ASSESSMENT-1

1. **gcloud command(s) to create IAM 2 Service accounts, assign the roles and principals (permissions) to the SA. One SA has to be complete Admin access and other service account has limited- create, view & delete but not assign any principles like assigning rules.**

- **Creating the first service account with complete admin access**

gcloud iam service-accounts create full-admin-sa --description="Service account with complete admin access" --display-name="Full Admin Service Account"

- **Assigning complete admin role to the first service account**

gcloud projects add-iam-policy-binding PROJECT_ID -- member="serviceAccount:full-admin-sa@PROJECT_ID.iam.gserviceaccount.com" --role="roles/owner

- **Creating the second service account with limited permissions**

gcloud iam service-accounts create limited-sa --description="Service account with limited permissions" --display-name="Limited Service Account"

- **Assigning limited permissions to the second service account**

gcloud projects add-iam-policy-binding PROJECT_ID -- member="serviceAccount:limitted-sa@PROJECT_ID.iam.gserviceaccount.com" -- role="roles/edito

---

2. **gcloud command(s) to create the Instance Group, Instances (2) using above SA(s), having disk size of 1Tb and startup script installing tomcat, python and apache**

#!/bin/bash

 # Create admin instance:

gcloud compute instances create admin-instance --project YOUR_PROJECT_ID --machine-type n1-standard-4 --zone us-central1-a --serviceAccount:full-admin-sa@PROJECT_ID.iam.gserviceaccount.com --boot-disk-size 1TB --boot-disk-auto-delete --scopes https://www.googleapis.com/auth/compute

 # Create limited instance:

gcloud compute instances create limited-instance --project YOUR_PROJECT_ID --machine-type n1-standard-4 --zone us-central1-a --serviceAccount:limitted-sa@PROJECT_ID.iam.gserviceaccount.com --boot-disk-size 1TB --boot-disk-auto-delete --scopes https://www.googleapis.com/auth/compute

 # Create unmanaged instance group

gcloud compute instance-groups unmanaged create custom-instance-group --zone=us-central1-a

 # Add instances to the unmanaged instance group

gcloud compute instance-groups unmanaged add-instances custom-instance-group --instances=admin-instance,limited-instance --zone=us-east1-b

Installing python3

sudo apt update

sudo apt install python3 -y

Installing apache

sudo apt install apache2 -y

 # Installing Tomcat

sudo apt install default-jdk -y

wget https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.88/bin/apache-tomcat-9.0.88.tar.gz

tar -zxvf apache-tomcat-9.0.88.tar.gz

sed -i '56 a\<role rolename="manager-gui"/>' apache-tomcat-9.0.88/conf/tomcat-users.xml

sed -i '57 a\<role rolename="manager-script"/>' apache-tomcat-9.0.88/conf/tomcat-users.xml

sed -i '58 a\<user username="tomcat" password="tomcat" roles="manager-gui, manager-script"/>' apache-tomcat-9.0.88/conf/tomcat-users.xml

sed -i '59 a\</tomcat-users>' apache-tomcat-9.0.88/conf/tomcat-users.xml

sed -i '56d' apache-tomcat-9.0.88/conf/tomcat-users.xml

sed -i '21d' apache-tomcat-9.0.88/webapps/manager/META-INF/context.xml

sed -i '22d' apache-tomcat-9.0.88/webapps/manager/META-INF/context.xml

sh apache-tomcat-9.0.88/bin/startup.sh

#PASSWORD=USERNAME=tomcat

---

3. **gcloud command(s) to create Bucket and assign the permissions to respective SA(s) with mentioned behavior**

- gsutil mb -c NEARLINE -p public -l us-central1 -p PROJECT_ID gs://BUCKET_NAME

- gsutil iam ch serviceAccount:full-admin-sa@PROJECT_ID.iam.gserviceaccount.com:roles/storage.admin gs://BUCKET_NAME

- gsutil iam ch serviceAccount:limitted-sa@PROJECT_ID.iam.gserviceaccount.com:roles/storage.objectUser gs://BUCKET_NAME

4. **gcloud commands to create single instance, and attaching the persistant disk(s) of size 512 gb and 1 tb disk (Drive Letter: D & E)**

- gcloud compute disks create disk-a --size=512GB --zone=us-central1-a

- gcloud compute disks create disk-b --size=1TB --zone=us-central1-a

---

Assessment-2

1. **Create CICD Pipeline to build and deploy java application using plain maven build and export war file to GCS Bucket. Deploy the War file from bucket to Server**

Approach1:

```
pipeline {
  agent any
  tools {
    maven 'maven' // Use the name configured in Jenkins
  }
  stages {
    stage('checkout') {
      steps {
```

```
            git 'https://github.com/summu97/monolythic-project.git'
        }
    }
    stage('package') {
        steps {
            sh 'mvn clean package'
        }
    }
    stage('storage') {
        steps {
            googleStorageUpload bucket: 'gs://YOUR_BUCKET_NAME/', credentialsId:
'YOUR_JSON_KEY_CREDENTIAL_ID', pattern: 'target/*.war'
        }
    }
    stage ('Download from GCS') {
        steps {
            googleStorageDownload bucketUri: ' YOUR_gsutil_URI ', credentialsId:
'YOUR_JSON_KEY_CREDENTIAL_ID', localDirectory: '.'
 // to gen
        }
    }
    stage('Deploy') {
        steps {
            deploy adapters: [
                tomcat9(
                    credentialsId: 'YOUR_TOMCAT_CREDENTIAL_ID',
                    path: '',
                    url: 'TOMCAT_SERVER_URL'
```

```
                )
                ],
                contextPath: 'tomcat-netflix',
                war: 'target/NETFLIX-1.2.2.war'
        }
    }
  }
}
```

- **ANOTHER APPROACH TO STORE WAR FILE TO STORAGE & THEN DOWNLOAD AND DEPLOY TO TOMCAT:**

- **In VM Instance Terminal - Set password to jenkins-Change permissions to sudoers file:**
- chmod 640 /etc/sudoers
- vim /etc/sudoers (add below line and :wq)
- jenkins ALL=(ALL:ALL) ALL
- chmod 400 /etc/sudoers

In Jenkins UI: Add above Jenkins password in credentials: Kind (secret text)

```
pipeline {
  agent any
  tools {
    maven 'maven' // Use the name configured in Jenkins
  }
    environment {
    MY_PASSWORD = credentials('Paste-jenkins-password-ID')
  }
  stages {
    stage('checkout') {
      steps {
        git 'https://github.com/summu97/monolythic-project.git'
      }
    }
    stage('package') {
```

```
        steps {
          sh 'mvn clean package'
        }
      }
      stage('storage') {
        steps {
          googleStorageUpload bucket: 'gs://YOUR_BUCKET_NAME', credentialsId: 'Paste-json-key-
ID', pattern: 'target/*.war'
        }
      }
      stage('Copying war') {
        steps {
          sh 'gsutil cp gs://YOUR_BUCKET_NAME/target/NETFLIX-1.2.2.war .'
        }
      }
      stage('Deploy to container') {
        steps {
          sh '''
            echo $MY_PASSWORD | sudo -S  mv  NETFLIX-1.2.2.war /root/apache-tomcat-
9.0.88/webapps
          '''
        }
      }
    }
}
```

---

2. **Create CICD Pipeline to build and deploy reactjs Application with the multiple branch pipeline.**

When i execute master branch, it should generate package_release.zip file (Expected output for master - package_release_1.X.X.zip)and

when i execute other branches, it should generate package_snapshot.zip file (Expected output for master - package_snapshot_1.X.X.zip)

```
pipeline {
  agent any
```

```
  stages {
    stage('Checkout') {
      steps {
        script {
          def gitInfo = checkout([$class: 'GitSCM', branches: [[name: '*/${branch}']], userRemoteConfigs: [[url: 'https://github.com/summu97/react.js.git']]])
          def branchName = gitInfo.GIT_BRANCH.tokenize('/')[1]
          echo "Branch name: ${branchName}"
        }
      }
    }
    stage('building code') {
      steps {
        sh '''
        npm install
        npm run build
        '''
      }
    }
    stage('creating zip') {
      steps {
        script {
          def version = "1.${env.BUILD_NUMBER}"
          def gitInfo = checkout([$class: 'GitSCM', branches: [[name: '*/${branch}']], userRemoteConfigs: [[url: 'https://github.com/summu97/react.js.git']]])
          def branchName = gitInfo.GIT_BRANCH.tokenize('/')[1]
```

```
                def fileName = branchName == 'master' ? 'package_release' :
'package_snapshot'

                fileName += "_${version}.zip"

                sh "zip -r ${fileName} build/"

            }

        }

    }

  }

}
```

Pass branch name using parameters: Build with parameters or you can directly pass the value i.e, master or main.

NOTE: Make sure to install node, npm, zip,git.