**EXPERIMENT NO: 9**

**DEVOPS**

**AIM:** To learn dockerfile instructions, build an image for a sample web application using dockerfile.

**LO:5 -** Explain concept of containerization and Analyze the Containerization of OS images and deployment of applications over Docker

**THEORY:**

**Dockerfile:**

* A Dockerfile is a text document that contains commands that are used to assemble an image. We can use any command that call on the command line. Docker builds images automatically by reading the instructions from the Dockerfile.
* The docker build command is used to build an image from the Dockerfile. You can use the -f flag with docker build to point to a Dockerfile anywhere in your file system.

The instructions are not case-sensitive but you must follow conventions which recommend to use uppercase.

* Docker runs instructions of Dockerfile in top to bottom order. The first instruction must be **FROM** in order to specify the Base Image.
* A statement begin with # treated as a comment. You can use RUN, CMD, FROM, EXPOSE, ENV etc instructions in your Dockerfile.
* Here, we are listing some commonly used instructions.

FROM

* This instruction is used to set the Base Image for the subsequent instructions. A valid Dockerfile must have FROM as its first instruction.
* Ex. FROM ubuntu

LABEL

* We can add labels to an image to organize images of our project. We need to use LABEL instruction to set label for the image.
* Ex. LABEL vendorl = "JavaTpoint"

RUN

* This instruction is used to execute any command of the current image.
* Ex. RUN /bin/bash -c 'source $HOME/.bashrc; echo $HOME'

CMD

* This is used to execute application by the image. We should use CMD always in the following form CMD ["executable", "param1", "param2"?]
* This is preferred way to use CMD. There can be only one CMD in a Dockerfile. If we use more than one CMD, only last one will execute.

COPY

* This instruction is used to copy new files or directories from source to the filesystem of the container at the destination.
* Ex. COPY abc/ /xyz

**Rules**

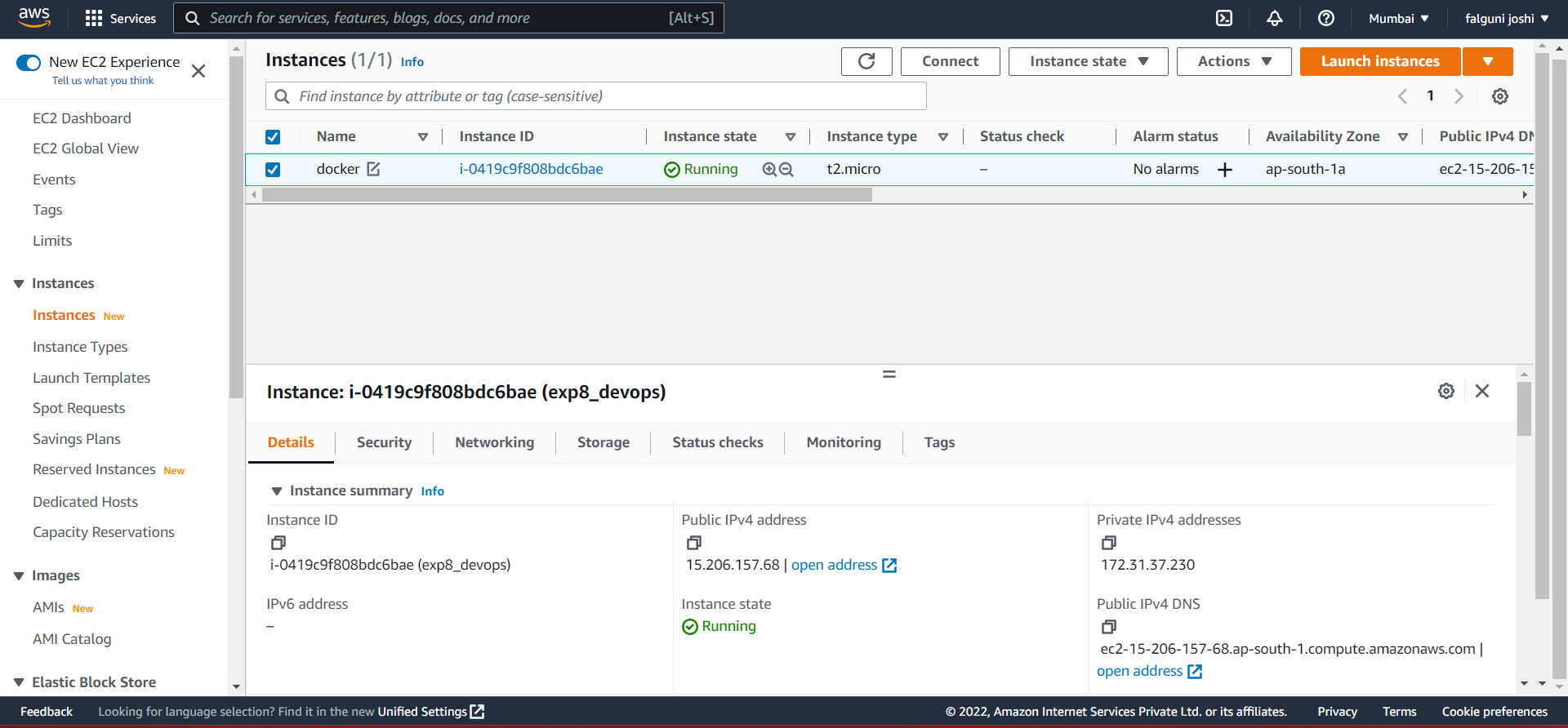
* The source path must be inside the context of the build. We cannot COPY ../something /something because the first step of a docker build is to send the context directory (and subdirectories) to the docker daemon.
* If source is a directory, the entire contents of the directory are copied including filesystem metadata.

WORKDIR

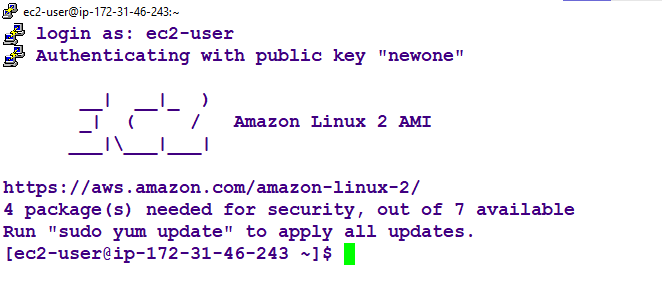
* The WORKDIR is used to set the working directory for any RUN, CMD and COPY instruction that follows it in the Dockerfile. If work directory does not exist, it will be created by default.
* We can use WORKDIR multiple times in a Dockerfile.
* Ex. WORKDIR /var/www/html

**OUTPUT**

Step 1: login into your AWS account and launch instance.



and then login this instance through puTTy.

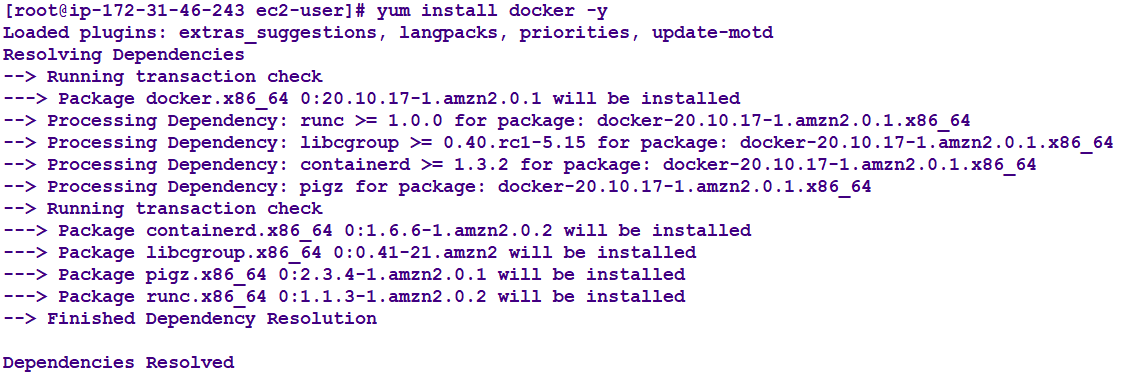


C

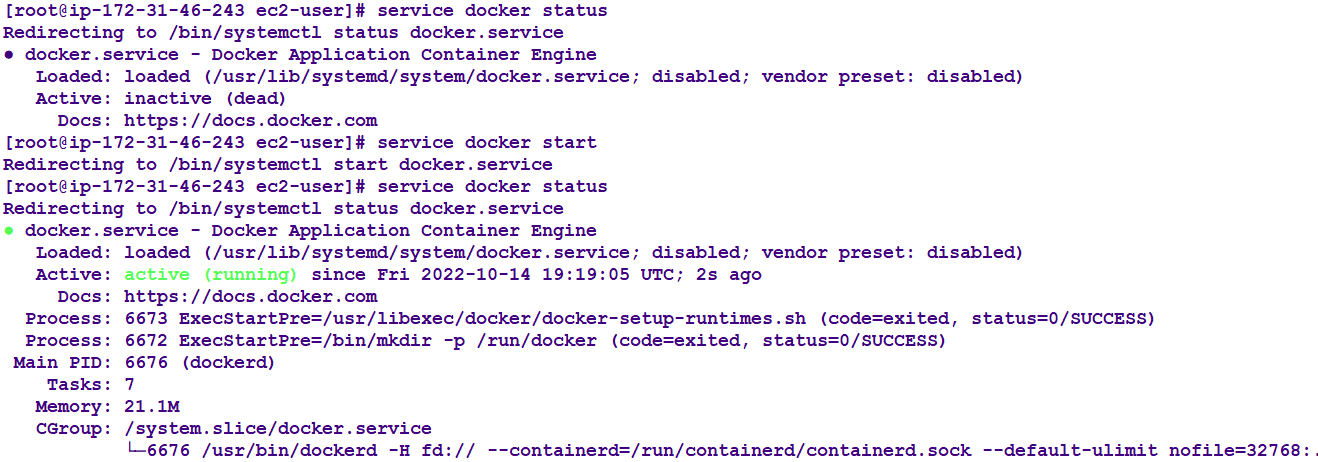
ome to the root and update your machine.

* sudo su
* yum update -y

Step 2: Install docker.



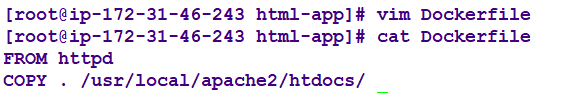
Start the docker services.



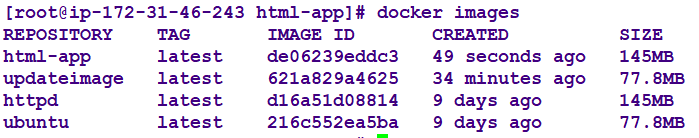
Step 3: making a directory which contains code related to our application.



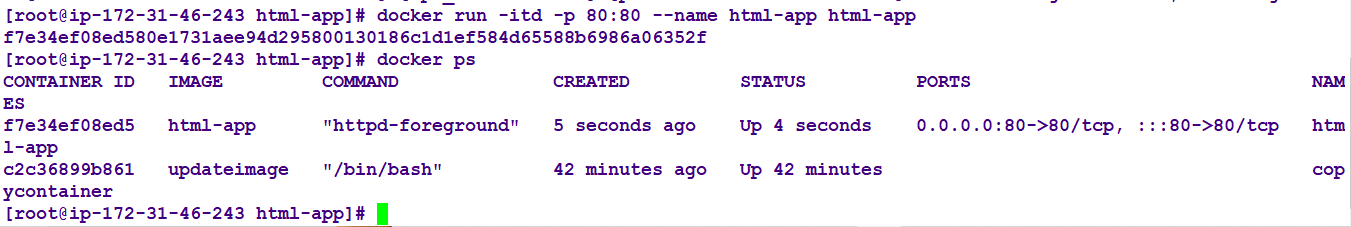
Step 4: create and configure docker file.



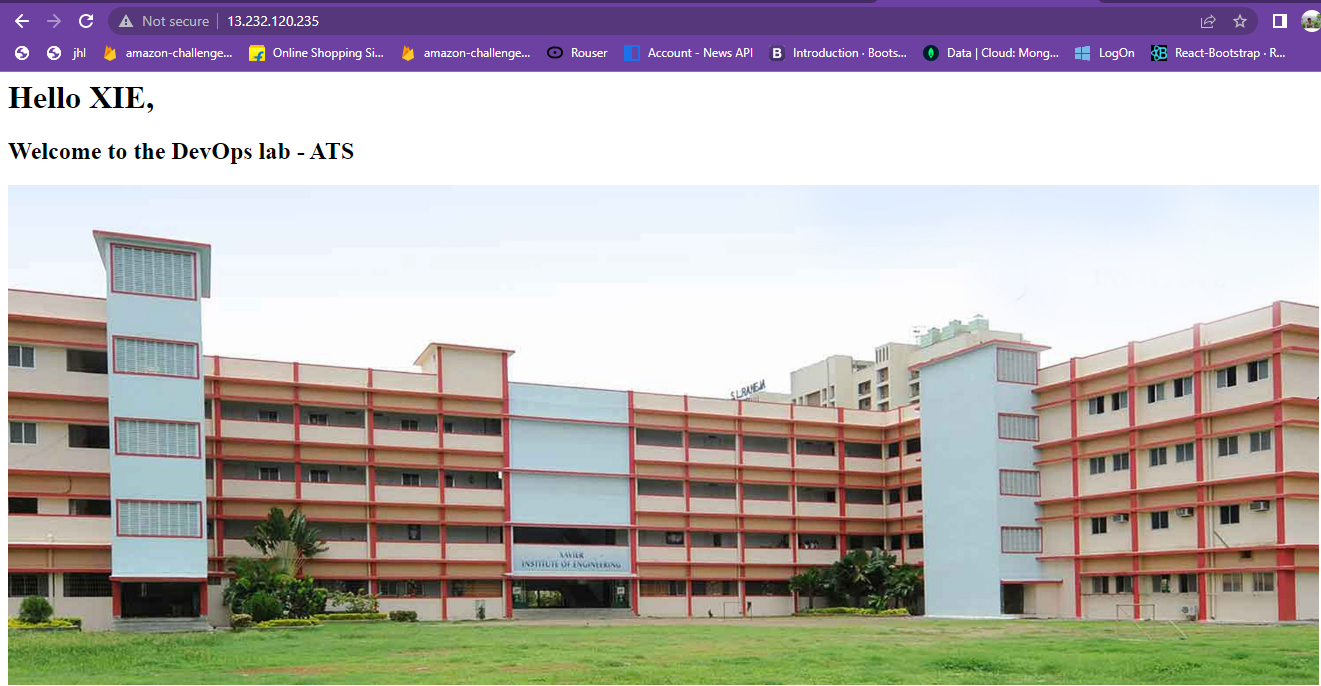
Step 5: Build an Images for the docker file.



Step 6: Run the image.



Step 7: Copy public IP of your instance and paste in the new tab of browser.



**CONCLUSION:**