Assignment

**Question 1)** Check if docker and docker-compose is installed on the system. If not present, install the missing packages.

**Installing docker and:**

* Launching a ec2-instance & login into server
* yum install docker -y
* using above command installed docker
* usermod -aG docker ec2-user – docker added into ec2-user group

**installing docker compose:**

* DOCKER\_CONFIG=${DOCKER\_CONFIG:-$HOME/.docker}
* mkdir -p $DOCKER\_CONFIG/cli-plugins
* curl -SL https://github.com/docker/compose/releases/download/v2.18.1/docker-compose-linux-x86\_64 -o $DOCKER\_CONFIG/cli-plugins/docker-compose

**apply executable permissions**

* chmod +x $DOCKER\_CONFIG/cli-plugins/docker-compose
* docker compose version – to check docker compose available or not

**Question 6&7)** Add another subcommand to enable/disable/delete the site (stopping/starting the containers)

* created a project directory **mkdir my-lemp-stack**
* cd my-lemp-stack
* in this directory empty file created using **touch docker\_delete.sh** and given the execution permissions usinf chmod +x docker\_delete.sh
* **./docker\_delete.sh** using this command executed the script

6th and 7th question

#!/bin/bash

start\_site() {

echo "Starting containers for the site..."

# Replace the following command with the appropriate command to start your containers

docker compose up -d

}

stop\_site() {

echo "Stopping containers for the site..."

# Replace the following command with the appropriate command to stop your containers

docker compose down

}

delete\_site() {

echo "Deleting the site..."

stop\_site

# Replace the following command with the appropriate command to delete local files/directories

rm -rf /path/to/site

}

enable\_site() {

echo "Enabling the site..."

start\_site

}

disable\_site() {

echo "Disabling the site..."

stop\_site

}

# Check the provided subcommand and call the corresponding function

case "$1" in

enable)

enable\_site

;;

disable)

disable\_site

;;

delete)

delete\_site

;;

\*)

echo "Usage: $0 {enable|disable|delete}"

exit 1

esac

**question 3)** It must be a LEMP stack running inside containers (Docker) and a docker-compose file is a must.

* Created a project directory: mkdir my-lemp-stack
* Cd My-lemp-stack & create a docker-compose.yml
* In docker-compose.yml

version: '3'

services:

nginx:

image: nginx:latest

ports:

- '8080:80'

volumes:

- ./nginx.conf:/etc/nginx/nginx.conf

- ./html:/var/www/html

depends\_on:

- php

php:

image: php:latest

volumes:

- ./html:/var/www/html

mysql:

image: mysql:latest

restart: always

environment:

MYSQL\_ROOT\_PASSWORD: anand

volumes:

- ./mysql-data:/var/lib/mysql

**Installation of mariaDB**

wget https://r.mariadb.com/downloads/mariadb\_repo\_setup

echo "3a562a8861fc6362229314772c33c289d9096bafb0865ba4ea108847b78768d2 mariadb\_repo\_setup" | sha256sum -c -

chmod +x mariadb\_repo\_setup

sudo ./mariadb\_repo\_setup --mariadb-server-version="mariadb-10.3"

yum install MariaDB-server MariaDB-backup

**installing php**

yum install php -y

**touch nginx.conf Created empty file**

events {}

http {

server {

listen 80;

root /var/www/html;

index index.php index.html;

location / {

try\_files $uri $uri/ =404;

}

location ~ \.php$ {

try\_files $uri =404;

fastcgi\_pass php:9000;

fastcgi\_index index.php;

fastcgi\_param SCRIPT\_FILENAME $document\_root$fastcgi\_script\_name;

include fastcgi\_params;

}

}

}

**docker compose up -d**

**question 4)** Create a /etc/hosts entry for example.com pointing to localhost. Here we are assuming the user has provided example.com as the site name.

* Created a server and login into terminal
* Changed ec2-user home directory to etc directory
* In etc we have hosts file. Opened it using vi hosts
* In hosts file given my private Ip of server and url

Example 172.31.17.210 <https://www.example.com> and saved it and exit the file

* Ping 172.31.17.210 command used

**question 5)** Prompt the user to open *example.com* in a browser if all goes well and the site is up and healthy.

* IN etc folder created a file **touch open\_website.sh**
* Chmod +x open\_website.sh
* In this file
* **#!/bin/bash**
* **website="http://example.com"**
* **# Check if the website is reachable**
* **if curl --output /dev/null --silent --head --fail "$website"; then**
* **echo "The website $website is up and healthy."**
* **echo "Opening $website in a browser..."**
* **xdg-open "$website"**
* **else**
* **echo "The website $website is not reachable or experiencing issues."**
* **Fi**
* To prompt the user to open a website in a browser using linux, uses the xdg-open command
* After installing the xdg-utils plugin I can use xdg-open
* To install xdg-utils I use **yum install xdg-units**

**Question2)** The script should be able to create a WordPress site using the latest WordPress Version. Please provide a way for the user to provide the site name as a command-line argument.

* Installed php, httpd & mariadb
* **Yum install httpd php -y** this command used to install php and httpd
* Steps to Installing mariadb
* wget https://r.mariadb.com/downloads/mariadb\_repo\_setup
* echo "3a562a8861fc6362229314772c33c289d9096bafb0865ba4ea108847b78768d2 mariadb\_repo\_setup" | sha256sum -c -
* chmod +x mariadb\_repo\_setup
* sudo ./mariadb\_repo\_setup --mariadb-server-version="mariadb-10.3"
* yum install MariaDB-server MariaDB-client -y

**cd /etc/yum.repo.d/ in this path we can see tha mariadb repo**

* created a file **vi create\_wp\_site.sh**

**#!/bin/bash**

**# Check if the site name is provided as a command-line argument**

**if [ -z "$1" ]; then**

**echo "Error: Site name not provided. Usage: create\_wp\_site.sh <site-name>"**

**exit 1**

**fi**

**# Variables**

**site\_name="$1"**

**wordpress\_dir="/var/www/$site\_name"**

**db\_name="$site\_name"**

**db\_user="tom" # Replace with your desired database username**

**db\_password="tomcat" # Replace with your desired database password**

**# Download the latest WordPress version**

**latest\_version=$(curl -s https://wordpress.org/latest.tar.gz | tar -xzv -C /var/www/)**

**# Create a new MySQL database for the WordPress site**

**mysql -uroot -p -e "CREATE DATABASE ${anand};"**

**# Configure file permissions**

**chown -R cat:cat $wordpress\_dir**

**chmod -R 755 $wordpress\_dir**

**# Create a wp-config.php file**

**cp $wordpress\_dir/wp-config-sample.php $wordpress\_dir/wp-config.php**

**# Update database connection details in wp-config.php**

**sed -i "s/database\_name\_here/${anand}/" $wordpress\_dir/wp-config.php**

**sed -i "s/username\_here/${tom}/" $wordpress\_dir/wp-config.php**

**sed -i "s/password\_here/${tomcat}/" $wordpress\_dir/wp-config.php**

**echo "WordPress site '$site\_name' created successfully!"**

* **useradd cat ; passwd tom**
* created a user and given password to it
* ./**create\_wp\_site.sh**