**LABSHEET 3**

**CREATING AND PLAYING WITH IMAGES USING DOCKERFILE**

1. Write a simple prompt application that takes the name as user input and leaves a welcome message with the name. Also create a dockerfile to tun the same in nginx

**Solution**

a. Create an HTML file with the following content

<!DOCTYPE html>

<html>

<head>

<title>Welcome Page</title>

</head>

<body>

<h1>Welcome Page</h1>

<form>

<label for="name">Enter your name:</label>

<input type="text" id="name" name="name" required>

<button type="button" onclick="displayWelcome()">Submit</button>

</form>

<p id="welcomeMessage"></p>

<script>

function displayWelcome() {

var name = document.getElementById("name").value;

var welcomeMessage = document.getElementById("welcomeMessage");

welcomeMessage.innerHTML = "Welcome, " + name + "!";

}

</script>

</body>

</html>

b. Create a dockerfile to serve this html file using NGINX server

# Dockerfile

# Use the official NGINX image as a parent image

FROM nginx

# Copy the HTML file into the container

COPY index.html /usr/share/nginx/html/index.html

# Expose port 80

EXPOSE 80

c. Build the docker image for your HTML application

docker build -t namratasdocker/html-app .

1. Run a docker container for the image

docker run -d -p 8086:80 namratasdocker/html-app

1. Push image to docker hub repository

docker push namratasdocker/html-app

2. Create a dockerfile that installs Ubuntu and displays a message immediately it runs.

**Solution**

FROM ubuntu:jammy-20230916

LABEL maintainer="namrata <dasnamrata795@gmail.com>"

LABEL description="This example Dockerfile installs ubuntu and displays a message."

RUN apt-get update

CMD ["echo", "Hello World...! from my first docker image"]

**Execute the following commands in your docker cli**

docker build –t myubuntuapp . // to build the Image

docker run - -name myubuntu myapp2 //to run a container for the image.

docker tag myubuntuapp namratasdocker/repository // *// tag your docker image with your dockerhub repository name. This means associating your image with your docker hub repo and when you push it to the docker hub, it will be stored in that repo under the name ‘namratasdocker/myapp’*

docker login //authenticate using credentials

docker push namratasdocker/repository //push image to dockerhub

3. Create a simple dockerfile that uses Ubuntu as the base image and installs apache v2

**Solution**

FROM ubuntu

RUN apt-get update

RUN apt-get -y install apache2

ADD . /var/www/html

#CMD apachectl -D FOREGROUND

ENTRYPOINT apachectl -D FOREGROUND //to start apache v2, apachectl is a frontend HTTP server

ENV name DevOps

//*creating the image, running container and pushing it to dockerhub*

docker build -t mydockerfile:latest .

docker run - -name container-c1 -d -p 8086:80 mydockerfile

docker tag mydockerfile namratasdocker/mydockerfile

docker login

docker push namratasdocker/mydockerfile:latest

4. Create a Dockerfile for an image that executes "script.sh" script, and displays a "Hello World...! from my first docker image" message in the container's standard output.

**Solution**

* 1. Create a script.sh file

**#! /bin/bash** *// This is called a shebang line. It specifies the interpreter to be used for the script, which is "/bin/bash" in this case*

**echo "Hello World...! from my first docker image"**  *// the actual command that will be executed when the "script.sh" script is run. It prints the "Hello World...! from my first docker image" message to the standard output of the container.*

Top of Form

* 1. Create a Dockerfile to build the above application

*#The base image you want to use and customize*

**FROM Ubuntu** *// This line specifies the base image for your Docker image. In this case, you are using the "ubuntu" base image, which provides the Ubuntu Linux operating system.*

*#Set the working directory inside container*

**WORKDIR** /app *// sets the working directory inside the container to "/app." All subsequent commands will be executed in this directory.*

*#Copy a script file to the container*

**COPY script.sh .** *// copies the "script.sh" file from the host system into the "/app" directory of the container. The "." refers to the current directory inside the container.*

*#Make the script executable*

**RUN chmod +x script.*sh*** *// makes the "script.sh" file executable within the container by using the "chmod" command.*

**CMD [ "./script.sh"]** *// specifies the command that should be executed when the container starts. It tells Docker to run the "script.sh" script.*

5. Create a static index.html file in a directory of your choice, to display any message of your choice

1. Create a dockerfile in the same directory
2. Use nginx as the base image
3. Copy the index.html file inside the container image
4. Create the image and push it to docker hub
5. Pull it into your docker server, create and run a container

6. Create a Dockerfile to build an image for a Python application that adds two numbers and displays the result