Infra Optimization Capstone Project Source Code

#Terraform Code

##Vars.tf

```
variable "AWS_ACCESS_KEY" {
  type = string
  default = "ASIA5ROHSDOADKV57ZFR"
}

variable "AWS_SECRET_KEY" {
  type = string
  default = "/ypAa7IbUF1lCHfe31VrUB1Id5WjPiMigq5u+6/l"
}

variable "AWS_TOKEN" {
  type = string
  default =
"FwoGZXIvYXdzED0aDLBqKKC0PnrwFct/0SK8Ace0qd4PotE9vpF6Mn5q9/gcMlRfK7rvJDu6LuTFu0UnFRZOz78
gzbmA6PE3pzo/OlasgGuqxfP2ahrx5jhTxKda/DeWxXjjmRY7DSxc2aSQyDuH0Lev1Ad6iKi+/
4zwmIOtB06T75G66JcT9mpiuskRX6+ItUDZ7O5uFwACUmxGOdZQsGvSw6M0OBSg2u6zfaSkstuH4lIIZvZYmgO/
DRCL+a3bFZWWuPEY83wS9Jxmrc6VYcB9s29/qIlRKL2D3JAGMi0V0PbCc/
WHomRL3IkYsV34pKVcrtMoXzMU1uvIFNcGbB+FAI+pN4xATGorqX0="
}

variable "AWS_REGION" {
  type = string
  default = "us-east-1"
}

variable "AMIS" {
  type = map(string)
  default = {
    us-east-1 = "ami-04505e74c0741db8d"
    us-west-2 = "ami-06b94666"
```

```
    eu-west-1 = "ami-0d729a60"

  }

}
variable "PATH_TO_PRIVATE_KEY" {
  default = "mykey"
}

variable "PATH_TO_PUBLIC_KEY" {
  default = "mykey.pub"
}

variable "INSTANCE_USERNAME" {
  default = "ubuntu"
}
```

##Provider.tf

```
provider "aws" {
  access_key = var.AWS_ACCESS_KEY
  secret_key = var.AWS_SECRET_KEY
  token = var.AWS_TOKEN
  region = var.AWS_REGION
}
```

##Instance.tf

```
resource "aws_key_pair" "mykey" {
  key_name   = "mykey"
  public_key = file(var.PATH_TO_PUBLIC_KEY)
}

resource "aws_instance" "kubernetes_master" {
  ami           = var.AMIS[var.AWS_REGION]
  instance_type = "t3.medium"
  key_name      = aws_key_pair.mykey.key_name
  vpc_security_group_ids = ["${aws_security_group.k8s.id}"]

 tags = {
    Name = "kubernetes_master"
  }
}
resource "aws_instance" "kubernetes_worker" {
  ami           = var.AMIS[var.AWS_REGION]
  instance_type = "t3.medium"
  key_name      = aws_key_pair.mykey.key_name
  vpc_security_group_ids = ["${aws_security_group.k8s.id}"]
  count = 2

 tags = {
    Name = "kubernetes_worker-${count.index}"
  }
}

resource "aws_security_group" "k8s" {
  name = "Ports 22"

  ingress {
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
```

```
    cidr_blocks = ["0.0.0.0/0"]
    self = true
  }

  ingress {
    from_port   = 80
    to_port     = 80
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
    self = true
  }

  ingress {
    protocol  = -1
    self      = true
    from_port = 0
    to_port   = 0
  }

  egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"]
    self = true
  }

 tags = {
    Name = "k8s"
  }
}

resource "local_file" "inventory" {
  filename = "./ansible_cm/inventory.ini"
  file_permission = "0644"
  content  = <<EOF
```

```
[kubernetes_master]
${aws_instance.kubernetes_master.public_dns}

[kubernetes_worker1]
${aws_instance.kubernetes_worker[0].public_dns}

[kubernetes_worker2]
${aws_instance.kubernetes_worker[1].public_dns}

EOF
}

resource "local_file" "host_script" {
  filename = "./add_host.sh"
  file_permission = "0700"
  content = <<EOF
#!/bin/bash
echo "Setting SSH Key"
#ssh-add ~/<PATH TO SSH KEYFILE>.pem
echo "Adding IPs"
ssh-keyscan -H ${aws_instance.kubernetes_master.public_dns} >> ~/.ssh/known_hosts
ssh-keyscan -H ${aws_instance.kubernetes_worker[0].public_dns} >> ~/.ssh/known_hosts
ssh-keyscan -H ${aws_instance.kubernetes_worker[1].public_dns} >>  ~/.ssh/known_hosts

EOF
}

resource "null_resource" "add_host_entry" {
  triggers = {
    order = local_file.host_script.id
  }
  provisioner "local-exec" {
    command = "sleep 10 && ./add_host.sh"
  }
}
```

#Ansible Code

##main.yaml

```
---
- hosts: kubernetes_master
  name: Kubernetes master control plane configuration
  become: yes
  user: ubuntu
  tags: master
  vars:
    ansible_ssh_private_key_file: "../mykey"
  tasks:
  - name: Run common tasks
    import_tasks: common.yaml
  - name: Configre k8s master node
    import_tasks: master.yaml

- hosts: kubernetes_worker1
  name: Kubernetes workde node configuration
  become: yes
  user: ubuntu
  tags: worker1
  vars:
    ansible_ssh_private_key_file: "../mykey"
    worker_hostname: "worker1"
  tasks:
  - name: Run common tasks
    import_tasks: common.yaml
  - name: Configre k8s worker node
    import_tasks: worker.yaml

- hosts: kubernetes_worker2
  name: Kubernetes workde node configuration
  become: yes
  user: ubuntu
  tags: worker2
```

```
  vars:
    ansible_ssh_private_key_file: "../mykey"
    worker_hostname: "worker2"
  tasks:
  - name: Run common tasks
    import_tasks: common.yaml
  - name: Configre k8s worker node
    import_tasks: worker.yaml

- hosts: kubernetes_worker3
  name: Kubernetes workde node configuration
  become: yes
  user: ubuntu
  tags: worker3
  vars:
    ansible_ssh_private_key_file: "../mykey"
    worker_hostname: "worker3"
  tasks:
  - name: Run common tasks
    import_tasks: common.yaml
  - name: Configre k8s worker node
    import_tasks: worker.yaml

- hosts: kubernetes_worker4
  name: Kubernetes workde node configuration
  become: yes
  user: ubuntu
  tags: worker4
  vars:
    ansible_ssh_private_key_file: "../mykey"
    worker_hostname: "worker4"
  tasks:
  - name: Run common tasks
    import_tasks: common.yaml
  - name: Configre k8s worker node
    import_tasks: worker.yaml
```

##common.yaml

```
---
 - name: Add an apt signing key for Kubernetes
   apt_key:
     url: https://packages.cloud.google.com/apt/doc/apt-key.gpg
     state: present

 - name: Adding apt repository for Kubernetes
   apt_repository:
     repo: deb https://apt.kubernetes.io/ kubernetes-xenial main
     state: present
     filename: kubernetes.list
```

##master.yaml

```
---
 - name: installing packages on k8s master
   apt:
     name: "{{ packages }}"
     state: present
     update_cache: yes
   vars:
     packages:
     - apt-transport-https
     - ca-certificates
     - curl
     - gnupg-agent
     - software-properties-common
     - kubelet
     - kubeadm
     - kubectl
     - docker.io
     - vim
     - net-tools
     - unzip

 - name: Update master hostname
   hostname:
     name: control-plane

 - name: Modify kubeadm config to match with docker info cgroups
   lineinfile:
     path: /etc/systemd/system/kubelet.service.d/10-kubeadm.conf
     regexp: '^Environment="KUBELET_KUBECONFIG_ARGS='
     line: Environment="KUBELET_KUBECONFIG_ARGS=--bootstrap-kubeconfig=/etc/kubernetes/bootstrap-kubelet.conf --kubeconfig=/etc/kubernetes/kubelet.conf --cgroup-driver=cgroupfs"

 - name: Restart kubelet
   service:
     name: kubelet
```

```
    daemon_reload: yes
    state: restarted

- name: Reset kubeadm
  command: kubeadm reset -f

- name: Initialize control plane master
  command: kubeadm init --node-name control-plane --ignore-preflight-errors=Mem --ignore-preflight-errors=NumCPU

- name: Setup kubeconfig for root user
  command: "{{ item }}"
  with_items:
   - mkdir -p /root/.kube
   - cp -i /etc/kubernetes/admin.conf /root/.kube/config
   - chown root:root /root/.kube/config

- name: Install calico pod network
  command: kubectl apply -f https://docs.projectcalico.org/manifests/calico.yaml

- name: Generate join command
  command: kubeadm token create --print-join-command
  register: join_command

- name: Copy join command to local file
  local_action: copy content="{{ join_command.stdout_lines[0] }}" dest="./join-command"

- name: Install metric server
  command: kubectl apply -f https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/components.yaml

- name: Modify kubeadm config to match with docker info cgroups
  lineinfile:
    path: /etc/systemd/system/kubelet.service.d/10-kubeadm.conf
    regexp: '^Environment="KUBELET_KUBECONFIG_ARGS='
    line: Environment="KUBELET_KUBECONFIG_ARGS=--bootstrap-kubeconfig=/etc/kubernetes/bootstrap-kubelet.conf --kubeconfig=/etc/kubernetes/kubelet.conf --cgroup-driver=cgroupfs"
```

```
- name: Copy code to master
  copy: src=two_tier_app_k8.tgz dest=/home/ubuntu/two_tier_app_k8.tgz mode=0644
```

##worker.yaml

```
---
  - name: installing packages on k8s worker
    apt:
      name: "{{ packages }}"
      state: present
      update_cache: yes
    vars:
      packages:
      - apt-transport-https
      - ca-certificates
      - curl
      - gnupg-agent
      - software-properties-common
      - kubelet
      - kubeadm
      - docker.io
      - vim
      - net-tools
  - name: update hostname
    hostname:
      name: "{{ worker_hostname }}"


  - name: Modify kubeadm config to match with docker info cgroups
    lineinfile:
      path: /etc/systemd/system/kubelet.service.d/10-kubeadm.conf
      regexp: '^Environment="KUBELET_KUBECONFIG_ARGS='
      line: Environment="KUBELET_KUBECONFIG_ARGS=--bootstrap-kubeconfig=/etc/kubernetes/bootstrap-kubelet.conf --kubeconfig=/etc/kubernetes/kubelet.conf --cgroup-driver=cgroupfs"


  - name: Restart kubelet
    service:
      name: kubelet
      daemon_reload: yes
      state: restarted
```

```
- name: Copy the join command to server location
  copy: src=join-command dest=/tmp/join-command.sh mode=0777

- name: Reset kubeadm
  command: kubeadm reset -f

- name: Join the node to cluster
  command: sh /tmp/join-command.sh
```

#Application Code

#Frontend Webapp

##redis.networkpolicy.yaml

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: redis
spec:
  podSelector:
    matchLabels:
      app: redis
  policyTypes:
  - Ingress
  ingress:
  - from:
    - podSelector:
        matchLabels:
          app: webapp
    ports:
    - protocol: TCP
      port: 6379
```

##redis-primary.deployment.yml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: redis-primary
spec:
  replicas: 1
  selector:
    matchLabels:
      app: redis
      role: primary
      tier: backend
  template:
    metadata:
      labels:
        app: redis
        role: primary
        tier: backend
    spec:
      containers:
      - name: redis
        image: gcr.io/google_containers/redis:e2e  # or just image: redis
        resources:
          requests:
            cpu: 100m
            memory: 100Mi
        ports:
        - containerPort: 6379
```

##redis-primary.service.yml

```
apiVersion: v1
kind: Service
metadata:
  name: redis-primary
  labels:
    app: redis
    role: primary
    tier: backend
spec:
  ports:
  - port: 6379
    targetPort: 6379
  selector:
    app: redis
    role: primary
    tier: backend
```

##redis-replica.deployment.yml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: redis-replica
spec:
  replicas: 1
  selector:
    matchLabels:
      app: redis
      role: replica
      tier: backend
  template:
    metadata:
      labels:
        app: redis
        role: replica
        tier: backend
    spec:
      containers:
      - name: replica
        image: gcr.io/google_samples/gb-redisslave:v2
        resources:
          requests:
            cpu: 100m
            memory: 100Mi
        env:
        - name: GET_HOSTS_FROM
          value: env
        - name: REDIS_MASTER_SERVICE_HOST
          value: redis-primary
        ports:
        - containerPort: 6379
```

##redis-replica.horizontal_pod_autoscaler.yml

```
apiVersion: autoscaling/v1
kind: HorizontalPodAutoscaler
metadata:
  name: redis-replica
spec:
  minReplicas: 3
  maxReplicas: 5
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: redis-replica
  targetCPUUtilizationPercentage: 20
```

##redis-replica.service.yml

```
apiVersion: v1
kind: Service
metadata:
  name: redis-replica
  labels:
    app: redis
    role: replica
    tier: backend
spec:
  ports:
  - port: 6379
  selector:
    app: redis
    role: replica
    tier: backend
```

#Application Database

##app.configmap.yml

apiVersion: v1

kind: ConfigMap

metadata:

  name: webapp

data:

  app.dependency.url: 'https://test.dependency.foo.bar/api/v1/'

  app.dependency.require_tls: truex

##app.deployment.yml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  annotations:
    flux.weave.works/automated: "true"
  name: webapp
spec:
  selector:
    matchLabels:
      app: webapp
  template:
    metadata:
      labels:
        app: webapp
    spec:
      containers:
        - name: webapp
          image: quay.io/fairwinds/k8s-workshop:latest
          command: ["ruby", "app/app.rb"]
          imagePullPolicy: IfNotPresent
          livenessProbe:
            failureThreshold: 3
            httpGet:
              path: /
              port: 8080
              scheme: HTTP
            initialDelaySeconds: 3
            periodSeconds: 3
            successThreshold: 1
            timeoutSeconds: 1
          readinessProbe:
            failureThreshold: 1
            httpGet:
              path: /
```

```
    port: 8080
    scheme: HTTP
ports:
- containerPort: 8080
  name: http
volumeMounts:
  - name: secrets
    mountPath: "/etc/secrets"
env:
  - name: REDIS_HOST
    value: 'redis-primary'
  - name: REDIS_PORT
    value: '6379'
  # - name: CHAOS
  #   value: true
  - name: SECRET1
    valueFrom:
      secretKeyRef:
        name: webapp
        key: val1
  - name: DEPENDENCY_URL
    valueFrom:
      configMapKeyRef:
        name: webapp
        key: app.dependency.url
  - name: DEPENDENCY_REQUIRE_TLS
    valueFrom:
      configMapKeyRef:
        name: webapp
        key: app.dependency.require_tls
resources:
  limits:
    cpu: 100m
    memory: 300Mi
  requests:
    cpu: 100m
```

```
        memory: 300Mi
volumes:
  - name: secrets
    secret:
      secretName: webapp
      items:
      - key: val2
        path: secret_file
        mode: 511
```

##app.horizontal_pod_autoscaler.yml

```
apiVersion: autoscaling/v1
kind: HorizontalPodAutoscaler
metadata:
  name: webapp
spec:
  minReplicas: 10
  maxReplicas: 20
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: webapp
  targetCPUUtilizationPercentage: 30
```

##app.secret.yml

apiVersion: v1

kind: Secret

metadata:

  name: webapp

type: Opaque

data:

  val1: aXRfaXNfYV9zZWNyZXRfdG9fZXZlcnlib2R5

  val2:
ZW5lbWllcz1hbGllbnMKbGl2ZXM9MwplbmVtaWVzLmNoZWF0PXRydWUKZW5lbWllcy5jaGVhdC5sZXZlbD1ub0dvb2RSb3R0ZW4Kc2VjcmV0LmNvZGUucGFzc3BocmFzZT1VVURETFJMUkJBQkFTCnNlY3JldC5jb2RlLmFsbG93ZWQ9dHJ1ZQpzZWNyZXQuY29kZS5saXZlcz0zMAoK

##app.service.yml

```
---
apiVersion: v1
kind: Service
metadata:
  name: webapp
  labels:
    app: webapp
spec:
  type: NodePort
  ports:
    - name: http
      protocol: TCP
      port: 80
      targetPort: http
      nodePort: 30007
  selector:
    app: webapp
```

#User Role and Role Binding

##csr.yaml

apiVersion: certificates.k8s.io/v1

kind: CertificateSigningRequest

metadata:

  name: csr-for-gk

spec:

  groups:

  - system:authenticated

  usages:

  - digital signature

  - key encipherment

  - client auth

  signerName: kubernetes.io/kube-apiserver-client

  request:
LS0tLS1CRUdJTiBDRVJUSUZJQ0FURSBSRVFVRVNULS0tLS0KTUlJQ1VqQ0NBVG9DQVFBd0RURUxNQWtH
QTFVRUF3d0NaMnN3Z2dFaU1BMEdDU3FHU0liM0RRRUJBUVVBQTRJQgpEd0F3Z2dFS0FvSUJBUURBOE9x
UFhqakt5QkZBQVFEVEhuUWVoNVZERXFsSjBhVCtwVUV6dnZLSkh6NTNleFJvCjJTb2lCY2lzQ1dsTWZwNmN
CbHF6WFQ0MnUwSU03ZFlzUTB0T3liU1hrV2ZScThhWHE0NXVrV3VjTWhBNkN1VFgKNTJhUzZONGxJZG5
MSTYrdzdaanpsOUw5bVNqSGRTL3l2b2hiR0g5RGpEaFpEODNhbDVQR3kzRFQ1a00ySUtQUgp4KzJMQW1GTnd
YNytMNzlaU0tPeUpnQmdySnFJS0ZLb1JxcTBaUGZGR0J0TVhTMk9JbGVwd1RPMHZoUVNOaDluCkJYSDNLZE
p1OUxCR0cyaC82empISzg0SHNXSUhyaGVOWDd1Y1dPYXRSS01nREkxajU3Y0NMbllDSTJoVUkwYlEKUXliVl
l5Qi9lUXU4MjErTnhlditObjNMdEVEcmhuYmNaZXM3QWdNQkFBR2dBREFOQmdrcWhraUc5dzBCQVFzRgpBQ
U9DQVFFQWFwaE1IbHdPWTdJTUNwcmM1R1JPN0lDZWJ1dnF1QjhlYmRCcDVXT3FuQUwrU0pRUUREL0Fve
Ep1CjJiamkrVzJrZUFWdXVrc2pGYmROVld2ZmVzZkxYUkM5bEhiVE9nVGswU0EybC9oRXJWZHNwYzBHcmp
3dWtFMUIKNkZYdkZKNWZWOVZYYnpHREozRWc5SWp5R3RFWHQrZ01DOGZnbFhaUFVYTm91ZEQ0c1N3
Y2hVeEFGdDBldWpHeApmS2o3bDZBL2k4QjZDRGEvNlpGWGVMZ0RwL01mbGNqc2pUTFRQTWdxTk1jbGg0b
TlSMFRIZ1YwQnQ4dVpFQmlMCm9heUpvUmxkdUlVQzQzb0pEclhaSlg0UUZpRGI1U0t3b1g0RFBDR2lEZDlzSV
JwNjNaWHZWNjhBY1BMb0lSNFQKV25haXVBSFpVb24veFVnRHgrYk5MNmRtSjFlK1JBPT0KLS0tLS1FTkQgQ
0VSVElGSUNBVEUgUkVRUVTVC0tLS0tCg==

##devrole.yaml

```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: developer
rules:
- apiGroups: [""]
  resources: ["pods"]
  verbs: ["create","update","get","list","delete"]
- apiGroups: [""]
  resources: ["secrets"]
  verbs: ["get","list","create"]
```

##rolebind.yaml

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: gk-developer-binding
subjects:
- kind: User
  name: gk # "name" is case sensitive
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: Role
  name: developer
  apiGroup: rbac.authorization.k8s.io
```

#Script to deploy all yaml files

##deploy_app_metric_userrole.sh

#!/bin/bash

kubectl apply -f metric_components.yaml

kubectl apply -f 01_redis/

kubectl apply -f 02_webapp/

kubectl apply -f user_role/