CURSO 23 - 24

Despliegue
BOOK 'N BOOK

REALIZADO POR:

MARÍA ESCRIBANO VERDE

2º DAW - IES ALIXAR

MANUAL DE DESPLIEGUE

En este apartado se describen todos los pasos necesarios para desplegar la aplicación *Book N' Book* utilizando Docker. La aplicación está compuesta por tres componentes principales: una base de datos MySQL, un backend desarrollado en Spring Boot y un frontend en Angular. El despliegue se realiza utilizando contenedores Docker y Docker Compose.

PRE-REQUISITOS

Antes de comenzar, asegúrate de tener instalados los siguientes componentes en tu máquina:

- Docker
- Docker Compose

ESTRUCTURA DE PROYECTO

La estructura del proyecto debe ser similar a la siguiente:

```
booknbook/
├── BooknBookBACK/
│   ├── Dockerfile
│   ├── target/
│   │   └── booknbook-0.0.1-SNAPSHOT.jar
│   └── ...
├── BookNBookFRONT/
│   ├── BooknBookFront/
│   │   ├── Dockerfile
│   │   ├── nginx.conf
│   │   └── ...
├── docker-compose.yml
└── init.sql
```

PASOS PARA EL DESPLIEGUE

Para desplegar la aplicación, desde una terminal en el directorio raíz del proyecto (booknbook) y ejecutar los siguientes comandos:

```
docker-compose down
docker-compose build --no-cache
docker-compose up
```

```
PS C:\Users\maria\Desktop\BNB> docker-compose down
[+] Running 4/4
 ✔Container frontcontainer  Removed
 ✔Container backcontainer   Removed
 ✔Container mysqlcontainer  Removed
 ✔Network bnb_default       Removed
```

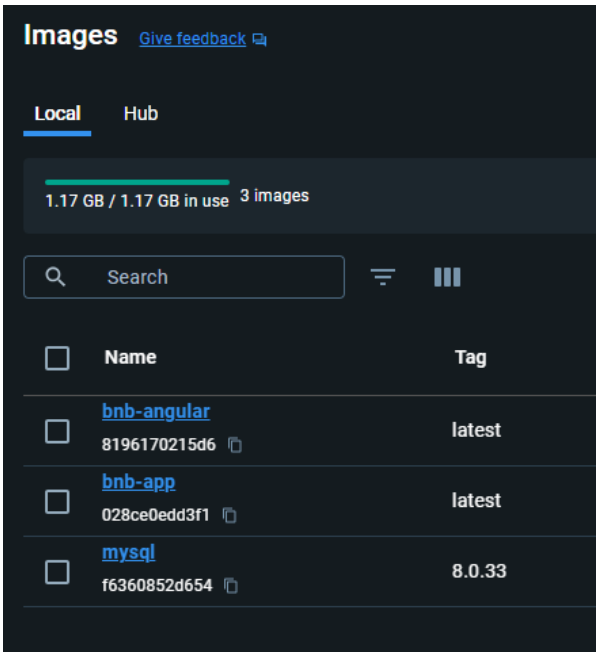
```
PS C:\Users\maria\Desktop\BNB> docker-compose build --no-cache
[+] Building 4.8s (8/8)                                                                                  docker:default
 => [app internal] load build definition from Dockerfile                                                          0.0s
[+] Building 5.1s (8/8)                                                                                  docker:default
 => [app internal] load build definition from Dockerfile                                                          0.0s
[+] Building 280.8s (24/24) FINISHED                                                                     docker:default
 => [app internal] load build definition from Dockerfile                                                          0.0s
 => => transferring dockerfile: 349B                                                                              0.0s
 => [app internal] load metadata for docker.io/library/openjdk:17                                                 1.2s
 => [app auth] library/openjdk:pull token for registry-1.docker.io                                                0.0s
 => [app internal] load .dockerignore                                                                             0.0s
 => => transferring context: 2B                                                                                   0.0s
 => [app internal] load build context                                                                             2.6s
 => => transferring context: 55.52MB                                                                              2.5s
 => CACHED [app 1/2] FROM docker.io/library/openjdk:17@sha256:528707081fdb9562eb819128a9f85ae7fe000e2fbaeaf9f8766  0.0s
 => [app 2/2] COPY target/booknbook-0.0.1-SNAPSHOT.jar app.jar                                                    0.5s
 => [app] exporting to image                                                                                      0.2se
 => => exporting layers                                                                                           0.2s
 => => writing image sha256:028ce0edd3f1d5f13845f0b38269aea7f58c15d563aecc92725fce6eb76f9441                      0.0se
 => => naming to docker.io/library/bnb-app                                                                        0.0s
 => [angular internal] load build definition from dockerfile                                                      0.0s
 => => transferring dockerfile: 312B                                                                              0.0s
 => [angular internal] load metadata for docker.io/library/nginx:alpine                                           1.0s
 => [angular internal] load metadata for docker.io/library/node:18-alpine                                         1.0s
 => [angular auth] library/nginx:pull token for registry-1.docker.io                                              0.0s
 => [angular auth] library/node:pull token for registry-1.docker.io                                               0.0s
 => [angular internal] load .dockerignore                                                                         0.0s
 => => transferring context: 52B                                                                                  0.0s
 => [angular build 1/5] FROM docker.io/library/node:18-alpine@sha256:cf350f8bb497d82471f1f735df5d6d3321138be3b9f7  0.0s
 => CACHED [angular stage-1 1/3] FROM docker.io/library/nginx:alpine@sha256:69f8c2c72671490607f52122be2af27d4fc09  0.0s
 => [angular internal] load build context                                                                       134.1s
 => => transferring context: 1.91GB                                                                             134.1s
 => CACHED [angular build 2/5] WORKDIR /app                                                                       0.0s
 => [angular build 3/5] COPY . .                                                                                 36.6s
 => [angular build 4/5] RUN npm install                                                                          43.4s
 => [angular build 5/5] RUN npm run build                                                                        58.5s
 => [angular stage-1 2/3] COPY --from=build /app/dist/BooknBookFront/ /usr/share/nginx/html                       0.4s
 => [angular stage-1 3/3] COPY nginx.conf /etc/nginx/conf.d/default.conf                                          0.2s
 => [angular] exporting to image                                                                                  0.9s
 => => exporting layers                                                                                           0.8s
 => => writing image sha256:8196170215d6a3846f4a29bc50e5997c630722d0d3b8729506045a657115688f                      0.0s
 => => naming to docker.io/library/bnb-angular                                                                    0.0s
```

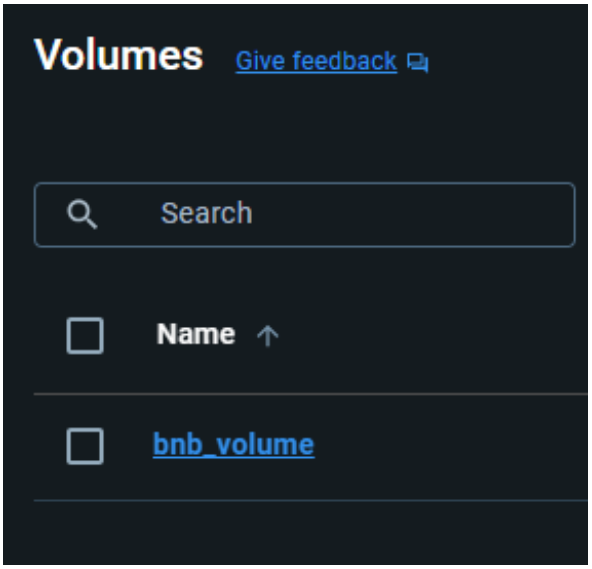
```
PS C:\Users\maria\Desktop\BNB> docker-compose up
[+] Running 4/4
 ✔Network bnb_default       Created
 ✔Container mysqlcontainer  Created
 ✔Container backcontainer   Created
 ✔Container frontcontainer  Created
```

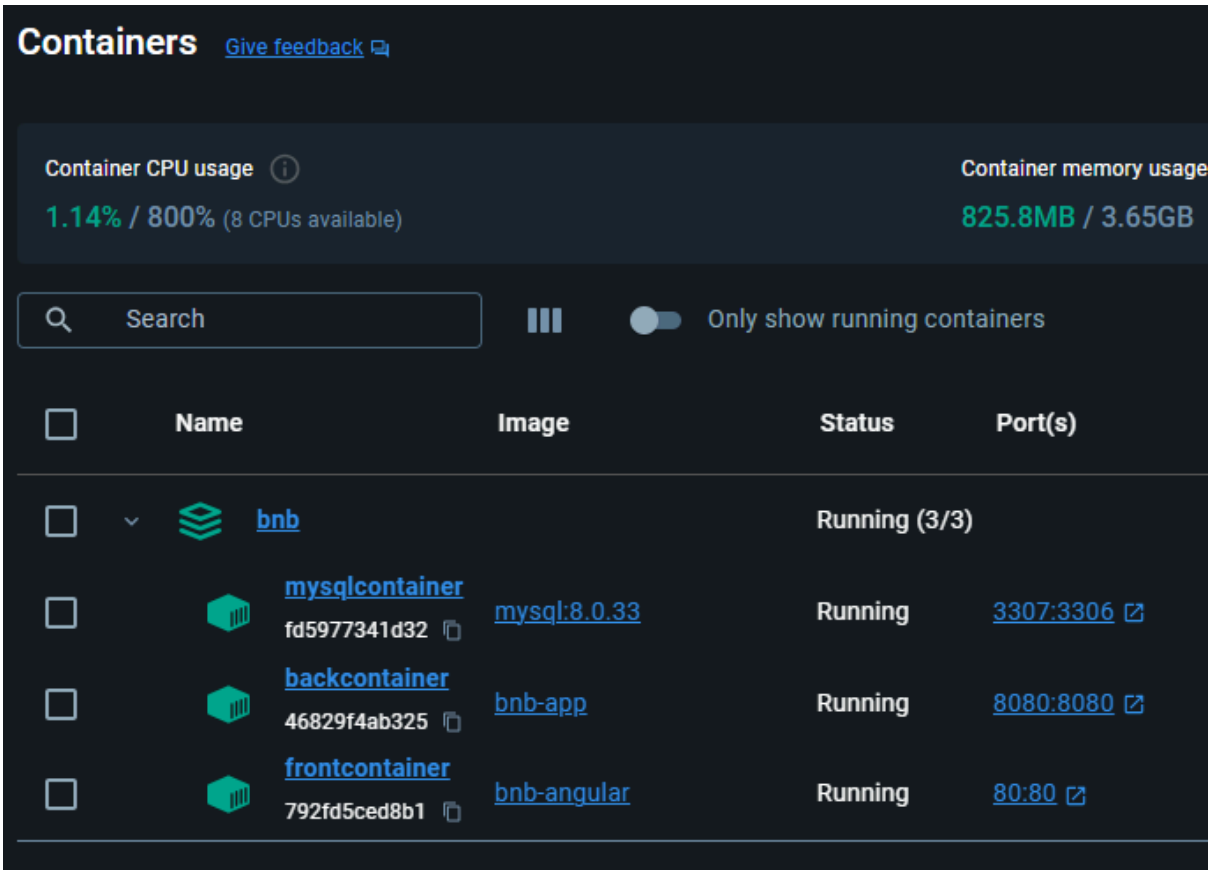
Estos comandos harán lo siguiente:

- docker-compose down: Detiene y elimina los contenedores existentes.
- docker-compose build --no-cache: Construye los contenedores sin utilizar la caché, asegurándose de que se utilicen las versiones más recientes de las imágenes.

- docker-compose up: Inicia los contenedores en modo adjunto, permitiendo ver los logs en tiempo real.

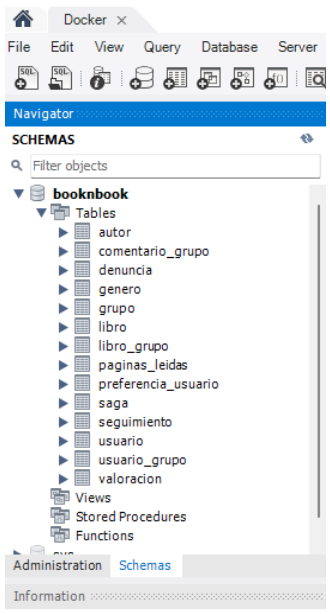


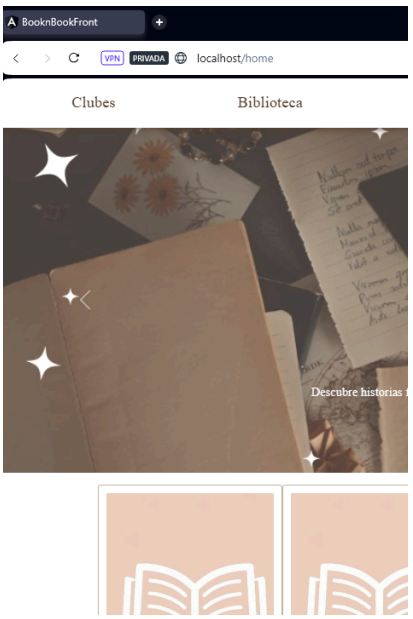




Tras ello, se puede verificar el despliegue de la siguiente forma:

- Backend: Acceder a http://localhost:8080/swagger para verificar que el backend de Spring Boot está corriendo correctamente.
- Frontend: Acceder a http://localhost para verificar que el frontend de Angular está corriendo correctamente.
- Base de Datos: Acceder a MySQL a través del puerto 3307 en tu máquina local con el usuario root y la contraseña 123456.





Por último a tener en cuenta, en este despliegue, se ha configurado un volumen para la base de datos MySQL, lo que permite la persistencia de datos incluso si los contenedores se detienen o reinician. Este volumen se define en el archivo docker-compose.yml y asegura que los datos de la aplicación no se pierdan, proporcionando una solución de almacenamiento confiable y persistente.

Además, el frontend de Angular se sirve utilizando Nginx, un servidor web ligero y de alto rendimiento. Nginx se utiliza para servir los archivos estáticos generados por Angular, asegurando una entrega rápida y eficiente de los recursos del frontend. La configuración de Nginx se especifica en el archivo nginx.conf, optimizando así la experiencia del usuario final al interactuar con la aplicación *Book N' Book*.