**Docker compose**

* Compose is a tool for defining and running multicontainer docker application
* Use YAML file to configure your applications service
* With single command, you create and start all the services from your configuration

Using Compose is basically a three-step process:

1. Define your app’s environment with a Dockerfile so it can be reproduced anywhere.
2. Define the services that make up your app in docker-compose.yml so they can be run together in an isolated environment.
3. Run docker-compose up and Compose starts and runs your entire app.

# docker-compose --version

open google --type install docker compose --open docs.docker --open/select linux --copy

command to download the latest version of docker compose : curl -L https://github .........

copy and paste in system

Again copy executable permission to the binary

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

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

#docker-compose --version

# mkdir compose

*# docker pull mysql:5.7*

*# docker pull wordpress*

Downloading images is optional. Docker compose finds the image in local system, if not available then pull it from dockerhub repository

**Configure Wordpress Application**

# cd compose

# nano wordpress.yml ( This is Sample file)

version: '3.3'

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: {}

# nano wordpress1.yml

**version**: '3.3'

**services**:

**db**:

**image**: mysql:5.7

**restart**: always

**volumes**:

- db\_data:/var/lib/mysql

**environment**:

**MYSQL\_ROOT\_PASSWORD**: password

**MYSQL\_DATABASE**: wordpress

**wordpress**:

**image**: wordpress

**restart**: always

**volumes**:

- ./wp\_data:/var/www/html

**ports**:

- "8080:80"

**environment**:

**WORDPRESS\_DB\_HOST**: db:3306

**WORDPRESS\_DB\_NAME**: wordpress

**WORDPRESS\_DB\_USER**: root

**WORDPRESS\_DB\_PASSWORD**: password

**depends\_on**:

- db

**volumes**:

**db\_data**:

**wp\_data**:

# docker-compose up -d

# mv wordpress.yaml docker-compose.yml

#ls

#docker-compose up -d

After installation -- copy public ip of the system and paste in browser

publicip:8000

# docker ps

**To Stop**

docker-compose stop

**To Start**

docker-compose start

**Shutdown and cleanup**

The command [docker-compose down](https://docs.docker.com/compose/reference/down/) removes the containers and default network, but preserves your WordPress database.

The command docker-compose down --volumes removes the containers, default network, and the WordPress database.

**To uninstall Docker Compose if you installed using curl**

# rm /usr/local/bin/docker-compose

**Configure Drupal Application**

# vi docker-compose.yml

version: '3.3'

services:

drupal:

image: drupal:latest

ports:

- 80:80

volumes:

- drupal\_modules:/var/www/html/modules

- drupal\_profiles:/var/www/html/profiles

- drupal\_themes:/var/www/html/themes

- drupal\_sites:/var/www/html/sites

restart: always

postgres:

image: postgres:10

environment:

POSTGRES\_PASSWORD: your\_postgres\_password

volumes:

- db\_data:/var/lib/postgresql/data

restart: always

volumes:

drupal\_modules:

drupal\_profiles:

drupal\_themes:

drupal\_sites:

db\_data:

#docker-compose up -d