Jayanthi

Assignment 21/10

1. Create a Dockerfile for a Custom Logging Container

• Inside the container:

• Install cron

• Create a cron job that writes the time to a log file every minute

• Mount a volume at /var/logs/ to persist logs

2. Set Up Nginx

• Mount a local HTML file (index.html) using a Docker volume

• Expose port 8080 and check that Nginx serves your content

3. Add Fluentd to Collect Logs

• Use fluent/fluentd image

• Configure it to:

• Read logs from the volume shared with your custom Alpine container

• Print those logs to stdout⸻

4. Docker Networking

• Create a custom bridge network

• Connect all containers to it

• Ensure Nginx and Fluentd can talk to other containers via container names

5. Logging Validation

• Check that your cron job logs are:

• Written to volume

• Picked up by Fluentd

• Printed to stdout

6. Clean Restart Test

• Stop and remove all containers

• Start again

• Ensure:

• Logs are still there (volume works)

• Cron restarts

• Nginx still serves HTML

7. Limit CPU and memory usage for each container

**1. Create a Dockerfile for a Custom Logging Container**

• Inside the container:

• Install cron

• Create a cron job that writes the time to a log file every minute

• Mount a volume at /var/logs/ to persist logs

Step1:

**Install Docker on EC2**

**sudo apt update -y**

**sudo apt install -y docker.io**

**sudo systemctl start docker**

**sudo systemctl enable docker**

**sudo usermod -aG docker $USER # allow running docker without sudo**

To Test Docker:

**docker --version**

**docker run hello-world**

Step 2: Create a custom logger container

**Folder structure:**

~/docker\_assign/

├── custom-logger/

│ └── Dockerfile

├── html/

│ └── index.html

**vi Dockerfile**

**Dockerfile for logger (custom-logger/Dockerfile):**

FROM ubuntu:latest

RUN apt-get update && apt-get install -y cron

RUN mkdir -p /var/logs

# Add cron job: write date to log every minute

RUN echo "\* \* \* \* \* date >> /var/logs/time.log 2>&1" > /etc/cron.d/logger-cron

# Give permission

RUN chmod 0644 /etc/cron.d/logger-cron

# Apply cron job

RUN crontab /etc/cron.d/logger-cron

# Expose volume

VOLUME ["/var/logs"]

# Start cron in foreground

CMD ["cron", "-f"]

A screenshot of a computer

AI-generated content may be incorrect.

Build and run logger container:

**docker build -t custom-logger ~/docker\_assign/custom-logger**

**docker run -d --name logger -v logs\_data:/var/logs custom-logger**

To Check logs:

**docker exec -it logger cat /var/logs/time.log**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

**2. Set Up Nginx**

• Mount a local HTML file (index.html) using a Docker volume

• Expose port 8080 and check that Nginx serves your content

Step1:

Create HTML page using below commands:

**mkdir -p ~/docker\_assign/html**

**echo "<h1>Hello from Docker Nginx!</h1>" > ~/docker\_assign/html/index.html**

**A screenshot of a computer

AI-generated content may be incorrect.**

we can proceed to run the Nginx container using below command:

**docker run -d --name my\_nginx -p 8080:80 -v ~/docker\_assign/html:/usr/share/nginx/html:ro nginx**

**A screenshot of a computer

AI-generated content may be incorrect.**

• Expose port 8080 and check that Nginx serves your content

**docker run -d --name my\_nginx -p 8080:80 -v ~/docker\_assign/html:/usr/share/nginx/html:ro nginx**

**A screenshot of a computer

AI-generated content may be incorrect.3. Add Fluentd to Collect Logs**

• Use fluent/fluentd image

• Configure it to:

• Read logs from the volume shared with your custom Alpine container

• Print those logs to stdout⸻

Step1: we need to Pull Fluentd image using below command

**docker pull fluent/fluentd**

**A screenshot of a computer

AI-generated content may be incorrect.**

Step2: we need to Create a minimal config file fluent.conf:

**vi fluent.conf**

<source>

@type tail

path /var/logs/time.log

pos\_file /var/log/log.pos

tag cron.logs

format none

</source>

<match cron.logs>

@type stdout

</match>

Change the permission:

**sudo chown -R 1000:1000 ~/docker\_assign/logs**

Step3: To run Fluentd container with the same volume using below command:

**docker run -d --name fluentd \**

**-v ~/docker\_assign/logs:/var/logs \**

**-v ~/docker\_assign/custom-logger:/fluentd/etc \**

**fluent/fluentd**

To Verify Fluentd logs

**docker logs -f fluentd**

**A screenshot of a computer

AI-generated content may be incorrect.**

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

4. Docker Networking

• Create a custom bridge network

• Connect all containers to it

• Ensure Nginx and Fluentd can talk to other containers via container names

**Step 1**:To Create a custom bridge network

**docker network create my-bridge-network**

**A screenshot of a computer

AI-generated content may be incorrect.**

**Step 2:** To Run our Alpine container on the custom network

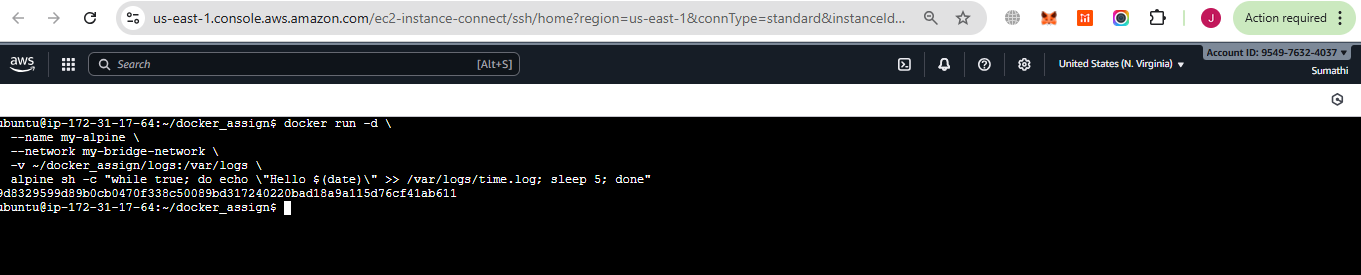
**docker run -d \**

**--name my-alpine \**

**--network my-bridge-network \**

**-v ~/docker\_assign/logs:/var/logs \**

**alpine sh -c "while true; do echo \"Hello $(date)\" >> /var/logs/time.log; sleep 5; done"**

****

**Step 3:** To Run Fluentd on the same network

**docker run -d \**

**--name fluentd \**

**--network my-bridge-network \**

**-v ~/docker\_assign/logs:/var/logs \**

**-v ~/docker\_assign/custom-logger/fluent.conf:/fluentd/etc/fluent.conf \**

**fluent/fluentd:v1.19-1**

**A screenshot of a computer

AI-generated content may be incorrect.**

**Step 4:** To Run Nginx on the same network

**docker run -d \**

**--name my-nginx2 \**

**--network my-bridge-network \**

**-p 8081:80 \**

**nginx:latest**

**A screenshot of a computer

AI-generated content may be incorrect.**

**Step 5:** we need to Verify network connections

**docker network inspect my-bridge-network**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

To verify Each container can ping the others using below commands

**docker exec -it my-nginx ping fluentd**

**docker exec -it fluentd ping my-alpine**

we can Use nslookup / getent to test name resolution

we need to Check if Fluentd can resolve the container name:

**docker exec -it fluentd sh**

**getent hosts my-alpine**

**docker exec -it fluentd sh**

**getent hosts my-nginx2**

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

To install ping to my-alpine container

**docker exec -it my-alpine sh**

inside container

**apk update**

**apk add iputils**

**A screenshot of a computer

AI-generated content may be incorrect.**

To install ping to fluentd container

**docker exec -it --user root fluentd sh**

**apt-get update**

**apt-get install -y iputils-ping**

To verify the connection

**docker exec -it fluentd sh**

**ping -c 3 my-alpine**

**ping -c 3 my-nginx**

**exit**

**A screenshot of a computer

AI-generated content may be incorrect.**

**5. Logging Validation**

• Check that your cron job logs are:

• Written to volume

• Picked up by Fluentd

• Printed to stdout

To Check logs are written to the volume

**ls -l ~/docker\_assign/logs**

**cat ~/docker\_assign/logs/time.log**

**A screenshot of a computer

AI-generated content may be incorrect.**

To Check logs are picked up by Fluentd

**docker logs -f fluentd**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

Confirm logs are printed to stdout

**docker exec -it fluentd sh**

**tail -f /var/logs/time.log**

**A screenshot of a computer

AI-generated content may be incorrect.**

6. Clean Restart Test

• Stop and remove all containers

• Start again

• Ensure:

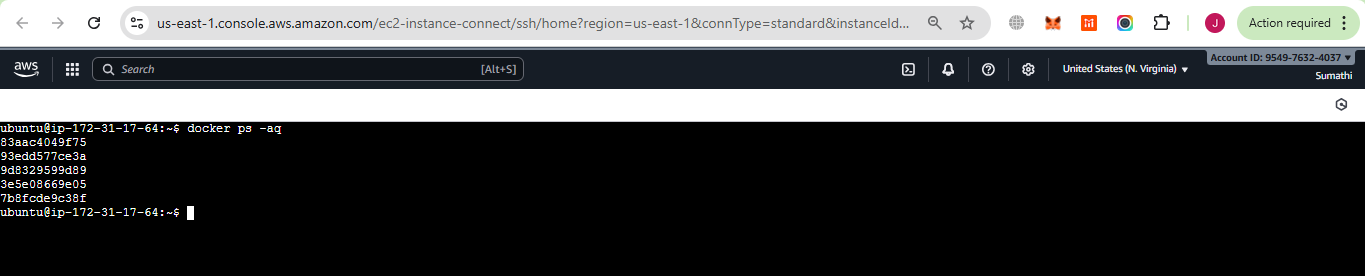
• Logs are still there (volume works)

• Cron restarts

• Nginx still serves HTML

**Step 1**: Stop and remove all containers using below commands

**docker ps -aq # List all container IDs**

****

**docker stop $(docker ps -aq)**

**docker rm $(docker ps -aq)**

A computer screen shot of a black screen

AI-generated content may be incorrect.

• Start again

**Step 2: Start containers again**

Make sure we need to recreate them **with the same volume mounts** so logs persist:

# Start cron logging container

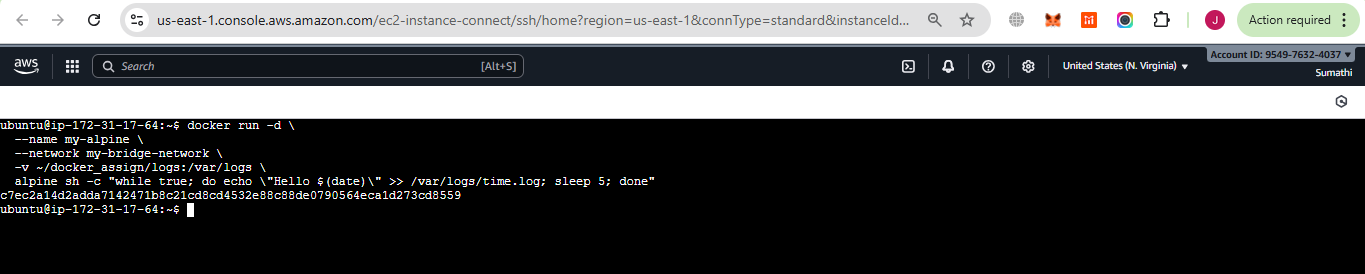
docker run -d \

--name my-alpine \

--network my-bridge-network \

-v ~/docker\_assign/logs:/var/logs \

alpine sh -c "while true; do echo \"Hello $(date)\" >> /var/logs/time.log; sleep 5; done"



# Start Fluentd container

**docker run -d \**

**--name fluentd \**

**--network my-bridge-network \**

**-v ~/docker\_assign/logs:/var/logs \**

**-v ~/docker\_assign/custom-logger/fluent.conf:/fluentd/etc/fluent.conf \**

**fluent/fluentd:v1.19-1**

A screenshot of a computer

AI-generated content may be incorrect.

# Start Nginx container

**docker run -d \**

**--name my-nginx \**

**--network my-bridge-network \**

**-p 8080:80 \**

**-v ~/docker\_assign/html:/usr/share/nginx/html \**

**nginx:latest**

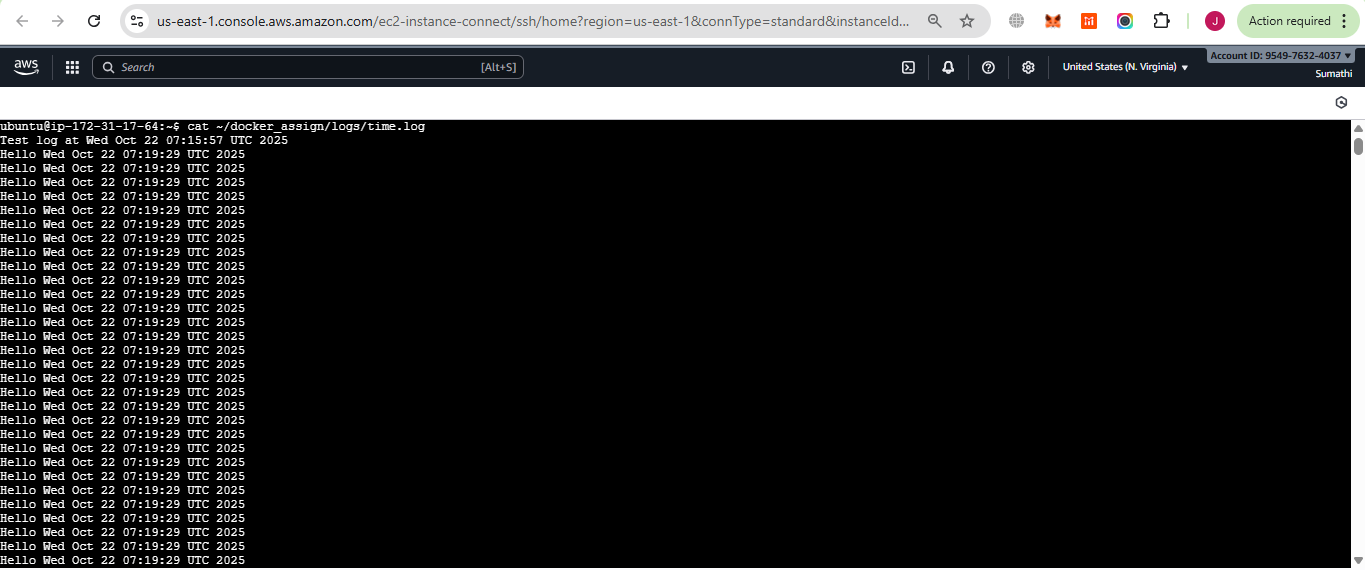
A screenshot of a computer

AI-generated content may be incorrect.

**Step 3: Verify everything**

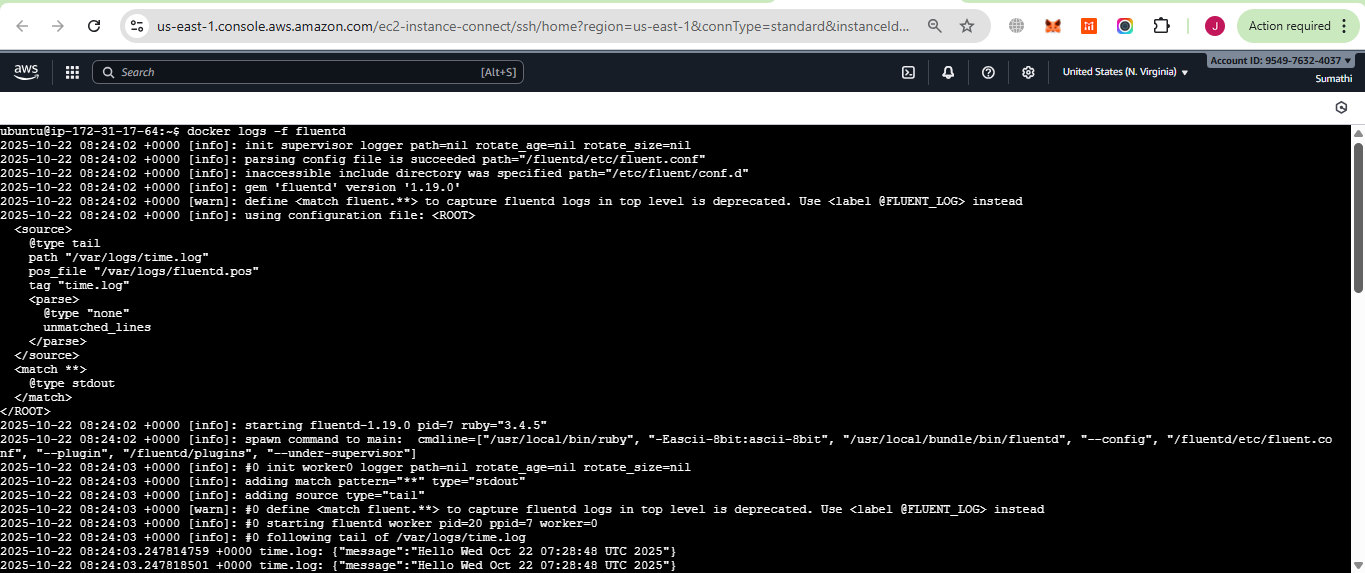
1. Check logs persist in volume

**cat ~/docker\_assign/logs/time.log**

****

Check Fluentd stdout

**docker logs -f fluentd**

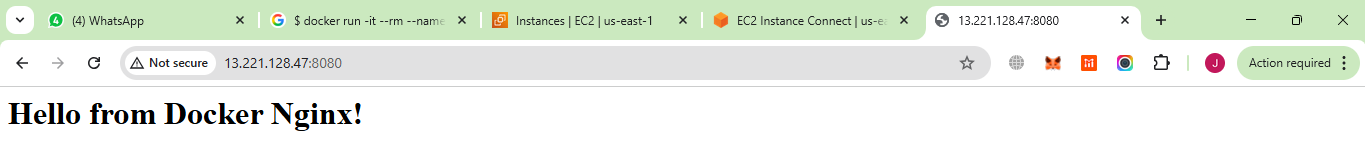
****

**A screenshot of a computer

AI-generated content may be incorrect.**

Check Nginx serves HTML **Open in browser: http://<server-ip>:8080**

**http://13.221.128.47:8080/**

****

7. Limit CPU and memory usage for each container

**docker rm -f my-alpine**

**docker run -d \**

**--name my-alpine \**

**--network my-bridge-network \**

**--cpus="0.5" \**

**--memory="256m" \**

**-v ~/docker\_assign/logs:/var/logs \**

**alpine sh -c "while true; do echo \"Hello $(date)\" >> /var/logs/time.log; sleep 5; done"**

**docker run -d \**

**--name fluentd \**

**--network my-bridge-network \**

**--cpus="0.5" \**

**--memory="256m" \**

**-v ~/docker\_assign/logs:/var/logs \**

**fluent/fluentd:v1.19-1**

**docker run -d \**

**--name my-nginx \**

**--network my-bridge-network \**

**--cpus="0.5" \**

**--memory="256m" \**

**-p 8080:80 \**

**nginx:latest**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

we can check it’s running and writing logs:

**docker ps**

**A screenshot of a computer

AI-generated content may be incorrect.**

**docker exec -it my-alpine tail -f /var/logs/time.log**

**A screen shot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

Stop and Remove All Containers

**docker stop $(docker ps -aq)**

**docker rm $(docker ps -aq)**

Verify Containers Are Removed

**docker ps -a**

**A screenshot of a computer

AI-generated content may be incorrect.**

Start Containers Again

**docker run -d \**

**--name my-alpine \**

**--network my-bridge-network \**

**--cpus="0.5" \**

**--memory="256m" \**

**-v ~/docker\_assign/logs:/var/logs \**

**alpine sh -c "while true; do echo \"Hello $(date)\" >> /var/logs/time.log; sleep 5; done"**

**A screenshot of a computer

AI-generated content may be incorrect.**

Start fluentd

**docker run -d \**

**--name fluentd \**

**--network my-bridge-network \**

**-v ~/docker\_assign/logs:/var/logs \**

**-v ~/docker\_assign/custom-logger/fluent.conf:/fluentd/etc/fluent.conf \**

**fluent/fluentd:v1.19-1**

**A screenshot of a computer

AI-generated content may be incorrect.**

Start nginx

**docker run -d \**

**--name my-nginx \**

**--network my-bridge-network \**

**--cpus="0.5" \**

**--memory="256m" \**

**-p 8080:80 \**

**nginx:latest**

**A screenshot of a computer

AI-generated content may be incorrect.**

Verify Logs Persist

**cat ~/docker\_assign/logs/time.log**

**A screenshot of a computer

AI-generated content may be incorrect.**

Verify Fluentd Picks Up Logs

**docker logs -f fluentd**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer screen

AI-generated content may be incorrect.**

Verify Nginx is Serving HTML

**curl http://localhost:8080**

A screenshot of a computer

AI-generated content may be incorrect.