

Despliegue aplicación gestionIES con contenedores Docker

Pasos previos

- Instalación de docker y, recomendado, portainer para gestionar desde entorno web nuestros contenedores. En xubuntu sería (puede variar según la versión/distribución):
 - Actualizar paquetes y dependencias:
`apt update`
`apt install -y apt-transport-https ca-certificates curl software-properties-common lsb-release`
 - Añadir GPG de Docker:
`curl -fsSL https://download.docker.com/linux/ubuntu/gpg | gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg`
 - Añadir repositorio de Docker:
`echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | tee /etc/apt/sources.list.d/docker.list > /dev/null`
 - Actualizar repos e instalar Docker:
`apt update`
`apt install -y docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin`

Para instalar Portainer:

- `docker volume create portainer_data`
- `docker run -d -p 9000:9443 --name portainer --restart=always -v /var/run/docker.sock:/var/run/docker.sock -v portainer_data:/data portainer/portainer-ce:latest`
- Y ya podemos abrir portainer en el navegador con `https://ip_equipo:9000`.
- Comprobar si la versión de docker que hemos instalado tiene el comando “docker-compose” o “docker compose” (probando a ejecutar ambos).
- Como la aplicación va a usar https, generaremos certificados autofirmados en el equipo que va a alojar la aplicación:

- `mkdir -p /etc/nginx/ssl`
 - `cd /etc/nginx/ssl`
 - `openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout nginx.key -out nginx.crt`
- Clonar repositorio en la máquina en la que vamos a desplegar los contenedores: git clone <https://github.com/Chisco77/gestionIES.git>. Os recomiendo hacerlo en una carpeta que llamaréis /docker. De esta forma, al clonar, tendréis /docker/gestionIES.
 - Actualizar estos archivos de configuración antes del despliegue:
 - Si nuestra versión de docker tiene el comando docker compose en lugar de docker-compose, hay que cambiar el archivo docker-compose.yml. La línea “docker-compose build” hay que cambiarla a “docker compose build” y la línea “docker-compose up -d” hay que cambiarla a “docker compose up”.
 - .env: Poner DB_PASSWORD al valor que queramos. Serán utilizados cuando se cree el contenedor de la base de datos postgresql, gestionIES.
 - ./backend/.env: Comentar NODE_ENV=development y descomentar NODE_ENV=production. Esta variable de entorno se usa para distinguir entre modo de desarrollo y producción y cargar otras variables en función del modo. Si vamos a desplegar la aplicación en contenedores, debe estar en modo production. Descomentar DB_HOST=db y comentar DB_HOST=172.16.218.52. En la línea LDAP_URL=ldap://ip_ldap:389 , poned la ip de vuestro LDAP. Añadir en ALLOWED_ORIGINS https://ip_equipo_despliegue. Esto es muy importante, tenéis que poner la ip del equipo en el que estáis desplegando los contenedores.

Despliegue

- Archivos del despliegue:
 - El propio `deploy_gestionIES.sh`. Es el archivo principal que ejecutaremos para desplegar los contenedores en la máquina que hemos elegido.
 - `docker-compose.yml`: Para la creación de los contenedores
 - `Dockerfile.frontend`: Para el despliegue del frontend.
 - `Dockerfile` del backend.
 - `gestionIES.sql`: Esquema de la base de datos `gestionIES`.

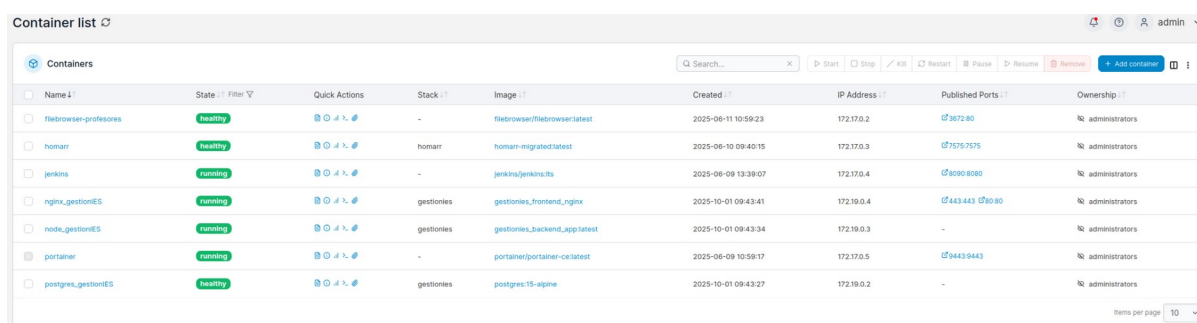
Antes de ejecutar `deploy_gestionIES.sh`, ejecutar `dos2unix deploy_gestionIES.sh` y `dos2unix reconstruir.sh` (si no está instalado, instalar con `apt-get install dos2unix`). Esto elimina caracteres “extraños” introducidos por windows en el caso de que desarrolléis en casa con windows.

Ya podemos realizar el despliegue:

```
# chmod +x deploy_gestionIES.sh
# ./deploy_gestionIES.sh
```

Si todo es correcto, finalizará con un mensaje de "Despliegue finalizado con éxito."

El script tarda unos minutos en finalizar, tiene que hacer muchas acciones (crear y desplegar contenedores, desplegar el frontend y backend, volúmenes, volcar la base de datos, etc.). Al finalizar, si todo ha ido correctamente, podremos ver en nuestro portainer los tres contenedores recién creados (`nginx_gestionIES`, `node_gestionIES` y `postgres_gestionIES`):



Name	State	Quick Actions	Stack	Image	Created	IP Address	Published Ports	Ownership
filebrowser-profesores	healthy		-	filebrowser/filebrowser:latest	2025-06-11 10:59:23	172.17.0.2	8080	administrators
homarr	healthy		homarr	homarr-migrated:latest	2025-06-10 09:40:15	172.17.0.3	7575	administrators
jenkins	running		-	jenkins/jenkins:its	2025-06-09 13:39:07	172.17.0.4	8080	administrators
nginx_gestionIES	running		gestionies	gestionies_frontend/nginx	2025-10-01 09:43:41	172.19.0.4	8080	administrators
node_gestionIES	running		gestionies	gestionies_backend/app:latest	2025-10-01 09:43:34	172.19.0.3	-	administrators
portainer	running		-	portainer/portainer-ce:latest	2025-06-09 10:59:17	172.17.0.5	9443	administrators
postgres_gestionIES	healthy		gestionies	postgres:15-alpine	2025-10-01 09:43:27	172.19.0.2	-	administrators

En `nginx_gestionIES` está el frontend y el `nginx`, en `node_gestionIES` está el backend y en `postgres_gestionIES` está la base de datos `gestionIES`.

Para utilizar la aplicación, probar en un navegador a poner: https://ip_equipo_despliegue/gestionIES. Debe informaros de que debéis aceptar los certificados para navegar y ya os dejará entrar.

El logo del IES está en la carpeta public, igual que los planos que se utilizan para el control de llaves.

Si vais a tocar el código de la aplicación, cualquier cosa, ya sea backend, frontend o cualquier archivo (como poner un nuevo logo para vuestro insti), hay que actualizar el código en los contenedores también. Para ello tengo el script [reconstruir.sh](#). Es decir, si cambiáis algo, después tenéis que ejecutar [reconstruir.sh](#) para que se despliegue en los contenedores. En este script, hay que verificar también qué comando, “docker-compose” o “docker compose” está activo en nuestra instalación de docker.