**Docker Compose and WordPress**

We can use Docker Compose to easily run WordPress in an isolated environment built with Docker containers.

**Install Docker Compose**

**Run this command to download the current stable release of Docker Compose:**

**sudo curl -L "https://github.com/docker/compose/releases/download/1.28.5/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose**

**Apply executable permissions to the binary:**

sudo chmod +x /usr/local/bin/docker-compose

Note: If the command docker-compose fails after installation, check your path. You can also create a symbolic link to /usr/bin or any other directory in your path.

**For example:**

sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose

**Test the installation.**

$ docker-compose --version

docker-compose version 1.25.3, build 1110ad01

**Creating Wordpress Application:**

**Create an empty project directory.**

**mkdir wordpress**

**Change into your project directory.**

cd wordpress

**Create a docker-compose.yml file that starts your WordPress blog and a separate MySQL instance with a volume mount for data persistence:**

vim docker-compose.yml

version: “3.9”

services:

db:

image: mysql:5.7

volumes:

- db\_data:/var/lib/mysql

restart: always

environment:

MYSQL\_ROOT\_PASSWORD: somewordpress

MYSQL\_DATABASE: wordpress

MYSQL\_USER: wordpress

MYSQL\_PASSWORD: wordpress

wordpress:

depends\_on:

- db

image: wordpress:latest

ports:

- "8000:80"

restart: always

environment:

WORDPRESS\_DB\_HOST: db:3306

WORDPRESS\_DB\_USER: wordpress

WORDPRESS\_DB\_PASSWORD: wordpress

WORDPRESS\_DB\_NAME: wordpress

volumes:

db\_data: {}

**Build the project:**

docker-compose up -d

Creating network "my\_wordpress\_default" with the default driver

Pulling db (mysql:5.7)...

5.7: Pulling from library/mysql

efd26ecc9548: Pull complete

a3ed95caeb02: Pull complete

...

Digest: sha256:34a0aca88e85f2efa5edff1cea77cf5d3147ad93545dbec99cfe705b03c520de

Status: Downloaded newer image for mysql:5.7

Pulling wordpress (wordpress:latest)...

latest: Pulling from library/wordpress

efd26ecc9548: Already exists

a3ed95caeb02: Pull complete

589a9d9a7c64: Pull complete

...

Digest: sha256:ed28506ae44d5def89075fd5c01456610cd6c64006addfe5210b8c675881aff6

Status: Downloaded newer image for wordpress:latest

Creating my\_wordpress\_db\_1

Creating my\_wordpress\_wordpress\_1

**Bring up WordPress in a web browser**

http://publicip:8000 external browser



**Check the mapping of volumes:**

* Observe the contents of under volumes folder on the host and /var/lib/mysql is same. Play around by creating files in host and containers and observe how they get mapped.

**Host:**

[root@ip-172-31-37-215 word\_db\_data]# cd \_data

[root@ip-172-31-37-215 \_data]# ls

auto.cnf client-cert.pem ibdata1 ibtmp1 private\_key.pem server-key.pem

ca-key.pem client-key.pem ib\_logfile0 mysql public\_key.pem sys

ca.pem ib\_buffer\_pool ib\_logfile1 performance\_schema server-cert.pem example.txt

**Container:**

root@2b2aa712085d:/# cd /var/lib/mysql

root@2b2aa712085d:/var/lib/mysql# ls

auto.cnf client-cert.pem ib\_buffer\_pool ibdata1 performance\_schema server-cert.pem wordpress

ca-key.pem client-key.pem ib\_logfile0 ibtmp1 private\_key.pem server-key.pem

ca.pem ib\_logfile1 mysql public\_key.pem sys

**Shutdown and cleanup (Ensure you are in wordpress folder)**

Shutdown the application by running the he command docker-compose down (Removes the containers but preserves WordPress volume)

Remove the volumes by running **docker-compose down --volumes** removes the volumes.