Sidrah Containerization Lab

Dockerfile:

```
FROM python:3.9-slim

# Set working directory inside the container

WORKDIR /app

# Copy everything into /app

COPY . /app

# Run the Python program

CMD ["python", "app.py"]
```

1) Hello World

```
print("Hello from Docker!")
```

2) Calculator

```
def calculator():

while True:

print("\nSimple Calculator")

print("1. Add")

print("2. Subtract")

print("3. Multiply")

print("4. Divide")

print("5. Exit")

choice = input("Enter your choice (1-5): ")



if choice == '5':
```

```
print("Exiting the calculator. Goodbye!")
break
if choice in ['1', '2', '3', '4']:
try:
num1 = float(input("Enter first number: "))
num2 = float(input("Enter second number: "))
except ValueError:
print("Invalid input! Please enter numeric values.")
continue
if choice == '1':
result = num1 + num2
print(f"Result: {result}")
elif choice == '2':
result = num1 - num2
print(f"Result: {result}")
elif choice == '3':
result = num1 * num2
print(f"Result: {result}")
elif choice == '4':
if num2 == 0:
print("Error: Division by zero!")
else:
result = num1 / num2
print(f"Result: {result}")
else:
print("Invalid choice! Please select a valid option.")
```

```
if __name__ == "__main__":
calculator()
```

3) CPU Delay

```
import time
print("Starting CPU-intensive task...")
start=time.time()
while time.time() - start < 30:
x=0
for i in range(1000000):
x+=i*i
print("Task completed!")
```

4) Factorial

```
def factorial(n):
return 1 if n==0 else n*factorial(n-1)
if __name__=="__main__":
num=int(input("Enter a number: "))
print(f"Factorial of {num} is {factorial(num)}")
```

5) Student Chart

```
import csv
def read_csv(filename):
    students = []
    with open(filename, newline='') as csvfile:
        reader = csv.DictReader(csvfile)
        for row in reader:
```

```
            students.append((row['Name'], int(row['Marks'])))
    return students
def draw_bar_chart(data):
    print("Student Marks Visualization\n")
    max_name_len = max(len(name) for name, _ in data)
    max_mark = max(mark for _, mark in data)
    for name, mark in data:
        bar = '' * (mark * 50 // max_mark) # scale to max 50 chars
        print(f"{name.ljust(max_name_len)} | {bar} {mark}")


if __name__ == "__main__":
    students = read_csv("students.csv")
    draw_bar_chart(students)
```

Dockerfile:

```
FROM python:3.9-slim
WORKDIR /app
COPY app.py students.csv ./
CMD ["python", "app.py"]
```

students.csv:

```
Name,Marks
Alice,58
Sidrah,65
Harshitha,70
XYZ,98
```

Note: to create .csv, $vi students.csv and then type each row separated by commas without space

6) Passing Environment Variables

```
import os
print("=== Environment Variables Passed to Container ===\n")
for key, value in os.environ.items():
    print(f"{key} = {value}")
```

7) Ping program

```
import os
import sys

if len(sys.argv)!=2:
    print("Usage: python ping.py <hostname_ip>")
    sys.exit(1)

host=sys.argv[1]
response=os.system(f"ping -c 4 {host}")

if response==0:
    print(f"{host} is reachable")
else:
    print(f"{host} is unreachable")
```

Dockerfile:

```
FROM python:3.9-slim
RUN apt-get update && apt-get install -y iputils-ping
WORKDIR /app
COPY app.py .
ENTRYPOINT ["python", "app.py"]
```

Output:

```
rit@rit:~/sidrah_containerization/exp7$ vi app.py
rit@rit:~/sidrah_containerization/exp7$ vi Dockerfile
rit@rit:~/sidrah_containerization/exp7$ sudo docker build -t ping-c .
[+] Building 0.7s (9/9) FINISHED
docker:default
 => [internal] load build definition from Dockerfile
0.0s
 => => transferring dockerfile: 211B
0.0s
 => [internal] load metadata for docker.io/library/python:3.9-slim
0.7s
 => [internal] load .dockerignore
0.0s
 => => transferring context: 2B
0.0s
 => [1/4] FROM
docker.io/library/python:3.9-slim@sha256:aeebfa2890da7819f1617ec9a5650669570ab0802e5f
51063aa8b7499da1ed26                    0.0s
```

```
 => [internal] load build context
0.0s
 => => transferring context: 296B
0.0s
 => CACHED [2/4] RUN apt-get update && apt-get install -y iputils-ping
0.0s
 => CACHED [3/4] WORKDIR /app
0.0s
 => CACHED [4/4] COPY app.py .
0.0s
 => exporting to image
0.0s
 => => exporting layers
0.0s
 => => writing image
sha256:63404404d7e72d872e23136d61db09e20d3022407d36de778a4b6ef9d42145ee
0.0s
 => => naming to docker.io/library/ping-c
0.0s
rit@rit:~/sidrah_containerization/exp7$ sudo docker run ping-c 127.0.0.1
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.019 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.050 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.051 ms
64 bytes from 127.0.0.1: icmp_seq=4 ttl=64 time=0.049 ms

--- 127.0.0.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3087ms
rtt min/avg/max/mdev = 0.019/0.042/0.051/0.013 ms
127.0.0.1 is reachable
```

8) Prime Checker

```
def is_prime(n):
    if n <= 1:
        return False
    for i in range(2, int(n ** 0.5) + 1):
        if n % i == 0:
            return False
    return True
if __name__ == "__main__":
    print("=== Prime Number Checker (Python in Docker) ===")
    num = int(input("Enter a number: "))
    if is_prime(num):
```

```
        print(f"{num} is a Prime Number.")
    else:
        print(f"{num} is NOT a Prime Number.")
```

Dockerfile:

```
FROM python:3.9-slim

WORKDIR /app

COPY . /app

CMD ["python", "app.py"]
```

Kubernetes setup:
Requirements:

1. sudo apt-get update -y
2. Install minikube
3. Install kubectl
4. Install Docker

curl -LO
https://github.com/kubernetes/minikube/releases/latest/download/minikube-linux-amd64
sudo install minikube-linux-amd64 /usr/local/bin/minikube && rm minikube-linux-amd64

curl -LO "https://dl.k8s.io/release/$(curl -s
https://storage.googleapis.com/kubernetes-release/release/stable.txt)/bin/linux/amd64/kubectl"
sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl

minikube start
Issue: minikube v1.37.0 on Ubuntu 22.04
👎 *Unable to pick a default driver. Here is what was considered, in preference order:*
▪ *docker: Not healthy: "docker version --format {{.Server.Os}}-{{.Server.Version}}:{{.Server.Platform.Name}}" exit status 1: permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Get "http://%2Fvar%2Frun%2Fdocker.sock/v1.48/version": dial unix /var/run/docker.sock: connect: permission denied*
▪ *docker: Suggestion: Add your user to the 'docker' group: 'sudo usermod -aG docker $USER && newgrp docker' <https://docs.docker.com/engine/install/linux-postinstall/>*
💡 *Alternatively you could install one of these drivers:*
▪ *kvm2: Not installed: exec: "virsh": executable file not found in $PATH*
▪ *podman: Not installed: exec: "podman": executable file not found in $PATH*
▪ *qemu2: Not installed: exec: "qemu-system-x86_64": executable file not found in $PATH*
▪ *virtualbox: Not installed: unable to find VBoxManage in $PATH*

❌ *Exiting due to DRV_NOT_HEALTHY: Found driver(s) but none were healthy. See above for suggestions how to fix installed drivers.*

Fix: # Add your user to the docker group
sudo usermod -aG docker $USER
# Apply the new group membership without logging out
newgrp docker

minikube start
Issue: /usr/local/bin/kubectl is version 1.31.0, which may have incompatibilities with Kubernetes 1.34.0.
▪ Want kubectl v1.34.0? Try 'minikube kubectl -- get pods -A'

Fix: curl -LO "https://dl.k8s.io/release/v1.34.0/bin/linux/amd64/kubectl"
sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl

minikube start

9) Kubernetes + Nginx + YAML

hello.yaml

```
apiVersion: v1
kind: Pod
metadata:
  name: hello-pod
  labels:
    app: hello
spec:
  containers:
    - name: hello-container
      image: nginx
      ports:
        - containerPort: 80
---
apiVersion: v1
kind: Service
metadata:
  name: hello-service
spec:
  selector:
    app: hello
  type: NodePort
  ports:
    - port: 80
      targetPort: 80
      nodePort: 30007
```

Note: Type “minikube start” in the terminal before starting this program.

10) Ubuntu Container

```
#!/bin/bash
echo "----------"
echo "Welcome to Docker Offline Demo!"
echo "This script is running inside a container."
echo "----------"
```

Dockerfile:

```
FROM ubuntu:latest
WORKDIR /app
COPY hello.sh .
RUN chmod +x hello.sh
CMD ["./hello.sh"]
```