Programming in Service and Processes

Rodrigo Arango Patiño
09/02/2025
DAM II

Multi-Container Application Deployment with Docker

Setting up the environment (Part 1)

The first step we will do is install Docker on our system and then we will verify the installation.

```
PS C:\Users\DAM2_Diurno> docker --version
Docker version 27.4.0, build bde2b89
PS C:\Users\DAM2_Diurno>
```

By using the command "Docker –version" , what we get is to see the version of Docker that is installed.

Creating the application (Part 2)

In this second part, we will first create our Docker network, so our network will be able to communicate with the 3 containers we will work with (MySQL database, Backend in Node.js, and Nginx web server).

To create the network, we will use the command: "docker network create my-network", in my case, "rodrigo-network".

```
PS C:\Users\DAM2_Diurno> docker network ls
NETWORK ID     NAME              DRIVER    SCOPE
62c44bcafeec   bridge            bridge    local
5356af2f891c   host              host      local
36bb93dc10ea   none              null      local
83ef09f57b92   rodrigo-network   bridge    local
PS C:\Users\DAM2_Diurno>
```

With the command docker network ls we can see how the network was successfully created.

Now in the second step we will launch the MySQL database, making the container run using the name "testdb".

To achieve this, we will need to use the following command: “docker run -d –name db –network rodrigo-network -e MYSQL_ROOT_PASSWORD=rootpassword -e MYSQL_DATABASE=testdb -p 3306:3306 mysql:5.7 ”

```
PS E:\> docker run -d --name db --network Rodrigo-network -e MYSQL_ROOT_PASSWORD=rootpassword -e MYSQL_DATABASE=testdb -p 3306:3306 mysql:5.7
Unable to find image 'mysql:5.7' locally
5.7: Pulling from library/mysql
20e4dcae4c69: Pull complete
1c56c3d4ce74: Pull complete
e9f03a1c24ce: Pull complete
68c3898c2015: Pull complete
6b95a940e7b6: Pull complete
90986bb8de6e: Pull complete
ae71319cb779: Pull complete
ffc89e9dfd88: Pull complete
43d05e938198: Pull complete
064b2d298fba: Pull complete
df9a4d85569b: Pull complete
Digest: sha256:4bc6bc963e6d8443453676cae56536f4b8156d78bae03c0145cbe47c2aad73bb
Status: Downloaded newer image for mysql:5.7
412e8eb9dd3c8c6f58ae040b0fdf27a3e351e6e883beaa701bbed4e37f962135
```

In the third step, we will create the JavaScript file called “server.js” in a folder named “backend". After creating this file, we will create a Dockerfile (a file with no extension and named “Dockerfile”). Once we have these, we will proceed to create the images.

```
\---Backend
    |   server.js
    |   package.json
    |   package-lock.json
    |   Dockerfile
    |
```

Now, what we will do is create the image with the following command:

“Docker build -t rodrigo-backend ./backend”

```
PS G:\Dam2\Asignaturas\Programación en servicios y procesos\Practica1-Tema3> docker image
s
REPOSITORY        TAG       IMAGE ID       CREATED          SIZE
rodrigo-backend   latest    67885639f958   39 seconds ago   1.58GB
alpine            latest    56fa17d2a7e7   3 weeks ago      12.1MB
ubuntu            latest    80dd3c3b9c6c   2 months ago     117MB
mysql             5.7       4bc6bc963e6d   13 months ago    689MB
```

With the command “Docker images” we can see all the images that we have created.

Now, we will launch the backend to make it start running, with this command:

```
PS G:\Dam2\Asignaturas\Programación en servicios y procesos\Practica1-Tema3> docker run -d --name backend --network rodrigo-network -p 3000:3000 rodrigo-backend
ae2aa5040d99ac5ba0c4cb130731a01c2938ee4e48833f55cbd2fe0bbc38b253
```

"Docker run -d –name backend –network Rodrigo-network -p 3000:3000 rodrigo-backend"

Now, we will configure the web server with Nginx. To do this, we will create a configuration file named "default.conf" in a new folder called "`nginx`".

Then, we will create the Dockerfile inside the "nginx"folder, and finally, we will build the image and launch it to run.

We will use the following commands: "docker build -t rodrigo-nginx ./nginx" and "docker run -d –name web –network -p 8080:80 rodrigo-nginx"

```
PS G:\Dam2\Asignaturas\Programación en servicios y procesos\Practica1-Tema3> docker images
REPOSITORY        TAG       IMAGE ID       CREATED          SIZE
rodrigo-nginx     latest    25341e8507d2   14 seconds ago   278MB
rodrigo-backend   latest    67885639f958   19 minutes ago   1.58GB
alpine            latest    56fa17d2a7e7   3 weeks ago      12.1MB
ubuntu            latest    80dd3c3b9c6c   2 months ago     117MB
mysql             5.7       4bc6bc963e6d   13 months ago    689MB
```

```
PS G:\Dam2\Asignaturas\Programación en servicios y procesos\Practica1-Tema3> docker run -d --name web --network rodrigo-network -p 8080:80 rodrigo-nginx
fada41842e33ddbe25581d82f2371fcfdec16089630c290d99f47b28763ad2f5
```

Finally, we will check that all the containers are running successfully, inspect the network, and verify the connections.

```
PS G:\Dam2\Asignaturas\Programación en servicios y procesos\Practica1-Tema3> docker ps -a
CONTAINER ID   IMAGE             COMMAND                  CREATED          STATUS                          PORTS     NAMES
fada41842e33   rodrigo-nginx     "/docker-entrypoint.…"   3 minutes ago    Exited (1) 3 minutes ago                  web
ae2aa5040d99   rodrigo-backend   "docker-entrypoint.s…"   9 minutes ago    Exited (1) 9 minutes ago                  backend
2594f046771e   mysql:5.7         "docker-entrypoint.s…"   9 minutes ago    Created                                   db2
509182f15f1b   mysql:5.7         "docker-entrypoint.s…"   27 hours ago     Created                                   db
b8803325f85d   alpine            "sh"                     28 hours ago     Exited (127) 28 hours ago                 awesome_mclean
bce51bd0bc29   ubuntu            "sh"                     4 days ago       Exited (255) 28 hours ago                 hardcore_banzai
bf60320e2445   ubuntu            "sh"                     4 days ago       Exited (255) 28 hours ago                 infallible_elion
b74c8ee6dff8   ubuntu            "/bin/bash"              4 days ago       Exited (0) 4 days ago                     thirsty_bhaskara
cebaeb058b40   alpine            "sh"                     4 days ago       Exited (0) 4 days ago                     amazing_visvesvaraya
a2a1fa931808   alpine            "/bin/sh"                4 days ago       Exited (127) 4 days ago                   objective_taussig
54856e847ae7   alpine            "/bin/sh"                4 days ago       Exited (0) 4 days ago                     eloquent_payne
48d34f4e1cdb   alpine            "/bin/sh"                4 days ago       Exited (0) 4 days ago                     naughty_kapitsa
PS G:\Dam2\Asignaturas\Programación en servicios y procesos\Practica1-Tema3>
```

We will use the following commands: "docker ps" and "docker network inspect rodrigo-network".

```
PS G:\Dam2\Asignaturas\Programación en servicios y procesos\Practica1-Tema3> docker network inspect rodrigo-network
[
    {
        "Name": "rodrigo-network",
        "Id": "83ef09f57b92b0ba9d72f61543dc3acebbf5750fdc5fff39eeae9f597923ab1a",
        "Created": "2025-02-03T07:10:10.653889825Z",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": {},
            "Config": [
                {
                    "Subnet": "172.18.0.0/16",
                    "Gateway": "172.18.0.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {},
        "Options": {},
        "Labels": {}
    }
]
```

End of the report.

Thank you for reading.

Rodrigo Arango Patiño-