**120A3051**

**Shreya Idate**

**Batch: E3**

# Experiment No: 8

**AIM**: To understand Docker Architecture and Container Life Cycle, install Docker and execute docker commands to manage images and interact with containers.

**THEORY**: Docker Engine is an open source containerization technology for building and containerizing your applications. Docker Engine acts as a client-server application with:

* A server with a long-running daemon process dockerd.
* APIs which specify interfaces that programs can use to talk to and instruct the Docker daemon.
* A command line interface (CLI) client docker.

The CLI uses Docker APIs to control or interact with the Docker daemon through scripting or direct CLI commands. Many other Docker applications use the underlying API and CLI. The daemon creates and manage Docker objects, such as images, containers, networks, and volumes..

**Installation of Docker:**

To get started with Docker Engine on Ubuntu, make sure you meet the prerequisites, and then install Docker.

## Prerequisites: OS requirements

To install Docker Engine, you need the 64-bit version of one of these Ubuntu versions:

* + Ubuntu Hirsute 21.04
  + Ubuntu Focal 20.04 (LTS)
  + Ubuntu Bionic 18.04 (LTS)

**Installation methods: You** can install Docker Engine in different ways, depending on your needs:

1. Most users set up Docker’s repositories and install from them
2. Some users download the DEB package and install it manually and manage upgrades completely manually.
3. In testing and development environments, some users choose to use automated convenience scripts to install Docker

**Install using the convenience script:** Docker provides a convenience script at get.docker.com to install Docker into development environments quickly and non-interactively. This example downloads the script from [get.docker.com](https://get.docker.com/) and runs it to install the latest stable release of Docker on Linux:

$ curl -fsSL https://get.docker.com -o get-docker.sh

$ sudo sh get-docker.sh

### To get OS detail and version

it77@it77-OptiPlex-3050 :~$ lsb\_release -a

## Uninstall old versions

it77@it77-OptiPlex-3050 :~$ sudo su

t77@it77-OptiPlex-305i0 :~$ sudo apt-get remove docker docker-engine docker.io containerd runc it77@it77-OptiPlex-3050 :~$ sudo apt install curl

root@it77-OptiPlex-3050:/home/it77# curl -fsSL https://get.docker.com -o get-docker.sh

### Examine scripts downloaded from the internet



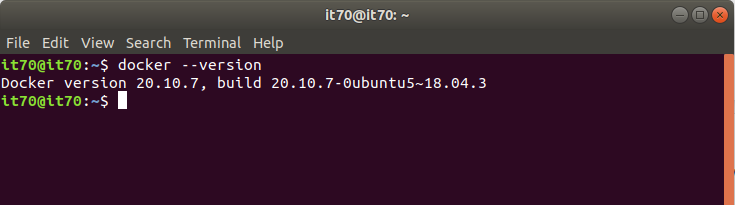
root@it77-OptiPlex-3050:/home/it77# ls

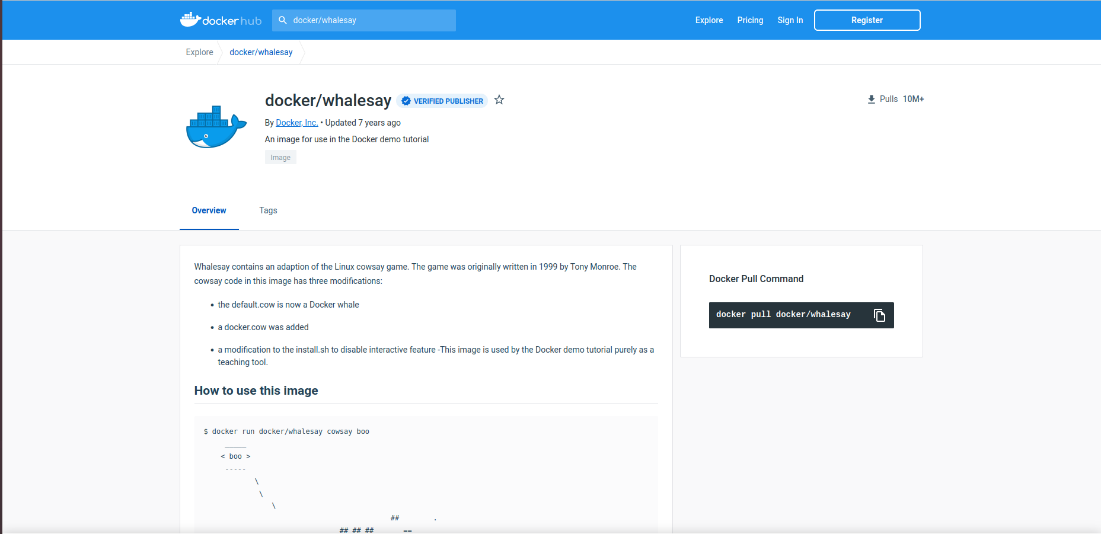
root@it77-OptiPlex-3050:/home/it77# sudo sh get-docker.sh

## Basic Docker Commands:

### Check the version of Docker installed

root@it77-OptiPlex-3050:/home/it77# docker –version

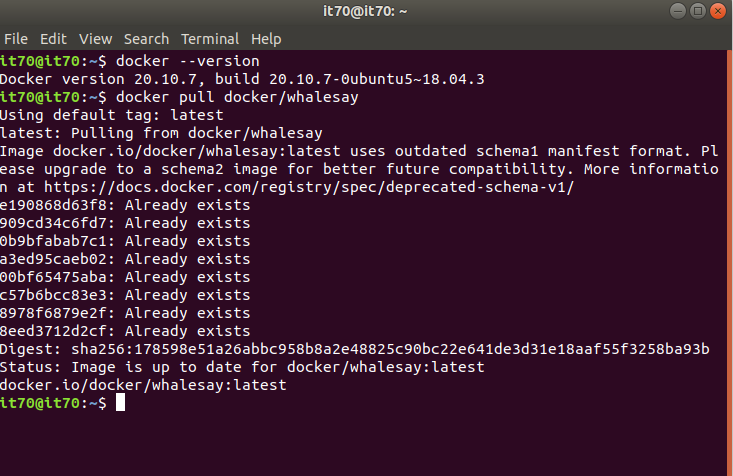




**Running existing Docker images: Go to Docker public repository at** [**https://hub.docker.com**](https://hub.docker.com/) to get the **official images available for testing purpose**

**Run docker image**

root@it77-OptiPlex-3050:/home/it77# docker run docker/whalesay cowsay hello\_you root@it77-OptiPlex-3050:/home/it77# docker run docker/whalesay cowsay hello\_me





**Check all pulled images**

root@it77-OptiPlex-3050:/home/it77# docker images

### Pull the sample images

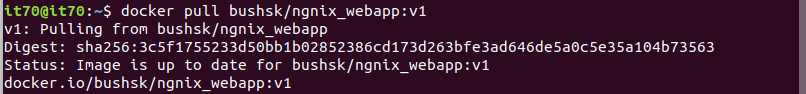
root@it77-OptiPlex-3050:/home/it77# sudo docker pull postgres root@it77-OptiPlex-3050:/home/it77# docker images

### Check all running container

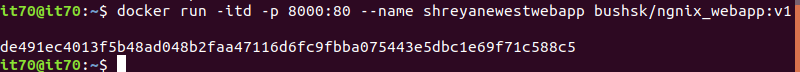
root@it77-OptiPlex-3050:/home/it77# docker ps // **note the container id**

root@it77-OptiPlex-3050:/home/it77# docker ps -a //**previously ran containers**

Pulling the created webapp project:



**Running the pulled image**





### Running or testing webapp on port 8989:

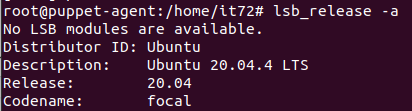
### 

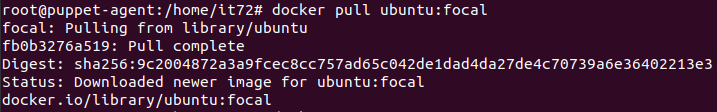
### 

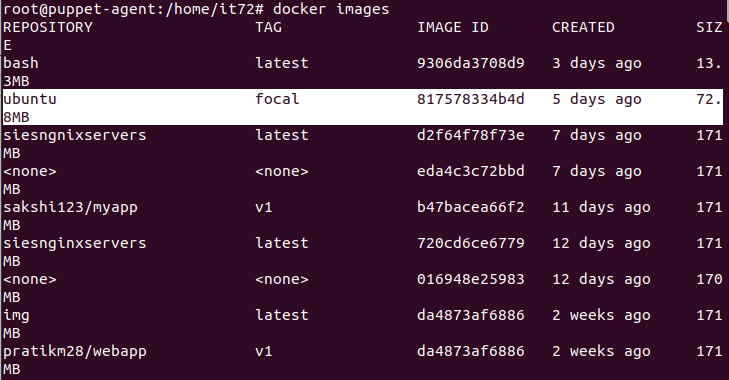
**Creating a new image from the running containers:**

### Pull the Ubuntu as a base image :

root@it77-OptiPlex-3050:/home/it77# docker pull ubuntu:latest root@it77-OptiPlex-3050:/home/it77# docker images







### Run the Ubuntu image with a command in a container: Getting a bash in Ubuntu

### root@it77-OptiPlex-3050:/home/it77# docker run -it ubuntu:latest bash

### root@it77-OptiPlex-3050:/home/it77# docker ps

### 

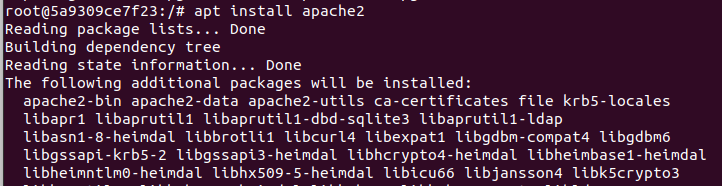
Note the <12 digit hash value> is the id of the shell.

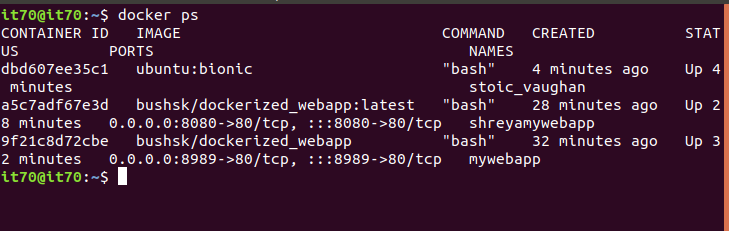
**Create an Apache Server and host index.html in the Containers**

root@67e9bd16d77b:/# apt update

root@67e9bd16d77b:/# apt install apache2

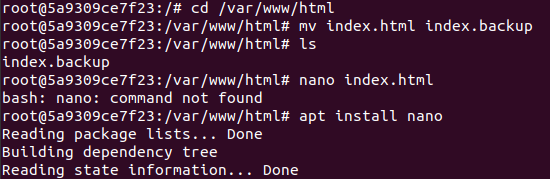
root@67e9bd16d77b:/# cd

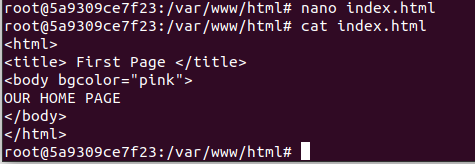




root@67e9bd16d77b:/var/www/html# nano index.html

root@67e9bd16d77b:/var/www/html# cat index.html





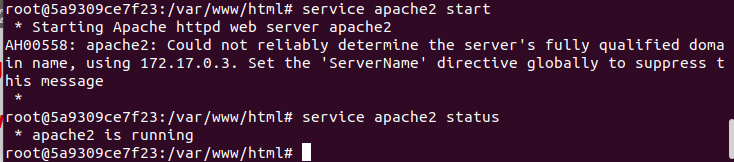
### <title> First page</title>

**<body bgcolor="pink"> Our home Page**

### </body>

**</html>**

root@67e9bd16d77b:/var/www/html# service apache2 start root@67e9bd16d77b:/var/www/html# service apache2 status

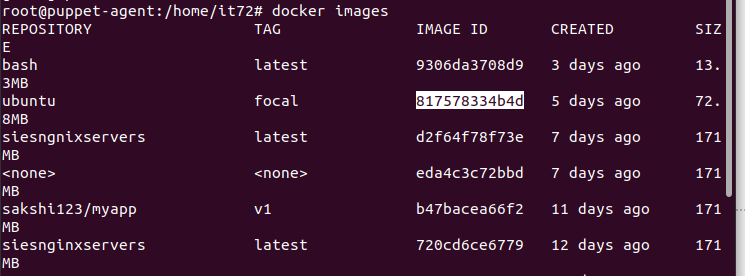


### Committing an image:

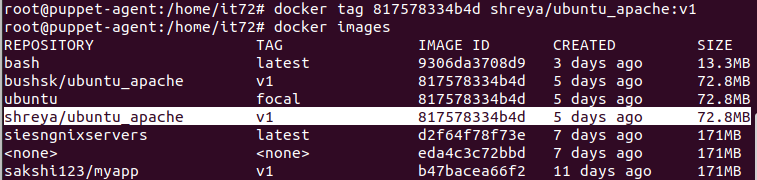
**Open a new terminal and run the foll. Commands:**

### Tagging <Ubuntu\_apache:v1> image using image id:

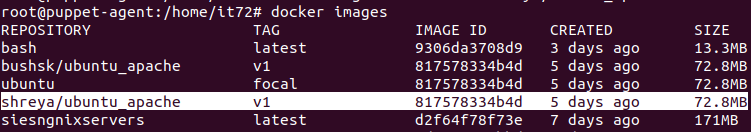
root@it77-OptiPlex-3050:/home/it77# docker tag 27941809078c bushsk/ubuntu\_apache:v1 27941809078c is the image id of the running container



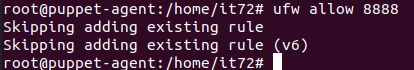
### Checking its size and committing an image



**Checking the size of committed image**



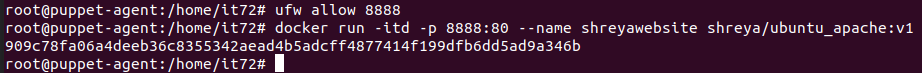
### Allowing port 8888

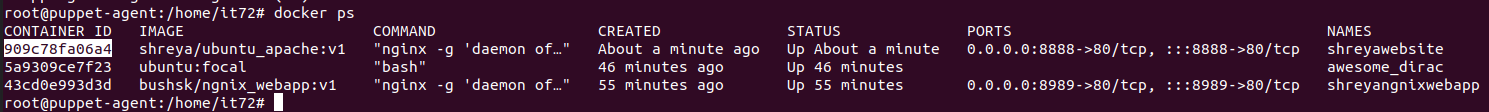


**Running a named <mywebsite> image Ubuntu\_apache:v1 image and note container id**



### Restarting an apache by attaching it into running containers using container id:



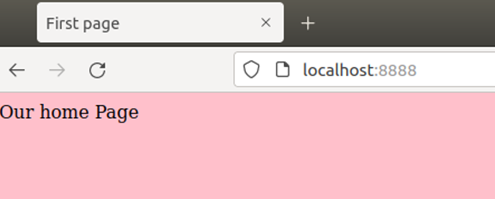


**Get the IP address of your system**

root@it77-OptiPlex-3050:/home/it77# ifconfig

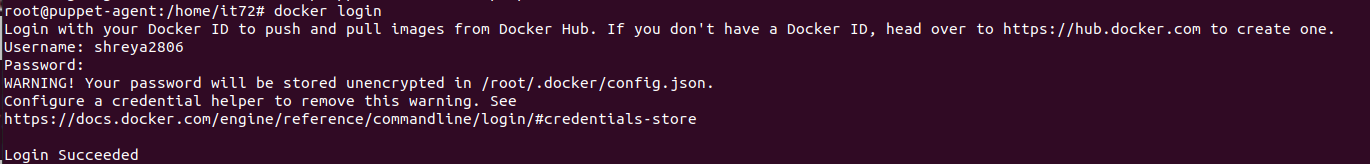
**Test your web application:**

**Open a browser and put <IP: port number> (as 8888) or type localhost:8888**

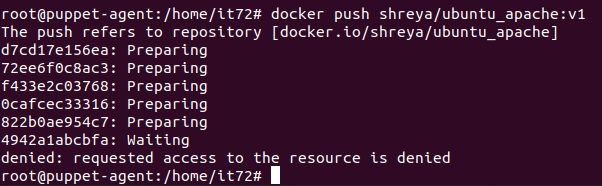


**Login in hub**

root@it77-OptiPlex-3050:/home/it77# docker login



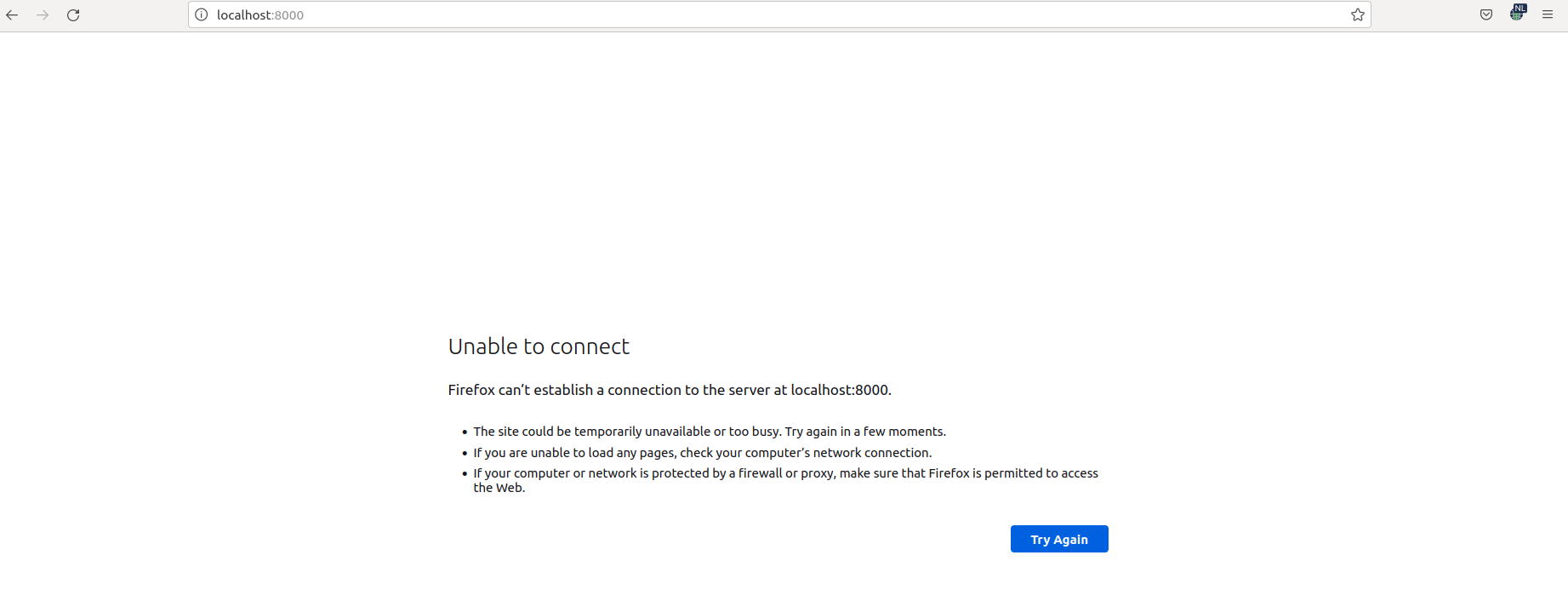
**Pushing the image on Docker Hub**



**Stopping and removing containers using container id**

root@it77-OptiPlex-3050:/home/it77# docker stop bd9fdf66daaf





root@it77-OptiPlex-3050:/home/it77# docker kill bd9fdf66daaf -- > not advisable root@it77-OptiPlex-3050:/home/it77# docker rm bd9fdf66daaf



root@it77-OptiPlex-3050:/home/it77# docker ps -a **Remove multiple containers using container ids:** root@it77-OptiPlex-3050:/home/it77# docker rm bd9 abc

root@it77-OptiPlex-3050:/home/it77# docker rm $(docker ps –aq) root@it77-OptiPlex-3050:/home/it77# docker ps -a

### Deleting the images

root@it77-OptiPlex-3050:/home/it77# docker images root@it77-OptiPlex-3050:/home/it77# docker rmi 5c6

### Creating an image using docker file script: DOCKER FILE

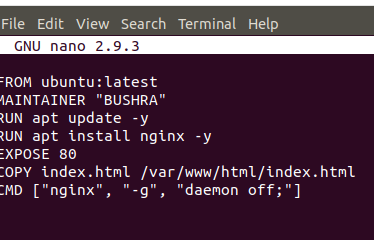


root@it77-OptiPlex-3050:/home/it77# mkdir dockertutorial root@it77-OptiPlex-3050:/home/it77# cd dockertutorial root@it77-OptiPlex-3050:/home/it77/dockertutorial# pwd

/home/it77/dockertutorial

root@it77-OptiPlex-3050:/home/it77/dockertutorial# nano index.html root@it77-OptiPlex-3050:/home/it77/dockertutorial# ls

index.html



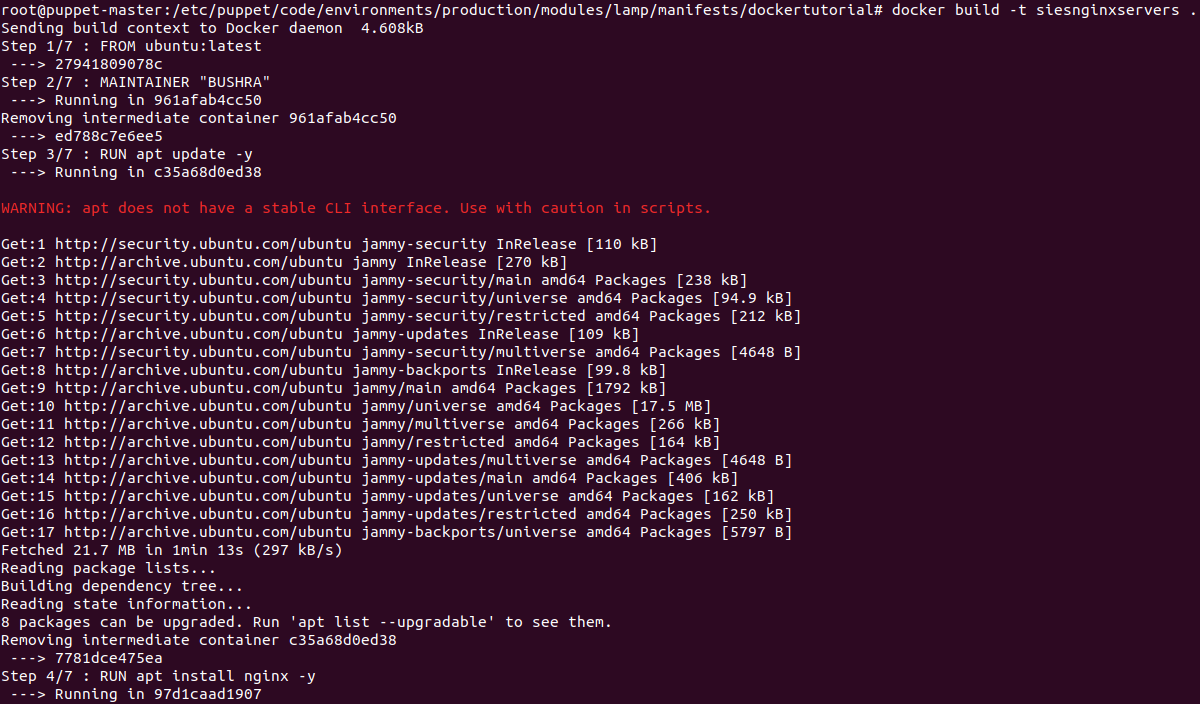
root@it77-OptiPlex-3050:/home/it77/dockertutorial# nano Dockerfile

**FROM ubuntu:latest MAINTAINER "BUSHRA"**

**RUN apt update -y RUN apt install nginx -y EXPOSE 80**

**COPY index.html /var/www/html/index.html CMD ["nginx", "-g", "daemon off;"]**

root@it77-OptiPlex-3050:/home/it77/dockertutorial# docker build -t siesnginxservers .

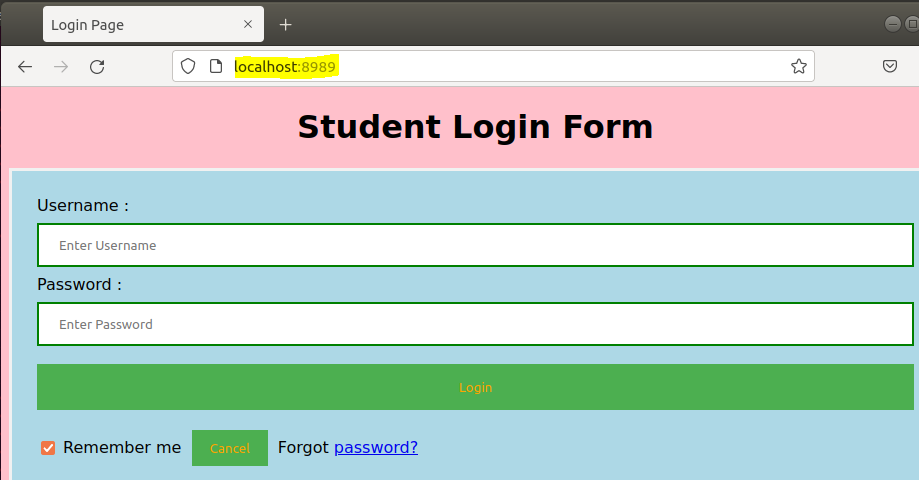


root@it77-OptiPlex-3050:/home/it77/dockertutorial# docker run -itd -p 8989:80 siesnginxservers



root@it77-OptiPlex-3050:/home/it77/dockertutorial# ufw allow 8989 //if required since already allowed

**Open browser and put ip with port number as 8989**



**Conclusion**: We are able to understand the concept of docker and have successfully pushed the repository on

docker hub.