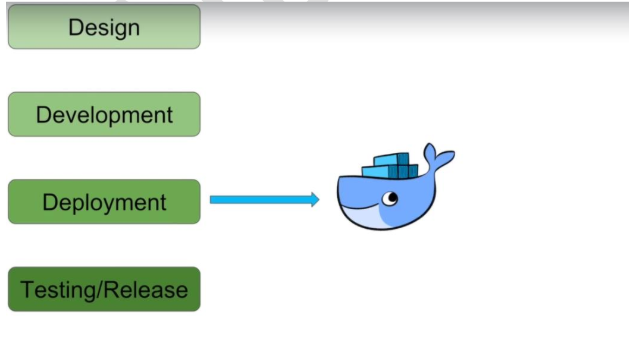
 **DOCKER TUTORIAL – 5**

**What covered in previous lecture:**

* **What is docker ?**
* **Why docker**
* **What is containerisation**
* **Docker setup in linux**
* **What is Dockerfile**
* **Dockerfile Keywords**
* **DockerHub account**
* **Dockerhub Integration with VMs -Docker login**
* **Dockerhub push & pull**
* **Working with Docker Images**
* **Java Dockerfile**
* **Next.js Dockerfile**
* **Python app with Docker**
* **Docker Commands**
* **Create & Containerize your first (Lab)**
* **Docker File – Docker Image**

**What you will Learn in this Tutorial:**

* **Docker Port mapping**
* **Docker Volumes**
* **Docker Compose**
* **Create & Containerize your first (Lab)**
* **Docker File – Docker Image**
* **Overview of Docker Hub**
* **What is docker-compose file?**
* **Why do we need this?**
* **Writing a docker-compose file**
* **Pre-requisite**
* **Starting the services**
* **Scale up or Scale down the services**
* **Inspect the container to see container details**
* **Stopping the services**
* **References**

****

**Docker overview**

**Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly. With Docker, you can manage your infrastructure in the same ways you manage your applications. By taking advantage of Docker's methodologies for shipping, testing, and deploying code, you can significantly reduce the delay between writing code and running it in production.**

**To install docker-compose tun the following commands:**

**sudo apt install docker.io  
sudo chown root:docker /var/run/docker.sock**

**sudo chmod 660 /var/run/docker.sock**

**sudo systemctl start docker**

**sudo systemctl enable docker**

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

To verify that your Docker Compose installation was successful, you can use the following command:

docker-compose --version

Running this command will display the version of Docker Compose that is currently installed on your system. If you see the version information, it means the installation was successful, and Docker Compose is ready for use.

**Writing a docker-compose file**

**Pre-requisite**

**Firstly, make sure to have docker-compose installed in you system. However If not already installed, install from here.**

**Create a file with name docker-compose.yml and use the below code snippet to begin,**

**version: "3"**

**services:**

**application:**

**image: app2**

**ports:**

**- "8080:7000"**

**networks:**

**- springboot-db-net**

**depends\_on:**

**- mysqldb**

**volumes:**

**- /data/springboot-app**

**mysqldb:**

**image: mysql:5.7**

**networks:**

**- springboot-db-net**

**environment:**

**- MYSQL\_ROOT\_PASSWORD=root**

**- MYSQL\_DATABASE=sbms**

**volumes:**

**- /data/mysql**

**networks:**

**springboot-db-net:**

**LAB**

**Docker Installation**

* **sudo apt install docker.io -y**
* sudo usermod -aG docker $USER
* sudo usermod -aG docker ubuntu
* newgrp docker
* sudo chown root:docker /var/run/docker.sock
* sudo chmod 660 /var/run/docker.sock
* sudo systemctl start docker
* sudo systemctl enable docker

**docker-compose Install the following commands:**

* sudo curl -L "https://github.com/docker/compose/releases/latest/download/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose  
  sudo chmod +x /usr/local/bin/docker-compose
* To verify that your Docker Compose installation was successful, you can use the following command:
* docker-compose --version
* Running this command will display the version of Docker Compose that is currently installed on your system. If you see the version information, it means the installation was successful, and Docker Compose is ready for use.

**Steps :**

**Git clone** [**https://github.com/Aseemakram19/ourspring-boot-mysql-docker-compose.git**](https://github.com/Aseemakram19/ourspring-boot-mysql-docker-compose.git)

* **sudo apt update**
* **sudo apt install maven**
* **cd ourspring-boot-mysql-docker-compose**
* **mvn clean package**
* **docker built -t spring-boot-mysql-app .**
* **docker images**
* **docker-compose up -d**
* **open port in security group 8080 port**
* **access ip:8080 on browser**
* **docker-compose ps**
* **Go inside the container to check data in tables of mysql**
* **Mysql -h localhost -u root -proot**
* **show databases ;**
* **use sbms ;**
* **show tables ;**
* **select \* from book;**

**Starting the services**

**docker-compose up -d**

**groups Jenkins**

**Scale up or Scale down the services**

**Scaling up database service**

**Similarly, scaling down database services**

**Inspect the container to see container details**

**docker container inspect <containerID>**

**This will give a json response containing all the details related to container. We can find network and volume associated with the container.**

* [**Stopping the services**](https://blog.nashtechglobal.com/understanding-the-docker-compose-file/#elementor-toc__heading-anchor-7)

**docker-compose down**

