**M2 Questions**

**Q. JENKINS-DOCKER**

**Question: Configure Jenkins to deploy a Docker container running Nginx to a local Docker host. The pipeline should build the Docker image and start the container.**

**On Docker terminal/cmd-Pull The image: docker pull jenkins/jenkins:lts**

****

**Create a container: docker run -d -p 8080:8080 -p 50000:50000 --name jenkins\_home:var/jenkins/jenkins\_home jenkins/jenkins:lts**

**A white background with black text

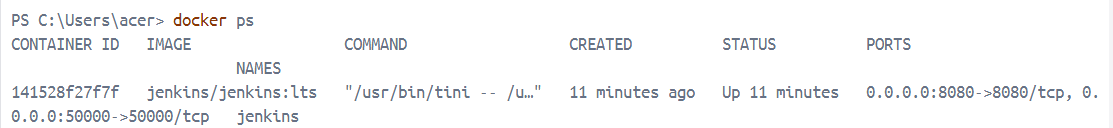
Description automatically generated**

**Check for images to verify: docker images**

**A close up of a number

Description automatically generated**

**Verify: docker ps**

****

**Go inside the container with root: docker exec -u root -it jenkins bash**

****

**Once you are in, run apt-get update**

**A close up of text

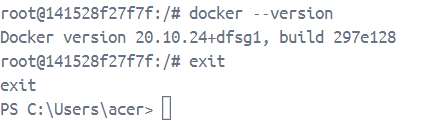
Description automatically generated**

**Install docker on your container: apt-get install -y docker .io**

**A screenshot of a computer

Description automatically generated**

**Verify installation: docker --version**

****

**Go to Jenkins->Manage Jenkins->Plug-ins->Docker commons->Install->Sign-in again->Check if installed successfully**

**A screenshot of a computer

Description automatically generated**

**Create a new folder:**

****

**Open it in VS code->**

**Dockerfile should be in Jenkins-Docker-Ngnix**

**Inside The Jenkins-Docker-Ngnix->Create a html folder-> Create your index.html inside the html folder:**

**A screenshot of a computer

Description automatically generated**

****

**Contents of Dockerfile:**

# Use the official Nginx image from the Docker Hub

FROM nginx:latest

# Copy custom Nginx configuration file (optional)

# COPY nginx.conf /etc/nginx/nginx.conf

# Copy HTML files or your web app into the container

# Make sure you have an 'index.html' or other web content to serve

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

# Expose port 80 for web traffic

EXPOSE 80

# Start Nginx in the foreground

CMD ["nginx", "-g", "daemon off;"]

**Contents of index.html:**

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<meta charset="UTF-8">**

**<meta name="viewport" content="width=device-width, initial-scale=1.0">**

**<title>Document</title>**

**</head>**

**<body>**

**<h1>Hello World</h1>**

**</body>**

**</html>**

**Push the project to git hub:**

**Commands:**

**git init**

**git add .**

**git status**

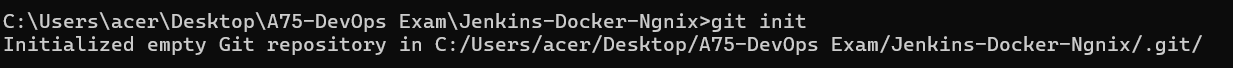
**git commit**

**git remote add origin <repo-url>**

**git remote -v**

**git branch**

**git push -u origin master**

****

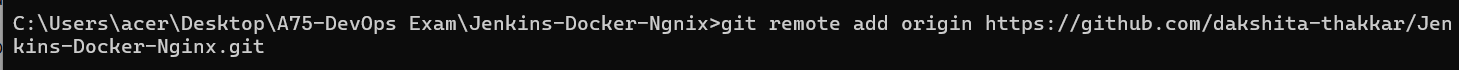
****

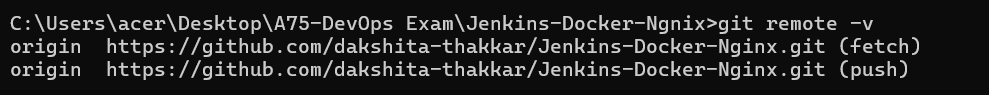
**A screen shot of a computer

Description automatically generated**

**A black screen with white text

Description automatically generated**

****

****

****

**A screen shot of a computer program

Description automatically generated**

**Pipeline:**

pipeline {

agent any

stages {

stage('Clone Repository') {

steps {

// Example: clone your repository that contains your Dockerfile

git 'https://github.com/dakshita-thakkar/Jenkins-Docker-Nginx.git'

}

}

stage('Build Docker Image') {

steps {

script {

// Build the Docker image using the Dockerfile in your repository

sh 'docker build -t my-nginx-image .'

}

}

}

stage('Run Docker Container') {

steps {

script {

// Stop and remove any previous containers

sh 'docker ps -aq --filter "name=my-nginx-container" | xargs -r docker stop | xargs -r docker rm'

// Run the new container

sh 'docker run -d --name my-nginx-container -p 8081:80 my-nginx-image'

}

}

}

}

}

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

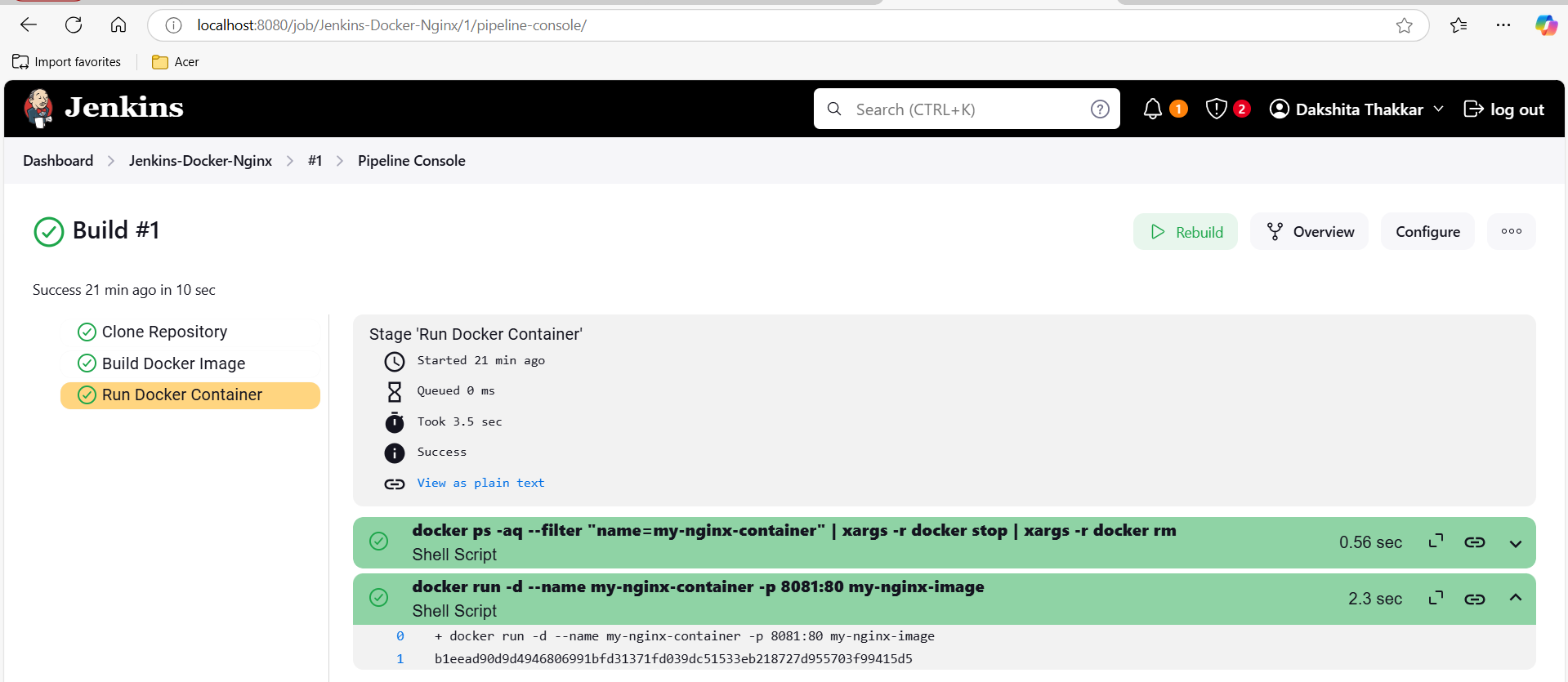
**A screenshot of a computer

Description automatically generated**

****

**A screenshot of a computer

Description automatically generated**

****

****

**A screenshot of a computer

Description automatically generated**

**Q. ANSIBLE-APACHE**

**Question:** **Write an Ansible playbook that installs Apache on a remote server and ensures the service is started.**

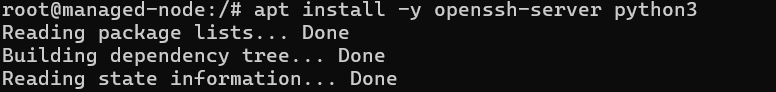
**A computer screen with white text

Description automatically generated**

****

**A screen shot of a computer

Description automatically generated**

****

****

**A black background with white text

Description automatically generated**

****

**A black screen with white text

Description automatically generated**

****

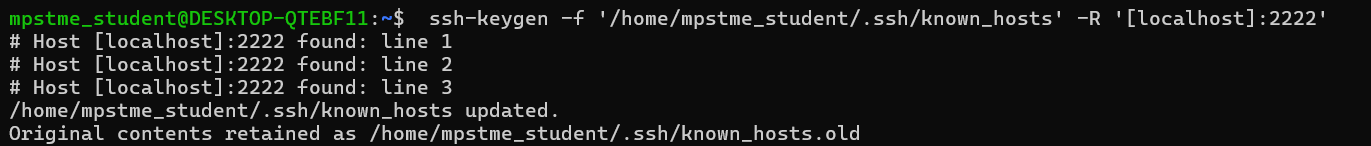
****

**A screenshot of a computer program

Description automatically generated**

**A screenshot of a computer error

Description automatically generated**

****

**A computer screen with white text

Description automatically generated**

****

**A black screen with white text

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**Run mkdir playbooks, to create a new directory to store your playbooks**

**Next run ls to verify**

****

****

****

**Playbook:**

---

- name: Install and ensure Apache is running

hosts: localhost

become: yes # Run tasks as sudo

tasks:

- name: Update the package repository

apt:

update\_cache: yes

- name: Install Apache

apt:

name: apache2

state: present

- name: Ensure Apache service is started and enabled

service:

name: apache2

state: started

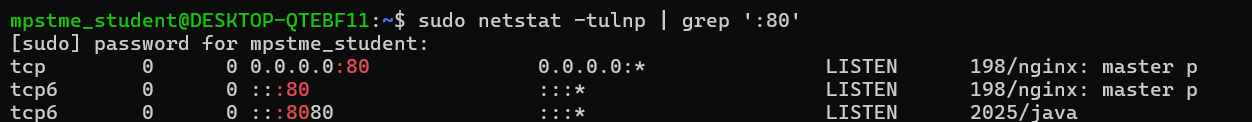
enabled: yes

**A black screen with white text

Description automatically generated**

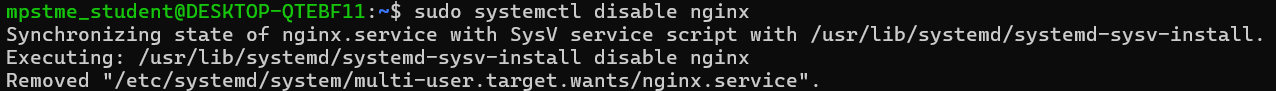
**A screen shot of a computer

Description automatically generated**

****

****

****

****

**A screenshot of a computer

Description automatically generated**

**A black screen with white text

Description automatically generated**

**Q. TERRAFORM-DOCKER**

**Question: Write a Terraform configuration to create a local Docker container running Nginx. The container should be created using the Docker provider in Terraform.**

**A black screen with white text

Description automatically generated**

**Go to downloads-> Create a new folder:**

**A screenshot of a computer

Description automatically generated**

**Open the folder in VS code and create a main.tf file:**

**A screenshot of a computer

Description automatically generated**

**Contents of main.tf file:**

# Specify the provider

terraform {

required\_providers {

docker = {

source = "kreuzwerker/docker"

version = "~> 3.0"

}

}

}

provider "docker" {

host = "npipe:////./pipe/docker\_engine"

}

# Create a Docker image for Nginx

resource "docker\_image" "nginx\_image" {

name = "nginx:latest"

keep\_locally = false

}

# Create a Docker container using the Nginx image

resource "docker\_container" "nginx\_container" {

name = "nginx\_container"

image = docker\_image.nginx\_image.name

ports {

internal = 80

external = 8080

}

}

# Output the container name and ID

output "container\_info" {

value = {

container\_name = docker\_container.nginx\_container.name

container\_id = docker\_container.nginx\_container.id

}

}

**A screenshot of a computer program

Description automatically generated**

**A screenshot of a computer program

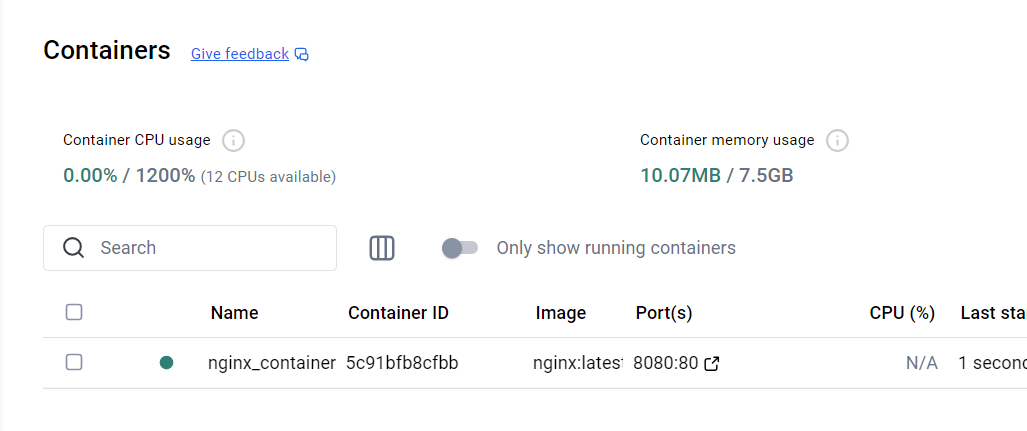
Description automatically generated**

**A screenshot of a computer program

Description automatically generated**

**A computer screen shot of text

Description automatically generated**

****

**STREAMLIT-GITHUB**

**Question: Integrate a basic Python project into SonarCloud and analyze its code quality. Configure the project to show issues like code smells, bugs, and security vulnerabilities.**

**GIT/GITHUB AND STREAMLIT**

1. **Installation**

**On Cmd: pip install streamlit**

**A screen shot of a computer code

Description automatically generated**

1. **Check if installed:**

**On Cmd: streamlit helloA screen shot of a computer

Description automatically generated**

1. **On Cmd you shld se:**

**A computer screen with white text

Description automatically generated**

1. **Streamlit Hello Page**

**A screenshot of a video game

Description automatically generated**

1. **Cmd to install venv: python -m venv myenv**

****

1. **Navigate to scripts: cd scripts**

****

1. **Check the directory**

**Command: dir**

**A computer screen with white text

Description automatically generated**

1. **Activate the streamlit environment:**

**Command: activate**

****

1. **Run: activate**

**A screenshot of a computer program

Description automatically generated**

1. **To verify, Run: streamlit hello**

A computer screen shot of a black screen

Description automatically generated

1. **Go to VS code, Open folder which contains myenv**

**A screenshot of a computer

Description automatically generated**

1. **Create a new file app.py in the folder**

**A screenshot of a computer program

Description automatically generated**

1. **Go to https://docs.streamlit.io/**

**A screenshot of a computer

Description automatically generated**

1. **Go to** [**https://docs.streamlit.io/get-started/tutorials/create-an-app**](https://docs.streamlit.io/get-started/tutorials/create-an-app)
2. **Create a title Page in Streamlit**

import streamlit as st

import pandas as pd

import numpy as np

st.title('Uber pickups in NYC')

A screenshot of a computer screen

Description automatically generated

1. **Go to cmd: streamlit run app.pyA screenshot of a computer

   Description automatically generated**
2. **Output**

**A screenshot of a computer

Description automatically generated**

1. **Using Magic Commands:**

import streamlit as st

import pandas as pd

st.write("Here's our first attempt at using data to create a table:")

st.write(pd.DataFrame({

    'first column': [1, 2, 3, 4],

    'second column': [10, 20, 30, 40]

}))

**A screenshot of a computer program

Description automatically generated**

1. **Refresh/Rerun**

**A screenshot of a computer

Description automatically generated**

1. **Creating Plots Using magic commands**

import streamlit as st

import numpy as np

import pandas as pd

chart\_data = pd.DataFrame(

     np.random.randn(20, 3),

     columns=['a', 'b', 'c'])

st.line\_chart(chart\_data)

**A screen shot of a computer program

Description automatically generated**

1. **Refresh/Rerun**

**A screenshot of a computer

Description automatically generated**

1. **Creating a Table Using Magic commands**

import streamlit as st

import numpy as np

import pandas as pd

dataframe = pd.DataFrame(

    np.random.randn(10, 20),

    columns=('col %d' % i for i in range(20)))

st.table(dataframe)

**A screen shot of a computer program

Description automatically generated**

1. **Refresh/Rerun**

**A screenshot of a computer screen

Description automatically generated**

1. **Making a Text Input:**

**A screen shot of a computer code

Description automatically generated**

1. **Rerun/Refresh**

**A black rectangular object with red line

Description automatically generated**

1. **Create a .evn file in your environment**

**A screenshot of a computer

Description automatically generated**

1. **On Cmd, run: pip install python-dotenv**

**A screen shot of a computer

Description automatically generatedPushing Your Code to GitHub:**

1. **Create a .gitignore file**

**A screenshot of a computer

Description automatically generated**

1. **Contents of .gitignore file:**

#.gitignore

# Ignore the virtual environment folder

myenv/

env/

# Ignore Python cache files

\_pycache\_/

\*.py[cod]

\*.pyo

# Ignore system files

.DS\_Store

Thumbs.db

# Ignore IDE/editor-specific files

.vscode/

.idea/

\*.sublime-workspace

\*.sublime-project

# Ignore logs and temporary files

\*.log

\*.tmp

# Ignore configuration files and secrets

.env

# Ignore Jupyter Notebook checkpoints (if applicable)

.ipynb\_checkpoints/

# Ignore compiled Python files

\*.pyc

1. **Initialize a repository:**
2. **Initialize a repository, Command: git init**

**A black screen with white text

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

1. **Move the app.py,.gitignore,.env file outside the myenv folder**

**A screenshot of a computer

Description automatically generated**

1. **Git add, command: git add .**

****

1. **Check status: git status**

**A screen shot of a computer

Description automatically generated**

1. **Create a New Git Hub repo**

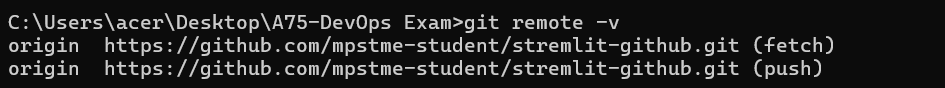
**A screenshot of a computer

Description automatically generated**

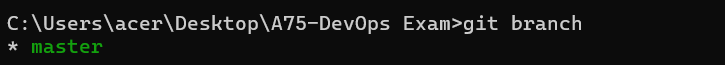
1. **Connect your repository: git remote add origin https://github.com/dakshita-thakkar/streamlit-github.git**

****

1. **Verify: git remote -v**

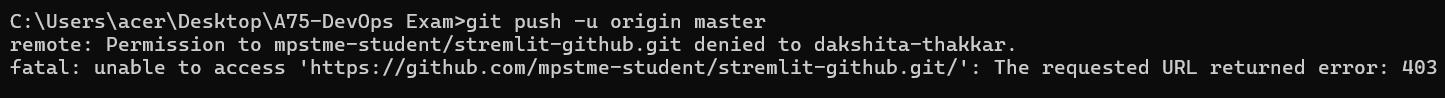


1. **Verify the branch, Command: git branch**

****

1. **During pushing, Git might throw a Permissions Denied Error**

**git push -u origin master**



**Run: git credential-manager clear**

**A screenshot of a computer program

Description automatically generated**

**Go To Control Panel>User Accounts>User Accounts>Manage Credentials**

**A screenshot of a computer

Description automatically generated**

**Remove your GitHub credentials**

**A screenshot of a computer

Description automatically generated**

1. **Push your files again: git push -u origin master -> Sign In again using browser**

**A screen shot of a computer program

Description automatically generated**

* **DEPLOYING YOUR APP ON STREAMLIT CLOUD**

1. **Go to Stramlit Community Cloud-> Login using GitHub->Create an app ->Deploy using Github**

**A screenshot of a computer

Description automatically generated**

1. **Fill all the details->Deploy**

**A screenshot of a computer

Description automatically generated**

**Deploy Your Streamlit App on Streamlit Cloud:**

**A screenshot of a computer

Description automatically generated**

**Q. KUBERNETES**

**Question: Scale a running Kubernetes deployment from 2 replicas to 5 replicas and verify that the scaling was successful.**

1. Open cmd🡪run as administrator🡪ensure minikube,Kubernetes and docker desktop is installed
2. Start minikube using docker driver, command: **minikube start --driver=docker**

A computer screen with white text

Description automatically generated

1. Run command: **docker ps** to verify that minikube has startedA computer code on a black background

   Description automatically generated
2. Run command: **minikube status**

A screen shot of a computer

Description automatically generated

1. Run command: **kubectl create deployment myy-nginx --image=nginx:latest** to pull the nginx image from docker hub

A black background with white text

Description automatically generated

1. Run: **kubectl get deployments**

A screenshot of a computer screen

Description automatically generated

1. Run: **kubectl get pods**

A black background with white text

Description automatically generated

1. Expose these pods to minikube:

Command: **kubectl expose deployment my-nginx --port=80 --type=LoadBalancer**

A black background with white text

Description automatically generated

1. Run: **kubectl get services**

A screenshot of a computer program

Description automatically generated

1. Run: **minikube service myy-nginx**

A screen shot of a computer

Description automatically generated

1. **Output:**

A screenshot of a computer

Description automatically generated

1. For creating 2 replicas, run: **kubectl scale deployment my-test-app --replicas=3**

A black background with white text

Description automatically generated

1. To scale it to 5 replicas, run: **kubectl scale deployment my-test-app --replicas=5**

**A black background with white text

Description automatically generated**

1. **To verify, run: kubectl get pods**

**A screenshot of a computer screen

Description automatically generated**