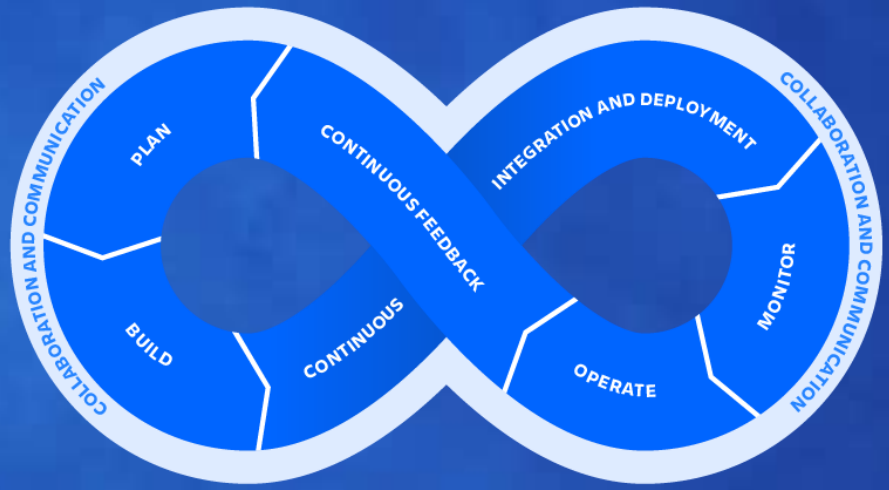

Session 15. Docker Compose

Ram N Sangwan

Agenda

- Installing Docker compose
- Terminology in Docker compose
- Build word press site using Docker compose

Docker Compose

- Compose is a tool for defining and running multi-container Docker applications.
- With Compose, you use a YAML file to configure your application’s services.
- Then, with a single command, you create and start all the services from your configuration.

Features

- Multiple isolated environments on a single host
- Preserve volume data when containers are created
- Only recreate containers that have changed
- Variables and moving a composition between environments

Install docker-compose

- Docker Compose relies on Docker Engine for any meaningful work.
- On desktop systems like Docker Desktop for Mac and Windows, Docker Compose is included as part of those desktop installs.
- On Linux systems, first install the Docker Engine, then come back here for instructions on installing Compose on Linux systems.

  # sudo curl -L "https://github.com/docker/compose/releases/download/1.28.6/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
- Change the permission of docker-compose binary

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

  # docker-compose --help
- Test the installation.

  $ docker-compose --version

  docker-compose version 1.28.6, build 1110ad01

https://docs.docker.com/compose/gettingstarted/

Using Compose

- Using Compose is basically a three-step process:
  - Define your app's environment with a Dockerfile so it can be reproduced anywhere.
  - Define the services that make up your app in docker-compose.yml so they can be run together in an isolated environment.
  - Run docker compose up and the Docker compose command starts and runs your entire app. You can alternatively run docker-compose up using the docker-compose binary.

A docker-compose.yml

```
version: "3.9"  # optional since v1.27.0
services:
  web:
    build: .
    ports:
      - "5000:5000"
    volumes:
      - .:/code
      - logvolume01:/var/log
    links:
      - redis
  redis:
    image: redis
volumes:
  logvolume01: {}
```

Terminology and Commands

- Compose has commands for managing the whole lifecycle of your application:
  - Start, stop, and rebuild services
  - View the status of running services
  - Stream the log output of running services
  - Run a one-off command on a service

Build Wordpress using Docker compose

1. Create an empty project directory.

   This project directory contains a docker-compose.yml file which is complete in itself for a good starter wordpress project.

2. Change into your project directory.

   For example, if you named your directory my_wordpress:

   $ cd my_wordpress/

3. Create a docker-compose.yml file that starts your WordPress blog and a separate MySQL instance with a volume mount for data persistence:
   - The docker volume db_data persists any updates made by WordPress to the database.
   - WordPress Multisite works only on ports 80 and 443.

```
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

- Now, run docker-compose up -d from your project directory.
- This runs docker-compose up in detached mode, pulls the needed Docker images, and starts the wordpress and database containers, as shown in the example below.

  $ docker-compose up -d

Bring up WordPress in a web browser

- At this point, WordPress should be running on port 8000 of your Docker Host.

Thank You