TP - Microservices with Docker
ING3-IMSI – Cloud for Data Analytics
2019–2020

1 Introduction

The objective of this hands-on exercise is to get familiar with networking, volumes management and microservices orchestration.

2 Linking containers

2.1 Using Dockerfiles

We have seen in the course slides how to set up a network and expose ports from different containers to create a multi-container, distributed application. Let us dive into it.

Set up the following infrastructure. Note that you will need one Dockerfile for each image to build:

— Start a container running a DBMS (a database). Use the `mysql:5.7` image from the Docker Hub.

— Once it is running, start a container running a CMS. You can user the `wordpress` image from the Docker Hub. Make sure that this container will be linked to the mysql container, so that Wordpress can use the MySQL database. Make sure as well that you open a port so that you can connect to the container.

— Open your browser and type the wordpress container's IP and port so that you can see the Wordpress installation page. Make sure it connects smoothly to the database. Take into account that, if you have not fixed the wordpress container's IP in the startup, you can find it with `docker inspect`.

2.2 Using docker-compose

Repeat the same exercise using a single `docker-compose.yml` file.

3 Setting up a development stack

In this exercise you are going to set up a development stack infrastructure. This means that you will configure a container as a platform for testing your code, and you will develop your code somewhere else (e.g. your laptop or another container).
You are free to chose the tools to install and the development environment, but you must adhere to the following requirements:

— Start from one or multiple existing image(s) in the Dockerhub (e.g. ubuntu, fedora, etc.).

— Use `docker-compose` to customise the image or images.

— The customised image will be the test environment where you will test and debug your code. But it will also become the production environment that you will give to the client, so make sure everything is well configured in the container. Think about everything you need to have installed (Java EE, JRE, Scala, Python, a database, etc.).

— You will develop your code outside the test container, which will mount a volume into the development directory so that you can run and test your code inside the container. The code to develop, as well as the tools you will use (your IDE, for example) can be in installed in your laptop or in another container.

— Once the code, the docker-compose.yml file and everything else you need work perfectly, commit those files to gitLab or gitHub and send your client (that's me) the link. I will clone the files and will run `docker-compose up`.