

# Module 8: Containerization, Docker, and Docker Hub

### Abhijit Lalasaheb Zende ###

## 1. L1 – Create Docker file and build the docker container Application Image for the application build in Jenkins Module

Ans.

(\*\*\* Note: Screen shots attached to end of each question \*\*\*)

### Step 1: Start the Jenkins slave node that we created in the Jenkins module assignment:

Boot up the EC2 instance configured as the Jenkins slave node. This instance, set up in our master-slave architecture, is necessary for executing the builds we have scheduled on the slave, as directed by the Jenkins master server.

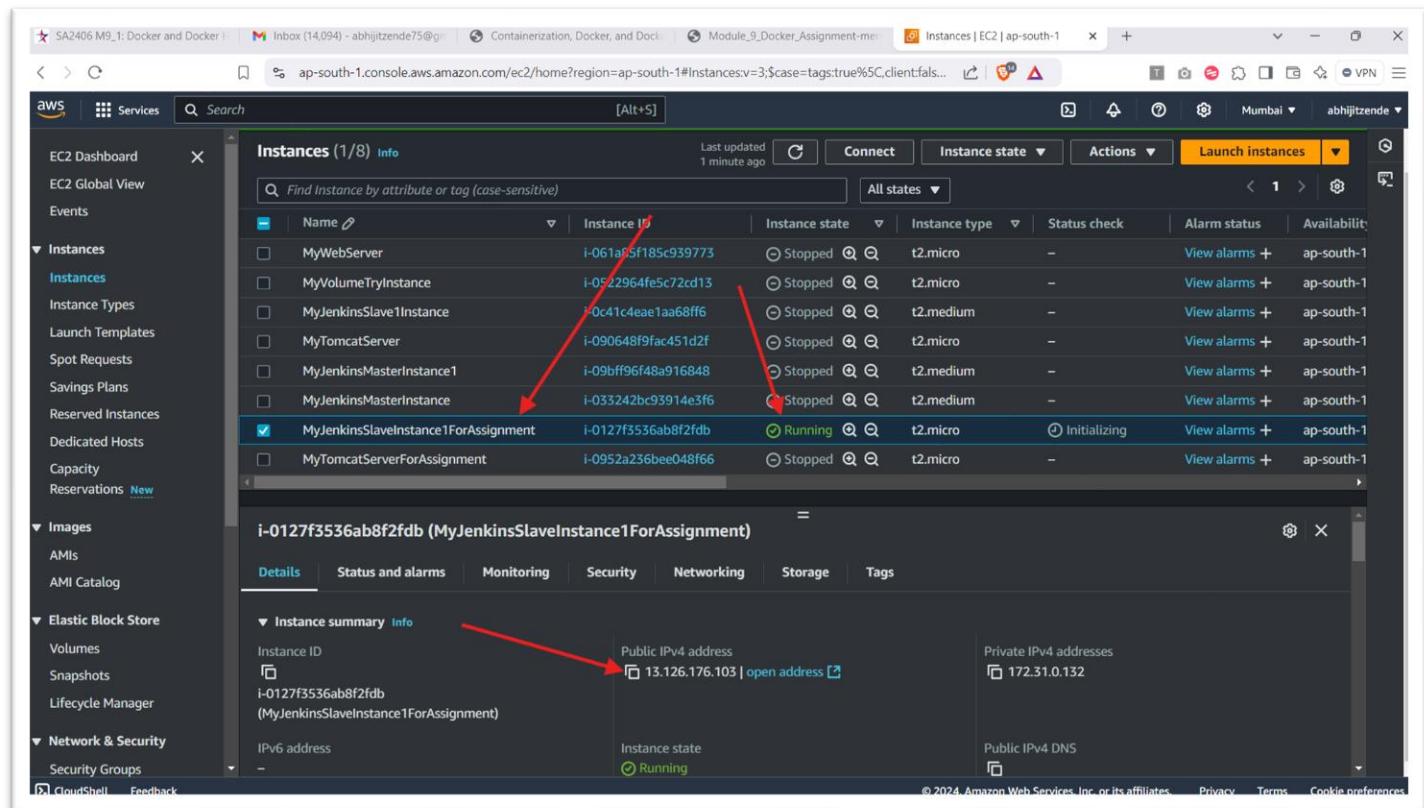


Fig. 1.01: Starting the Jenkins slave node EC2 instance

## Step 2: SSH into the started instance:

Launch MobaXterm and gain root privileges. Then, switch to the '`devopsadmin`' user account we previously created. This account has access to the build artifacts, which are stored in the '`workspace/sample_pipeline-project/target/`' directory. In this location, you will find the WAR files generated during our application's build process.

```
git: devopsadmin@ip-172-31-0-132:~/workspace/DemoPipelineProject/target  
ubuntu@ip-172-31-0-132:~$ sudo -i  
root@ip-172-31-0-132:~# su - devopsadmin  
devopsadmin@ip-172-31-0-132:~$ ls  
aches remoting remoting.jar workspace  
devopsadmin@ip-172-31-0-132:~$ cd workspace/  
devopsadmin@ip-172-31-0-132:~/workspace$ ls  
emoFreestyleProject1 DemoPipelineProject1 DemoPipelineProject1@tmp  
devopsadmin@ip-172-31-0-132:~/workspace$ cd DemoPipelineProject1  
devopsadmin@ip-172-31-0-132:~/workspace/DemoPipelineProject1$ ls  
Dockerfile bin maven pom.xml src target  
devopsadmin@ip-172-31-0-132:~/workspace/DemoPipelineProject1$ cd target/  
devopsadmin@ip-172-31-0-132:~/workspace/DemoPipelineProject1/target$ ls  
lasses demo-1.0-SNAPSHOT demo-1.0-SNAPSHOT.war demo-1.0-SNAPSHOT.war.original generated-sources generated-test-sources maven-archiver maven-status surefire-reports test-classes  
devopsadmin@ip-172-31-0-132:~/workspace/DemoPipelineProject1/target$ |
```

*Fig. 1.02: Project Preview*

### Step 3: Install Docker:

Install Docker by switching to root and running `apt install docker.io`. Verify the installation by checking the Docker version. Grant the 'devopsadmin' user Docker access by adding them to the docker group with `usermod -aG docker devopsadmin`. With Docker set up, we're now ready to create our application image by packaging the app and its dependencies using a **Dockerfile**.

The screenshot shows a terminal window with several lines of command-line output. Red arrows point to two specific lines: one at the top pointing to the command 'apt-get update' and another further down pointing to the command 'apt-get install apt-transport-https ca-certificates curl software-properties-common gnupg'. The terminal shows the progress of package updates, including dependency resolution and the download of various files from the 'jammy' repository.

```
root@ip-172-31-0-132:~# apt-get update
...
root@ip-172-31-0-132:~# apt-get install apt-transport-https ca-certificates curl software-properties-common gnupg

```

Fig. 1.03: Update packages

*Fig. 1.04: Add Docker’s Official GPG Key and Add Docker Repository*

```
root@ip-172-31-0-132:~# docker --version
docker version 27.3.1, build cef2230
root@ip-172-31-0-132:~# systemctl enable docker
Synchronizing state of docker.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable docker
root@ip-172-31-0-132:~# systemctl start docker
root@ip-172-31-0-132:~# usermod -aG docker devopsadmin
root@ip-172-31-0-132:~# su - devopsadmin
devopsadmin@ip-172-31-0-132:~$ docker run hello-world
Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
   (amd64)
3. The Docker daemon created a new container from that image which runs the
   executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it
   to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas:
https://docs.docker.com/get-started/
devopsadmin@ip-172-31-0-132:~$ |
```

*Fig. 1.05: Install Docker Engine and enable it on startup*

#### Step 4: Create Dockerfile for docker image:

Change to the **DemoPipelineProject1** directory. Create a Dockerfile using '**vi Dockerfile**'. In this file, use the '**FROM**' instruction to specify the base image, '**COPY**' to transfer WAR files from their source to the target location within the image, and '**EXPOSE**' to define the port for internet access. After adding these instructions, save the Dockerfile.

The screenshot shows a terminal window with a black background and white text. The terminal session starts with the command 'ls' in the root directory, followed by navigating into the 'workspace/DemoPipelineProject1' directory. Inside this directory, the user runs 'mvnw cmd pom.xml site target' to build the project. Then, they navigate to the 'target' directory and run 'ls' again. Finally, the user types 'vi Dockerfile' to open the Dockerfile for editing. A red arrow points from the text 'vi Dockerfile' back up towards the beginning of the terminal session, indicating the path taken to reach the Dockerfile creation step.

```
devopsadmin@ip-172-31-0-132:~$ ls
caches remoting remoting.jar workspace/
devopsadmin@ip-172-31-0-132:~$ ls workspace/
devopsFreestyleProject1 DemoPipelineProject1 DemoPipelineProject10tmp
devopsadmin@ip-172-31-0-132:~$ cd workspace/
devopsadmin@ip-172-31-0-132:~/workspace$ mvnw cmd pom.xml site target
devopsadmin@ip-172-31-0-132:~/workspace$ ls
classes demo-1.0-SNAPSHOT demo-1.0-SNAPSHOT.war demo-1.0-SNAPSHOT.war.original generated-sources generated-test-sources maven-archiver maven-status surefire-reports test-classes
devopsadmin@ip-172-31-0-132:~/workspace$ cd ..
devopsadmin@ip-172-31-0-132:~/workspace$ vi Dockerfile
```

Fig. 1.06: Navigation to Dockerfile

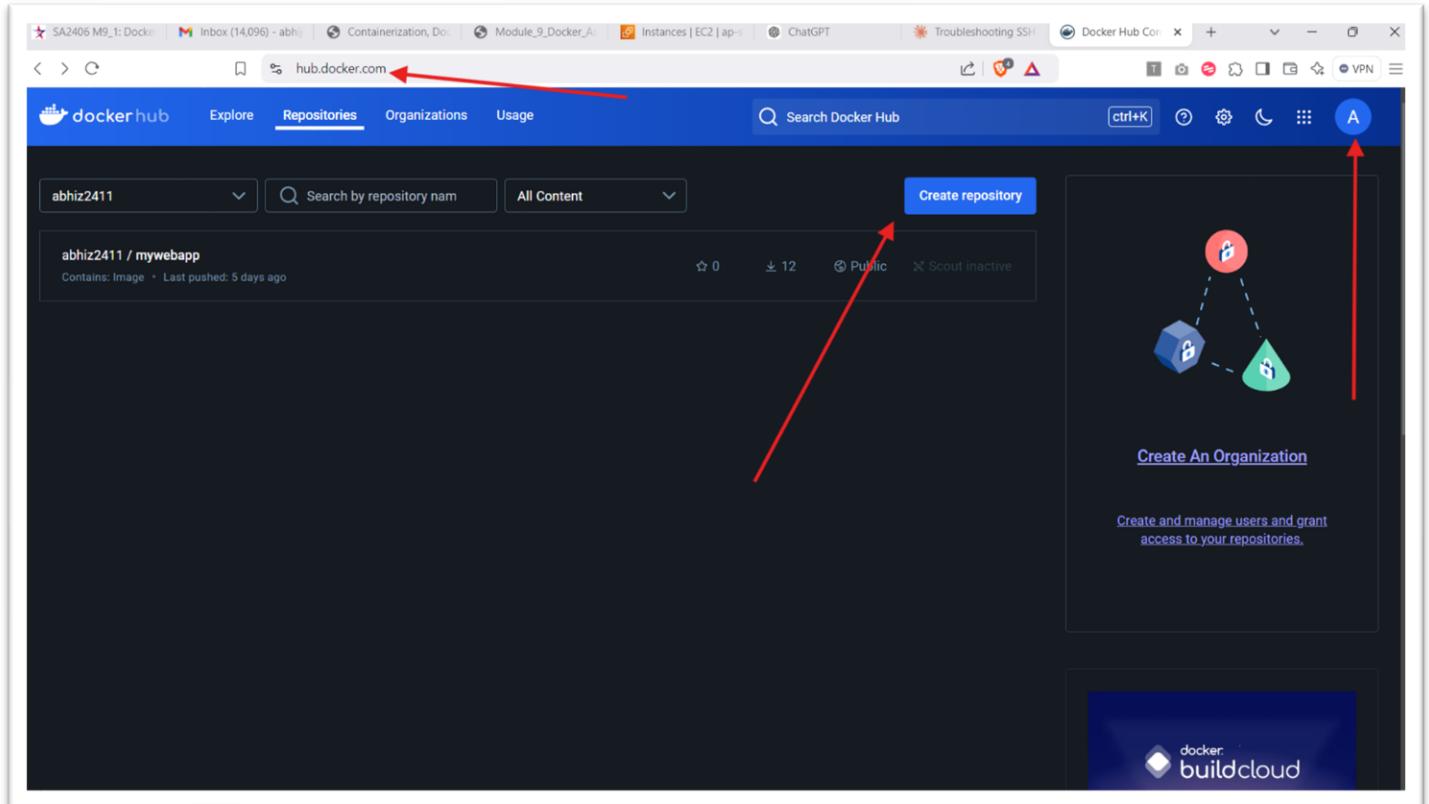


```
devopsadmin@DESKTOP-UJUJL: ~ /workspace/temopimplementuje
FROM tomcat:8.0
COPY ./target/*.war /usr/local/tomcat/webapps
EXPOSE 8080
```

*Fig. 1.07: Contents of Dockerfile*

## Step 5: Build the docker image:

Execute the command `docker build -t abhiz2411/tomcat\_web\_img:8.1 .` to create a new Docker image based on our Dockerfile. In this command, 'abhiz2411' is the DockerHub registry, 'tomcat\_web\_img:8.1' specifies the image name and its tag/version, and the final '.' indicates that the Dockerfile is in the current directory.



*Fig. 1.08: Create new repository on Docker Hub*

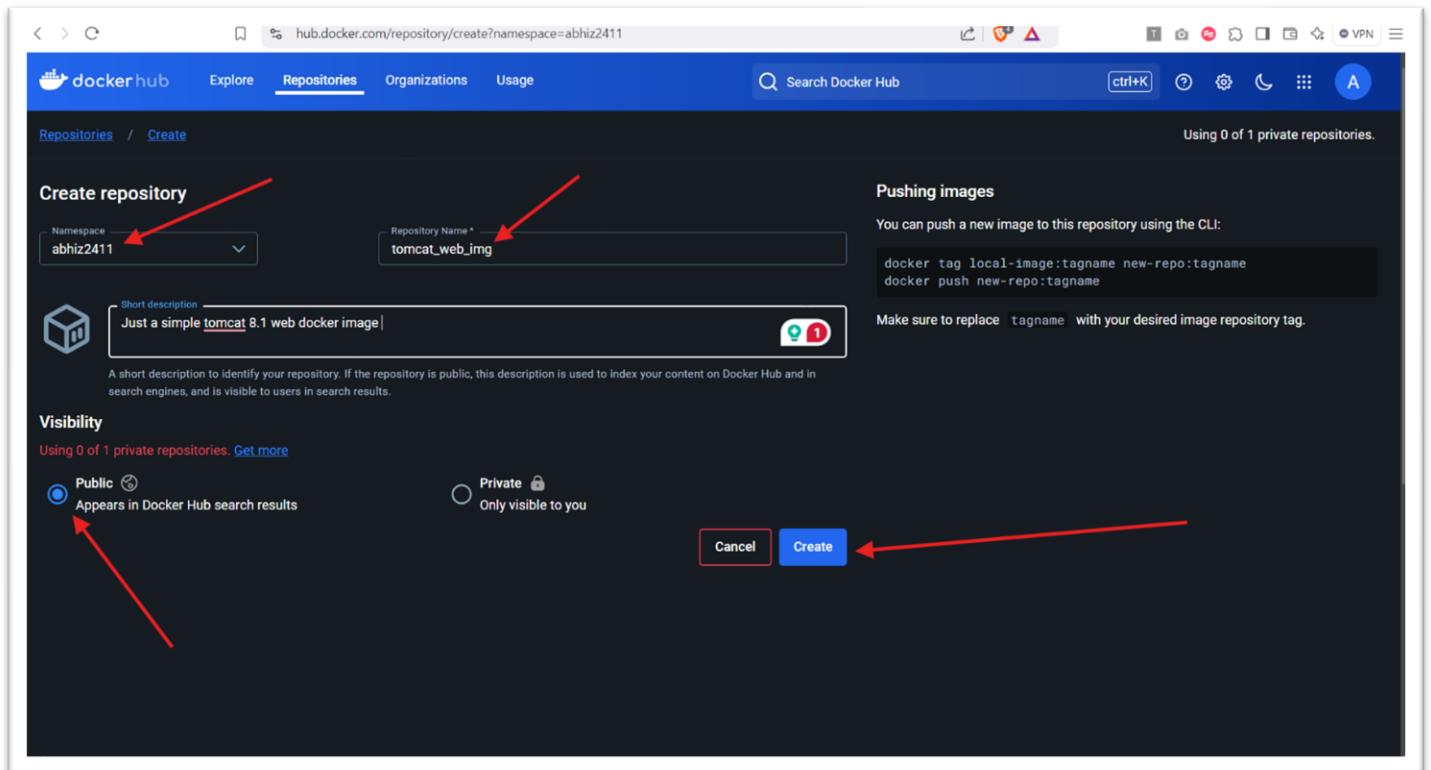


Fig. 1.09: Contents of Docker Hub repository

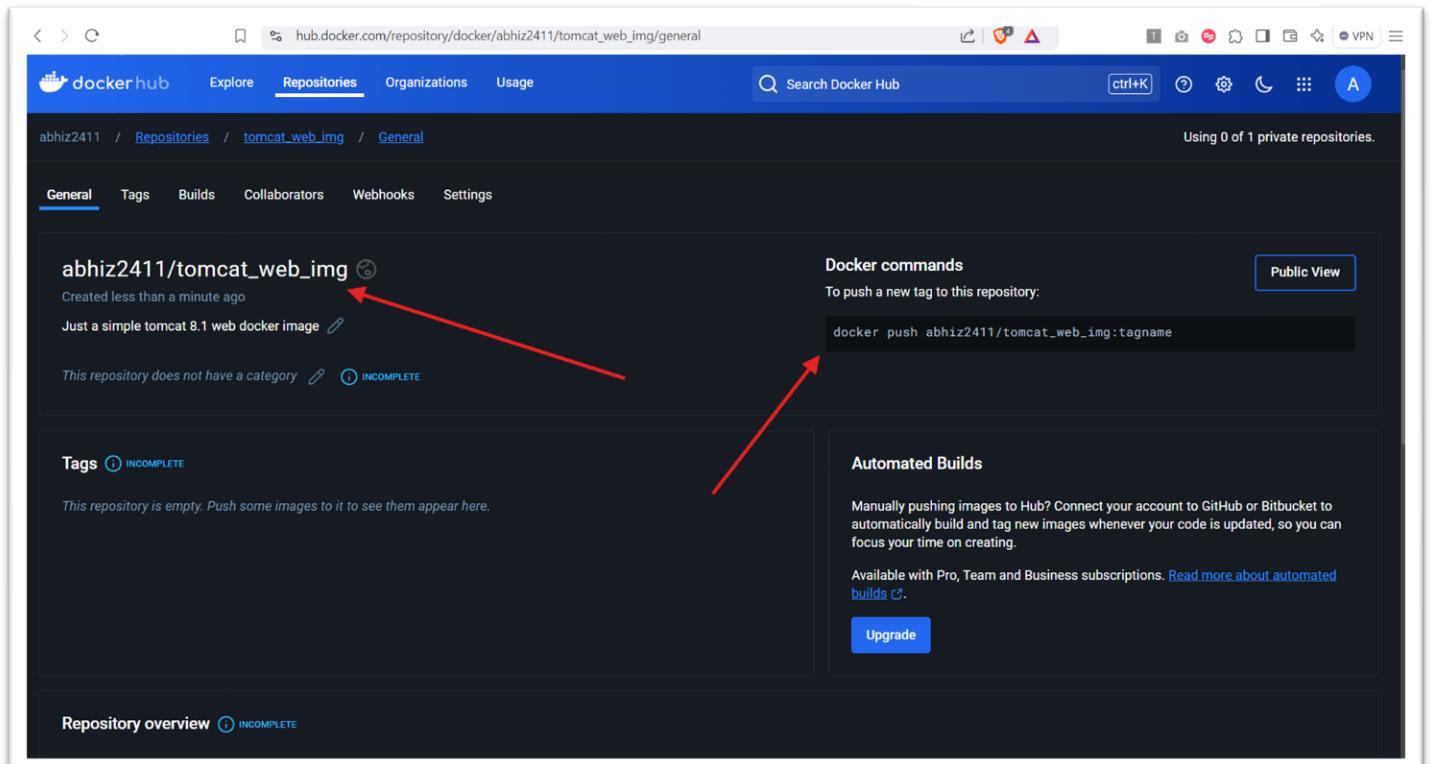


Fig. 1.10: Successful Docker Hub repository creation

*Fig. 1.11: Build Docker image*

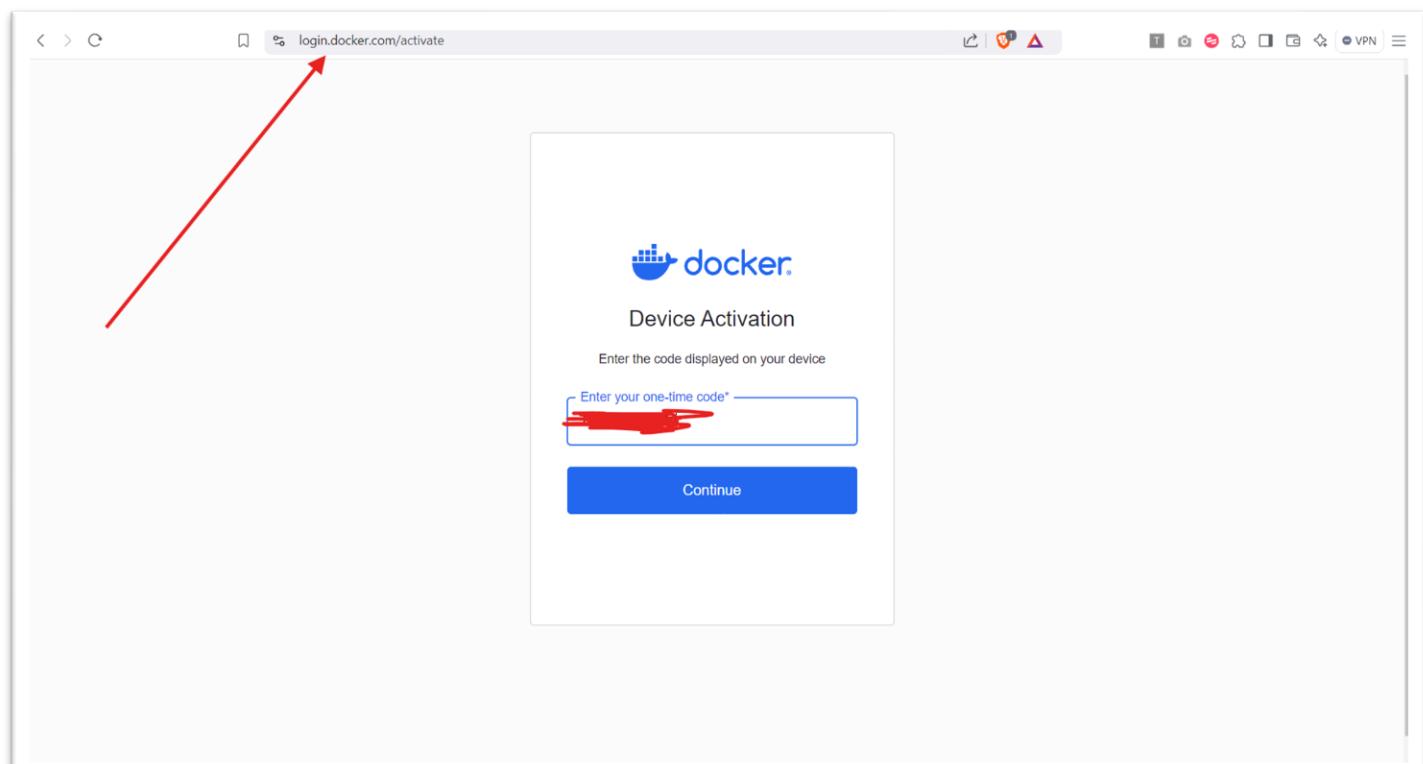
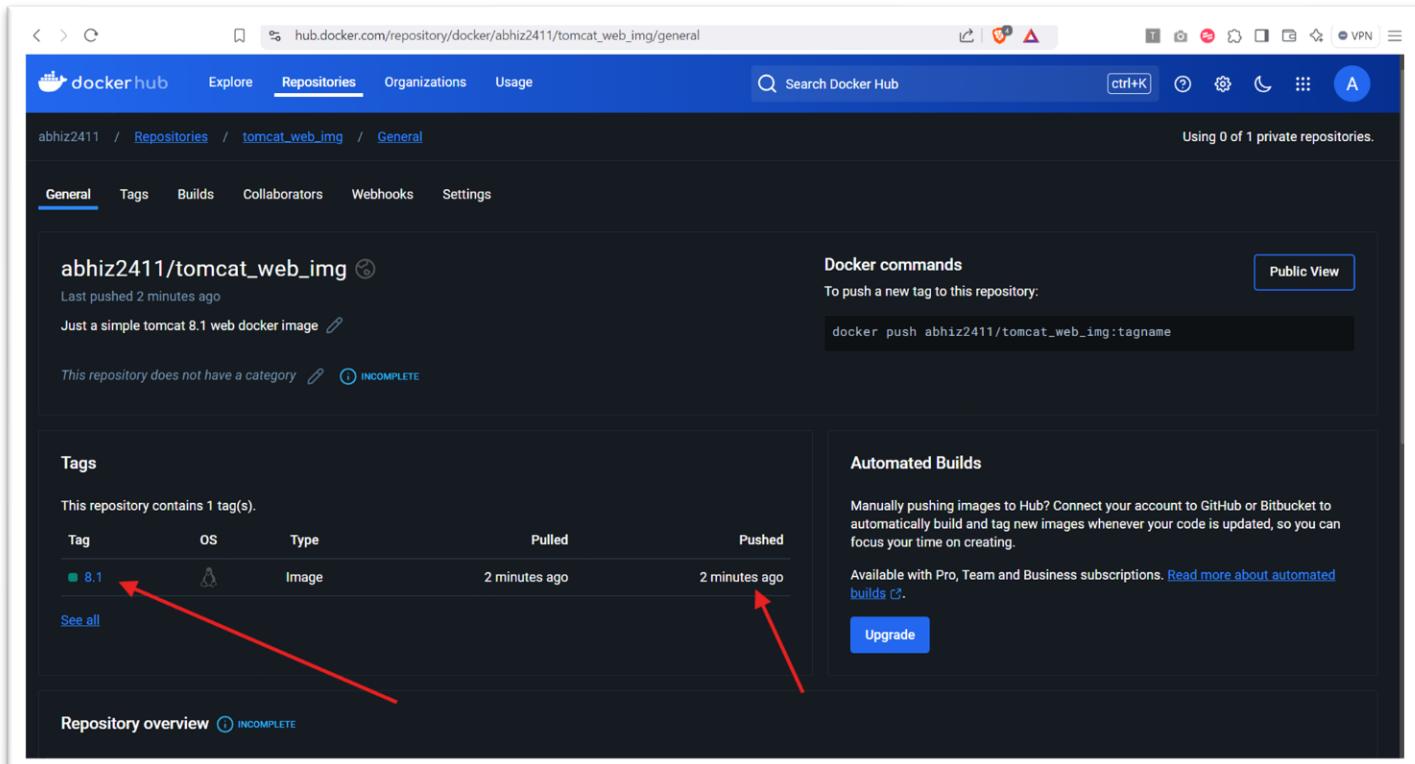


Fig. 1.12: Login Docker on the machine using command '`docker login`'

*Fig. 1.13: Pushing Docker image to Docker Hub*



*Fig. 1.14: Web view of the docker image we pushed*

## 2. L2 - Create the Container using the same Application Image and run the application in a Web Browser using container port mapping

Ans.

### Step 1: Creating and running a Docker container from the image:

To create and run a container from our newly built image, use the command `docker run -it [image\_name:tag]`. This starts the container in interactive mode. To make the application accessible via web browser, we need to bind the container's port to a host port. Use the command `docker run -d -p [host\_port]:[container\_port] [image\_name:tag]`. This maps the container's port to a port on the host, allowing web browser access to the application.

The screenshot shows a terminal window with the following command history:

```
devopsadmin@ip-172-31-0-132:~/workspace/DemoPipelineProject$ docker images
REPOSITORY          TAG      IMAGE ID      CREATED             SIZE
abhi2411/tomcat_web_img  8.1      0e3c5fcf0e80  54 minutes ago  402MB
hello-world          latest   d2c94e258dc8  17 months ago   13.3kB
devopsadmin@ip-172-31-0-132:~/workspace/DemoPipelineProject$ docker run -d -p 8090:8080 abhi2411/tomcat_web_img:8.1
b890840d26e57c9573cf0701445ff7d1ba1b05f0483cd0e9cc985edc4c
devopsadmin@ip-172-31-0-132:~/workspace/DemoPipelineProject$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS               NAMES
b890840d26e       abhi2411/tomcat_web_img:8.1   "catalina.sh run"   5 seconds ago     Up 4 seconds      0.0.0.0:8090->8080/tcp   heuristic_gagarin
devopsadmin@ip-172-31-0-132:~/workspace/DemoPipelineProject$
```

Red arrows highlight the first two lines of the command history, specifically the `docker run` command and its output.

Fig. 2.01: Running docker container and port mapping

## Step 2: Verifying container creation and configuring security group for web access:

Confirm the container is running by using `docker ps -a` to list all containers. To access the web application from a browser, modify the instance's security group. Edit the inbound rules, adding a new rule that either specifies ports 8080-8089 or allows all traffic. Save these changes to enable external access to your containerized application.

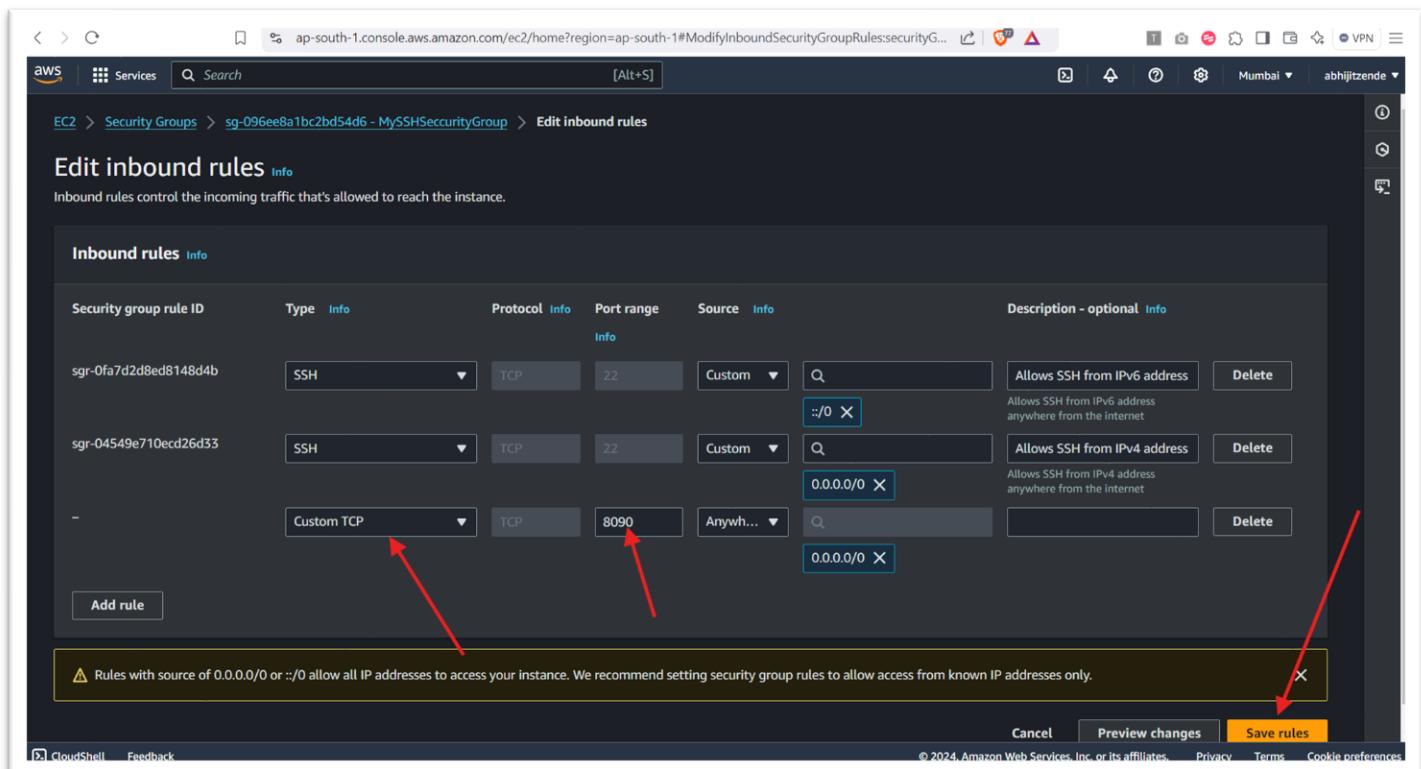
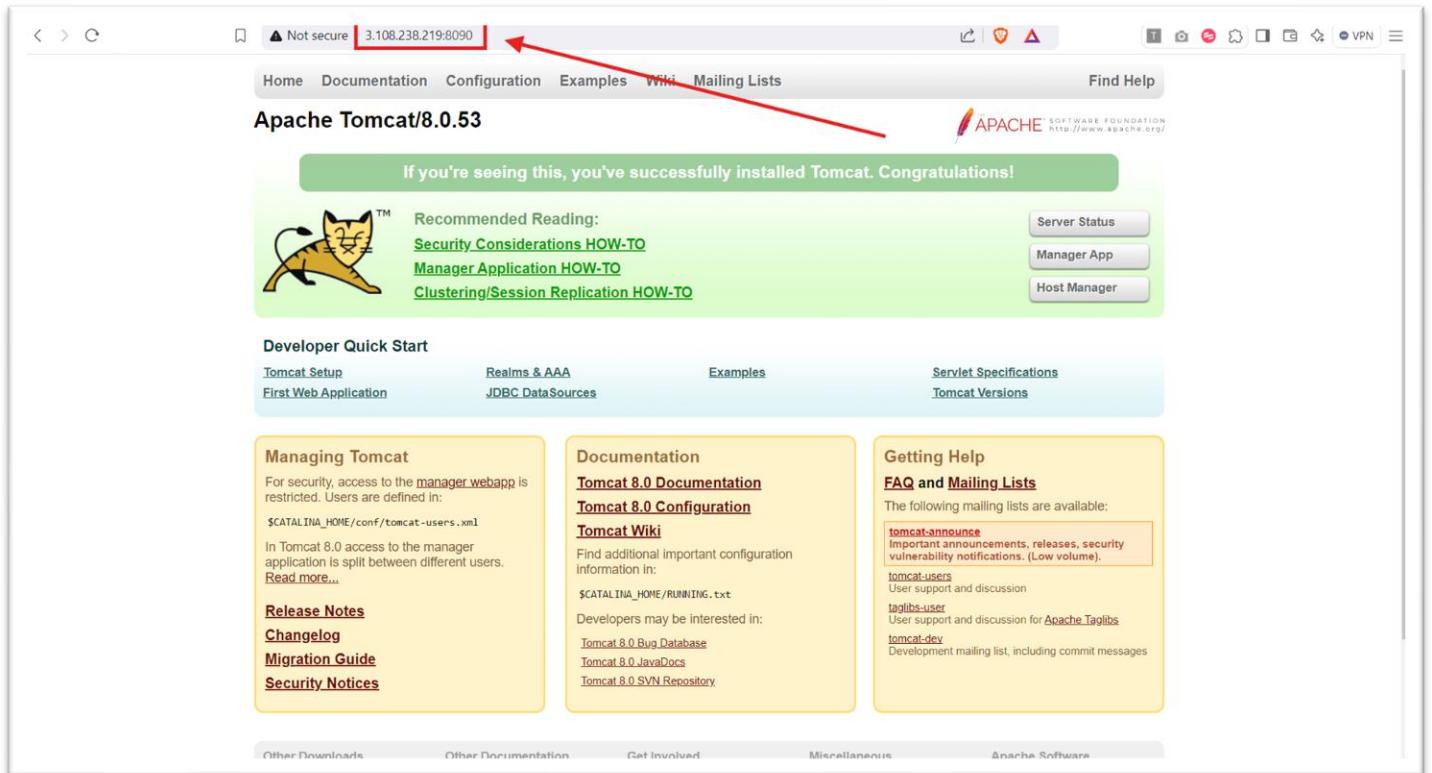


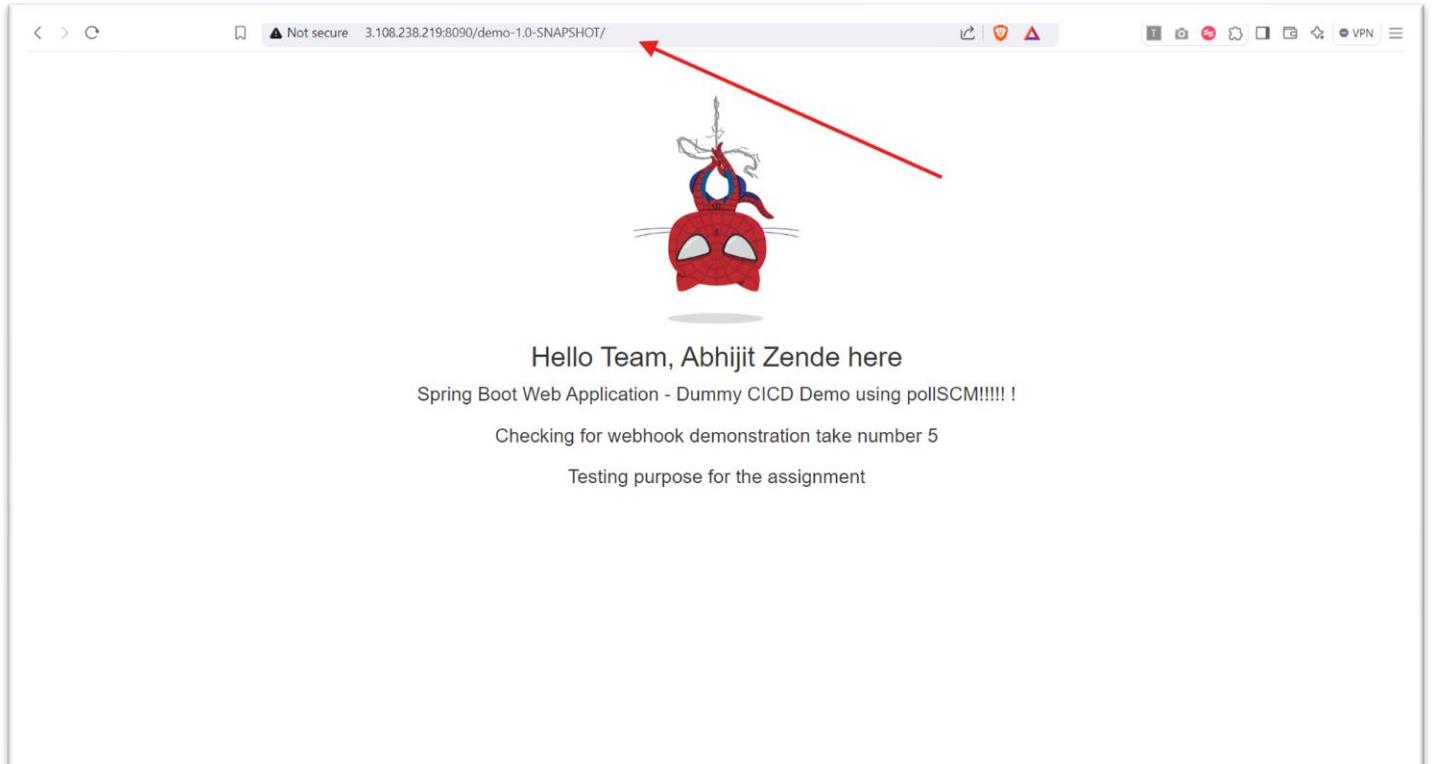
Fig. 2.02: Edit inbound rule of attached security group to allow traffic to host port

## Step 3: Viewing the application in the browser:

Copy the EC2 instance's public IP address, open a browser, and enter the IP along with the Host Port and the path to the WAR file to access the application.



*Fig. 3.01: Web view of tomcat application*



*Fig. 3.02: Application web preview*

```
devopsadmin@ip-172-31-0-132:~/workspace/DemoPipelineProject1/target$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS               NAMES
b890840d26e        abhi22411/tomcat_web_img:8.1   "catalina.sh run"   28 minutes ago   Up 27 minutes   0.0.0.0:8090->8080/tcp, [::]:8090->8080/tcp   heuristic_gagarin
devopsadmin@ip-172-31-0-132:~/workspace/DemoPipelineProject1/target$ docker stop AC
devopsadmin@ip-172-31-0-132:~/workspace/DemoPipelineProject1/target$ docker stop heuristic_gagarin
devopsadmin@ip-172-31-0-132:~/workspace/DemoPipelineProject1/target$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS               NAMES
devopsadmin@ip-172-31-0-132:~/workspace/DemoPipelineProject1/target$ docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS               NAMES
b890840d26e        abhi22411/tomcat_web_img:8.1   "catalina.sh run"   28 minutes ago   Exited (143) 8 seconds ago   heuristic_gagarin
devopsadmin@ip-172-31-0-132:~/workspace/DemoPipelineProject1/target$ docker rm heuristic_gagarin
devopsadmin@ip-172-31-0-132:~/workspace/DemoPipelineProject1/target$ docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS               NAMES
```

Fig. 3.03: Stop and remove container once done

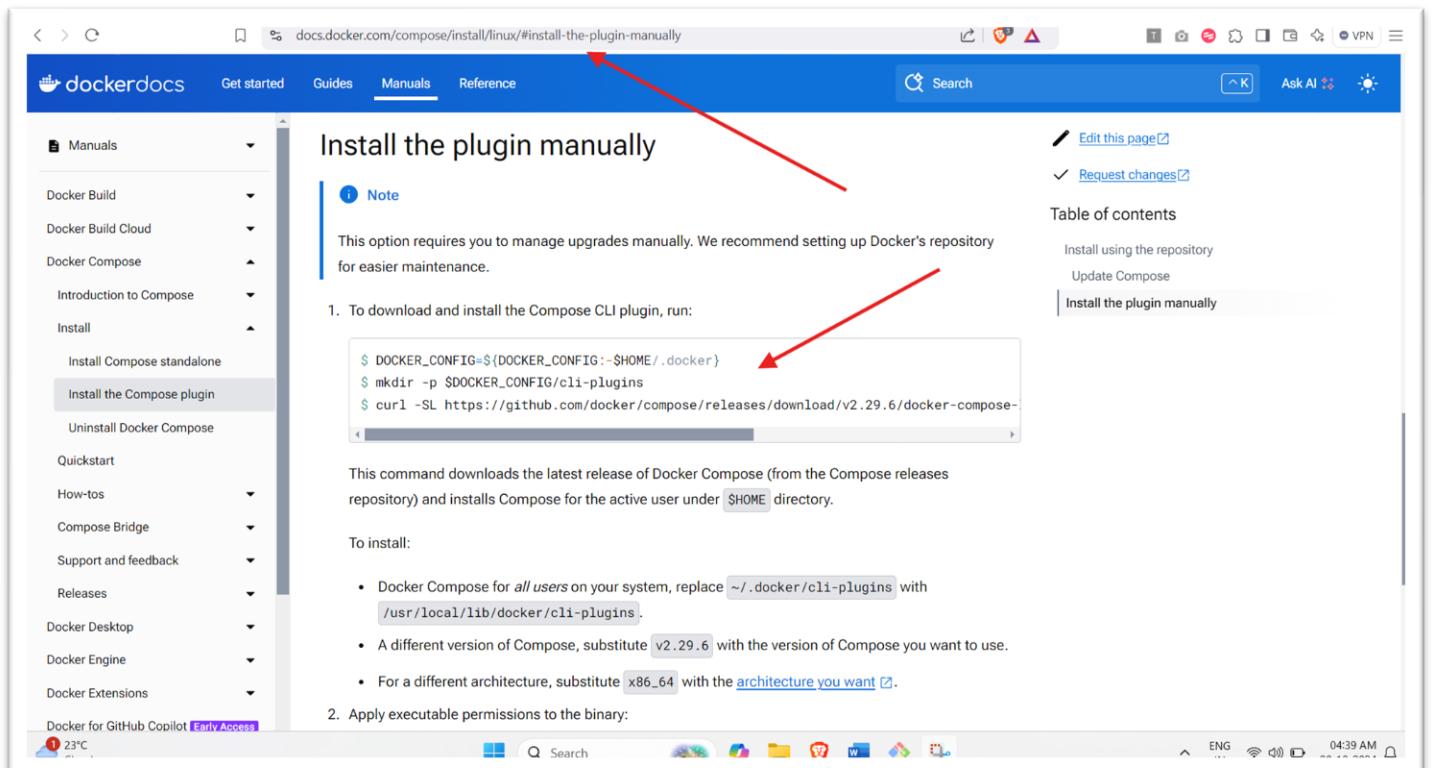
### 3. L3 - Demonstrate Docker Compose using the Application Image and MySQL Image to start and stop all container services

Ans.

#### Step 1: Install Docker compose:

Install Docker Compose on your machine by following the steps provided in Docker's official documentation. After installation, validate the setup by checking the version with:

' docker compose version'



Fig/ 3.01: Docker compose installation official manual

The screenshot shows a terminal window with a black background and white text. It displays a series of commands being run by a root user on an Ubuntu system. Red arrows point from the right margin to specific command lines:

- An arrow points to the first command: `sudo -i`.
- An arrow points to the command: `curl -sL https://github.com/docker/compose/releases/download/v2.29.6/docker-compose-linux-x86_64 -o $DOCKER_CONFIG/cli-plugins/docker-compose`.
- An arrow points to the command: `docker --version`.
- An arrow points to the command: `chmod +x $DOCKER_CONFIG/cli-plugins/docker-compose`.
- An arrow points to the command: `docker compose version`.

```
ubuntu@ip-172-31-0-132:~$ sudo -i
root@ip-172-31-0-132:~# DOCKER_CONFIG=${HOME}/.docker
mkdir -p $DOCKER_CONFIG/cli-plugins
curl -sL https://github.com/docker/compose/releases/download/v2.29.6/docker-compose-linux-x86_64 -o $DOCKER_CONFIG/cli-plugins/docker-compose
% Total    % Received % Xferd  Average Speed   Time   Time  Current
0          0     0      0      0      0      0      0      0      0
100 60.8M  100 60.8M  0      0      18.2M      0      0:00:03  0:00:03  30.9M
root@ip-172-31-0-132:~# docker --version
docker version 27.3.1, build ce12230
root@ip-172-31-0-132:~# chmod +x $DOCKER_CONFIG/cli-plugins/docker-compose
root@ip-172-31-0-132:~# docker compose version
Docker Compose version v2.29.6
root@ip-172-31-0-132:~# |
```

Fig. 3.02: Docker compose installation

## Step 2: Create a docker-compose.yml file:

To run multiple container services simultaneously with Docker Compose, create a .yaml file that defines all the services. Use the following command:

` vi docker-compose.yaml`

In the file, specify all the necessary services, then save and exit



```
version: '3.8'
services:
  webserver1:
    image: "abhishek2411/tomcat_web_img:8.1"
    ports:
      - 8090:8080
  dbserver1:
    image: "mysql:oracle"
```

*Fig. 3.03: Contents of docker-compose.yml file*

The screenshot shows a terminal window with a black background and white text. It displays the following command and its output:

```
root@ip-172-31-0-132:~# docker images
REPOSITORY          TAG      IMAGE ID      CREATED     SIZE
phiz2411/tomcat_web_img  8.1      8c3c3fef0c8d  About an hour ago  402MB
ello-world          latest   d2c94e258dc9  17 months ago  13.3kB
root@ip-172-31-0-132:~# vi docker-compose.yml
root@ip-172-31-0-132:~# vi docker-compose.yml
root@ip-172-31-0-132:~# ls
docker-compose.yml  snap
root@ip-172-31-0-132:~# |
```

Two red arrows point from the right margin towards the terminal window. One arrow points to the line "13.3kB" in the Docker image list, and another arrow points to the line "vi docker-compose.yml" in the command history.

Fig. 3.04: *docker-compose.yml* file creation

### Step 3: Compose up the container:

To start the container services defined in the docker-compose.yaml file, use the following command:

'**docker compose up -d**'

This will initiate the containers and run them in detached mode.

```
root@ip-172-31-0-132:~# docker images
REPOSITORY          TAG      IMAGE ID      CREATED        SIZE
phiz2411/tomcat_web_img    8.1      8c3c3fef0c8d  About an hour ago   402MB
phiz2411/tomcat_web_img    latest   d2c94e258dbc  17 months ago  13.3kB

root@ip-172-31-0-132:~# vi docker-compose.yml
root@ip-172-31-0-132:~# docker-compose.yml
root@ip-172-31-0-132:~# ls
root@ip-172-31-0-132:~# docker images
REPOSITORY          TAG      IMAGE ID      CREATED        SIZE
phiz2411/tomcat_web_img    8.1      8c3c3fef0c8d  2 hours ago   402MB
phiz2411/tomcat_web_img    latest   d2c94e258dbc  17 months ago  13.3kB

root@ip-172-31-0-132:~# docker-compose up -d
command 'docker-compose' not found, but can be installed with:
apt install docker-compose # version 24.0.5, or
apt install docker-compose # version 1.29.2-1
See 'snap info docker' for additional versions.
root@ip-172-31-0-132:~# docker compose up -d
[!] [0000] /root/docker-compose.yml: the attribute 'version' is obsolete, it will be ignored, please remove it to avoid potential confusion
[+] Running 111
  ↗ dbserver1 Pulled
    ✓ ab9a3c6198b7 Pull complete
    ✓ 97fc8c33abe Pull complete
    ✓ aa23d877fa04 Pull complete
    ✓ a143609ddd2d Pull complete
    ✓ 78308a3437c4 Pull complete
    ✓ c0880e4b3737 Pull complete
    ✓ 4bab2679ce1 Pull complete
    ✓ e57fffd9d974 Pull complete
    ✓ 607f40000553 Pull complete
    ✓ cd65ca5a5fee Pull complete
[+] Running 3/3
  ↗ Network root_default Created
  ↗ Container root-webserver1-1 Started
  ↗ Container root-dbserver1-1 Started
root@ip-172-31-0-132:~# docker images
REPOSITORY          TAG      IMAGE ID      CREATED        SIZE
phiz2411/tomcat_web_img    8.1      8c3c3fef0c8d  2 hours ago   402MB
phiz2411/tomcat_web_img    latest   d2c94e258dbc  17 months ago  13.3kB
phiz2411/oracle           latest   be960704dfac  5 days ago   602MB

root@ip-172-31-0-132:~# docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS               NAMES
b9de8e62bea        abhiz2411/tomcat_web_img:8.1   "catalina.sh run"   About a minute ago   Up About a minute   0.0.0.0:8090->8080/tcp, [::]:8090->8080/tcp   root-webserver1-1
```

Fig. 3.05: *docker compose up command*

## Step 4: docker compose down:

The containers are now running in detached mode. To stop and remove the services, execute the following command:

' docker compose down'

This command will halt the running containers and remove them from the containers list.

The screenshot shows a terminal session on a Linux system (Ubuntu 12.04 LTS). The user runs 'docker images' to list available images, then 'docker ps' to list running containers. A red arrow points to the 'root-webserver1-1' container in the ps output. The user then runs 'docker compose down' (also highlighted by a red arrow), which stops the container and removes it from the list. Red double-headed arrows highlight the removal of the container from the ps output. Finally, 'docker ps -a' is run to show all containers, including the removed one, which is also highlighted by a red arrow.

```
root@ip-172-31-0-132:~# docker images
REPOSITORY          TAG      IMAGE ID      CREATED       SIZE
phiz2411/tomcat_web_img    8.1      8c3c3fef0c8d  About an hour ago  402MB
ello-world          latest   d2c94e258dbc  17 months ago  13.3kB
root@ip-172-31-0-132:~# vi docker-compose.yml
root@ip-172-31-0-132:~# vi docker-compose.yml
root@ip-172-31-0-132:~# ls
docker-compose.yml  snap
root@ip-172-31-0-132:~# docker images
REPOSITORY          TAG      IMAGE ID      CREATED       SIZE
phiz2411/tomcat_web_img    8.1      8c3c3fef0c8d  2 hours ago  402MB
ello-world          latest   d2c94e258dbc  17 months ago  13.3kB
root@ip-172-31-0-132:~# docker compose up
error: 'docker-compose' not found. You can be installed with:
apt install docker     # version 24.0.5, or
apt install docker-compose # version 1.29.2-1
see 'snap info docker' for additional versions.
root@ip-172-31-0-132:~# docker compose up -d
[!] [0000] /root/docker-compose.yml: the attribute 'version' is obsolete, it will be ignored, please remove it to avoid potential confusion
[j] Running 1/1
[+] Running 1/1
  ✓ dbserver1-1 Pulled
  ✓ dbserver1-1 Pull complete
  ✓ 97fc8c23abe Pull complete
  ✓ aa23db77fa04 Pull complete
  ✓ a143609dd2d Pull complete
  ✓ 78308a3437c4 Pull complete
  ✓ c0880e4b3737 Pull complete
  ✓ 4bab267f9c1e Pull complete
  ✓ e575f6d9b17a Pull complete
  ✓ 40f78cc01553 Pull complete
  ✓ cd66a5a3febe Pull complete
[j] Running 0/3
[+] Network root_default Created
[+] Container root-webserver1-1 Started
[+] Container root-dbserver1-1 Started
root@ip-172-31-0-132:~# docker images
REPOSITORY          TAG      IMAGE ID      CREATED       SIZE
phiz2411/tomcat_web_img    8.1      8c3c3fef0c8d  2 hours ago  402MB
phiz2411/tomcat_web_img    oracle   be960704dfac  5 days ago  602MB
ello-world          latest   d2c94e258dbc  17 months ago  13.3kB
root@ip-172-31-0-132:~# docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS               NAMES
5d6de862bea        phiz2411/tomcat_web_img:8.1   "sh /run.sh run"   About a minute ago   Up About a minute   0.0.0.0:8090->8080/tcp, :::8080->8080/tcp   root-webserver1-1
root@ip-172-31-0-132:~# docker compose down
[!] [0000] /root/docker-compose.yml: the attribute 'version' is obsolete, it will be ignored, please remove it to avoid potential confusion
[j] Running 3/2
[+] Container root-webserver1-1 Removed
[+] Container root-dbserver1-1 Removed
[+] Network root_default Removed
root@ip-172-31-0-132:~# docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS               NAMES
root@ip-172-31-0-132:~# docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS               NAMES
root@ip-172-31-0-132:~# docker images
REPOSITORY          TAG      IMAGE ID      CREATED       SIZE
phiz2411/tomcat_web_img    8.1      8c3c3fef0c8d  2 hours ago  402MB
ello-world          latest   d2c94e258dbc  17 months ago  13.3kB
root@ip-172-31-0-132:~|
```

Fig. 3.06: docker compose down