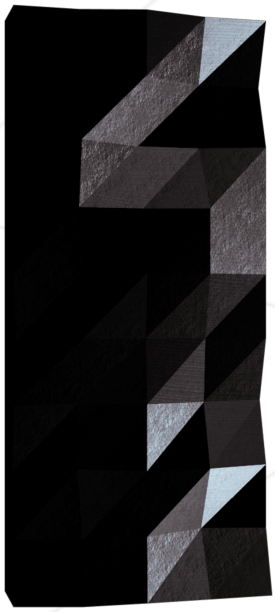


Inception

Summary: This document is a System Administration related exercise.

Contents

Chapter I

Preamble

Chapter II

Introduction

This project aims to broaden your knowledge of system administration by using Docker. You will virtualize several Docker images, creating them in your new personal virtual machine.

Chapter III

General guidelines

- All the files required for the configuration of your project must be placed in a `srcs` folder.
- A `Makefile` is also required and must be located at the root of your directory. It must set up your entire application (i.e., build the Docker images using `docker-compose`).
- This subject requires putting into practice concepts that, depending on your background, you may not have learned yet. Therefore, we advise you not to hesitate to read a lot of documentation related to Docker usage, as well as anything else you will find helpful to complete this assignment.

Chapter IV

Mandatory part

This project consists in having you set up a small infrastructure composed of different services under specific rules. To do so, you have to use `docker-compose`.

Each Docker image must have the same name as its corresponding service.
Each service has to run in a dedicated container.
For performance matters, the containers must be built either from Alpine Linux with the penultimate stable version, or from Debian Buster. The choice is yours.
You must also write your own `Dockerfiles`, one per service. The `Dockerfiles` must be called in your `docker-compose.yaml` by your `Makefile`.
It means you have to build the Docker images that you will be using by yourself. It is then forbidden to pull ready-made Docker images, as well as using services such as DockerHub (Alpine/Debian being excluded from this rule).

You then have to set up:

- A Docker container that contains NGINX with TLSv1.2 or TLSv1.3 only.
- A Docker container that contains WordPress + php-fpm (it must be installed and configured).
- A Docker container that contains MariaDB.
- A volume that contains your WordPress database.
- A second volume that contains your WordPress website files.
- A `docker-network` that establishes the connection between your containers.

A Docker container is not a virtual machine. Thus, it is not recommended to use any hacky patch based on 'tail -f' and so forth when trying to run it. Read about how daemons work and whether it's a good idea to use them or not.

Of course, using network: host or --link is forbidden.

- In your WordPress database, there must be two users, one of them being the administrator. The administrator's username can't contain admin/Admin or administrator/Administrator.

Your containers have to restart in case of a crash.

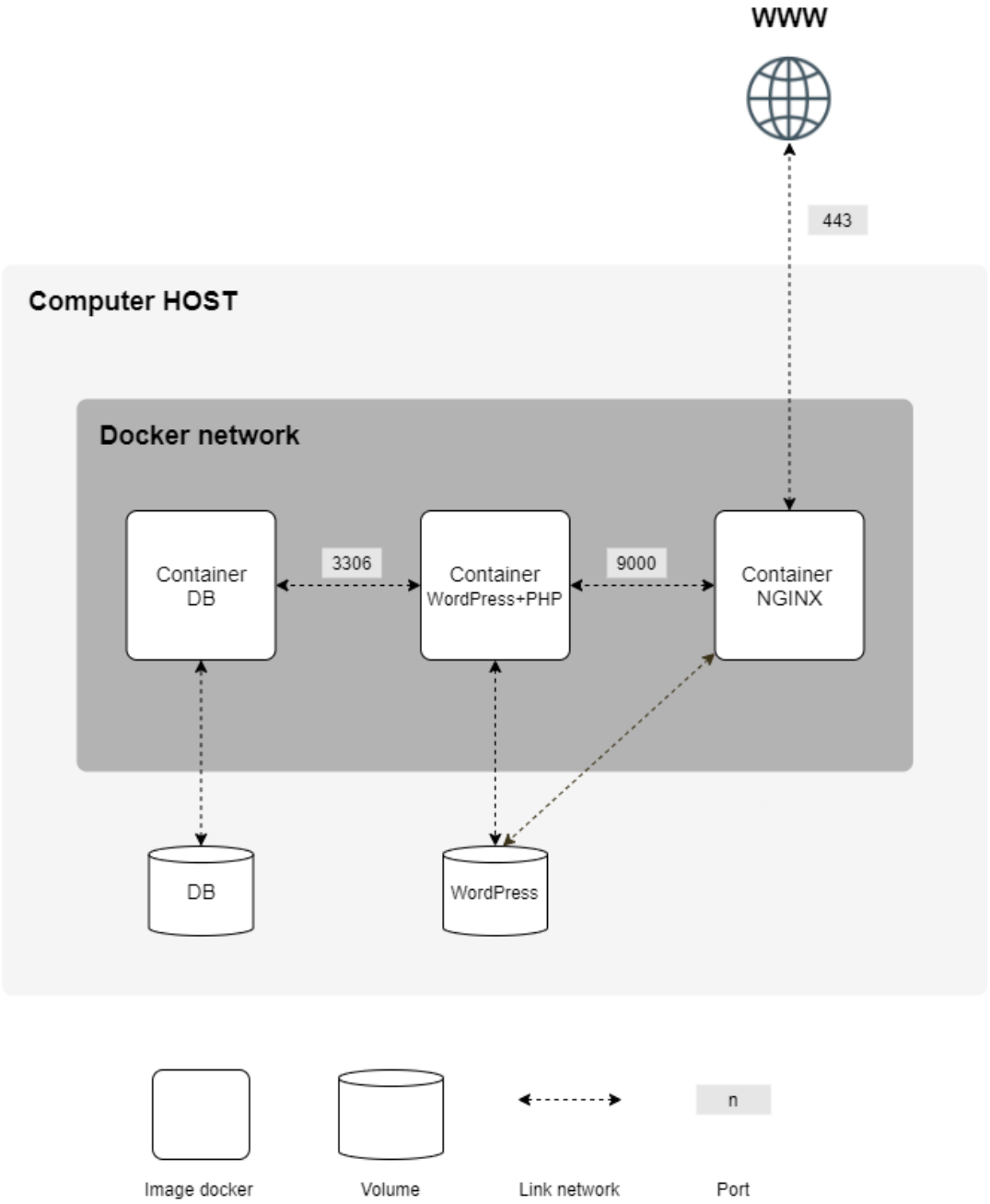
Your volumes will be available in the /home/login/data folder of the host machine using docker. Of course, you have to replace the login with yours.

To make this project simpler, you have to configure your domain name so it points to your local IP address.
This domain name must be `login.42.fr`. Again, you have to use your own login. For example, if your login is wil, `wil.42.fr` will redirect to the IP address pointing to wil's website.

The latest tag is prohibited. No password must be present in your Dockerfiles. It is mandatory to use environment variables. Also, it is strongly recommended to use a .env file to store environment variables. The .env file should be located at the root of the srcs directory.
Your NGINX container must be the only entrypoint into your infrastructure via the port 443 only and using the TLSv1.2 or TLSv1.3 protocol.

To complete the bonus part, you have to set up more services. In this case, more ports can be open as needed.

Here is an example diagram of the expected result:



Below is an example of the expected directory structure:

```
$> ls -alR
total XX
drwxrwxr-x 3 wil wil 4096 avril 42 20:42 .
drwxrwxrwt 17 wil wil 4096 avril 42 20:42 ..
-rw-rw-r-- 1 wil wil XXXX avril 42 20:42 Makefile
drwxrwxr-x 3 wil wil 4096 avril 42 20:42 srcs

./srcs:
total XX
drwxrwxr-x 3 wil wil 4096 avril 42 20:42 .
drwxrwxr-x 3 wil wil 4096 avril 42 20:42 ..
-rw-rw-r-- 1 wil wil XXXX avril 42 20:42 docker-compose.yml
-rw-rw-r-- 1 wil wil XXXX avril 42 20:42 .env
drwxrwxr-x 5 wil wil 4096 avril 42 20:42 requirements

./srcs/requirements:
total XX
drwxrwxr-x 5 wil wil 4096 avril 42 20:42 .
drwxrwxr-x 3 wil wil 4096 avril 42 20:42 ..
drwxrwxr-x 4 wil wil 4096 avril 42 20:42 bonus
drwxrwxr-x 4 wil wil 4096 avril 42 20:42 mariadb
drwxrwxr-x 4 wil wil 4096 avril 42 20:42 nginx
drwxrwxr-x 4 wil wil 4096 avril 42 20:42 tools
drwxrwxr-x 4 wil wil 4096 avril 42 20:42 wordpress

./srcs/requirements/mariadb:
total XX
drwxrwxr-x 4 wil wil 4096 avril 42 20:45 .
drwxrwxr-x 5 wil wil 4096 avril 42 20:42 ..
drwxrwxr-x 2 wil wil 4096 avril 42 20:42 conf
-rw-rw-r-- 1 wil wil XXXX avril 42 20:42 Dockerfile
-rw-rw-r-- 1 wil wil XXXX avril 42 20:42 .dockerignore
drwxrwxr-x 2 wil wil 4096 avril 42 20:42 tools
[...]
./srcs/requirements/nginx:
total XX
drwxrwxr-x 4 wil wil 4096 avril 42 20:42 .
drwxrwxr-x 5 wil wil 4096 avril 42 20:42 ..
drwxrwxr-x 2 wil wil 4096 avril 42 20:42 conf
-rw-rw-r-- 1 wil wil XXXX avril 42 20:42 Dockerfile
-rw-rw-r-- 1 wil wil XXXX avril 42 20:42 .dockerignore
drwxrwxr-x 2 wil wil 4096 avril 42 20:42 tools
[...]

$> cat srcs/.env
DOMAIN_NAME=wil.42.fr
# certificates
CERTS_=./XXXXXXXXXXX
# MYSQL SETUP
MYSQL_ROOT_PASSWORD=XXXXXXXXXXX
MYSQL_USER=XXXXXXXXXXX
MYSQL_PASSWORD=XXXXXXXXXXX
[...]
$>
```

Chapter V

Bonus part

For this project, bonuses are aimed to be simple.
A Dockerfile must be written for each extra service. Thus, each one of them will run inside its own container and will have, if necessary, its dedicated volume.

Bonus list:

- Set up `redis cache` for your WordPress website in order to properly manage the cache.
- Set up a `FTP server` container pointing to the volume of your WordPress website.
- Create a simple static website in the language of your choice except PHP (Yes, PHP is excluded!). For example, a showcase site or a site for presenting your resume.
- Set up `Adminer`.
- Set up a service of your choice that you think is useful.

`Bonuses will only be assessed if the mandatory part is PERFECT. Perfect means the mandatory part has been integrally done and everything works with no dysfunction. If you have not passed ALL the mandatory requirements, your bonus part will not be evaluated at all.`

Chapter VI

Submission and peer-evaluation

Turn in your assignment in your `Git` repository as usual. Only the work present in your repository will be evaluated during the defense. Don't hesitate to double check the names of your folders and files to ensure they comply with the expected ones.