Running a DOCKER COMPOSE

Step 1: (BACK)

Create an image able to run a backend API (python, java, nodejs, php…).

1. The image should expose an api (/hello) with a json output:
    a. Ex: {message: 'Hello World'}
2. Database credentials should be added with a secret
3. The back app should not be exposed to the host
4. The app logs should be mount to a folder (ex: /logs/myapp) on your host filesystem

→ *Image should be created within a docker compose file.*

Step 2: (DB)

Create an image able to run a sql database (mysql, postgres…).

- The image should include an init script, to create a table (hello) with a text column containing the greeting message (Hello world)
- A volume (db_volume) should be added
- The password must be added from a secret

Step 3: (Proxy)

Create an image with a web application (angular, react, …) or a simple (html and JavaScript file) that:

1. Consumes the API and display the received message.
2. Use a web server (nginx or httpd) to create your reverse proxy.

→ *Image should be created within a docker compose file.*

Step 4: (Networking)

1. Proxy service should have access only to the back service.
2. Back service should communicate with both proxy and database service.

<u>TIPS:</u>

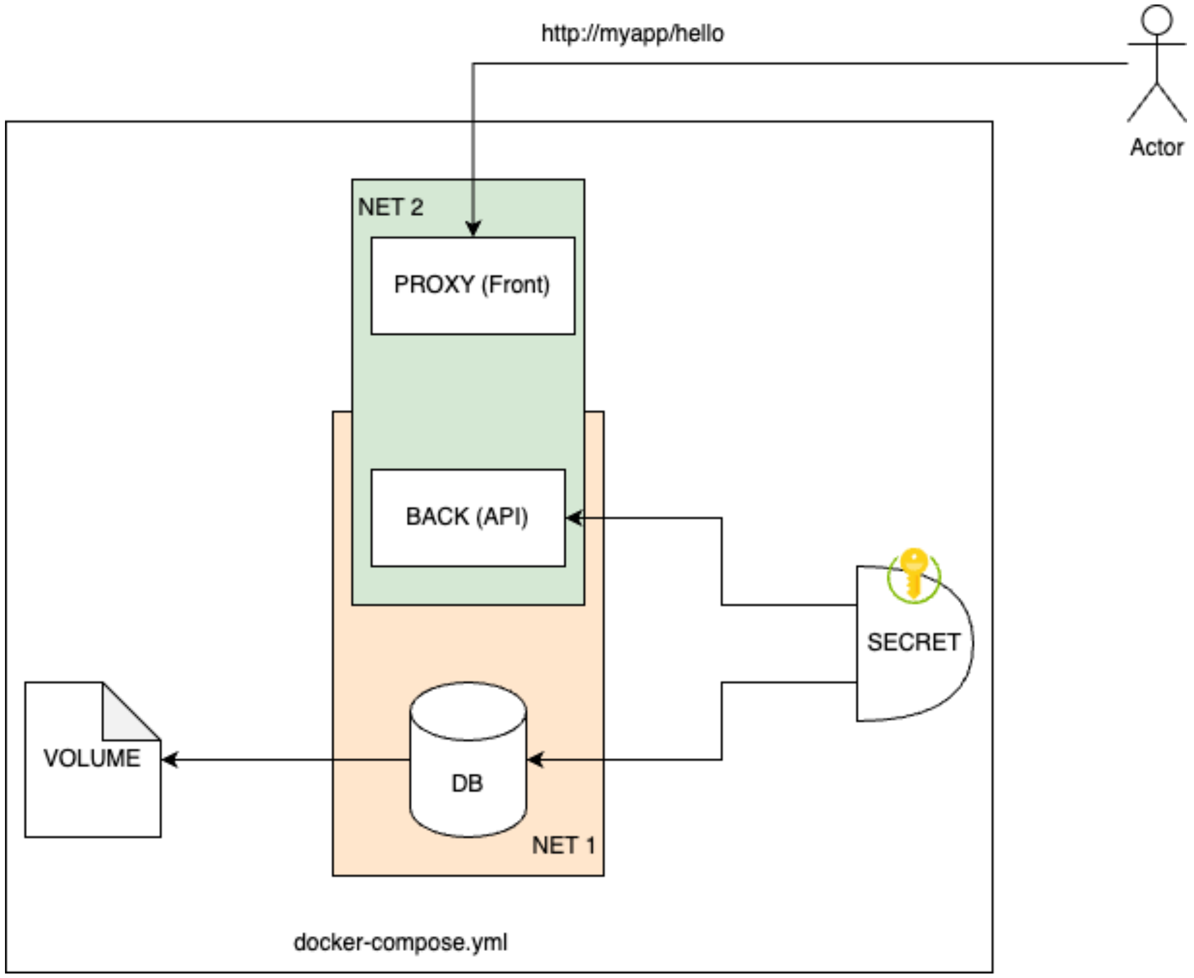
docker compose up: build and initialize the environment.

docker compose build: refresh images after updating the code or the Dockerfile.

IF container UP → docker compose start: to start the service with newer image.

IF container DOWN → docker compose restart: to start the service with newer image.

Docker compose down: to kill all the resources created with the compose file.


http://myapp/hello
Actor
NET 2
PROXY (Front)
BACK (API)
SECRET
VOLUME
DB
NET 1
docker-compose.yml