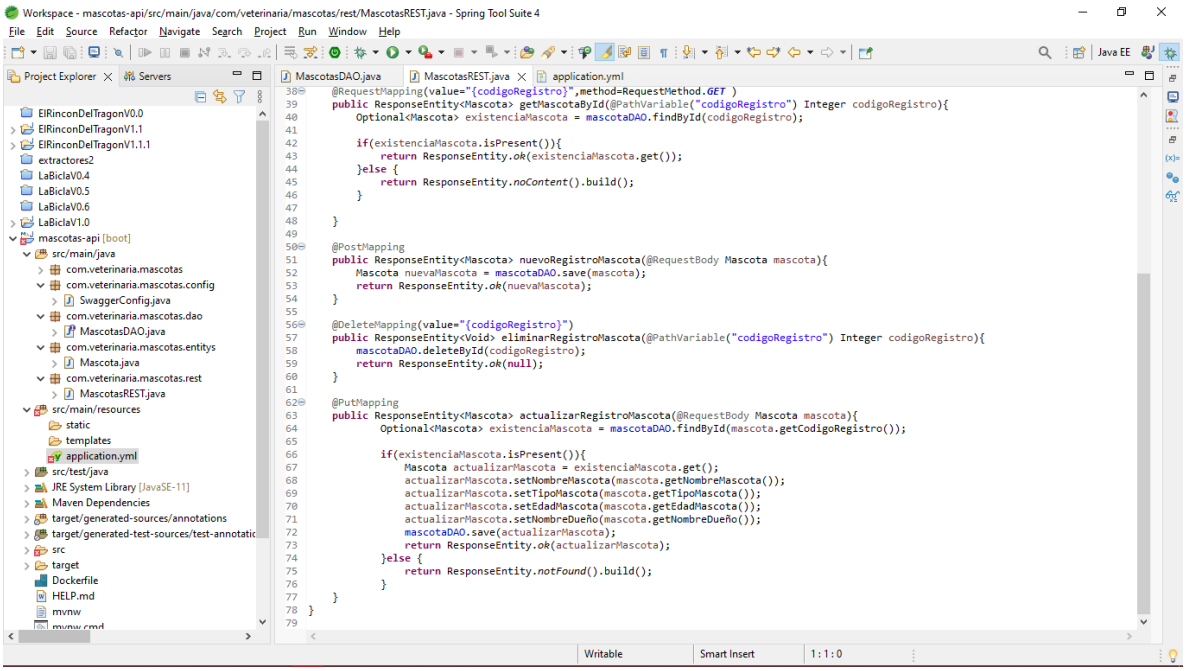



```
[{"codigoRegistro":1,"nombreMascota":"Drako","tipoMascota":"Canino","edadMascota":5,"nombreDueño":"Kevin"},
{"codigoRegistro":2,"nombreMascota":"Daisy","tipoMascota":"Canino","edadMascota":2,"nombreDueño":"Sthepany"},
{"codigoRegistro":3,"nombreMascota":"Chiki","tipoMascota":"Canino","edadMascota":14,"nombreDueño":"Alexandra"},
{"codigoRegistro":4,"nombreMascota":"Neron","tipoMascota":"Canino","edadMascota":1,"nombreDueño":"Diego"},
{"codigoRegistro":5,"nombreMascota":"Mai","tipoMascota":"Canino","edadMascota":2,"nombreDueño":"Pedro"},
{"codigoRegistro":6,"nombreMascota":"Mailo","tipoMascota":"Canino","edadMascota":6,"nombreDueño":"Maria Luisa"},
{"codigoRegistro":7,"nombreMascota":"Lola","tipoMascota":"Canino","edadMascota":5,"nombreDueño":"Emma"},
{"codigoRegistro":8,"nombreMascota":"Orus","tipoMascota":"Canino","edadMascota":2,"nombreDueño":"Marco"},
{"codigoRegistro":9,"nombreMascota":"Sombra","tipoMascota":"Canino","edadMascota":7,"nombreDueño":"Francisco"},
{"codigoRegistro":10,"nombreMascota":"Shaylo","tipoMascota":"Canino","edadMascota":5,"nombreDueño":"Mario"},
{"codigoRegistro":11,"nombreMascota":"Flash","tipoMascota":"Felino","edadMascota":9,"nombreDueño":"Miguel"},
{"codigoRegistro":12,"nombreMascota":"Bigotes","tipoMascota":"Felino","edadMascota":8,"nombreDueño":"Yazmin"},
{"codigoRegistro":13,"nombreMascota":"Pelusa","tipoMascota":"Felino","edadMascota":10,"nombreDueño":"Brian"},
{"codigoRegistro":14,"nombreMascota":"Zisou","tipoMascota":"Felino","edadMascota":3,"nombreDueño":"Alejandro"},
{"codigoRegistro":15,"nombreMascota":"Moka","tipoMascota":"Felino","edadMascota":6,"nombreDueño":"Antonio"},
{"codigoRegistro":16,"nombreMascota":"Uzui","tipoMascota":"Felino","edadMascota":5,"nombreDueño":"Maria"},
{"codigoRegistro":17,"nombreMascota":"Mora","tipoMascota":"Felino","edadMascota":3,"nombreDueño":"Corina"},
{"codigoRegistro":18,"nombreMascota":"Creed","tipoMascota":"Felino","edadMascota":9,"nombreDueño":"Antonio"},
{"codigoRegistro":19,"nombreMascota":"Drago","tipoMascota":"Felino","edadMascota":5,"nombreDueño":"Angel"},
{"codigoRegistro":20,"nombreMascota":"Boss","tipoMascota":"Felino","edadMascota":7,"nombreDueño":"Jaen"}]
```



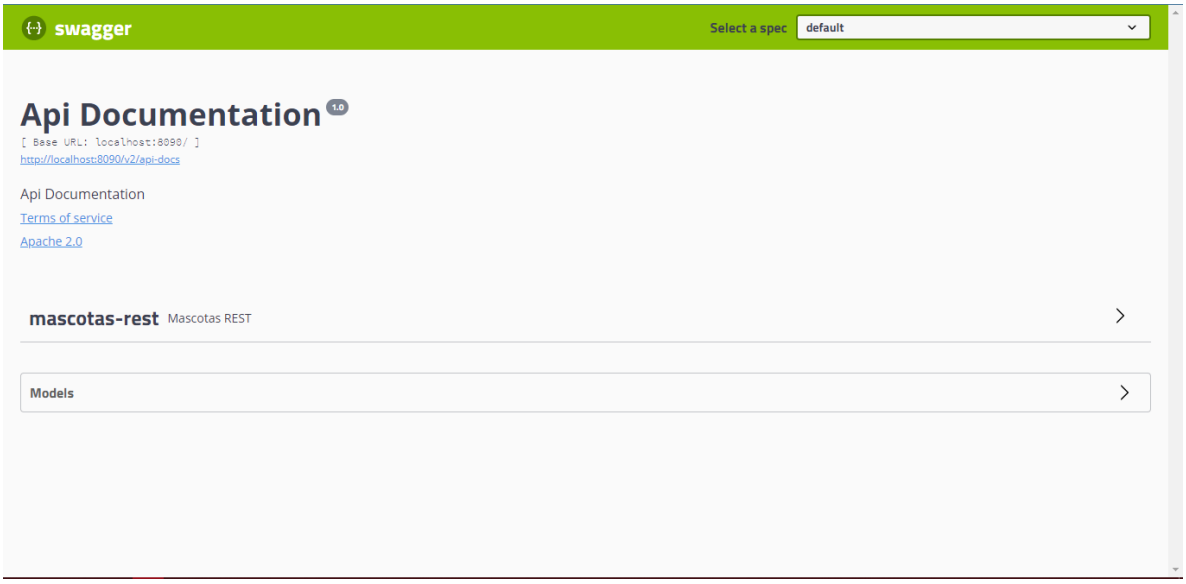
```
{"codigoRegistro":5,"nombreMascota":"Mai","tipoMascota":"Canino","edadMascota":2,"nombreDueño":"Pedro"}
```

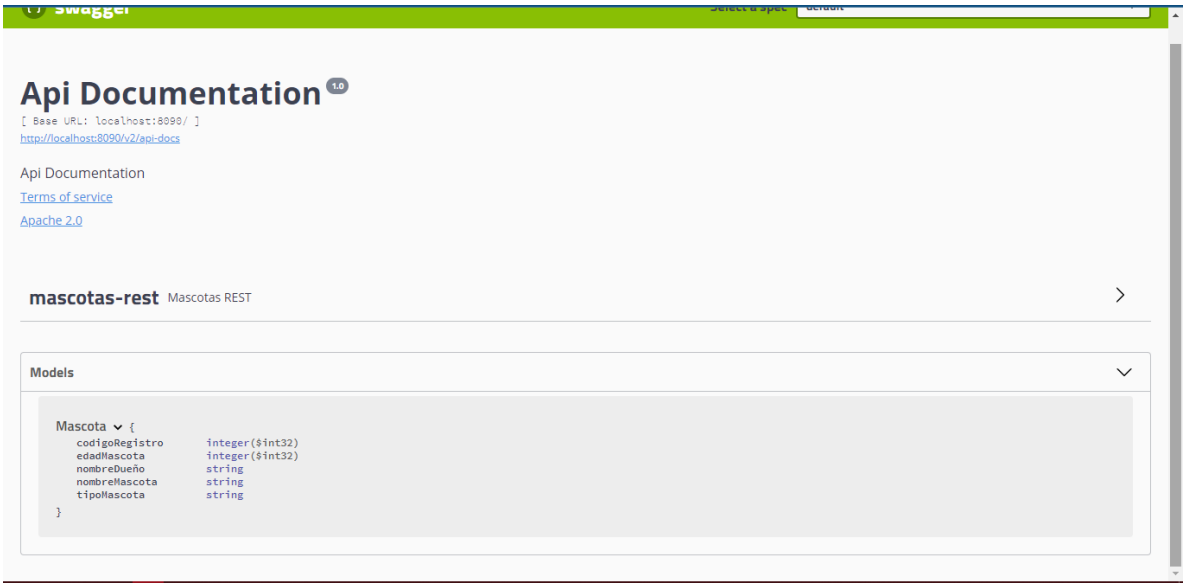
Se continua con los métodos **POST, PUT, DELETE**, con los cuales se obtiene una API RESTful, y para definirlos según su tipo de utiliza el etiquetado que ofrece Spring, dentro de esas etiquetas se ingresan las variables correspondientes al servicio, como lo son la ruta en donde se encontrara y el parámetro de entrada que solicitara para otorgar el servicio.

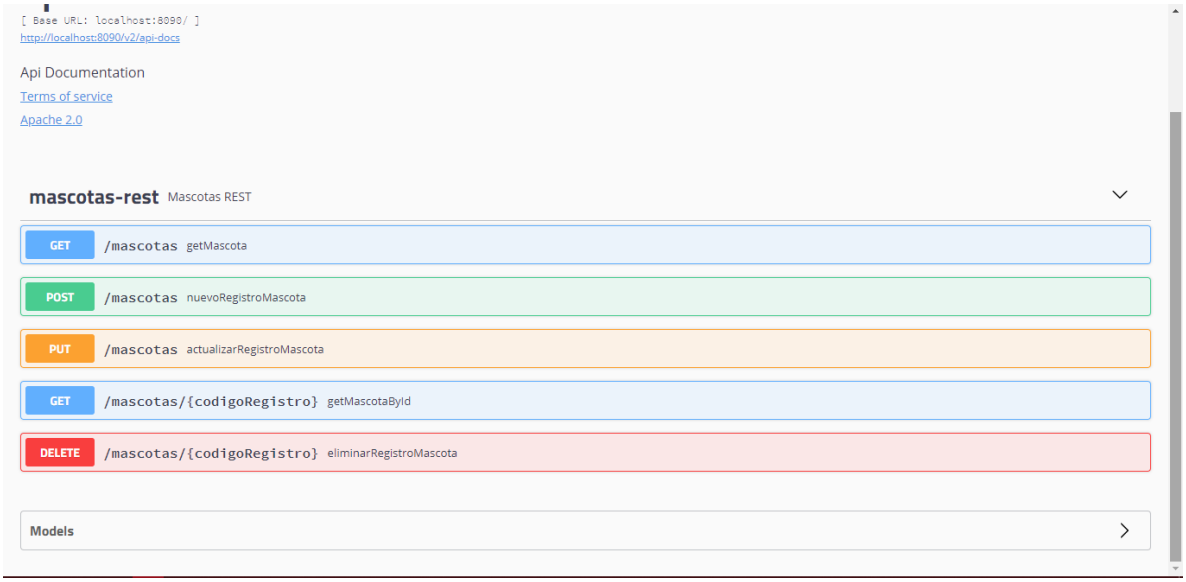


Ya que están creados los servicios se pueden probar y hacer su documentación, la documentación o contrato de la API se crea en Swagger, utilizando una herramienta del mismo Spring que permite integrar a Swagger en las dependencias, así como hacer que esta funcione documentando todo el proyecto de forma automatiza.

La documentación se puede ver en la ruta de **“servidor”/swagger-ui.html**




swagger
Api Documentation
1.0
[Base URL: localhost:8090/]
http://localhost:8090/v2/api-docs
Api Documentation
Terms of service
Apache 2.0
mascotas-rest Mascotas REST
Models
Mascota {
codigoRegistro integer($int32)
edadMascota integer($int32)
nombreDueño string
nombreMascota string
tipoMascota string
}


[Base URL: localhost:8090/]
http://localhost:8090/v2/api-docs
Api Documentation
Terms of service
Apache 2.0
mascotas-rest Mascotas REST
GET /mascotas getMascota
POST /mascotas nuevoRegistroMascota
PUT /mascotas actualizarRegistroMascota
GET /mascotas/{codigoRegistro} getMascotaById
DELETE /mascotas/{codigoRegistro} eliminarRegistroMascota
Models

Code

Details

200

Response body


```
[
  {
    "codigoRegistro": 1,
    "nombreMascota": "Drako",
    "tipoMascota": "Canino",
    "edadMascota": 5,
    "nombreDueño": "Kevin"
  },
  {
    "codigoRegistro": 2,
    "nombreMascota": "Daisy",
    "tipoMascota": "Canino",
    "edadMascota": 2,
    "nombreDueño": "Sthepany"
  },
  {
    "codigoRegistro": 3,
    "nombreMascota": "Chiki",
    "tipoMascota": "Canino",
    "edadMascota": 14,
    "nombreDueño": "Alexandra"
  },
  {
    "codigoRegistro": 4,
    "nombreMascota": "Neron",
    "tipoMascota": "Canino",
```


Download

Response headers


```
connection: keep-alive
content-type: application/json
date: Mon, 24 Jan 2022 17:39:35 GMT
keep-alive: timeout=60
transfer-encoding: chunked
```


Responses

Name

Description

mascota * required

(body)

mascota

Example Value | Model


```
{
  "edadMascota": 6,
  "nombreDueño": "Jamarcus",
  "nombreMascota": "Taquito",
  "tipoMascota": "Canino"
}
```


Cancel

Parameter content type

application/json

Execute

Clear

Curl

```
curl -X POST "http://localhost:8090/mascotas" -H "accept: */*" -H "Content-Type: application/json" -d "{ \"edadMascota\": 6, \"nombreDueño\": \"Jamarcus\",
\"nombreMascota\": \"Taquito\", \"tipoMascota\": \"Canino\"}"
```

Request URL

```
http://localhost:8090/mascotas
```

Server response

Code	Details
200	Response body

```
{
  "codigoRegistro": 21,
  "nombreMascota": "Taquito",
  "tipoMascota": "Canino",
  "edadMascota": 6,
  "nombreDueño": "Jamarcus"
}
```

Download

Response headers

```
connection: keep-alive
content-type: application/json
date: Mon, 24 Jan 2022 17:41:54 GMT
keep-alive: timeout=60
transfer-encoding: chunked
```

Responses

Code	Description

Si volvemos a ejecutar el servicio **GET**

Code	Details
200	Response body

```
    "tipoMascota": "Felino",
    "edadMascota": 9,
    "nombreDueño": "Antonio"
  },
  {
    "codigoRegistro": 19,
    "nombreMascota": "Drago",
    "tipoMascota": "Felino",
    "edadMascota": 5,
    "nombreDueño": "Angel"
  },
  {
    "codigoRegistro": 20,
    "nombreMascota": "Boss",
    "tipoMascota": "Felino",
    "edadMascota": 7,
    "nombreDueño": "Jaen"
  },
  {
    "codigoRegistro": 21,
    "nombreMascota": "Taquito",
    "tipoMascota": "Canino",
    "edadMascota": 6,
    "nombreDueño": "Jamarcus"
  }
]
```

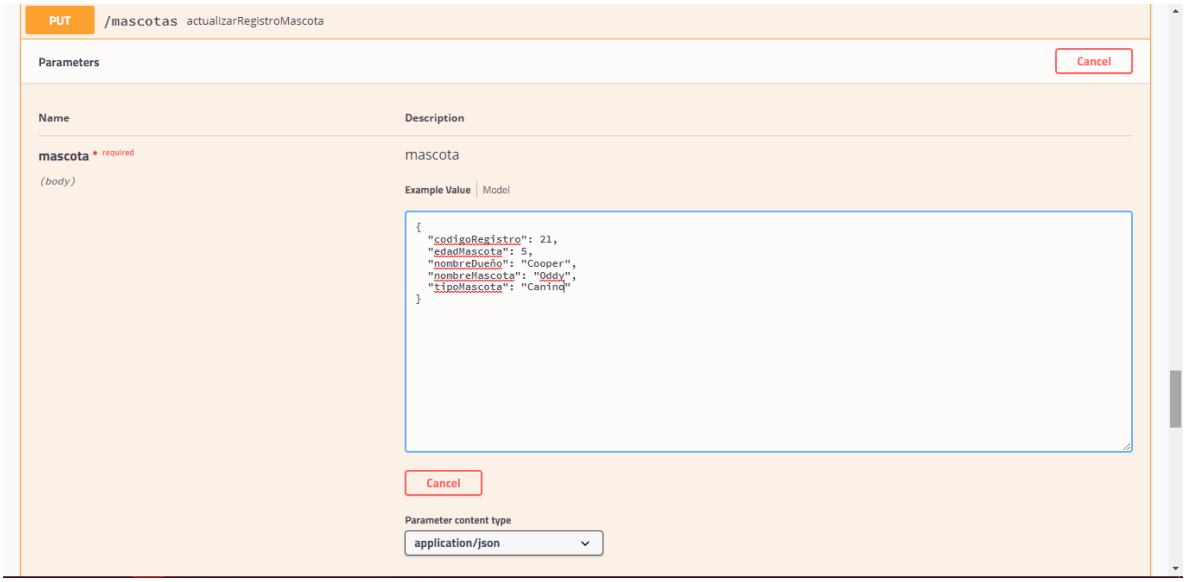
Download

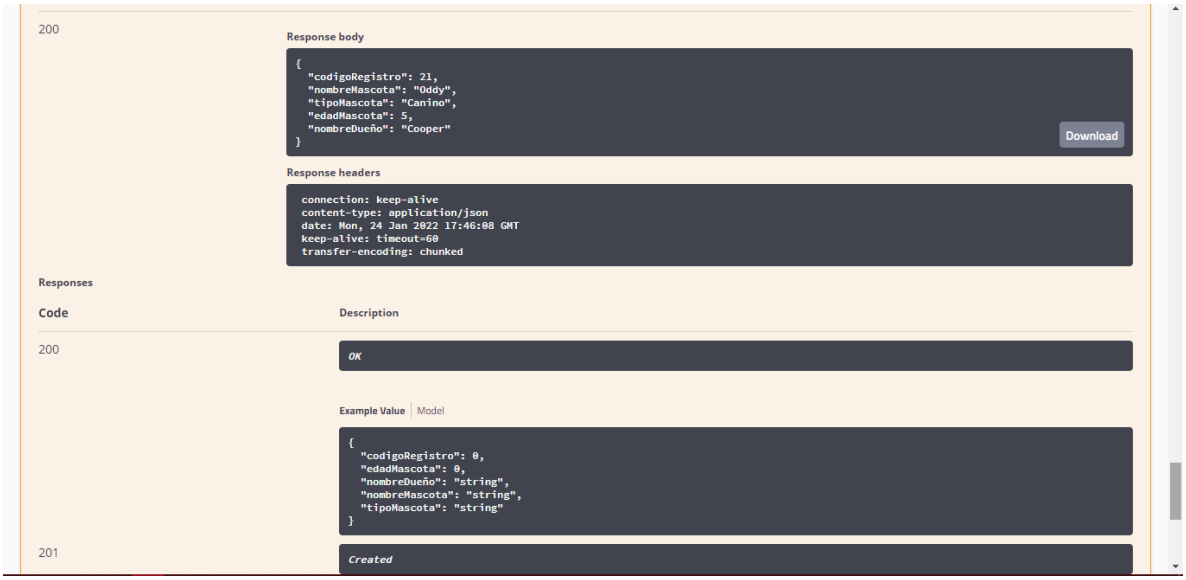
Response headers

```
connection: keep-alive
content-type: application/json
date: Mon, 24 Jan 2022 17:43:17 GMT
keep-alive: timeout=60
transfer-encoding: chunked
```

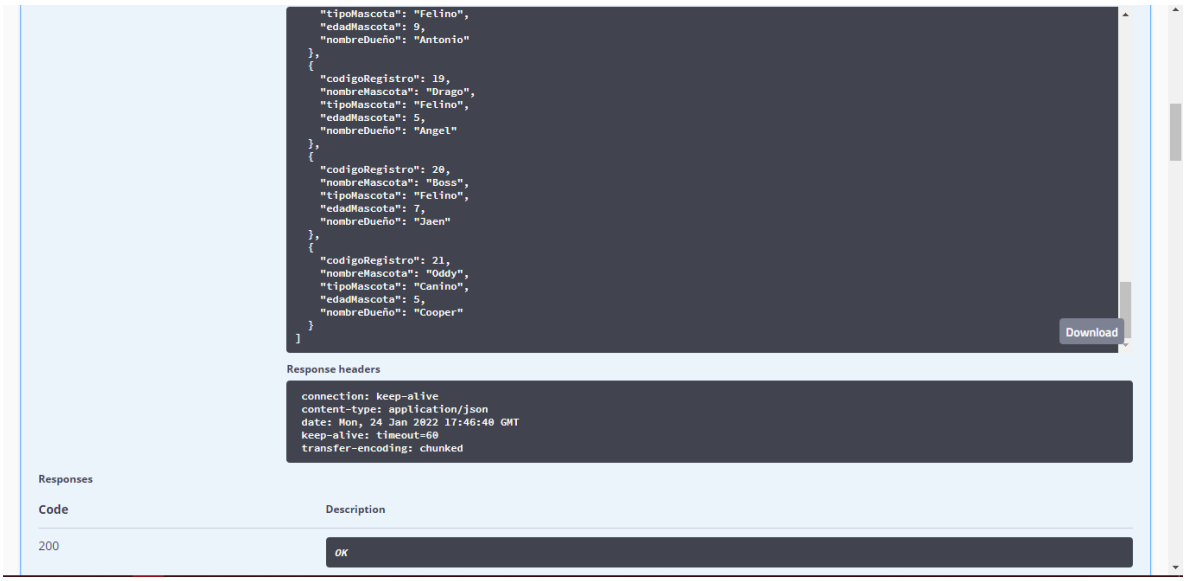
Responses

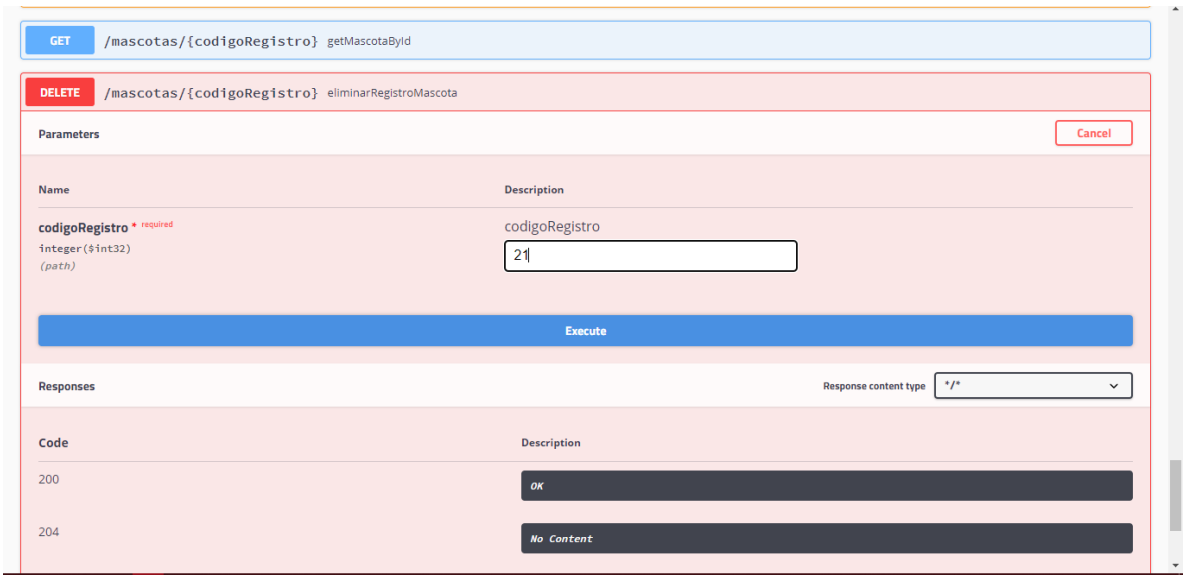
Code	Description

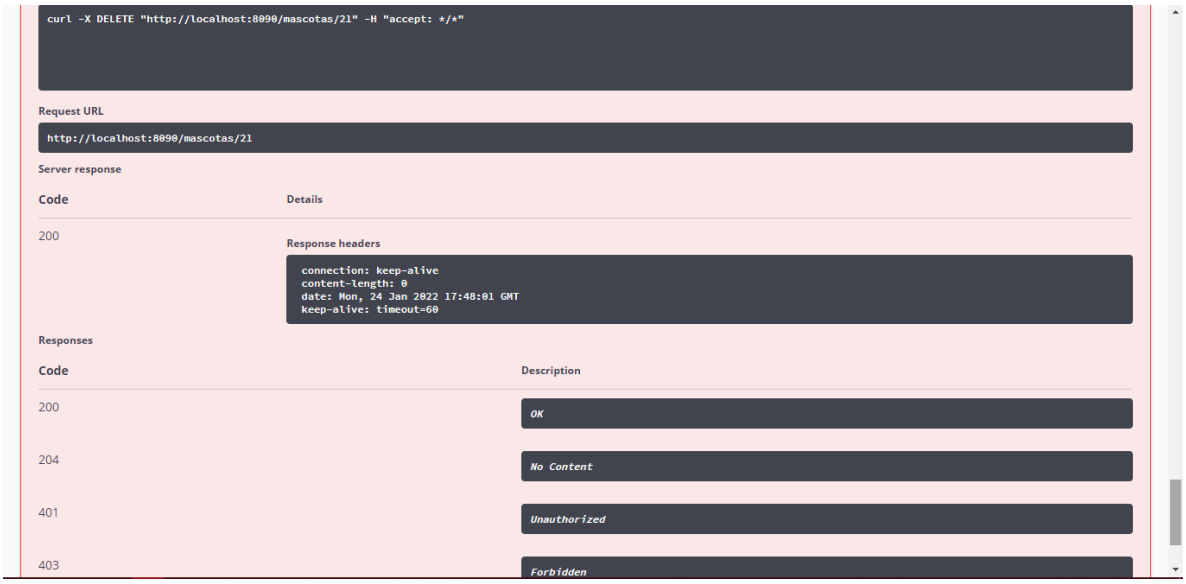

PUT
/mascotas actualizarRegistroMascota
Parameters
Cancel
Name
Description
mascota * required
(body)
mascota
Example Value | Model
{
"codigoRegistro": 21,
"edadMascota": 5,
"nombreDueño": "Cooper",
"nombreMascota": "Oddy",
"tipoMascota": "Canino"
}
Cancel
Parameter content type
application/json


200
Response body
{
"codigoRegistro": 21,
"nombreMascota": "Oddy",
"tipoMascota": "Canino",
"edadMascota": 5,
"nombreDueño": "Cooper"
}
Download
Response headers
connection: keep-alive
content-type: application/json
date: Mon, 24 Jan 2022 17:46:08 GMT
keep-alive: timeout=60
transfer-encoding: chunked
Responses
Code
Description
200
OK
Example Value | Model
{
"codigoRegistro": 0,
"edadMascota": 0,
"nombreDueño": "string",
"nombreMascota": "string",
"tipoMascota": "string"
}
201
Created

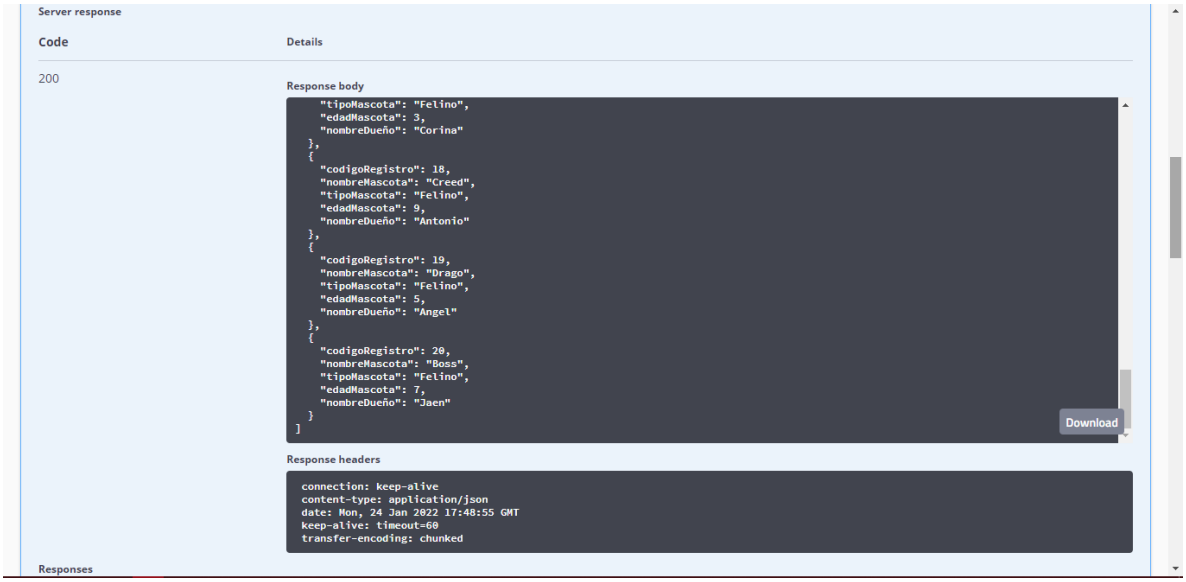
Si volvemos a ejecutar el servicio **GET**





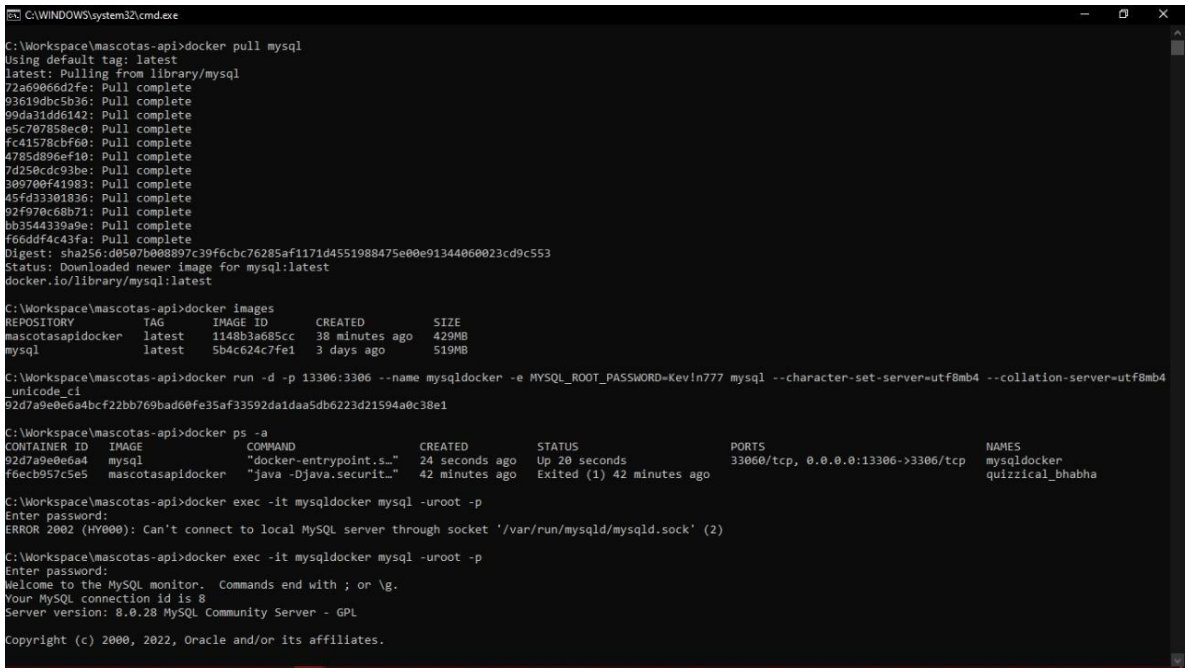


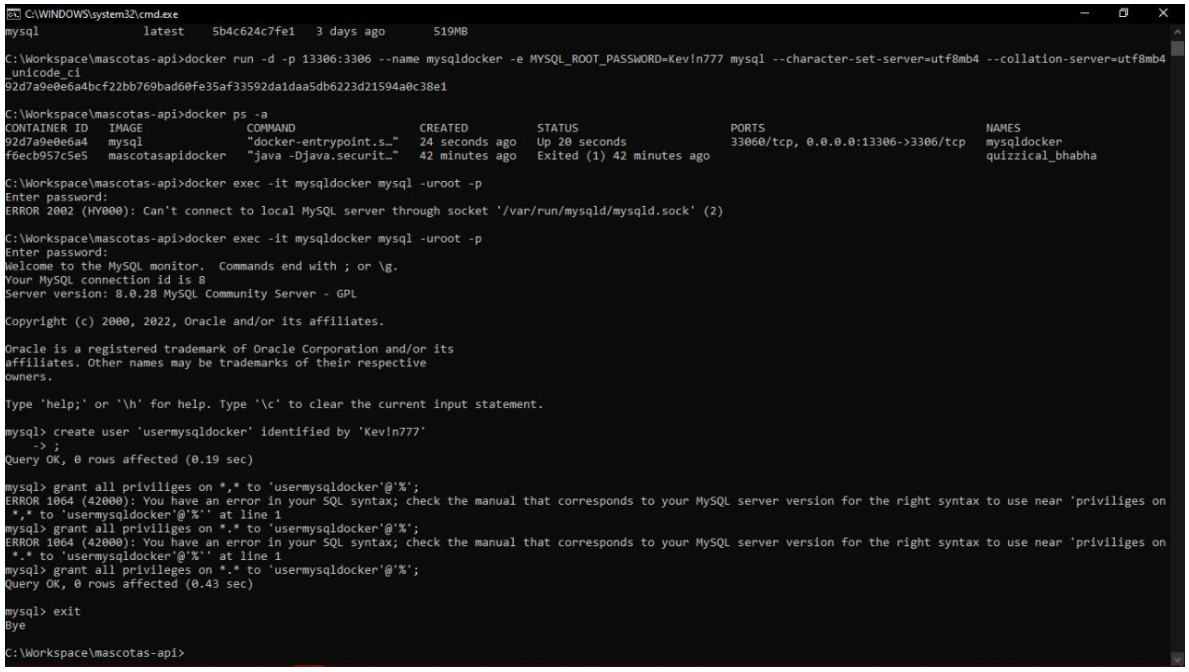
Si volvemos a ejecutar el servicio **GET**



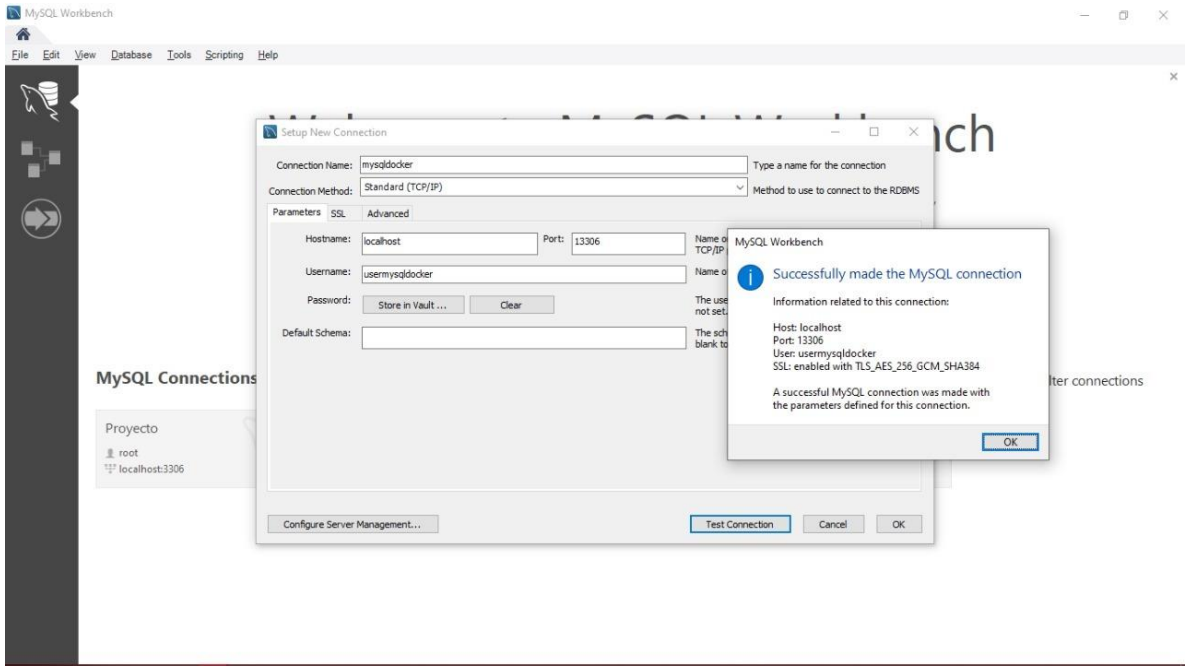
Ya que comprobamos que nuestra API funciona, lo siguiente es descargar Docker, con lo que podremos crear un contenedor para nuestra API. Y Maven con lo que podremos empaquetar nuestra aplicación en un archivo JAR.

Ya se instaló Docker, ahora se crea un contenedor para MySQL y se configura como se muestra a continuación.

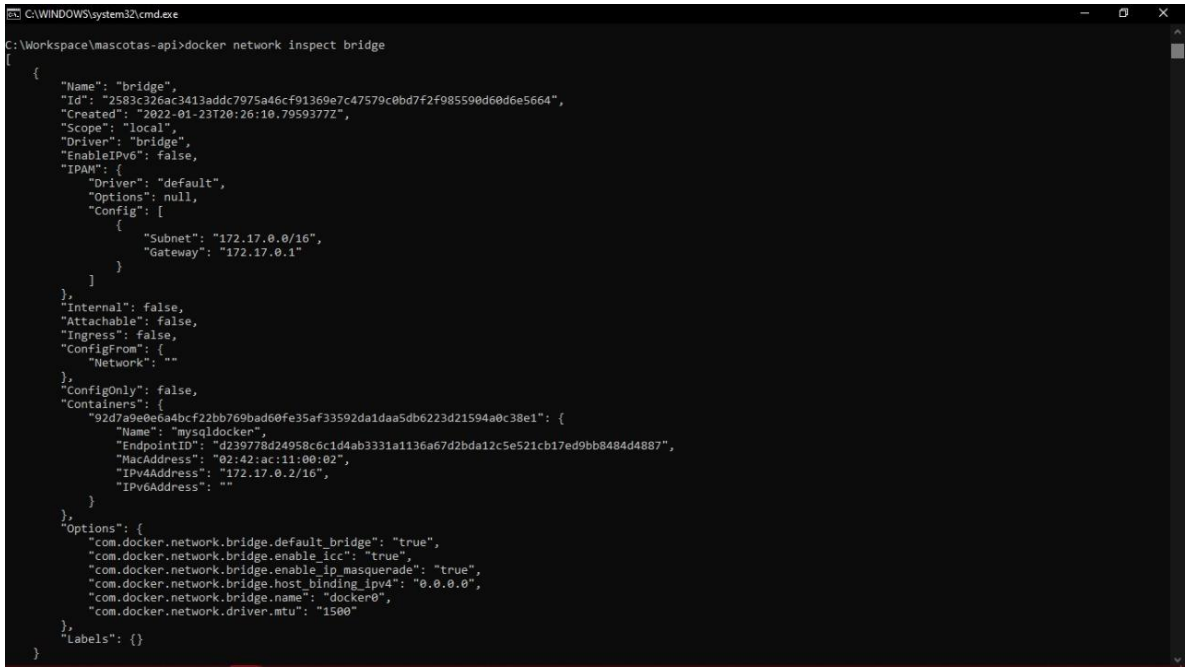


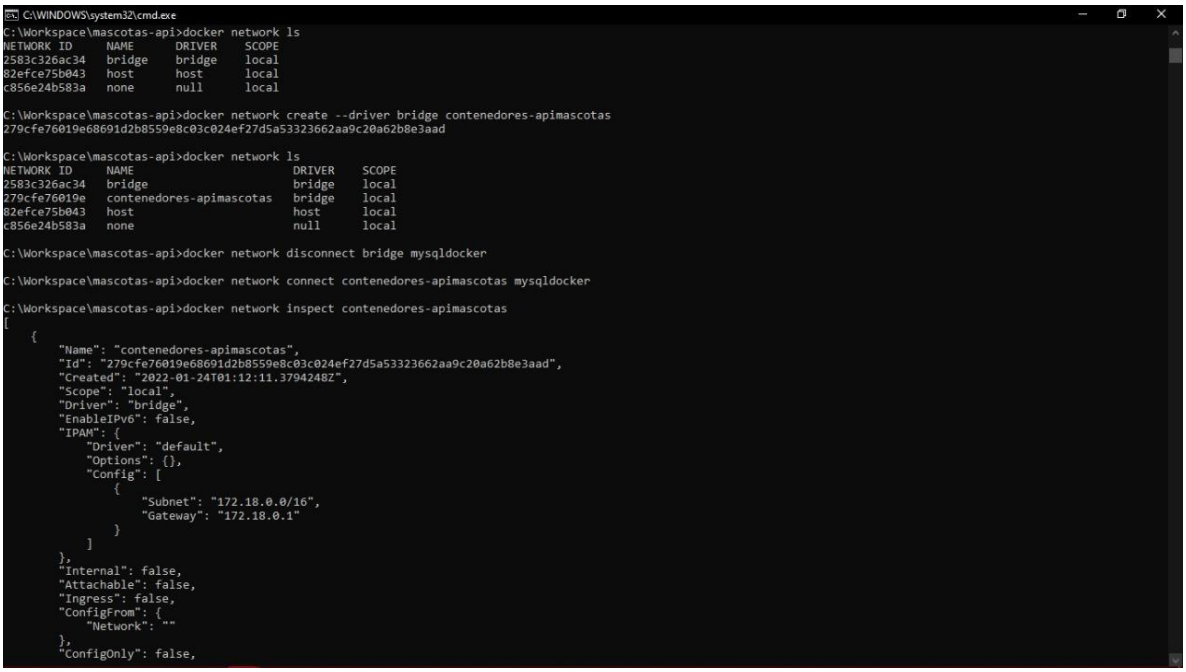


Probamos la conexión al contenedor mediante la herramienta de Workbench

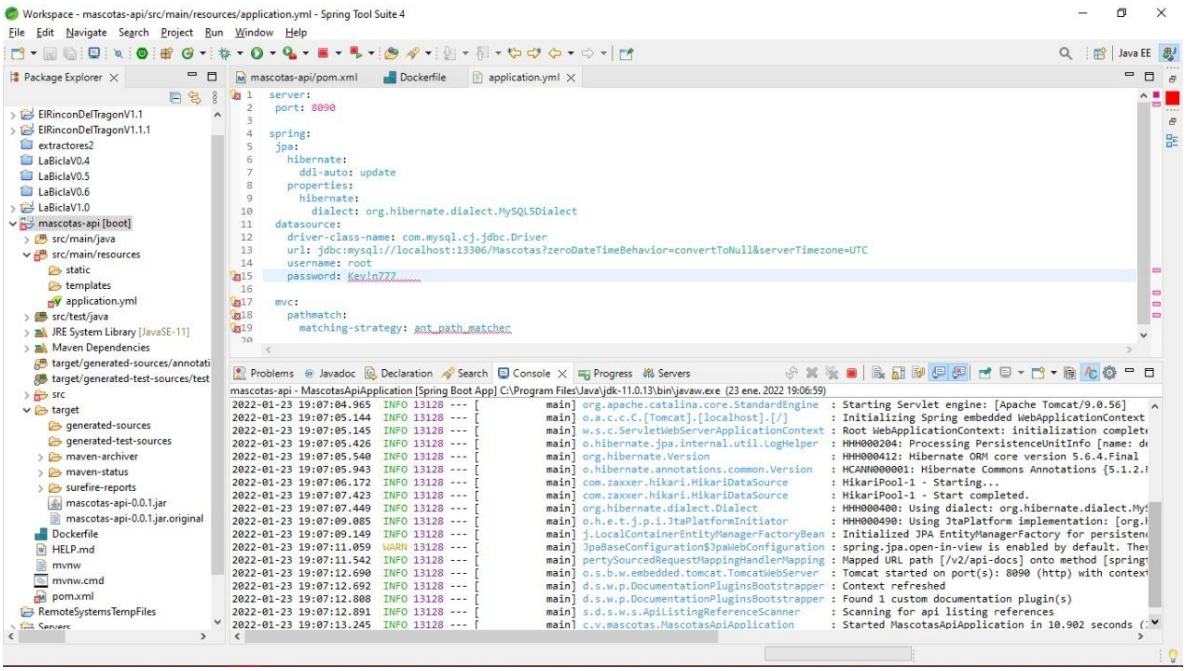


Ya que se comprueba que se puede acceder y hay conexión, procedemos a cambiarla de la red **bridge** que utiliza Docker, para introducirla en una red propia que llamamos **contenedores-apimascotas**.

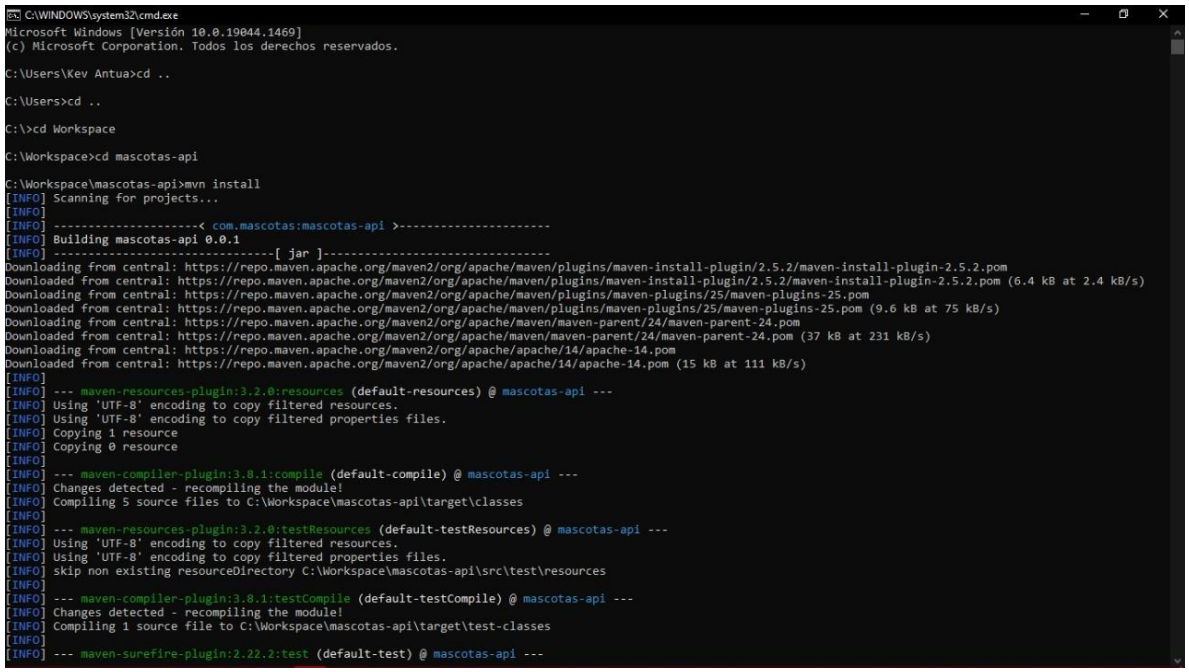




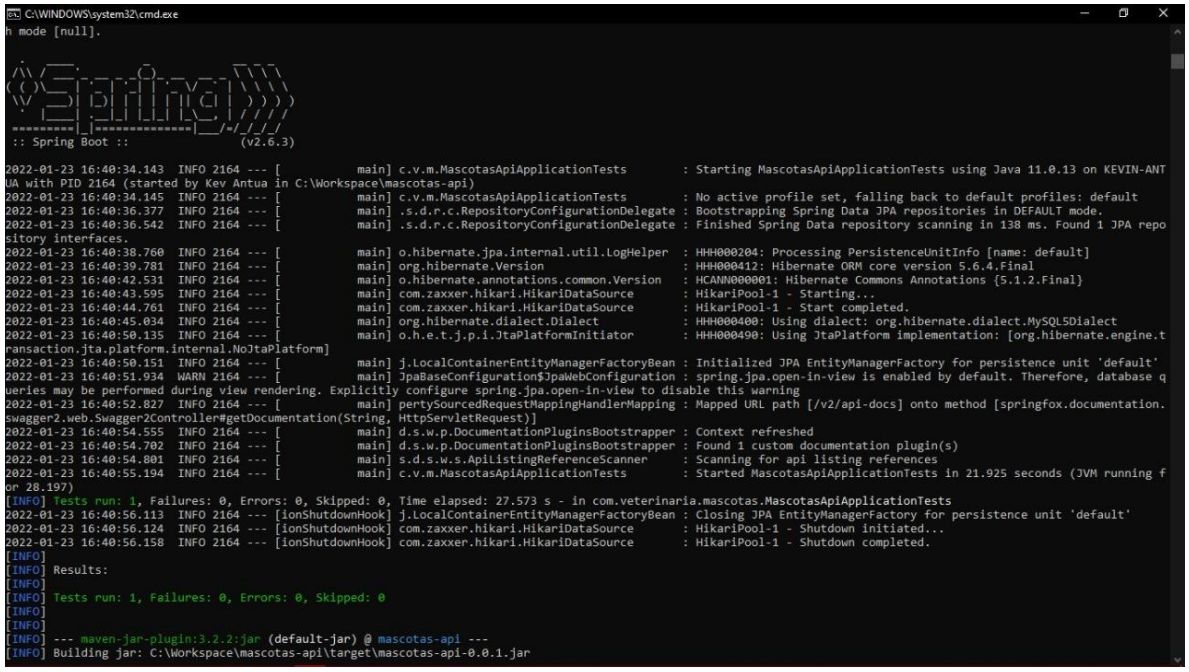
Ahora probamos la conexión a nuestro contenedor de MySQL desde nuestra API, apuntando a la dirección que le asignamos.



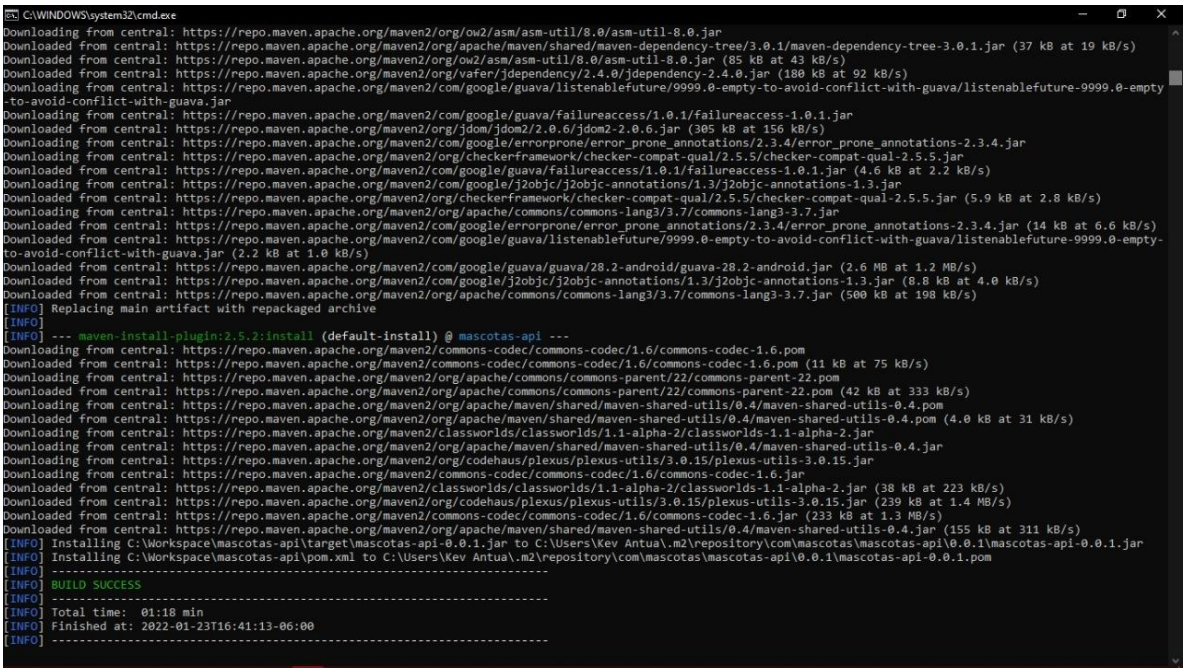
El Log muestra que se ha establecido la conexión sin ningún problema por lo que ahora se puede compilar un JAR.



```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Versión 10.0.19044.1469]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\Kev Antua>cd ..

C:\Users>cd ..

C:\>cd Workspace

C:\Workspace>cd mascotas-api

C:\Workspace\mascotas-api>mvn install
[INFO] Scanning for projects...
[INFO]
[INFO] ---------------------< com.mascotas:mascotas-api >----------------------
[INFO] Building mascotas-api 0.0.1
[INFO] --------------------------------[ jar ]---------------------------------
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-install-plugin/2.5.2/maven-install-plugin-2.5.2.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-install-plugin/2.5.2/maven-install-plugin-2.5.2.pom (6.4 kB at 2.4 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-plugins/25/maven-plugins-25.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-plugins/25/maven-plugins-25.pom (9.6 kB at 75 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-parent/24/maven-parent-24.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-parent/24/maven-parent-24.pom (37 kB at 231 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/apache/14/apache-14.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/apache/14/apache-14.pom (15 kB at 111 kB/s)
[INFO]
[INFO] --- maven-resources-plugin:3.2.0:resources (default-resources) @ mascotas-api ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] Using 'UTF-8' encoding to copy filtered properties files.
[INFO] Copying 1 resource
[INFO] Copying 0 resource
[INFO]
[INFO] --- maven-compiler-plugin:3.8.1:compile (default-compile) @ mascotas-api ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 5 source files to C:\Workspace\mascotas-api\target\classes
[INFO]
[INFO] --- maven-resources-plugin:3.2.0:testResources (default-testResources) @ mascotas-api ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] Using 'UTF-8' encoding to copy filtered properties files.
[INFO] skip non existing resourceDirectory C:\Workspace\mascotas-api\src\test\resources
[INFO]
[INFO] --- maven-compiler-plugin:3.8.1:testCompile (default-testCompile) @ mascotas-api ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 1 source file to C:\Workspace\mascotas-api\target\test-classes
[INFO]
[INFO] --- maven-surefire-plugin:2.22.2:test (default-test) @ mascotas-api ---
```

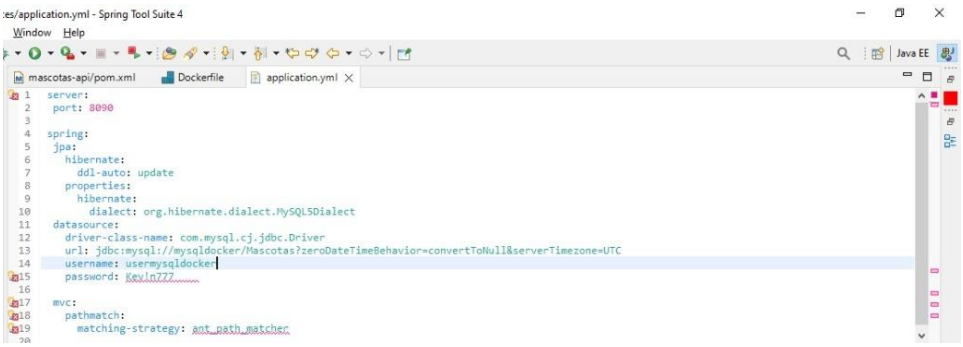


```
C:\WINDOWS\system32\cmd.exe
h mode [null].

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::                (v2.6.3)

2022-01-23 16:40:34.143  INFO 2164 --- [           main] c.v.m.MascotasApiApplicationTests        : Starting MascotasApiApplicationTests using Java 11.0.13 on KEVIN-ANT
UA with PID 2164 (started by Kev Antua in C:\Workspace\mascotas-api)
2022-01-23 16:40:34.145  INFO 2164 --- [           main] c.v.m.MascotasApiApplicationTests        : No active profile set, falling back to default profiles: default
2022-01-23 16:40:36.377  INFO 2164 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFAULT mode.
2022-01-23 16:40:36.542  INFO 2164 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 138 ms. Found 1 JPA repo
sitory interfaces.
2022-01-23 16:40:38.760  INFO 2164 --- [           main] o.hibernate.jpa.internal.util.LogHelper  : HHH000204: Processing PersistenceUnitInfo [name: default]
2022-01-23 16:40:39.781  INFO 2164 --- [           main] org.hibernate.Version                    : HHH000412: Hibernate ORM core version 5.6.4.Final
2022-01-23 16:40:42.531  INFO 2164 --- [           main] o.hibernate.annotations.common.Version   : HCANN000001: Hibernate Commons Annotations {5.1.2.Final}
2022-01-23 16:40:43.595  INFO 2164 --- [           main] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Starting...
2022-01-23 16:40:44.761  INFO 2164 --- [           main] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Start completed.
2022-01-23 16:40:45.034  INFO 2164 --- [           main] org.hibernate.dialect.Dialect            : HHH000400: Using dialect: org.hibernate.dialect.MySQL5Dialect
2022-01-23 16:40:50.135  INFO 2164 --- [           main] o.h.e.t.j.p.i.JtaPlatformInitiator       : HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.t
ransaction.jta.platform.internal.NoJtaPlatform]
2022-01-23 16:40:50.151  INFO 2164 --- [           main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'
2022-01-23 16:40:51.934  WARN 2164 --- [           main] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default. Therefore, database q
ueries may be performed during view rendering. Explicitly configure spring.jpa.open-in-view to disable this warning
2022-01-23 16:40:52.827  INFO 2164 --- [           main] pertySourcedRequestMappingHandlerMapping : Mapped URL path [/v2/api-docs] onto method [springfox.documentation.
swagger2.web.Swagger2Controller#getDocumentation(String, HttpServletRequest)]
2022-01-23 16:40:54.555  INFO 2164 --- [           main] d.s.w.p.DocumentationPluginsBootstrapper : Context refreshed
2022-01-23 16:40:54.702  INFO 2164 --- [           main] d.s.w.p.DocumentationPluginsBootstrapper : Found 1 custom documentation plugin(s)
2022-01-23 16:40:54.801  INFO 2164 --- [           main] s.d.s.w.s.ApiListingReferenceScanner     : Scanning for api listing references
2022-01-23 16:40:55.194  INFO 2164 --- [           main] c.v.m.MascotasApiApplicationTests        : Started MascotasApiApplicationTests in 21.925 seconds (JVM running f
or 28.197)
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 27.573 s - in com.veterinaria.mascotas.MascotasApiApplicationTests
2022-01-23 16:40:56.113  INFO 2164 --- [ionShutdownHook] j.LocalContainerEntityManagerFactoryBean : Closing JPA EntityManagerFactory for persistence unit 'default'
2022-01-23 16:40:56.124  INFO 2164 --- [ionShutdownHook] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Shutdown initiated...
2022-01-23 16:40:56.158  INFO 2164 --- [ionShutdownHook] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Shutdown completed.
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO]
[INFO] --- maven-jar-plugin:3.2.2:jar (default-jar) @ mascotas-api ---
[INFO] Building jar: C:\Workspace\mascotas-api\target\mascotas-api-0.0.1.jar
```

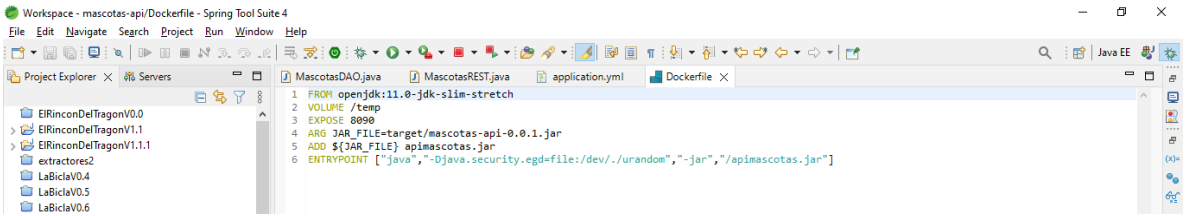


Una vez que se compila, el API, puede ser utilizado, aún sin estar dentro del contenedor, sin embargo, también se trata de meter en uno, por lo que ahora, se deberá cambiar la ruta de conexión a la base de datos, en lugar de poner la dirección y el puerto, se pondrá el nombre del contenedor creado que es **mysqldocker.**

De igual manera se crea el archivo **Dockerfile,** quedando ambas configuraciones de la siguiente manera.

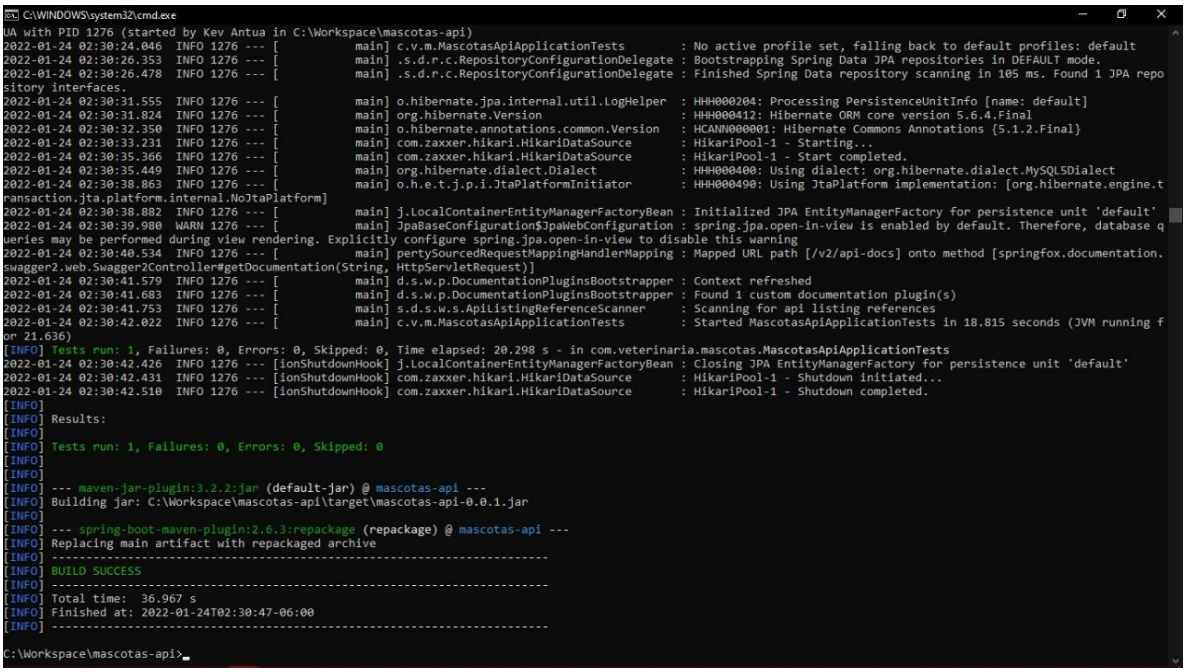


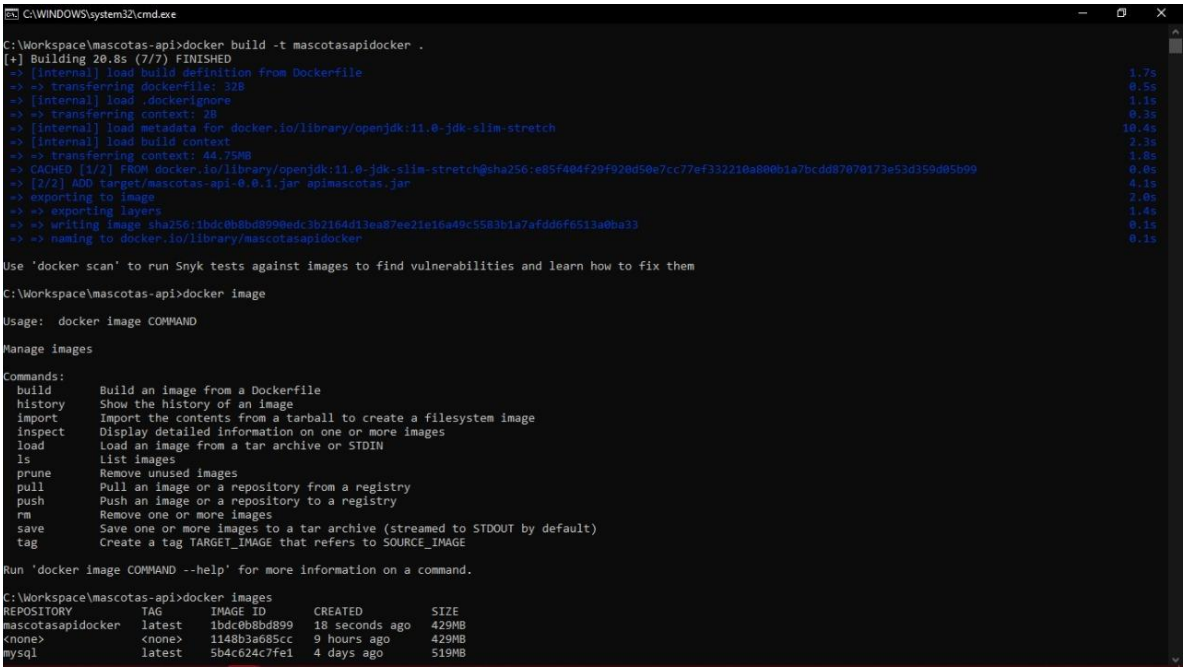
En el Docker file describimos la imagen que queremos para nuestro contendor, así como de que archivo será el que ejecute, su ubicación y como se identificará.



Sin embargo, aquí nos encontramos con un problema desconocido, que podría tener su origen en las versiones de algunas dependencias.

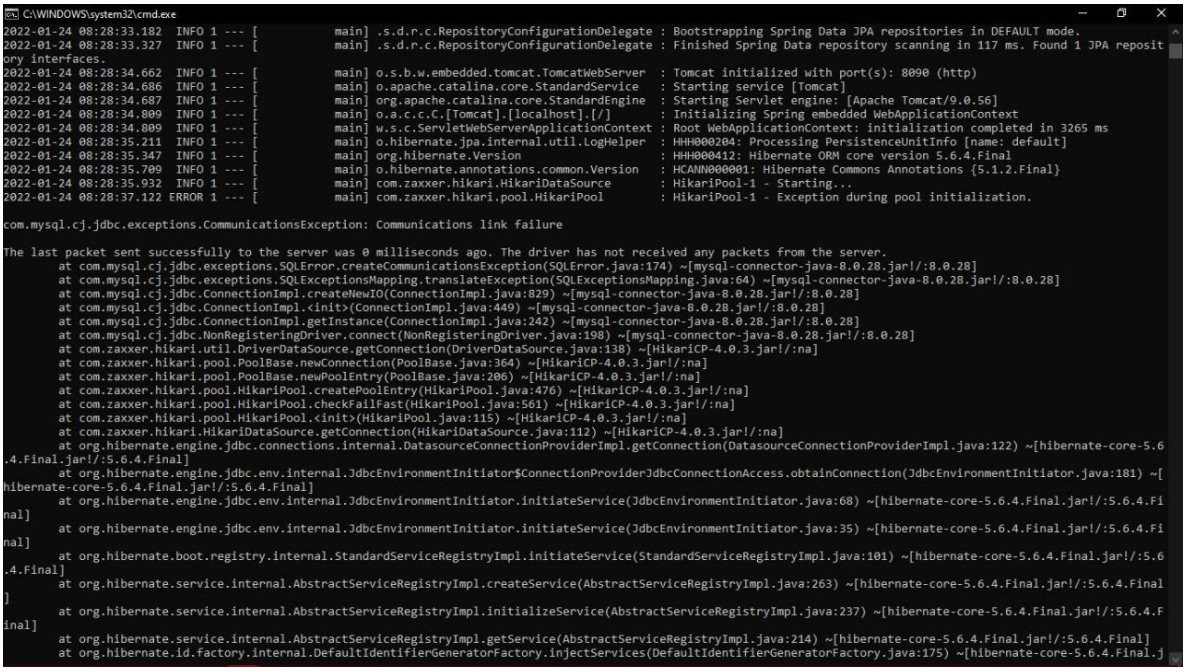
Ya que se hicieron pruebas para crear los contenedores de la API, arrojando que al poner la ruta **/localhost:13036/…** el proyecto se puede compilar y crear el archivo JAR sin ningún problema, incluso se pudo crear su contenedor.





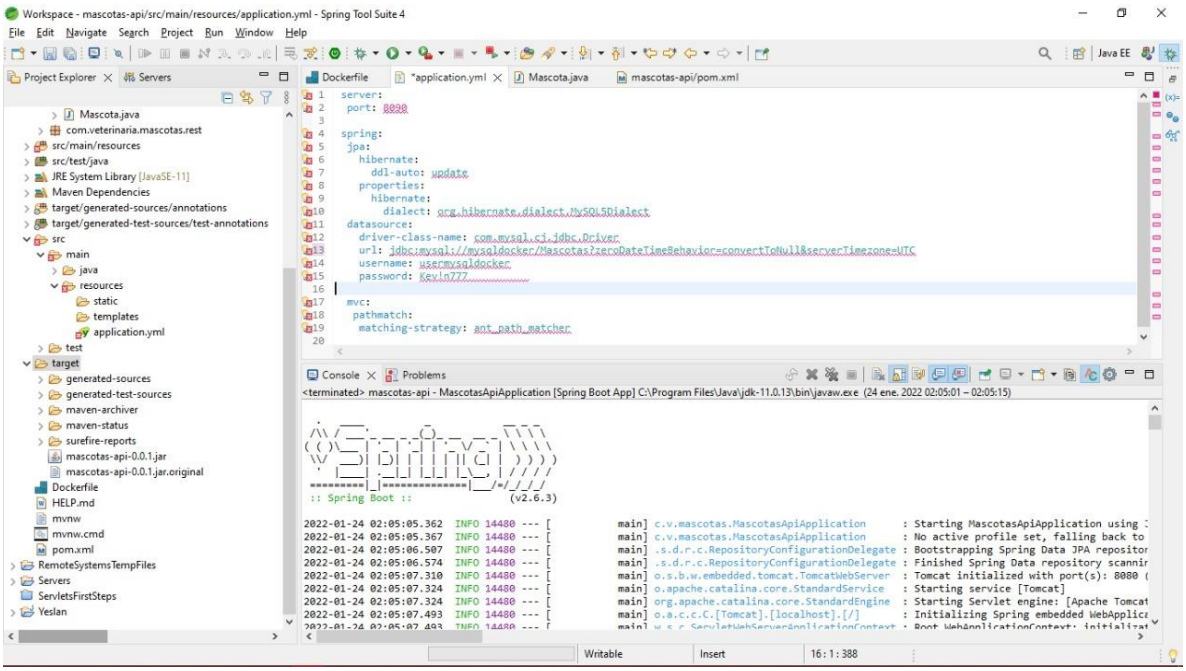
Sin embargo, aunque se puede crear su contenedor, no le es posible conectase a la base de datos, ya que no esta apuntando de forma correcta, pues lo que necesita

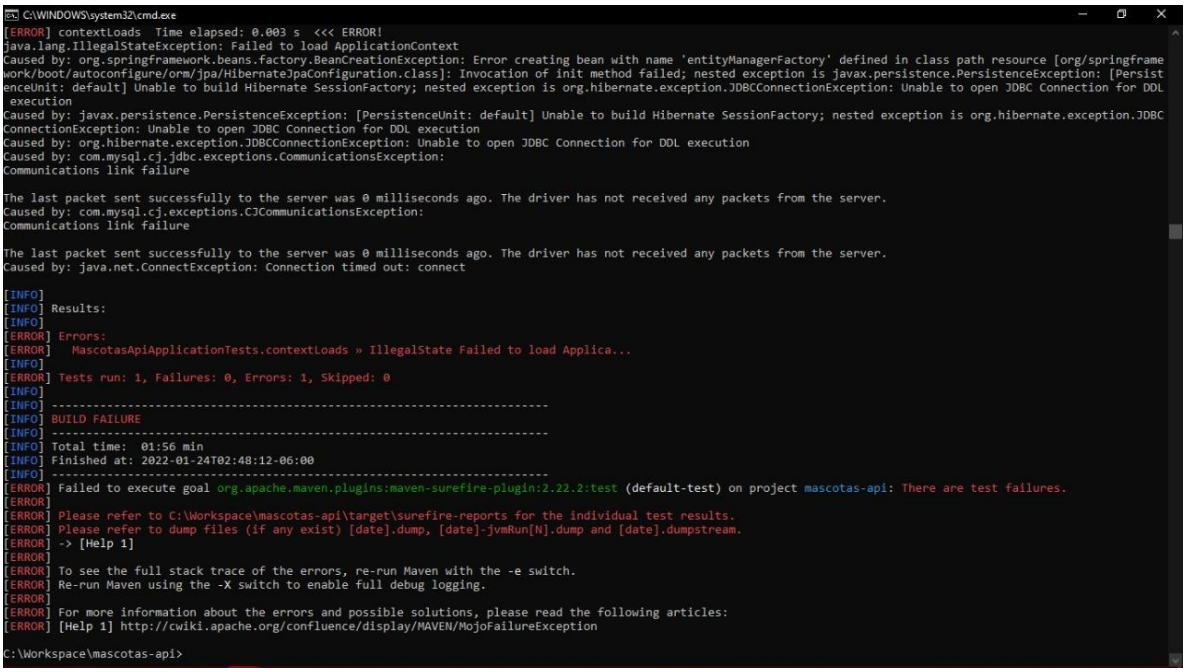
en lugar de **/localhost,** es el nombre del contenedore de MySQL, que para este caso es **mysqldocker.**



Y el problema final es que ahora que sabemos que se debe de cambiar la ruta, por el nombre del contenedor, así como el usuario y contraseña de la conexión.

Nos encontramos que, al hacer estos cambios, Maven ya no nos permite crear el archivo JAR, pues tiene conflicto con dependencias que desconozco hasta el momento.





Finalmente, los puntos que se han desarrollado para esta práctica han sido:

- La creación de una API RESTful.
- Conexión a una base de datos en MySQL.
- La creación y despliegue de contenedor de MySQL, para la base de datos.
- La creación y despliegue de contenedor de la API.

Se busca la solución al problema de la negación de creación del archivo JAR que no permite la conexión entre los contenedores, pues de no crearse el archivo con las características necesarias en la URL de conexión, no se podrá acceder desde dentro de un contenedor hacia otro.