

LAMP Stack Deployment using Docker Compose

Ankit Jangir

Department of Information Technology
Dronacharya College of Engineering, Gurugram, India

Abstract: *Small businesses typically use a single cloud server to host multiple projects, and their databases are sometimes hosted on the same server. As a result, all projects are hosted in the same environment. This will cause problems in the future, such as the inability to upgrade the server's environment because all projects are dependent on the same environment; in the future, that environment support will be discontinued; or in the future, if the database fails, it will affect all projects that hosted on the server.*

Keywords: Cloud Server, Databases, Hosted, Environment

I. INTRODUCTION

Docker Compose is a service that allows you to run several containers as one. Each container operates alone, but they can communicate with one another if necessary. The scripting language YAML, which stands for Yet Another Markup Language and is based on XML, makes writing Docker Compose files a breeze. Another excellent feature of Docker Compose is that users may use a single command to enable all of the services (containers).

1.1 Docker Compose's Advantages

- The term "single host deployment" refers to the ability to execute everything on a single piece of hardware.
- YAML scripts provide for quick and easy configuration.
- Security - Because all of the containers are isolated from one another, the threat landscape is reduced. (Simpli learn, 2020)

LAMP are acronyms for Linux, Apache, MySQL, and PHP. Using docker-compose, I built three containers for each project. Container one includes PHP, Apache, and composer; container two includes MySQL as the database; and container three includes PhpMyAdmin for web-based database access. All three containers are linked to a particular project, allowing you to create a unique environment for each application and keeping your database separate from other databases.

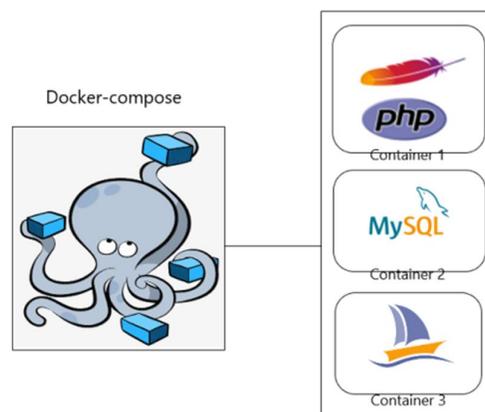


Fig. 1: LAMP stack on docker-compose

With the above scenarios, if one of the project databases fails, it will not affect the other projects on the same server, and if we want to upgrade the environment for a specific project in the future, we may do so without affecting the environment of other projects.

II. MATERIALS AND METHODS

I generated a 'docker-compose.yml' file to create the LAMP stack environment, in which I indicated three services: PHP server, MYSQL, and PhpMyAdmin.

To meet the project's requirements, I generated a custom image for the PHP Webserver. I also connected the database container to the application container, which I did by creating a network with the application container as the frontend and the database container as the backend. We may access the MYSQL database within the application container by constructing this network. On port 3000, we may connect to the application server. I utilised official MySQL images from Docker Hub for the MySQL server, and I also included some variables in the file, so we don't have to generate root or user authentication credentials.

```
version: "3.2"
services:
  PHP_Webserver:
    image: ankitjangir1690/apache-php:7.0
    networks:
      - frontend
      - backend
    environment:
      - MYSQL_HOST=mysql-app
      - MYSQL_USER=username
      - MYSQL_PASSWORD=password
      - MYSQL_DB=database_name
    volumes:
      - ./application:/var/www/html/
    ports:
      - "3000:80"

    container_name: application_container
  mysql:
    image: mysql:latest
    networks:
      - backend
    environment:
      - MYSQL_ROOT_PASSWORD=secret
      - MYSQL_USER=username
      - MYSQL_PASSWORD=password
      - MYSQL_DATABASE=atabase_name
    container_name: mysql-app
  phpmyadmin:
    image: phpmyadmin/phpmyadmin
    depends_on:
      - mysql
    networks:
      - backend
    ports:
      - "8080:80"
    environment:
      - PMA_HOST=mysql-app
      - PMA_PORT= 3306
    volumes:
      - /sessions
    container_name: phpmyadmin
networks:
  frontend:
  backend:
```

Fig. 2: docker-compose.yml file

Source: (docker docs)

I also utilised phpMyAdmin official images from docker hub for PhpMyAdmin, and I connected it to a MySQL container so that developers could access the database from the web. On port 8080, we can access phpMyAdmin or the database. We simply need to place this file in the empty folder and run the command "docker-compose up -d" from the terminal. The -d option allows you to run the script in the background while executing the 'docker-compose.yml file'.

This docker file first fetches images from the docker hub, and then uses those images to build our three containers on the server with the desired environment. We only need to place the code on the running PHP Webserver, modify the database credentials, then start the Apache service using the command 'service apache2 start' once the container is up and running. After that, our application will be able to access on port 3000, which we can map to our domain name. As a result, we

can only access it via the domain name. We also need to upload dump of the database on the mysql server so that the application may access the data.

III. RESULTS AND DISCUSSIONS

1. **Simple and Quick Configuration:** The LAMP application's environment can be quickly set up using YAML scripts and environment variables. On the same server, we might use multiple environments for different applications.
2. **Internal Communication that is Secure:** Compose establishes a network through which all of the services can be shared. Because the services can't be accessible from the outside, this gives an extra degree of protection to the programme.
3. **Resource Utilization that is Efficient as Possible:** We can use Docker Compose to host several isolated environments on a single host. You can save a lot of resources by running everything on a single piece of hardware. It has capabilities that allow it to cache configurations and reuse existing containers, which helps it save resources.
4. **Setting up Environment Variable:** We may use Docker Compose to construct environment variables from a single file. Users for the database can be created from the outside.

IV. CONCLUSION

In this article, we discussed how small businesses and start-ups are using the same environment for many projects and the risks they may face in the future in this post. After that, we spoke about how to use docker compose to fix the problem. We created a docker-compose file and put it to the test for setting up various environments for various projects. We worked on it together, and I used the same architecture in my internship company. All the LAMP stack projects were moved from the staging server to the docker containers.

REFERENCES

- [1]. docker docs. (n.d.). Retrieved from <https://docs.docker.com/>: <https://docs.docker.com/compose/>
- [2]. Simpli learn. (2020, 03 01). Retrieved from www.simplilearn.com: <https://www.simplilearn.com/tutorials/docker-tutorial/docker-compose>