**Assignments by Aditya** 13-sep-2023

1. Create a install-tomcat.sh file for tomcat that installs and configures and make it available on browser and also it should be opened from manager app on browser.
2. **Write a shell script to kill processes that are using high memory and making system slow and also check for shell scripts that are commonly used**.

#!/bin/bash

# Set the memory threshold in kilobytes (adjust as needed)

MEMORY\_THRESHOLD=50 # For example, 50%

# Get a list of processes sorted by memory usage (descending)

PID\_LIST=$(ps aux --sort=-%mem | awk -v mem\_thresh="$MEMORY\_THRESHOLD" '$4 >= mem\_thresh {print $2}')

# Loop through the list of processes and kill them

for PID in $PID\_LIST; do

# Check if the PID belongs to the script itself (to avoid self-destruction)

if [ "$$" -ne "$PID" ]; then

# Send a SIGTERM signal to gracefully terminate the process

kill -TERM "$PID"

sleep 5 # Wait for a few seconds

# If the process is still running, forcefully terminate it

if ps -p "$PID" > /dev/null; then

kill -KILL "$PID"

fi

echo "Killed process with PID $PID"

fi

done

echo "Memory cleanup script completed."

1. java source code to be setup in git hub, pipeline cicd to pull source code and build into jar/war file and push to artifactory, binary is pulled from artifactory and is pushed into the correct folder of tomcat (should be setup via dockerfile as a container).
2. **Create custome dockerfile (image) for tomcat which will have sample war/jar files preloaded**.

Setp1: Create a directory and add jar or war file into the directory and create a Dockerfile in same directory and get into the Dockerfile using vim command

# Use the official Tomcat base image

FROM tomcat:latest

# Create a directory to store your application files

WORKDIR /usr/local/tomcat/webapps

# Copy your sample WAR/JAR files to the Tomcat webapps directory

COPY sample.war .

# ADD sample.jar .

# Expose the Tomcat port

EXPOSE 8080

# Start Tomcat when the container starts

CMD ["catalina.sh", "run"]

Step2: build the image

Command: docker build –t custom\_tomcat .

Step3: Run the container

Command: docker run –d –p 8080:8080 --name tomcat\_jar custom\_tomcat

1. **Integrate loadbalancer in to Route53, Create a certificate in AWS certificate manager(ssl certificate) and use to access your web application on htpps://yourdomain-name.com**

Done.

1. Create a lambda function for pushing cloud trail logs into S3
2. What are services in K8s, link the deployment to service

4 types of Services in k8s

1. ClusterIP Service
2. NodePort Service
3. LoadBalancer Service
4. Headless Service

Demo:

**Create a deployment yaml file for simple-python-app**

apiVersion: apps/v1

kind: Deployment

metadata:

  name: python-sample-app

  labels:

    app: python-sample-app

spec:

  replicas: 2

  selector:

    matchLabels:

      app: python-sample-app

  template:

    metadata:

      labels:

        app: python-sample-app

    spec:

      containers:

      - name: python-app

        image: anand1565/python-sample-app:v1

        ports:

        - containerPort: 8000

**Create a NodePort Service to expose app to the External world.**

apiVersion: v1

kind: Service

metadata:

  name: python-sample-app

spec:

  type: NodePort

  selector:

    app: python-sample-app

  ports:

  - targetPort: 8000 #application running on this port

    port: 8080 #custerIP port

    nodePort: 30010 #rage from 30000 – 32767

1. how to filter the way an image is being pulled, only pull when there is change
2. find out what are the local kubernetes setups that can be done and which would be your preferred local setup to use.(K8s localsetup)