



# Inception

*Summary: This document is a System Administration related exercise.*

*Version: 3.2*

# Contents

|            |                                       |           |
|------------|---------------------------------------|-----------|
| <b>I</b>   | <b>Preamble</b>                       | <b>2</b>  |
| <b>II</b>  | <b>Introduction</b>                   | <b>3</b>  |
| <b>III</b> | <b>General guidelines</b>             | <b>4</b>  |
| <b>IV</b>  | <b>Mandatory part</b>                 | <b>5</b>  |
| <b>V</b>   | <b>Bonus part</b>                     | <b>10</b> |
| <b>VI</b>  | <b>Submission and peer-evaluation</b> | <b>11</b> |

# Chapter I

## Preamble



# Chapter II

## Introduction

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

# Chapter III

## General guidelines

- This project must be completed on a `Virtual Machine`.
- 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., it has to build the Docker images using `docker-compose.yml`).
- This subject requires putting into practice concepts that, depending on your background, you may not have learned yet. Therefore, we advise you to read extensive documentation related to Docker usage, as well as any other resources you find helpful to complete this assignment.

# Chapter IV

## Mandatory part

This project involves setting up a small infrastructure composed of different services under specific rules. The whole project has to be done in a virtual machine. You must 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 reasons, the containers must be built from either the penultimate stable version of Alpine or Debian. The choice is yours.

You also have to write your own Dockerfiles, one per service. The Dockerfiles must be called in your docker-compose.yml by your Makefile.

This means you must build the Docker images for your project yourself. It is then forbidden to pull ready-made Docker images or use 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 with php-fpm (it must be installed and configured) only, without nginx.
- A Docker container that contains only MariaDB, without nginx.
- 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.

Your containers must restart automatically in case of a crash.



A Docker container is not a virtual machine. Thus, it is not recommended to use any hacky patches based on `'tail -f'` and similar methods 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` or `links:` is forbidden. The network line must be present in your `docker-compose.yml` file. Your containers must not be started with a command running an infinite loop. Thus, this also applies to any command used as `entrypoint`, or used in `entrypoint` scripts. The following are a few prohibited hacky patches: `tail -f`, `bash`, `sleep infinity`, `while true`.



Read about PID 1 and the best practices for writing Dockerfiles.

- In your `WordPress database`, there must be two users, one of them being the `administrator`. The administrator's username must not contain `'admin'`, `'Admin'`, `'administrator'`, or `'Administrator'` (e.g., `admin`, `administrator`, `Administrator`, `admin-123`, etc.).



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.

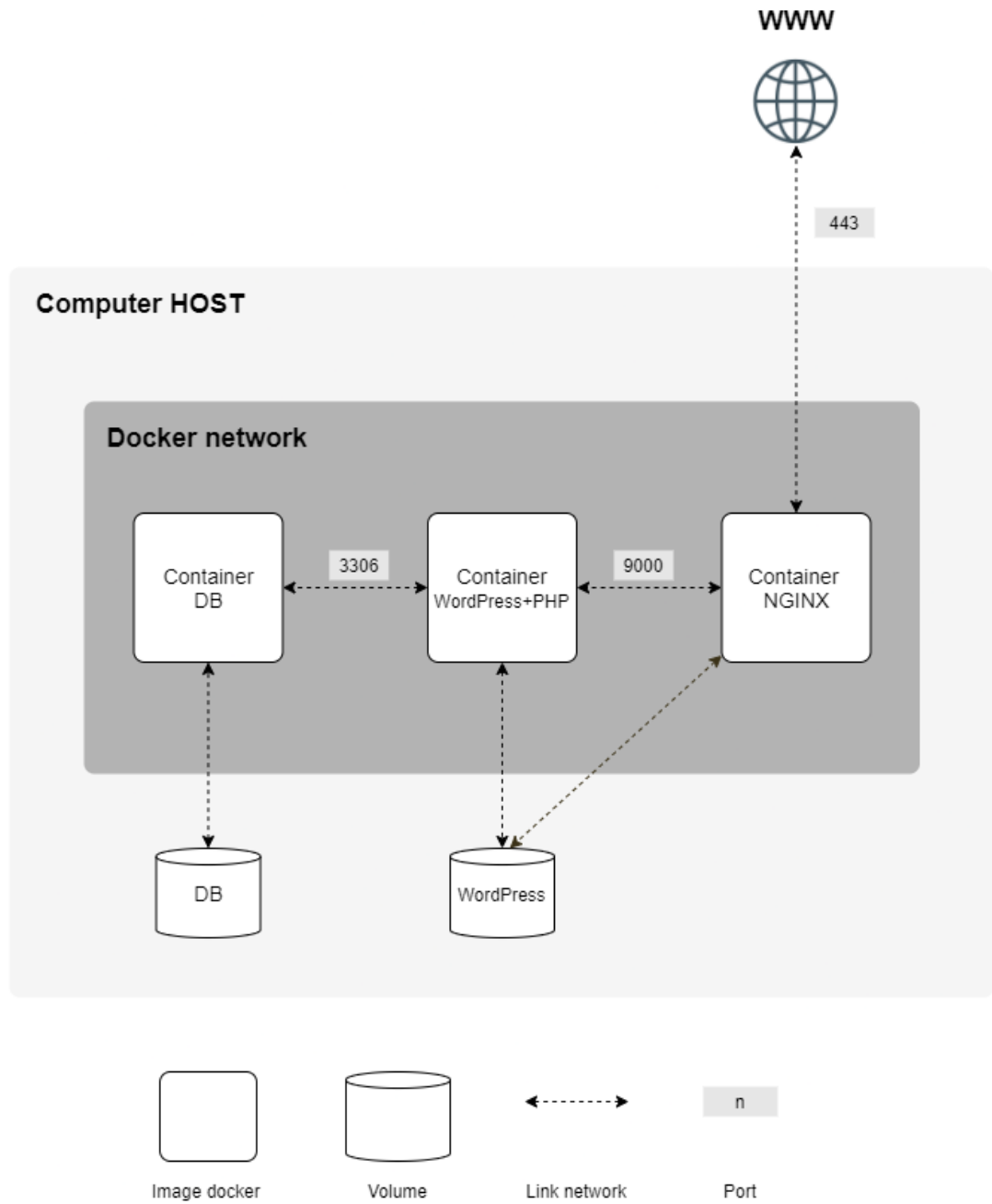
To simplify the process, you must configure your domain name to point to your local IP address.

This domain name must be `login.42.fr`. Again, you must 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.  
Passwords must not be present in your Dockerfiles.  
The use of environment variables is mandatory.  
It is also strongly recommended to use a `.env` file to store environment variables and to use the Docker secrets to store any confidential information.  
Your NGINX container must be the sole entry point into your infrastructure, accessible only via port 443, using the TLSv1.2 or TLSv1.3 protocol.

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 secrets
drwxrwxr-x 3 wil wil 4096 avril 42 20:42 srcs

./secrets:
total XX
drwxrwxr-x 2 wil wil 4096 avril 42 20:42 .
drwxrwxr-x 6 wil wil 4096 avril 42 20:42 ..
-rw-r--r-- 1 wil wil XXXX avril 42 20:42 credentials.txt
-rw-r--r-- 1 wil wil XXXX avril 42 20:42 db_password.txt
-rw-r--r-- 1 wil wil XXXX avril 42 20:42 db_root_password.txt

./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
# MYSQL SETUP
MYSQL_USER=XXXXXXXXXXXX
[...]
$>
```



For obvious security reasons, any credentials, API keys, passwords, etc., must be saved locally in various ways / files and ignored by git. Publicly stored credentials will lead you directly to a failure of the project.



You can store your variables (as a domain name) in an environment variable file like `.env`

# Chapter V

## Bonus part

For this project, the bonus part is intended to be simple.

A Dockerfile must be written for each additional service. Thus, each service 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. During the defense, you will have to justify your choice.



To complete the bonus part, you have the possibility to set up extra services. In this case, you may open more ports to suit your needs.



The bonus part will only be assessed if the mandatory part is completed perfectly. Perfect means the mandatory part has been fully completed and functions without any malfunctions. If you have not passed ALL the mandatory requirements, your bonus part will not be evaluated at all.

# Chapter VI

## Submission and peer-evaluation

Submit your assignment to your `Git` repository as usual. Only the work inside your repository will be evaluated during the defense. Do not hesitate to double-check the names of your folders and files to ensure they are correct.



16D85ACC441674FBA2DF65190663EC3C3C258FEA065D090A715F1B62F5A57F0B75403  
61668BD6823E2F873124B7E59B5CE94BB7ABD71CD01F65B959E14A3838E414F1E871  
F7D91730B