Probelm Statement

Background of the problem statement:

A popular payment application,**EasyPay** where users add money to their wallet accounts, faces an issue in its payment success rate. The timeout that occurs with the connectivity of the database has been the reason for the issue.

While troubleshooting, it is found that the database server has several downtime instances at irregular intervals. This situation compels the company to create their own infrastructure that runs in high-availability mode.

Given that online shopping experiences continue to evolve as per customer expectations, the developers are driven to make their app more reliable, fast, and secure for improving the performance of the current system.

Implementation requirements:

1. Create the cluster (EC2 instances with load balancer and elastic IP in case of AWS)
2. Automate the provisioning of an EC2 instance using Ansible or Chef Puppet
3. Install Docker and Kubernetes on the cluster
4. Implement the network policies at the database pod to allow ingress traffic from the front-end application pod
5. Create a new user with permissions to create, list, get, update, and delete pods
6. Configure application on the pod
7. Take snapshot of ETCD database
8. Set criteria such that if the memory of CPU goes beyond 50%, environments automatically get scaled up and configured

The following tools must be used:

1. EC2
2. Kubernetes
3. Docker
4. Ansible or Chef or Puppet

The following things to be kept in check:

1. You need to document the steps and write the algorithms in them.
2. The submission of your GitHub repository link is mandatory. In order to track your tasks, you need to share the link of the repository.
3. Document the step-by-step process starting from creating test cases, then executing them, and recording the results.
4. You need to submit the final specification document, which includes:

•Project and tester details
•Concepts used in the project
•Links to the GitHub repository to verify the project completion
•Your conclusion on enhancing the application and defining the USPs (Unique Selling Points)

Final Specification Details

Project and tester details:-

• Project Name : Infra Optimization

• Developer and Tester Name : Ganesh Kale

Concepts used in the project:-

• Terraform IAC,Ansible CM,AWS Webconsole,EC2,Nodeport,LoadBalancer,Kubernetes self managed cluster,Docker Container Engine,HPA,AWS Target Group,Web Application load testing,AWS Security Groups Ingress, Network Policy Pod level ingress, VPC,AWS AMI

Links to the GitHub repository to verify the project completion:-

• https://github.com/ganeshjkale/terraform_aws

Conclusion on enhancing the Project and defining the USPs (Unique Selling Points):-

• AS load is genererate frontend application is autoscaled with database pod also autoscale as defined in deployment yaml files.

•Infra provisioning ,Installing Packages ,Configuting self managed kubernetes cluster with 3 nodes requires only nearly 10 minutes and only two commands are required which can also be automated through script file.

•No manual intervention is needed for application deployment.

•Loadbalancer configuration with Target Group can also be automated using Terraform as further enhancement

•Self managed kubernetes cluster with NodePort Service and LoadBalancer on it, High Availability of application service is achieved through HPA (Horizontal Pod Autoscaler).

•Project is developed with Maximum Automation and HA as Goal.

Solution and Project Outline

Pre-Requisite:-

1:-Ubuntu-20.04_x64 with terraform and Ansible installed
2:-High Speed Internet (50-100Mbps)
3:-Simplilearn AWS Lab with Credentials to be used in Terraform
4:-Aws Web console to visually verify Terraform IAC working
5:-Gnome Terminal to create and execute code

#project is completed in 4 stages as below :-

Stage-A

Infra Provisioning with Terraform and Coniguration management with Ansible.

1:-Create ssh keypair on local machine
2:-Write terrafom plan to create and launch 3 aws ec2 instance with type t3.medium ,
Write a vars.tf file which will have all variables required by instance.tf (such as aws secrets, Region and custom variables)

Refer below table for more details about resources used in terraform plan.

AWS Resources Name	Details
resource "aws_instance" "kubernetes_master"	Instance count : 1 Kubernetes Control Plane Node. This is required for customized resource such as tags,AMI,count,security group,instance type
resource "aws_instance" "kubernetes_worker"	Instance count : 2 Kubernetes Worker Node. This is required for customized resource such as tags,AMI,count,security group,instance type
resource "aws_security_group" "k8s"	Created 3 ingress and 1 egress. Ingress : 22,80 for outside VPC Ingress: -1 for within VPC Egress: ALL Traffic allowed over inernet and VPC Port 80 will be used to access application exposed through AWS ALB (Application load balancer HTTP traffic)
resource "aws_key_pair" "mykey"	Use precreated sshkeypair for taking ssh of ec2 instance. Keys are uploaded to AWS cloud.
resource "local_file" "inventory"	This will create Local Ansible inventory file with tags and ec2 public dns entry to be used by ansible-playbook -i <inventroy.ini>.
resource "local_file" "host_script"	To create a script with ssh-keyscan -H to scan ec2 instance public dns keys and add it in ssh known_hosts file, so that ansible-playbook can run without interruptions
resource "null_resource" "add_host_entry"	Run locally created script to add known host entry . This resource depends on resource "local_file" "host_script" . For that added triggers = { order = local_file.host_script.id}. Used provisioner “local-exec” to execute script file.

3:-Run Terraform command as below :-
“terraform init” This will download required plugins based of resources used in terraform files.
“terraform plan -out k8s.zip” This will save plan output to k8s.zip
“terraform apply k8s.zip” This will apply plan and start provisioning
4:-Now write ansible playbooks for **self managed kubernetes cluster.**

Devops Infra Optimization Capstone Project

Details of each ansible yaml file are as below :-

common.yaml : common task used for master and worker kubernetes task

main.yaml : main file to be provided to ansible-playbook ,includes hosts name ,tasks,vars,tags customized for kubernetes master and worker nodes configuration.

master.yaml : kubernetes master node tasks such as installing packages , intializing kubeadm ,create prinjoin command file, and configuring kubernetes clustes master node setup completely with hostname.

worker.yaml : kubernetes worker nodes tasks such as installing packages, copying join command file , joining node to master control plane, restarting kubelet, configuring kubeadm.conf file for cgroup driver,modifying hostname.

5:-Using mykey from step 1 and inventory.ini file created by terraform in steps 3 ,
Run ansible playbook imperative command as below :

ansible-playbook -i inventory.ini main.yaml

Note: mykey ssh key path is provided in vars of main.yaml

4:-Test and verify configuration by visiting AWS web console and taking ssh of kubernetes master node using public dns name and mykey.

Eg:-

```
cat inventory.ini
ssh -i mykey  ubuntu@ec2-54-84-226-118.compute-1.amazonaws.com
sudo su
alias k=kubectl
k get nodes -o wide
```

you should get output of one master node and two worker node.

Note:-

Username for ubuntu OS is “ubuntu”

Username for amazon OS is “ec2-user”

5:- Refer Terminal Output in **Annexure A**

Stage-B

Write Yaml files of applications,user role, role binding, HPA for deployments,services and other task.

1:-Write Application Frontend and Database Yaml files.
2:-Using Network policy ingress and application pod label selector only frontend application pods will able to communicate with database pods.
3:-Forntend application deployment will have auto scalling capability **HPA from 10 pods to maximum 20 pods when CPU load is greater than 30%** (set has 30% for now can be increased as required, due to resource limitation have done this).
4:-Database redis replicas deployment will have auto scalling capability **HPA from 3 pods to 5 pods when CPU load is greater than 20%**
4:-We will require metric server deployment to track pod resources utilization and HPA to work.
Download metric server YAML file from wget
https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/components.yaml
Modify metric server yaml file to include additional argument under args "- --kubelet-insecure-tls"
Rename components.yaml to metric_components.yaml
5:-Created a script to deploy metric server , applications frontend and database,user role for role based access control and role binding .

deploy_app_metric_userrole.sh

6:-Create directory structure for all above components with script added and make a **two_tier_app_k8.tgz** file which will be automatically copied to kubernetes master node through **ansible playbook in STAGE-A step 5**
7:-With completion of stage-A we are all ready to deploy our applications and other realted components to kubernetes cluster.
Take ssh of kubernetes master node and run "sudo su",
Goto user home path and see **two_tier_app_k8.tgz file**

Extract it "tar -xvf two_tier_app_k8.tgz"
"cd two_tier_app_k8"
"./deploy_app_metric_userrole.sh"

8:-Verify deployments , pods , services, HPA with using kubernetes-admin@kubernetes context with followup below command.

Kubectl get po,deploy,nodes,svc,hpa

Application Frontend service is exposed on NodePort 30007 and will be utilized by AWS ALB-Application LoadBalancer

9:-Create and Verify user role, role binding ,user context switching, user permission to get,list,update,create,delete pods in default namespace.

Preconfigured steps :-

```
openssl genrsa -out gk.key 2048
openssl req -new -key gk.key -subj "/CN=gk" -out gk.csr
```

cat gk.csr |base64 -w 0

Automated Steps in yaml file :-

Yaml file to generate certificate signing request (csr-for-gk) with gk.csr encoded with base64 and added in csr.yaml file.
Yaml file for developer role of get,list,update,create,delete pods only
Yaml file for role binding “gk” user with developer role

Manual Steps imprerative way :-

```
cd /home/ubuntu/two_tier_app_k8/user_role
kubectl get csr
kubectl certificate approve csr-for-gk
kubectl get csr csr-for-gk -o jsonpath='{.status.certificate}' | base64 --decode > gk.cer
kubectl config set-credentials gk –client-key /home/ubuntu/two_tier_app_k8/user_role/gk.key  --client-certificate /home/ubuntu/two_tier_app_k8/user_role/gk.cer
kubectl config set-context gk@kubernetes --cluster kubernetes --user gk
kubectl config view
kubectl config use-context gk@kubernetes
```

Test Authorization :-

```
kubectl auth can-i --as gk get pods
kubectl auth can-i --as gk delete pods
kubectl auth can-i --as gk create pods
kubectl auth can-i --as gk update pods
kubectl auth can-i --as gk list pods
```

10:- Refer Terminal output in **Annexure B**

Stage-C

ETCD Snapshot and Setup ALB (Application loadbalancer on AWS with Target group)

#ETCD Backup :-

```
root@control-plane:/home/ubuntu/two_tier_app_k8/etcd_backup# apt install etcd-client
root@control-plane:/home/ubuntu/two_tier_app_k8/etcd_backup# ETCDCTL_API=3 etcdctl snapshot save snapshot.db \
> --endpoints=https://127.0.0.1:2379 \
> --cacert=/etc/kubernetes/pki/etcd/ca.crt \
> --cert=/etc/kubernetes/pki/etcd/server.crt \
> --key=/etc/kubernetes/pki/etcd/server.key
Snapshot saved at snapshot.db
root@control-plane:/home/ubuntu/two_tier_app_k8/etcd_backup# ls
snapshot.db
root@control-plane:/home/ubuntu/two_tier_app_k8/etcd_backup# ETCDCTL_API=3 \
> etcdctl --write-out=table snapshot status snapshot.db
+----------+----------+------------+------------+
|   HASH   | REVISION | TOTAL KEYS | TOTAL SIZE |
+----------+----------+------------+------------+
| 89e301f3 |     8999 |       1377 |     4.3 MB |
+----------+----------+------------+------------+
```

#Setup ALB

1:-Goto AWS webconsole and create Target Group of instances

EC2 >> Target groups >> Create target group >> Instances (Protocol : Http , Port : 30007) >> Target Group Name (httpALB) >> Next >> Register targets (select instances with Port :30007 for routing traffic) >> Include as pending below >> Create target group

2:-Now create LoadBalancer as below : -

Load Balancing >> Load Balancers >> Create Load Balancer >> Application Load Balancer >> Create >> Load Balancer Name (myalb) >>Scheme (Internet Facing) >>Mapping (Select us-east-1a and us-east-1d) >> Security Group (Named Port 22) >> Listeners and Routing >> Listener (Http:80) with Default Action (Forward to httpALB) >> Create load balancer >> View load balancer

3:-Now goto Target group and check ec2 instaces are intialized and healthy (It will take nearly 5minutes)

4:-Now goto Load Balancer and copy DNS name and use in new tab of browser and check frontend application is reachable or not.

5:- Refer screenshots in **Annexure C**

Stage-D

Test ALB with ab tool and verify HPA (Horizontal Pod Autoscaling)

1:-We will use Apache ab loadtesting command line tool on local machine to generate load on frontend application which is exposed to internet on port 80 via LoadBalancer .
2:-Frontend webapp deployment will autoscale from 10 pods to max 20 pods when cpu of target pads reaches 30% and cpu limit mentioned in deployment yaml file 100m exceed.
#Command to generate load .

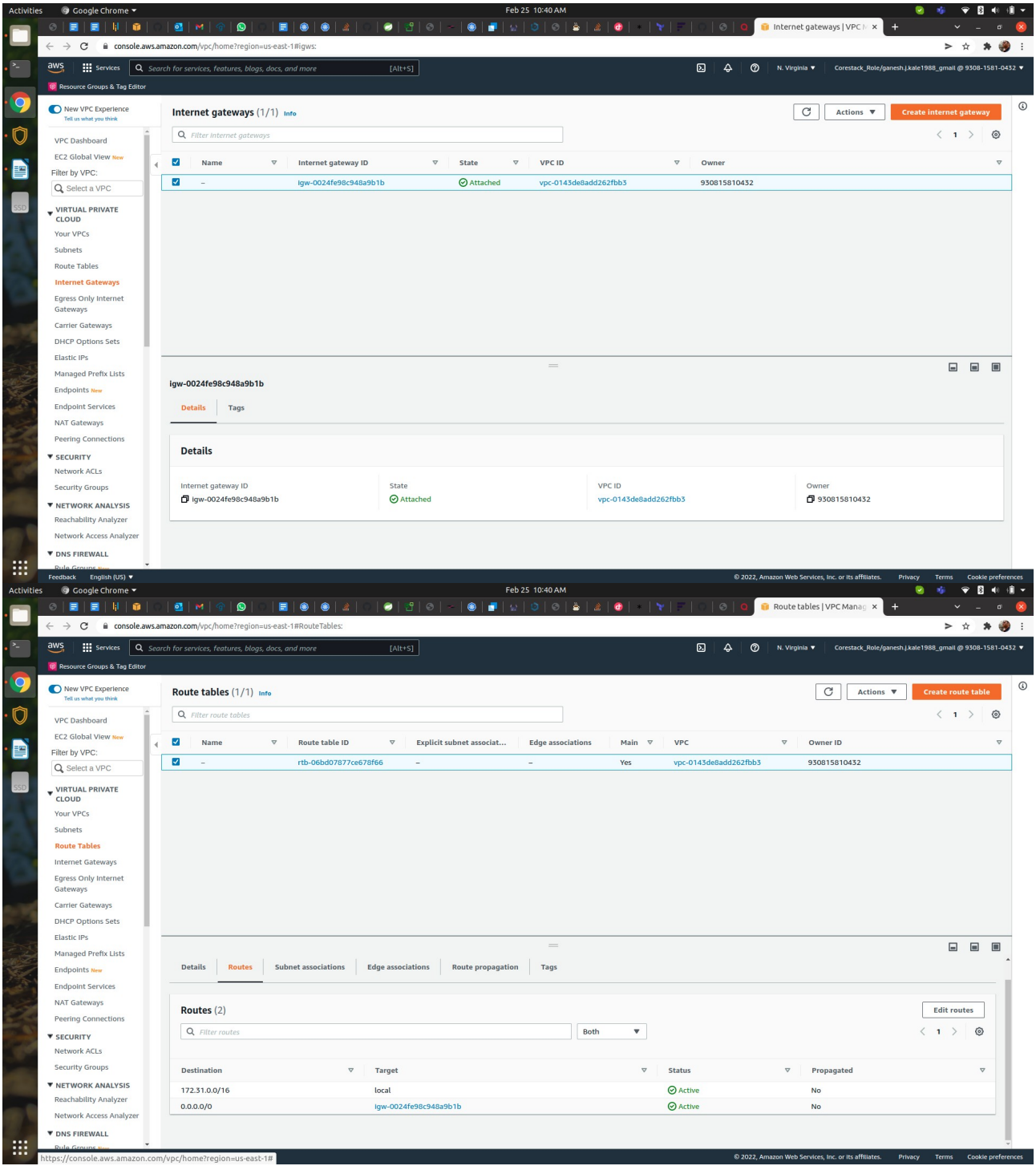
```
sudo apt install apache2-utils
ab -n 50000 -c 500 http://myalb-1047253744.us-east-1.elb.amazonaws.com/
```

3:-Refer Terminal Output Screenshots in **Annexure D**

Additional Stuff

#Reverse SCP from ec2 to local machine :-
scp -i ../mykey ubuntu@ec2-44-193-197-119.compute-1.amazonaws.com:/home/ubuntu/kubernetes_project.tgz /home/gk/kubernetes_project.tgz

#EC2 instance not reachable from local machine solution:-



Annexure A

#Stage-A Output

#Terraform Output :-

##################Terraform Started

```
root@gk-ThinkPad-E15-Gen-2:/home/gk/terraform_project/devops_capstone_project# terraform plan -out
^C
root@gk-ThinkPad-E15-Gen-2:/home/gk/terraform_project/devops_capstone_project# vi vars.tf
root@gk-ThinkPad-E15-Gen-2:/home/gk/terraform_project/devops_capstone_project# terraform plan -out
k8s.zip
aws_key_pair.mykey: Refreshing state... [id=mykey]
aws_security_group.k8s: Refreshing state... [id=sg-0050288bf137298f5]
aws_instance.kubernetes_worker[1]: Refreshing state... [id=i-06899f8284ba73761]
aws_instance.kubernetes_master: Refreshing state... [id=i-0af40014020493f40]
aws_instance.kubernetes_worker[0]: Refreshing state... [id=i-078043d5d9f5e1619]
local_file.host_script: Refreshing state... [id=4017cb94b49b8f4f81d7d0a474146da7bac4c56c]
local_file.inventory: Refreshing state... [id=bbbb5ef91b2e6c9799fbc8321592d38e33220bd6]
null_resource.add_host_entry: Refreshing state... [id=7046571710382423961]

Note: Objects have changed outside of Terraform

Terraform detected the following changes made outside of Terraform since the last "terraform apply":

  # aws_instance.kubernetes_master has been deleted
  - resource "aws_instance" "kubernetes_master" {
      - ami                                  = "ami-04505e74c0741db8d" -> null
      - arn                                  = "arn:aws:ec2:us-east-1:930815810432:instance/i-0af40014020493f40" ->
null
      - associate_public_ip_address          = true -> null
      - availability_zone                    = "us-east-1a" -> null
      - cpu_core_count                       = 1 -> null
      - cpu_threads_per_core                 = 2 -> null
      - disable_api_termination              = false -> null
      - ebs_optimized                        = false -> null
      - get_password_data                    = false -> null
      - hibernation                          = false -> null
      - id                                   = "i-0af40014020493f40" -> null
      - instance_initiated_shutdown_behavior = "stop" -> null
      - instance_state                       = "running" -> null
      - instance_type                        = "t3.medium" -> null
      - ipv6_address_count                   = 0 -> null
      - ipv6_addresses                       = [] -> null
      - key_name                             = "mykey" -> null
      - monitoring                           = false -> null
      - primary_network_interface_id         = "eni-0b4d46d2f33c7fe13" -> null
      - private_dns                          = "ip-172-31-90-184.ec2.internal" -> null
      - private_ip                           = "172.31.90.184" -> null
      - public_dns                           = "ec2-54-84-226-118.compute-1.amazonaws.com" -> null
      - public_ip                            = "54.84.226.118" -> null
      - secondary_private_ips                = [] -> null
      - security_groups                      = [
          - "Ports 22",
        ] -> null
      - source_dest_check                    = true -> null
      - subnet_id                            = "subnet-0248d9d300b0980f7" -> null
      - tags                                 = {
          - "Name" = "kubernetes_master"
        } -> null
      - tags_all                             = {
          - "Name" = "kubernetes_master"
        } -> null
      - tenancy                              = "default" -> null
      - vpc_security_group_ids               = [
```

```
      - "sg-0050288bf137298f5",
    ] -> null

  - capacity_reservation_specification {
      - capacity_reservation_preference = "open" -> null
    }

  - credit_specification {
      - cpu_credits = "unlimited" -> null
    }

  - enclave_options {
      - enabled = false -> null
    }

  - metadata_options {
      - http_endpoint               = "enabled" -> null
      - http_put_response_hop_limit = 1 -> null
      - http_tokens                 = "optional" -> null
      - instance_metadata_tags      = "disabled" -> null
    }

  - root_block_device {
      - delete_on_termination = true -> null
      - device_name           = "/dev/sda1" -> null
      - encrypted             = false -> null
      - iops                  = 100 -> null
      - tags                  = {} -> null
      - throughput            = 0 -> null
      - volume_id             = "vol-06841181b1f41ae64" -> null
      - volume_size           = 8 -> null
      - volume_type           = "gp2" -> null
    }
  }

  # aws_instance.kubernetes_worker[0] has been deleted
  - resource "aws_instance" "kubernetes_worker" {
      - ami                                  = "ami-04505e74c0741db8d" -> null
      - arn                                  = "arn:aws:ec2:us-east-1:930815810432:instance/i-078043d5d9f5e1619" -
> null
      - associate_public_ip_address          = true -> null
      - availability_zone                    = "us-east-1a" -> null
      - cpu_core_count                       = 1 -> null
      - cpu_threads_per_core                 = 2 -> null
      - disable_api_termination              = false -> null
      - ebs_optimized                        = false -> null
      - get_password_data                    = false -> null
      - hibernation                          = false -> null
      - id                                   = "i-078043d5d9f5e1619" -> null
      - instance_initiated_shutdown_behavior = "stop" -> null
      - instance_state                       = "running" -> null
      - instance_type                        = "t3.medium" -> null
      - ipv6_address_count                   = 0 -> null
      - ipv6_addresses                       = [] -> null
      - key_name                             = "mykey" -> null
      - monitoring                           = false -> null
      - primary_network_interface_id         = "eni-0c2d5f5cca4a2878d" -> null
      - private_dns                          = "ip-172-31-89-72.ec2.internal" -> null
      - private_ip                           = "172.31.89.72" -> null
      - public_dns                           = "ec2-54-91-147-95.compute-1.amazonaws.com" -> null
      - public_ip                            = "54.91.147.95" -> null
      - secondary_private_ips                = [] -> null
      - security_groups                      = [
          - "Ports 22",
        ] -> null
      - source_dest_check                    = true -> null
```

```
    - subnet_id                     = "subnet-0248d9d300b0980f7" -> null
    - tags                          = {
        - "Name" = "kubernetes_worker-0"
      } -> null
    - tags_all                      = {
        - "Name" = "kubernetes_worker-0"
      } -> null
    - tenancy                       = "default" -> null
    - vpc_security_group_ids        = [
        - "sg-0050288bf137298f5",
      ] -> null

    - capacity_reservation_specification {
        - capacity_reservation_preference = "open" -> null
      }

    - credit_specification {
        - cpu_credits = "unlimited" -> null
      }

    - enclave_options {
        - enabled = false -> null
      }

    - metadata_options {
        - http_endpoint               = "enabled" -> null
        - http_put_response_hop_limit = 1 -> null
        - http_tokens                 = "optional" -> null
        - instance_metadata_tags      = "disabled" -> null
      }

    - root_block_device {
        - delete_on_termination = true -> null
        - device_name           = "/dev/sda1" -> null
        - encrypted             = false -> null
        - iops                  = 100 -> null
        - tags                  = {} -> null
        - throughput            = 0 -> null
        - volume_id             = "vol-04b13e68a1ec15c3c" -> null
        - volume_size           = 8 -> null
        - volume_type           = "gp2" -> null
      }
  }

  # aws_instance.kubernetes_worker[1] has been deleted
  - resource "aws_instance" "kubernetes_worker" {
      - ami                                  = "ami-04505e74c0741db8d" -> null
      - arn                                  = "arn:aws:ec2:us-east-1:930815810432:instance/i-06899f8284ba73761" -
> null
      - associate_public_ip_address          = true -> null
      - availability_zone                    = "us-east-1a" -> null
      - cpu_core_count                       = 1 -> null
      - cpu_threads_per_core                 = 2 -> null
      - disable_api_termination              = false -> null
      - ebs_optimized                        = false -> null
      - get_password_data                    = false -> null
      - hibernation                          = false -> null
      - id                                   = "i-06899f8284ba73761" -> null
      - instance_initiated_shutdown_behavior = "stop" -> null
      - instance_state                       = "running" -> null
      - instance_type                        = "t3.medium" -> null
      - ipv6_address_count                   = 0 -> null
      - ipv6_addresses                       = [] -> null
      - key_name                             = "mykey" -> null
      - monitoring                           = false -> null
      - primary_network_interface_id         = "eni-04c9be190e2b1510d" -> null
```

```
    - private_dns                        = "ip-172-31-83-211.ec2.internal" -> null
    - private_ip                         = "172.31.83.211" -> null
    - public_dns                         = "ec2-3-82-251-124.compute-1.amazonaws.com" -> null
    - public_ip                          = "3.82.251.124" -> null
    - secondary_private_ips              = [] -> null
    - security_groups                    = [
        - "Ports 22",
      ] -> null
    - source_dest_check                  = true -> null
    - subnet_id                          = "subnet-0248d9d300b0980f7" -> null
    - tags                               = {
        - "Name" = "kubernetes_worker-1"
      } -> null
    - tags_all                           = {
        - "Name" = "kubernetes_worker-1"
      } -> null
    - tenancy                            = "default" -> null
    - vpc_security_group_ids             = [
        - "sg-0050288bf137298f5",
      ] -> null

    - capacity_reservation_specification {
        - capacity_reservation_preference = "open" -> null
      }

    - credit_specification {
        - cpu_credits = "unlimited" -> null
      }

    - enclave_options {
        - enabled = false -> null
      }

    - metadata_options {
        - http_endpoint               = "enabled" -> null
        - http_put_response_hop_limit = 1 -> null
        - http_tokens                 = "optional" -> null
        - instance_metadata_tags      = "disabled" -> null
      }

    - root_block_device {
        - delete_on_termination = true -> null
        - device_name           = "/dev/sda1" -> null
        - encrypted             = false -> null
        - iops                  = 100 -> null
        - tags                  = {} -> null
        - throughput            = 0 -> null
        - volume_id             = "vol-0cd8e8515dee01207" -> null
        - volume_size           = 8 -> null
        - volume_type           = "gp2" -> null
      }
  }

  # aws_key_pair.mykey has been deleted
  - resource "aws_key_pair" "mykey" {
      - arn         = "arn:aws:ec2:us-east-1:930815810432:key-pair/mykey" -> null
      - fingerprint = "02:f5:67:48:c1:54:76:f8:db:8c:f0:51:6a:80:a1:20" -> null
      - id          = "mykey" -> null
      - key_name    = "mykey" -> null
      - key_pair_id = "key-0d5f6ff25515c0c85" -> null
      - public_key  = "ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABgQCehe8o02ZWFNSmviDxwt4KlqJW6HxBp2zhw7BBc5PjJ9nNfr+Cp0i
FRM/S6g7bIUyn5j+4sz+6A51ogCw6rn1+j44pU/
NI9R41zDElBtetaipXZ1CElBedYg4zIdbiCje5YA7dJm80jCSFWe8eKtgPAGuLh5ZFaFUC8WCHwNj2qt7PFojn9ieB
8BoILoQ/
8ODWAgskVNSLejjYmTvotWNg6V5582sqaKJOoPqKts9guG1k4OP8T9mJafif6NX2aJFsd2SYNxm3PlTU9iyg38n
```

```
pVVvg3YxjtFpPXO2mxqOtKCt4eQRDUvzyAJ1dLPxtjva1+Hp+i4YPBooURTXdHNFdbhpUO0ZBPvohqZby+A3k
ULCo1yyKY9q6pJK13aP1svhitPzH74Gb3pqbQ+nI0xf6ekS54PZufF/tysu+A6lagx0BhGDbK1GErvusI3aZG/
OxEbwoLkq2LGwYlw0iygbKZh6QYV7VOLeG+/1zvZ4XkROyd2cSsK7zb/p6I0A8ogE= root@gk-ThinkPad-E15-
Gen-2" -> null
    - tags       = {} -> null
    - tags_all   = {} -> null
  }

 # aws_security_group.k8s has been deleted
 - resource "aws_security_group" "k8s" {
    - arn                    = "arn:aws:ec2:us-east-1:930815810432:security-group/sg-0050288bf137298f5" ->
null
    - description            = "Managed by Terraform" -> null
    - egress                 = [
        - {
            - cidr_blocks      = [
                - "0.0.0.0/0",
              ]
            - description      = ""
            - from_port        = 0
            - ipv6_cidr_blocks = []
            - prefix_list_ids  = []
            - protocol         = "-1"
            - security_groups  = []
            - self             = true
            - to_port          = 0
          },
      ] -> null
    - id                     = "sg-0050288bf137298f5" -> null
    - ingress                = [
        - {
            - cidr_blocks      = [
                - "0.0.0.0/0",
              ]
            - description      = ""
            - from_port        = 22
            - ipv6_cidr_blocks = []
            - prefix_list_ids  = []
            - protocol         = "tcp"
            - security_groups  = []
            - self             = true
            - to_port          = 22
          },
        - {
            - cidr_blocks      = [
                - "0.0.0.0/0",
              ]
            - description      = ""
            - from_port        = 80
            - ipv6_cidr_blocks = []
            - prefix_list_ids  = []
            - protocol         = "tcp"
            - security_groups  = []
            - self             = true
            - to_port          = 80
          },
        - {
            - cidr_blocks      = []
            - description      = ""
            - from_port        = 0
            - ipv6_cidr_blocks = []
            - prefix_list_ids  = []
            - protocol         = "-1"
            - security_groups  = []
            - self             = true
            - to_port          = 0
```

```
        },
      ] -> null
    - name                 = "Ports 22" -> null
    - owner_id             = "930815810432" -> null
    - revoke_rules_on_delete = false -> null
    - tags                 = {
        - "Name" = "k8s"
      } -> null
    - tags_all             = {
        - "Name" = "k8s"
      } -> null
    - vpc_id               = "vpc-03593c082c3c76dc6" -> null
  }
```

Unless you have made equivalent changes to your configuration, or ignored the relevant attributes using ignore_changes, the following plan may include actions to undo or respond to these changes.

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  + create
-/+ destroy and then create replacement

Terraform will perform the following actions:

```
  # aws_instance.kubernetes_master will be created
  + resource "aws_instance" "kubernetes_master" {
      + ami                                  = "ami-04505e74c0741db8d"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + cpu_core_count                       = (known after apply)
      + cpu_threads_per_core                 = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + get_password_data                    = false
      + host_id                              = (known after apply)
      + id                                   = (known after apply)
      + instance_initiated_shutdown_behavior = (known after apply)
      + instance_state                       = (known after apply)
      + instance_type                        = "t3.medium"
      + ipv6_address_count                   = (known after apply)
      + ipv6_addresses                       = (known after apply)
      + key_name                             = "mykey"
      + monitoring                           = (known after apply)
      + outpost_arn                          = (known after apply)
      + password_data                        = (known after apply)
      + placement_group                      = (known after apply)
      + placement_partition_number           = (known after apply)
      + primary_network_interface_id         = (known after apply)
      + private_dns                          = (known after apply)
      + private_ip                           = (known after apply)
      + public_dns                           = (known after apply)
      + public_ip                            = (known after apply)
      + secondary_private_ips                = (known after apply)
      + security_groups                      = (known after apply)
      + source_dest_check                    = true
      + subnet_id                            = (known after apply)
      + tags                                 = {
          + "Name" = "kubernetes_master"
        }
      + tags_all                             = {
```

```
      + "Name" = "kubernetes_master"
    }
  + tenancy                            = (known after apply)
  + user_data                          = (known after apply)
  + user_data_base64                   = (known after apply)
  + vpc_security_group_ids             = (known after apply)

  + capacity_reservation_specification {
      + capacity_reservation_preference = (known after apply)

      + capacity_reservation_target {
          + capacity_reservation_id = (known after apply)
        }
    }

  + ebs_block_device {
      + delete_on_termination = (known after apply)
      + device_name           = (known after apply)
      + encrypted             = (known after apply)
      + iops                  = (known after apply)
      + kms_key_id            = (known after apply)
      + snapshot_id           = (known after apply)
      + tags                  = (known after apply)
      + throughput            = (known after apply)
      + volume_id             = (known after apply)
      + volume_size           = (known after apply)
      + volume_type           = (known after apply)
    }

  + enclave_options {
      + enabled = (known after apply)
    }

  + ephemeral_block_device {
      + device_name  = (known after apply)
      + no_device    = (known after apply)
      + virtual_name = (known after apply)
    }

  + metadata_options {
      + http_endpoint               = (known after apply)
      + http_put_response_hop_limit = (known after apply)
      + http_tokens                 = (known after apply)
      + instance_metadata_tags      = (known after apply)
    }

  + network_interface {
      + delete_on_termination = (known after apply)
      + device_index          = (known after apply)
      + network_interface_id  = (known after apply)
    }

  + root_block_device {
      + delete_on_termination = (known after apply)
      + device_name           = (known after apply)
      + encrypted             = (known after apply)
      + iops                  = (known after apply)
      + kms_key_id            = (known after apply)
      + tags                  = (known after apply)
      + throughput            = (known after apply)
      + volume_id             = (known after apply)
      + volume_size           = (known after apply)
      + volume_type           = (known after apply)
    }
}
```

```
# aws_instance.kubernetes_worker[0] will be created
+ resource "aws_instance" "kubernetes_worker" {
    + ami                                  = "ami-04505e74c0741db8d"
    + arn                                  = (known after apply)
    + associate_public_ip_address          = (known after apply)
    + availability_zone                    = (known after apply)
    + cpu_core_count                       = (known after apply)
    + cpu_threads_per_core                 = (known after apply)
    + disable_api_termination              = (known after apply)
    + ebs_optimized                        = (known after apply)
    + get_password_data                    = false
    + host_id                              = (known after apply)
    + id                                   = (known after apply)
    + instance_initiated_shutdown_behavior = (known after apply)
    + instance_state                       = (known after apply)
    + instance_type                        = "t3.medium"
    + ipv6_address_count                   = (known after apply)
    + ipv6_addresses                       = (known after apply)
    + key_name                             = "mykey"
    + monitoring                           = (known after apply)
    + outpost_arn                          = (known after apply)
    + password_data                        = (known after apply)
    + placement_group                      = (known after apply)
    + placement_partition_number           = (known after apply)
    + primary_network_interface_id         = (known after apply)
    + private_dns                          = (known after apply)
    + private_ip                           = (known after apply)
    + public_dns                           = (known after apply)
    + public_ip                            = (known after apply)
    + secondary_private_ips                = (known after apply)
    + security_groups                      = (known after apply)
    + source_dest_check                    = true
    + subnet_id                            = (known after apply)
    + tags                                 = {
        + "Name" = "kubernetes_worker-0"
      }
    + tags_all                             = {
        + "Name" = "kubernetes_worker-0"
      }
    + tenancy                              = (known after apply)
    + user_data                            = (known after apply)
    + user_data_base64                     = (known after apply)
    + vpc_security_group_ids               = (known after apply)

    + capacity_reservation_specification {
        + capacity_reservation_preference = (known after apply)

        + capacity_reservation_target {
            + capacity_reservation_id = (known after apply)
          }
      }

    + ebs_block_device {
        + delete_on_termination = (known after apply)
        + device_name           = (known after apply)
        + encrypted             = (known after apply)
        + iops                  = (known after apply)
        + kms_key_id            = (known after apply)
        + snapshot_id           = (known after apply)
        + tags                  = (known after apply)
        + throughput            = (known after apply)
        + volume_id             = (known after apply)
        + volume_size           = (known after apply)
        + volume_type           = (known after apply)
      }
```

```
    + enclave_options {
        + enabled = (known after apply)
      }

    + ephemeral_block_device {
        + device_name  = (known after apply)
        + no_device    = (known after apply)
        + virtual_name = (known after apply)
      }

    + metadata_options {
        + http_endpoint               = (known after apply)
        + http_put_response_hop_limit = (known after apply)
        + http_tokens                 = (known after apply)
        + instance_metadata_tags      = (known after apply)
      }

    + network_interface {
        + delete_on_termination = (known after apply)
        + device_index          = (known after apply)
        + network_interface_id  = (known after apply)
      }

    + root_block_device {
        + delete_on_termination = (known after apply)
        + device_name           = (known after apply)
        + encrypted             = (known after apply)
        + iops                  = (known after apply)
        + kms_key_id            = (known after apply)
        + tags                  = (known after apply)
        + throughput            = (known after apply)
        + volume_id             = (known after apply)
        + volume_size           = (known after apply)
        + volume_type           = (known after apply)
      }
  }

# aws_instance.kubernetes_worker[1] will be created
+ resource "aws_instance" "kubernetes_worker" {
    + ami                                  = "ami-04505e74c0741db8d"
    + arn                                  = (known after apply)
    + associate_public_ip_address          = (known after apply)
    + availability_zone                    = (known after apply)
    + cpu_core_count                       = (known after apply)
    + cpu_threads_per_core                 = (known after apply)
    + disable_api_termination              = (known after apply)
    + ebs_optimized                        = (known after apply)
    + get_password_data                    = false
    + host_id                              = (known after apply)
    + id                                   = (known after apply)
    + instance_initiated_shutdown_behavior = (known after apply)
    + instance_state                       = (known after apply)
    + instance_type                        = "t3.medium"
    + ipv6_address_count                   = (known after apply)
    + ipv6_addresses                       = (known after apply)
    + key_name                             = "mykey"
    + monitoring                           = (known after apply)
    + outpost_arn                          = (known after apply)
    + password_data                        = (known after apply)
    + placement_group                      = (known after apply)
    + placement_partition_number           = (known after apply)
    + primary_network_interface_id         = (known after apply)
    + private_dns                          = (known after apply)
    + private_ip                           = (known after apply)
    + public_dns                           = (known after apply)
    + public_ip                            = (known after apply)
```

```
+ secondary_private_ips               = (known after apply)
+ security_groups                     = (known after apply)
+ source_dest_check                   = true
+ subnet_id                           = (known after apply)
+ tags                                = {
    + "Name" = "kubernetes_worker-1"
  }
+ tags_all                            = {
    + "Name" = "kubernetes_worker-1"
  }
+ tenancy                             = (known after apply)
+ user_data                           = (known after apply)
+ user_data_base64                    = (known after apply)
+ vpc_security_group_ids              = (known after apply)

+ capacity_reservation_specification {
    + capacity_reservation_preference = (known after apply)

    + capacity_reservation_target {
        + capacity_reservation_id = (known after apply)
      }
  }

+ ebs_block_device {
    + delete_on_termination = (known after apply)
    + device_name           = (known after apply)
    + encrypted             = (known after apply)
    + iops                  = (known after apply)
    + kms_key_id            = (known after apply)
    + snapshot_id           = (known after apply)
    + tags                  = (known after apply)
    + throughput            = (known after apply)
    + volume_id             = (known after apply)
    + volume_size           = (known after apply)
    + volume_type           = (known after apply)
  }

+ enclave_options {
    + enabled = (known after apply)
  }

+ ephemeral_block_device {
    + device_name  = (known after apply)
    + no_device    = (known after apply)
    + virtual_name = (known after apply)
  }

+ metadata_options {
    + http_endpoint               = (known after apply)
    + http_put_response_hop_limit = (known after apply)
    + http_tokens                 = (known after apply)
    + instance_metadata_tags      = (known after apply)
  }

+ network_interface {
    + delete_on_termination = (known after apply)
    + device_index          = (known after apply)
    + network_interface_id  = (known after apply)
  }

+ root_block_device {
    + delete_on_termination = (known after apply)
    + device_name           = (known after apply)
    + encrypted             = (known after apply)
    + iops                  = (known after apply)
    + kms_key_id            = (known after apply)
```

```
      + tags               = (known after apply)
      + throughput         = (known after apply)
      + volume_id          = (known after apply)
      + volume_size        = (known after apply)
      + volume_type        = (known after apply)
    }
  }

  # aws_key_pair.mykey will be created
  + resource "aws_key_pair" "mykey" {
    + arn            = (known after apply)
    + fingerprint    = (known after apply)
    + id             = (known after apply)
    + key_name       = "mykey"
    + key_name_prefix = (known after apply)
    + key_pair_id    = (known after apply)
    + public_key     = "ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABgQCehe8o02ZWFNSmviDxwt4KlqJW6HxBp2zhw7BBc5PjJ9nNfr+Cp0i
FRM/S6g7bIUyn5j+4sz+6A51ogCw6rn1+j44pU/
NI9R41zDElBtetaipXZ1CElBedYg4zIdbiCje5YA7dJm80jCSFWe8eKtgPAGuLh5ZFaFUC8WCHwNj2qt7PFojn9ieB
8BoILoQ/
8ODWAgskVNSLejjYmTvotWNg6V5582sqaKJOoPqKts9guG1k4OP8T9mJafif6NX2aJFsd2SYNxm3PlTU9iyg38n
pVVvg3YxjtFpPXO2mxqOtKCt4eQRDUvzyAJ1dLPxtjva1+Hp+i4YPBooURTXdHNFdbhpUO0ZBPvohqZby+A3k
ULCo1yyKY9q6pJK13aP1svhitPzH74Gb3pqbQ+nI0xf6ekS54PZufF/tysu+A6lagx0BhGDbK1GErvusI3aZG/
OxEbwoLkq2LGwYlw0iygbKZh6QYV7VOLeG+/1zvZ4XkROyd2cSsK7zb/p6I0A8ogE= root@gk-ThinkPad-E15-
Gen-2"
    + tags_all       = (known after apply)
  }

  # aws_security_group.k8s will be created
  + resource "aws_security_group" "k8s" {
    + arn                    = (known after apply)
    + description            = "Managed by Terraform"
    + egress                 = [
      + {
        + cidr_blocks      = [
          + "0.0.0.0/0",
        ]
        + description      = ""
        + from_port        = 0
        + ipv6_cidr_blocks = []
        + prefix_list_ids  = []
        + protocol         = "-1"
        + security_groups  = []
        + self             = true
        + to_port          = 0
      },
    ]
    + id                     = (known after apply)
    + ingress                = [
      + {
        + cidr_blocks      = [
          + "0.0.0.0/0",
        ]
        + description      = ""
        + from_port        = 22
        + ipv6_cidr_blocks = []
        + prefix_list_ids  = []
        + protocol         = "tcp"
        + security_groups  = []
        + self             = true
        + to_port          = 22
      },
      + {
        + cidr_blocks      = [
          + "0.0.0.0/0",
```

```
            ]
          + description      = ""
          + from_port        = 80
          + ipv6_cidr_blocks = []
          + prefix_list_ids  = []
          + protocol         = "tcp"
          + security_groups  = []
          + self             = true
          + to_port          = 80
        },
      + {
          + cidr_blocks      = []
          + description      = ""
          + from_port        = 0
          + ipv6_cidr_blocks = []
          + prefix_list_ids  = []
          + protocol         = "-1"
          + security_groups  = []
          + self             = true
          + to_port          = 0
        },
    ]
  + name                 = "Ports 22"
  + name_prefix          = (known after apply)
  + owner_id             = (known after apply)
  + revoke_rules_on_delete = false
  + tags                 = {
      + "Name" = "k8s"
    }
  + tags_all             = {
      + "Name" = "k8s"
    }
  + vpc_id               = (known after apply)
}

  # local_file.host_script must be replaced
-/+ resource "local_file" "host_script" {
    ~ content             = <<-EOT
          #!/bin/bash
          echo "Setting SSH Key"
          #ssh-add ~/<PATH TO SSH KEYFILE>.pem
          echo "Adding IPs"
          ssh-keyscan -H ec2-54-84-226-118.compute-1.amazonaws.com >> ~/.ssh/known_hosts
          ssh-keyscan -H ec2-54-91-147-95.compute-1.amazonaws.com >> ~/.ssh/known_hosts
          ssh-keyscan -H ec2-3-82-251-124.compute-1.amazonaws.com >>  ~/.ssh/known_hosts

      EOT -> (known after apply) # forces replacement
    ~ id                  = "4017cb94b49b8f4f81d7d0a474146da7bac4c56c" -> (known after apply)
      # (3 unchanged attributes hidden)
  }

  # local_file.inventory must be replaced
-/+ resource "local_file" "inventory" {
    ~ content             = <<-EOT
          [kubernetes_master]
          ec2-54-84-226-118.compute-1.amazonaws.com

          [kubernetes_worker1]
          ec2-54-91-147-95.compute-1.amazonaws.com

          [kubernetes_worker2]
          ec2-3-82-251-124.compute-1.amazonaws.com

      EOT -> (known after apply) # forces replacement
    ~ id                  = "bbbb5ef91b2e6c9799fbc8321592d38e33220bd6" -> (known after apply)
      # (3 unchanged attributes hidden)
```

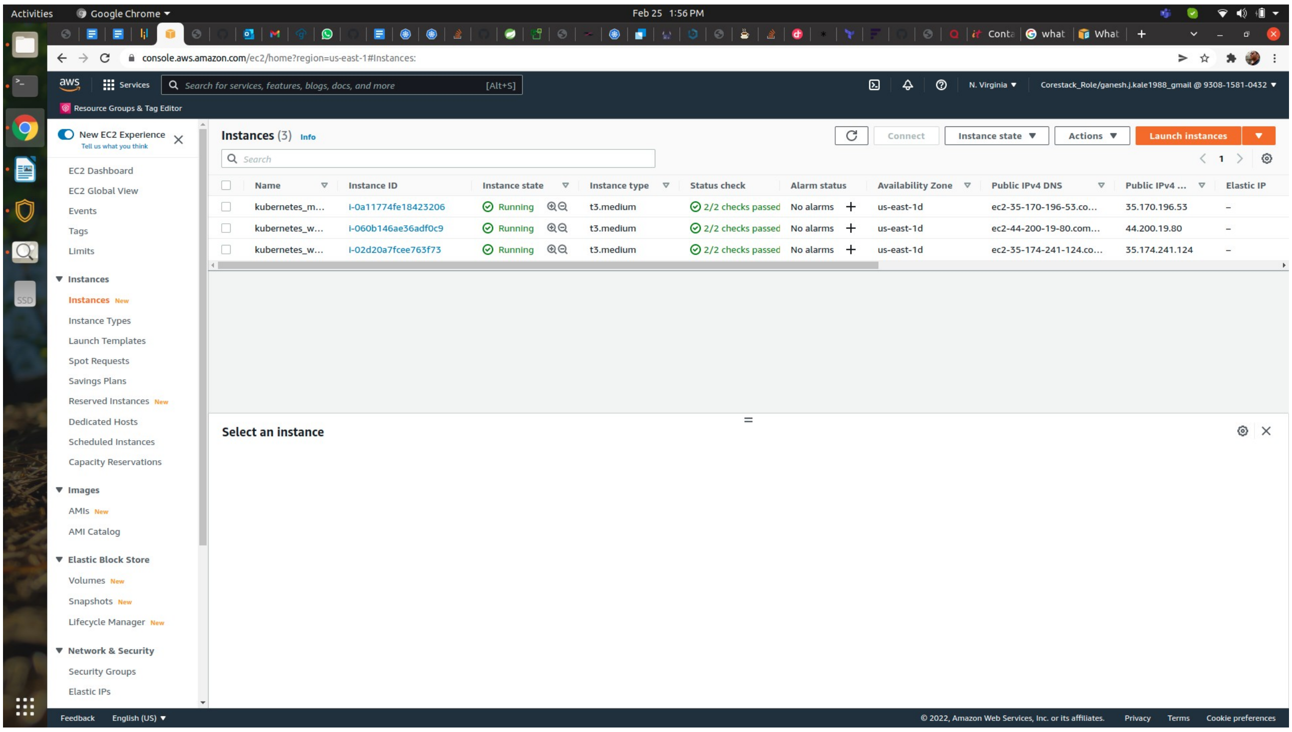
```
    }

  # null_resource.add_host_entry must be replaced
-/+ resource "null_resource" "add_host_entry" {
      ~ id       = "7046571710382423961" -> (known after apply)
      ~ triggers = {
          - "order" = "4017cb94b49b8f4f81d7d0a474146da7bac4c56c"
        } -> (known after apply) # forces replacement
    }

Plan: 8 to add, 0 to change, 3 to destroy.
```

```
Saved the plan to: k8s.zip

To perform exactly these actions, run the following command to apply:
    terraform apply "k8s.zip"
root@gk-ThinkPad-E15-Gen-2:/home/gk/terraform_project/devops_capstone_project# terraform apply k8s.zip
null_resource.add_host_entry: Destroying... [id=7046571710382423961]
null_resource.add_host_entry: Destruction complete after 0s
local_file.inventory: Destroying... [id=bbbb5ef91b2e6c9799fbc8321592d38e33220bd6]
local_file.host_script: Destroying... [id=4017cb94b49b8f4f81d7d0a474146da7bac4c56c]
local_file.inventory: Destruction complete after 0s
local_file.host_script: Destruction complete after 0s
aws_key_pair.mykey: Creating...
aws_security_group.k8s: Creating...
aws_key_pair.mykey: Creation complete after 2s [id=mykey]
aws_security_group.k8s: Creation complete after 7s [id=sg-083cd0f5dbae00066]
aws_instance.kubernetes_worker[0]: Creating...
aws_instance.kubernetes_master: Creating...
aws_instance.kubernetes_worker[1]: Creating...
aws_instance.kubernetes_master: Still creating... [10s elapsed]
aws_instance.kubernetes_worker[0]: Still creating... [10s elapsed]
aws_instance.kubernetes_worker[1]: Still creating... [10s elapsed]
aws_instance.kubernetes_worker[1]: Creation complete after 15s [id=i-060b146ae36adf0c9]
aws_instance.kubernetes_worker[0]: Creation complete after 16s [id=i-02d20a7fcee763f73]
aws_instance.kubernetes_master: Creation complete after 16s [id=i-0a11774fe18423206]
local_file.host_script: Creating...
local_file.inventory: Creating...
local_file.host_script: Creation complete after 0s [id=68e65ee736b4eb2ddb94accfd1305d304131b0c0]
null_resource.add_host_entry: Creating...
local_file.inventory: Creation complete after 0s [id=75ee1410ba34ae2a33f1add40359a6593bb03fc3]
null_resource.add_host_entry: Provisioning with 'local-exec'...
null_resource.add_host_entry (local-exec): Executing: ["/bin/sh" "-c" "sleep 10 && ./add_host.sh"]
null_resource.add_host_entry: Still creating... [10s elapsed]
null_resource.add_host_entry (local-exec): Setting SSH Key
null_resource.add_host_entry (local-exec): Adding IPs
null_resource.add_host_entry (local-exec): # ec2-35-170-196-53.compute-1.amazonaws.com:22 SSH-2.0-OpenSSH_8.2p1 Ubuntu-4ubuntu0.3
null_resource.add_host_entry (local-exec): # ec2-35-170-196-53.compute-1.amazonaws.com:22 SSH-2.0-OpenSSH_8.2p1 Ubuntu-4ubuntu0.3
null_resource.add_host_entry (local-exec): # ec2-35-170-196-53.compute-1.amazonaws.com:22 SSH-2.0-OpenSSH_8.2p1 Ubuntu-4ubuntu0.3
null_resource.add_host_entry (local-exec): # ec2-35-170-196-53.compute-1.amazonaws.com:22 SSH-2.0-OpenSSH_8.2p1 Ubuntu-4ubuntu0.3
null_resource.add_host_entry (local-exec): # ec2-35-170-196-53.compute-1.amazonaws.com:22 SSH-2.0-OpenSSH_8.2p1 Ubuntu-4ubuntu0.3
null_resource.add_host_entry (local-exec): # ec2-35-174-241-124.compute-1.amazonaws.com:22 SSH-2.0-OpenSSH_8.2p1 Ubuntu-4ubuntu0.3
null_resource.add_host_entry (local-exec): # ec2-35-174-241-124.compute-1.amazonaws.com:22 SSH-2.0-OpenSSH_8.2p1 Ubuntu-4ubuntu0.3
```

null_resource.add_host_entry (local-exec): # ec2-35-174-241-124.compute-1.amazonaws.com:22 SSH-2.0-OpenSSH_8.2p1 Ubuntu-4ubuntu0.3
null_resource.add_host_entry (local-exec): # ec2-35-174-241-124.compute-1.amazonaws.com:22 SSH-2.0-OpenSSH_8.2p1 Ubuntu-4ubuntu0.3
null_resource.add_host_entry (local-exec): # ec2-35-174-241-124.compute-1.amazonaws.com:22 SSH-2.0-OpenSSH_8.2p1 Ubuntu-4ubuntu0.3
null_resource.add_host_entry: Still creating... [20s elapsed]
null_resource.add_host_entry (local-exec): # ec2-44-200-19-80.compute-1.amazonaws.com:22 SSH-2.0-OpenSSH_8.2p1 Ubuntu-4ubuntu0.3
null_resource.add_host_entry (local-exec): # ec2-44-200-19-80.compute-1.amazonaws.com:22 SSH-2.0-OpenSSH_8.2p1 Ubuntu-4ubuntu0.3
null_resource.add_host_entry (local-exec): # ec2-44-200-19-80.compute-1.amazonaws.com:22 SSH-2.0-OpenSSH_8.2p1 Ubuntu-4ubuntu0.3
null_resource.add_host_entry (local-exec): # ec2-44-200-19-80.compute-1.amazonaws.com:22 SSH-2.0-OpenSSH_8.2p1 Ubuntu-4ubuntu0.3
null_resource.add_host_entry (local-exec): # ec2-44-200-19-80.compute-1.amazonaws.com:22 SSH-2.0-OpenSSH_8.2p1 Ubuntu-4ubuntu0.3
null_resource.add_host_entry: Creation complete after 24s [id=1433643244391658260]

Apply complete! Resources: 8 added, 0 changed, 3 destroyed.

##################Terraform End



#Ansible playbook Output :-
####################Ansible Started

root@gk-ThinkPad-E15-Gen-2:/home/gk/terraform_project/devops_capstone_project/ansible_cm# ansible-playbook -i inventory.ini main.yaml
[WARNING]: Could not match supplied host pattern, ignoring: kubernetes_worker3
[WARNING]: Could not match supplied host pattern, ignoring: kubernetes_worker4

PLAY [Kubernetes master control plane configuration]
****

TASK [Gathering Facts]
****
ok: [ec2-35-170-196-53.compute-1.amazonaws.com]

TASK [Add an apt signing key for Kubernetes]
****
changed: [ec2-35-170-196-53.compute-1.amazonaws.com]

TASK [Adding apt repository for Kubernetes]
****
changed: [ec2-35-170-196-53.compute-1.amazonaws.com]

TASK [installing packages on k8s master]
****
changed: [ec2-35-170-196-53.compute-1.amazonaws.com]

TASK [Update master hostname]
****
changed: [ec2-35-170-196-53.compute-1.amazonaws.com]

TASK [Modify kubeadm config to match with docker info cgroups]
****
changed: [ec2-35-170-196-53.compute-1.amazonaws.com]

TASK [Restart kubelet]
****
changed: [ec2-35-170-196-53.compute-1.amazonaws.com]

TASK [Reset kubeadm]
****
changed: [ec2-35-170-196-53.compute-1.amazonaws.com]

TASK [Initialize control plane master]
****
changed: [ec2-35-170-196-53.compute-1.amazonaws.com]

TASK [Setup kubeconfig for root user]
****
changed: [ec2-35-170-196-53.compute-1.amazonaws.com] => (item=mkdir -p /root/.kube)
changed: [ec2-35-170-196-53.compute-1.amazonaws.com] => (item=cp -i /etc/kubernetes/admin.conf /root/.kube/config)
changed: [ec2-35-170-196-53.compute-1.amazonaws.com] => (item=chown root:root /root/.kube/config)

Devops Infra Optimization Capstone Project

[WARNING]: Consider using the file module with state=directory rather than running 'mkdir'. If you need to use command because file is insufficient you can add 'warn: false' to this command task or set 'command_warnings=False' in ansible.cfg to get rid of this message.
[WARNING]: Consider using the file module with owner rather than running 'chown'. If you need to use command because file is insufficient you can add 'warn: false' to this command task or set 'command_warnings=False' in ansible.cfg to get rid of this message.

TASK [Install calico pod network]
****
changed: [ec2-35-170-196-53.compute-1.amazonaws.com]

TASK [Generate join command]
****
changed: [ec2-35-170-196-53.compute-1.amazonaws.com]

TASK [Copy join command to local file]
****
changed: [ec2-35-170-196-53.compute-1.amazonaws.com -> localhost]

TASK [Install metric server]
****
changed: [ec2-35-170-196-53.compute-1.amazonaws.com]

TASK [Modify kubeadm config to match with docker info cgroups]
****
ok: [ec2-35-170-196-53.compute-1.amazonaws.com]

TASK [Copy code to master]
****
changed: [ec2-35-170-196-53.compute-1.amazonaws.com]

PLAY [Kubernetes workde node configuration]
****

TASK [Gathering Facts]
****
ok: [ec2-35-174-241-124.compute-1.amazonaws.com]

TASK [Add an apt signing key for Kubernetes]
****
changed: [ec2-35-174-241-124.compute-1.amazonaws.com]

TASK [Adding apt repository for Kubernetes]
****
changed: [ec2-35-174-241-124.compute-1.amazonaws.com]

TASK [installing packages on k8s worker]
****
changed: [ec2-35-174-241-124.compute-1.amazonaws.com]

TASK [update hostname]
****
changed: [ec2-35-174-241-124.compute-1.amazonaws.com]

TASK [Modify kubeadm config to match with docker info cgroups]
****
changed: [ec2-35-174-241-124.compute-1.amazonaws.com]

TASK [Restart kubelet]
****
changed: [ec2-35-174-241-124.compute-1.amazonaws.com]

TASK [Copy the join command to server location]
****
changed: [ec2-35-174-241-124.compute-1.amazonaws.com]

TASK [Reset kubeadm]
****
changed: [ec2-35-174-241-124.compute-1.amazonaws.com]

TASK [Join the node to cluster]
****
changed: [ec2-35-174-241-124.compute-1.amazonaws.com]

PLAY [Kubernetes workde node configuration]
****

TASK [Gathering Facts]
****
ok: [ec2-44-200-19-80.compute-1.amazonaws.com]

TASK [Add an apt signing key for Kubernetes]
****
changed: [ec2-44-200-19-80.compute-1.amazonaws.com]

TASK [Adding apt repository for Kubernetes]
****
changed: [ec2-44-200-19-80.compute-1.amazonaws.com]

TASK [installing packages on k8s worker]
****
changed: [ec2-44-200-19-80.compute-1.amazonaws.com]

TASK [update hostname]
****
changed: [ec2-44-200-19-80.compute-1.amazonaws.com]

TASK [Modify kubeadm config to match with docker info cgroups]
****
changed: [ec2-44-200-19-80.compute-1.amazonaws.com]

TASK [Restart kubelet]
****
changed: [ec2-44-200-19-80.compute-1.amazonaws.com]

```
TASK [Copy the join command to server location]
*********************************************************************************************************************
*************************************************
changed: [ec2-44-200-19-80.compute-1.amazonaws.com]

TASK [Reset kubeadm]
*********************************************************************************************************************
*****************************************************************************
changed: [ec2-44-200-19-80.compute-1.amazonaws.com]

TASK [Join the node to cluster]
*********************************************************************************************************************
*******************************************************************
changed: [ec2-44-200-19-80.compute-1.amazonaws.com]

PLAY [Kubernetes workde node configuration]
*********************************************************************************************************************
*****************************************************
skipping: no hosts matched

PLAY [Kubernetes workde node configuration]
*********************************************************************************************************************
*****************************************************
skipping: no hosts matched

PLAY RECAP
*********************************************************************************************************************
*********************************************************************************************
ec2-35-170-196-53.compute-1.amazonaws.com : ok=16   changed=14   unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
ec2-35-174-241-124.compute-1.amazonaws.com : ok=10   changed=9    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
ec2-44-200-19-80.compute-1.amazonaws.com : ok=10   changed=9    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
```

######################Ansible END#########################

Annexure B

#Stage-B Output on AWS EC2 Master Node Terminal

Stage-B Started#####################

```
root@gk-ThinkPad-E15-Gen-2:/home/gk/terraform_project/devops_capstone_project/ansible_cm# cat
inventory.ini
[kubernetes_master]
ec2-35-170-196-53.compute-1.amazonaws.com

[kubernetes_worker1]
ec2-35-174-241-124.compute-1.amazonaws.com

[kubernetes_worker2]
ec2-44-200-19-80.compute-1.amazonaws.com

root@gk-ThinkPad-E15-Gen-2:/home/gk/terraform_project/devops_capstone_project/ansible_cm# ssh -
i ../mykey ubuntu@ec2-35-170-196-53.compute-1.amazonaws.com
Welcome to Ubuntu 20.04.3 LTS (GNU/Linux 5.11.0-1022-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Fri Feb 25 05:54:33 UTC 2022

  System load:  0.63              Users logged in:          0
  Usage of /:   52.8% of 7.69GB   IPv4 address for docker0: 172.17.0.1
  Memory usage: 27%               IPv4 address for ens5:    172.31.2.133
  Swap usage:   0%                IPv4 address for tunl0:   192.168.235.128
  Processes:    160


79 updates can be applied immediately.
53 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable


Last login: Fri Feb 25 05:19:14 2022 from 103.93.198.164
ubuntu@control-plane:~$ sudo su
root@control-plane:/home/ubuntu# ls
two_tier_app_k8.tgz
root@control-plane:/home/ubuntu# tar -xvf two_tier_app_k8.tgz
two_tier_app_k8/
two_tier_app_k8/metric_components.yaml
two_tier_app_k8/etcd_backup/
two_tier_app_k8/etcd_backup/snapshot.db
two_tier_app_k8/etcd_backup/all_keys/
two_tier_app_k8/etcd_backup/all_keys/pki/
two_tier_app_k8/etcd_backup/all_keys/pki/apiserver-kubelet-client.crt
two_tier_app_k8/etcd_backup/all_keys/pki/front-proxy-ca.key
two_tier_app_k8/etcd_backup/all_keys/pki/front-proxy-client.crt
two_tier_app_k8/etcd_backup/all_keys/pki/front-proxy-client.key
two_tier_app_k8/etcd_backup/all_keys/pki/apiserver.key
two_tier_app_k8/etcd_backup/all_keys/pki/sa.pub
two_tier_app_k8/etcd_backup/all_keys/pki/ca.crt
two_tier_app_k8/etcd_backup/all_keys/pki/ca.key
two_tier_app_k8/etcd_backup/all_keys/pki/etcd/
two_tier_app_k8/etcd_backup/all_keys/pki/etcd/healthcheck-client.key
two_tier_app_k8/etcd_backup/all_keys/pki/etcd/ca.crt
two_tier_app_k8/etcd_backup/all_keys/pki/etcd/server.key
two_tier_app_k8/etcd_backup/all_keys/pki/etcd/ca.key
two_tier_app_k8/etcd_backup/all_keys/pki/etcd/peer.crt
two_tier_app_k8/etcd_backup/all_keys/pki/etcd/healthcheck-client.crt
```

```
two_tier_app_k8/etcd_backup/all_keys/pki/etcd/server.crt
two_tier_app_k8/etcd_backup/all_keys/pki/etcd/peer.key
two_tier_app_k8/etcd_backup/all_keys/pki/apiserver.crt
two_tier_app_k8/etcd_backup/all_keys/pki/apiserver-kubelet-client.key
two_tier_app_k8/etcd_backup/all_keys/pki/front-proxy-ca.crt
two_tier_app_k8/etcd_backup/all_keys/pki/apiserver-etcd-client.key
two_tier_app_k8/etcd_backup/all_keys/pki/apiserver-etcd-client.crt
two_tier_app_k8/etcd_backup/all_keys/pki/sa.key
two_tier_app_k8/deploy_app_metric_userrole.sh
two_tier_app_k8/user_role/
two_tier_app_k8/user_role/csr.yaml
two_tier_app_k8/user_role/gk.csr
two_tier_app_k8/user_role/gk.cer
two_tier_app_k8/user_role/gk.key
two_tier_app_k8/user_role/devrole.yaml
two_tier_app_k8/user_role/rolebind.yaml
two_tier_app_k8/02_webapp/
two_tier_app_k8/02_webapp/app.configmap.yml
two_tier_app_k8/02_webapp/app.service.yml
two_tier_app_k8/02_webapp/app.deployment.yml
two_tier_app_k8/02_webapp/app.horizontal_pod_autoscaler.yml
two_tier_app_k8/02_webapp/app.secret.yml
two_tier_app_k8/01_redis/
two_tier_app_k8/01_redis/redis-primary.service.yml
two_tier_app_k8/01_redis/redis.networkpolicy.yml
two_tier_app_k8/01_redis/redis-primary.deployment.yml
two_tier_app_k8/01_redis/redis-replica.deployment.yml
two_tier_app_k8/01_redis/redis-replica.service.yml
two_tier_app_k8/01_redis/redis-replica.horizontal_pod_autoscaler.yml
root@control-plane:/home/ubuntu# k get po
k: command not found
root@control-plane:/home/ubuntu# alias k=kubectl
root@control-plane:/home/ubuntu# k get po
No resources found in default namespace.
root@control-plane:/home/ubuntu# k get nodes
NAME           STATUS   ROLES                  AGE   VERSION
control-plane   Ready    control-plane,master   37m   v1.23.4
worker1        Ready    <none>                 33m   v1.23.4
worker2        Ready    <none>                 30m   v1.23.4
root@control-plane:/home/ubuntu# ls
two_tier_app_k8  two_tier_app_k8.tgz
root@control-plane:/home/ubuntu# cd two_tier_app_k8
root@control-plane:/home/ubuntu/two_tier_app_k8# ls
01_redis  02_webapp  deploy_app_metric_userrole.sh  etcd_backup  metric_components.yaml  user_role
root@control-plane:/home/ubuntu/two_tier_app_k8# ./deploy_app_metric_userrole.sh
serviceaccount/metrics-server unchanged
clusterrole.rbac.authorization.k8s.io/system:aggregated-metrics-reader unchanged
clusterrole.rbac.authorization.k8s.io/system:metrics-server unchanged
rolebinding.rbac.authorization.k8s.io/metrics-server-auth-reader unchanged
clusterrolebinding.rbac.authorization.k8s.io/metrics-server:system:auth-delegator unchanged
clusterrolebinding.rbac.authorization.k8s.io/system:metrics-server unchanged
service/metrics-server unchanged
deployment.apps/metrics-server configured
apiservice.apiregistration.k8s.io/v1beta1.metrics.k8s.io unchanged
deployment.apps/redis-primary created
service/redis-primary created
deployment.apps/redis-replica created
horizontalpodautoscaler.autoscaling/redis-replica created
service/redis-replica created
networkpolicy.networking.k8s.io/redis created
configmap/webapp created
deployment.apps/webapp created
horizontalpodautoscaler.autoscaling/webapp created
secret/webapp created
service/webapp created
certificatesigningrequest.certificates.k8s.io/csr-for-gk created
```

```
role.rbac.authorization.k8s.io/developer created
rolebinding.rbac.authorization.k8s.io/gk-developer-binding created

root@control-plane:/home/ubuntu/two_tier_app_k8/user_role# k config use-context kubernetes-admin@kubernetes
Switched to context "kubernetes-admin@kubernetes".
root@control-plane:/home/ubuntu/two_tier_app_k8/user_role# k get svc
NAME            TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
kubernetes      ClusterIP   10.96.0.1       <none>        443/TCP        49m
redis-primary   ClusterIP   10.107.111.199  <none>        6379/TCP       11m
redis-replica   ClusterIP   10.110.31.237   <none>        6379/TCP       11m
webapp          NodePort    10.105.185.152  <none>        80:30007/TCP   11m
root@control-plane:/home/ubuntu/two_tier_app_k8/user_role# k get-context
error: unknown command "get-context" for "kubectl"
root@control-plane:/home/ubuntu/two_tier_app_k8/user_role# Kubectl get po,deploy,nodes,svc,hpa
Kubectl: command not found
root@control-plane:/home/ubuntu/two_tier_app_k8/user_role# kubectl get po,deploy,nodes,svc,hpa
NAME                                READY   STATUS    RESTARTS   AGE
pod/redis-primary-58c7df987d-28tnl  1/1     Running   0          13m
pod/redis-replica-8c6c65b47-gxt4x   1/1     Running   0          13m
pod/redis-replica-8c6c65b47-r9qh7   1/1     Running   0          13m
pod/redis-replica-8c6c65b47-swvhp   1/1     Running   0          13m
pod/webapp-556bbb797c-6x98z         1/1     Running   0          13m
pod/webapp-556bbb797c-9jwhk         1/1     Running   0          13m
pod/webapp-556bbb797c-bbmwt         1/1     Running   0          13m
pod/webapp-556bbb797c-bfkgd         1/1     Running   0          13m
pod/webapp-556bbb797c-bsjlm         1/1     Running   0          13m
pod/webapp-556bbb797c-dcrcp         1/1     Running   0          13m
pod/webapp-556bbb797c-h2td9         1/1     Running   0          13m
pod/webapp-556bbb797c-r74c5         1/1     Running   0          13m
pod/webapp-556bbb797c-rg9w5         1/1     Running   0          13m
pod/webapp-556bbb797c-vq8lq         1/1     Running   0          13m

NAME                           READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/redis-primary  1/1     1            1           13m
deployment.apps/redis-replica  3/3     3            3           13m
deployment.apps/webapp         10/10   10           10          13m

NAME                 STATUS   ROLES                  AGE   VERSION
node/control-plane   Ready    control-plane,master   51m   v1.23.4
node/worker1         Ready    <none>                 46m   v1.23.4
node/worker2         Ready    <none>                 44m   v1.23.4

NAME                    TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
service/kubernetes      ClusterIP   10.96.0.1       <none>        443/TCP        51m
service/redis-primary   ClusterIP   10.107.111.199  <none>        6379/TCP       13m
service/redis-replica   ClusterIP   10.110.31.237   <none>        6379/TCP       13m
service/webapp          NodePort    10.105.185.152  <none>        80:30007/TCP   13m

NAME                                                        REFERENCE                TARGETS   MINPODS   MAXPODS   REPLICAS   AGE
horizontalpodautoscaler.autoscaling/redis-replica   Deployment/redis-replica   1%/20%    3         5         3          13m
horizontalpodautoscaler.autoscaling/webapp          Deployment/webapp          2%/30%    10        20        10         13m

root@control-plane:/home/ubuntu/two_tier_app_k8# k get po
NAME                            READY   STATUS              RESTARTS   AGE
redis-primary-58c7df987d-28tnl  0/1     ContainerCreating   0          6s
redis-replica-8c6c65b47-swvhp   1/1     Running             0          6s
webapp-556bbb797c-dcrcp         0/1     ContainerCreating   0          6s
```

#Preconfigured Steps Output:-

```
root@control-plane:/home/ubuntu# cd user_role/
root@control-plane:/home/ubuntu/user_role# openssl genrsa -out gk.key 2048
```

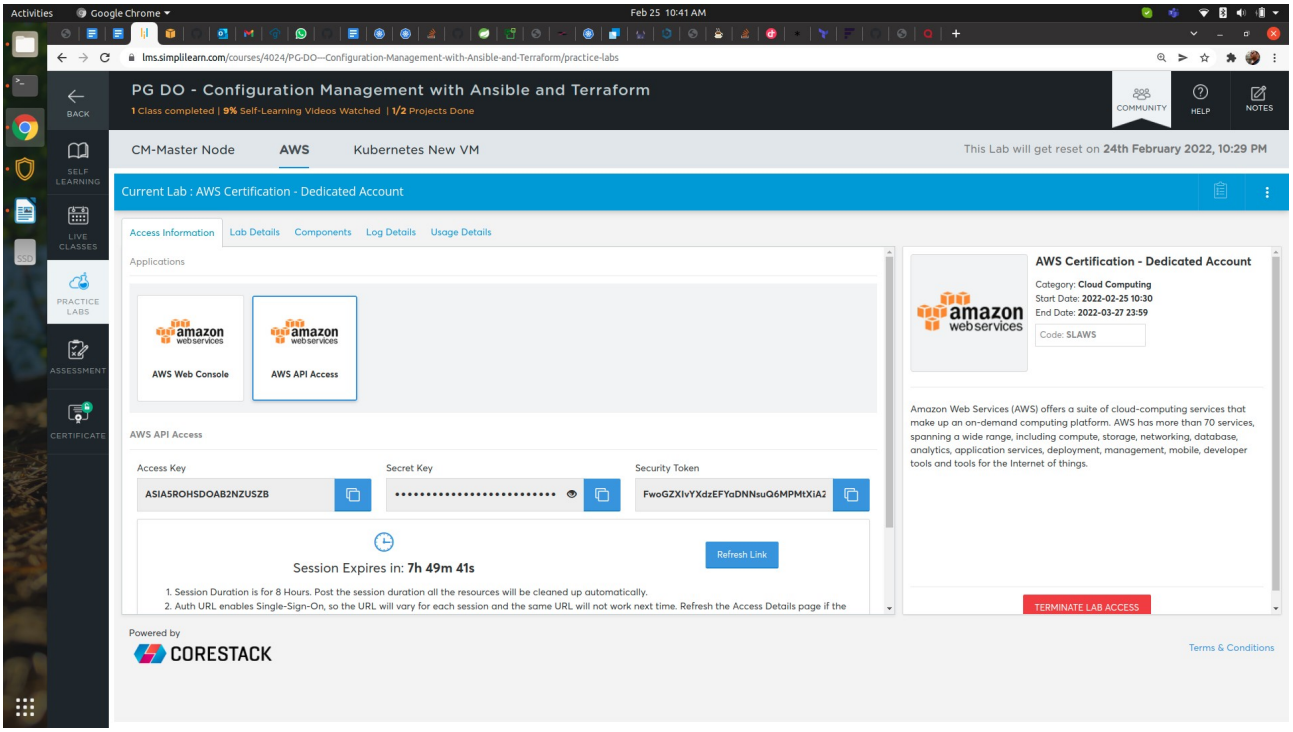
```
Generating RSA private key, 2048 bit long modulus (2 primes)
.............................+++++
..+++++
e is 65537 (0x010001)
root@control-plane:/home/ubuntu/user_role# openssl req -new -key gk.key -subj "/CN=gk" -out gk.csr
root@control-plane:/home/ubuntu/user_role# cat gk.
gk.csr  gk.key
root@control-plane:/home/ubuntu/user_role# cat gk.
gk.csr  gk.key
root@control-plane:/home/ubuntu/user_role# cat gk.csr  |base64 -w 0
LS0tLS1CRUdJTiBDRVJUSUZJQ0FURSBSRVFVRVNULS0tLS0KTUlJQ1VqQ0NBVG9DQVFBd0RURUxNQWtHQTF
VRUF3d0NaMnN3Z2dFaU1BMEdDU3FHU0liM0RRRUJBUVVBQTRJQgpEd0F3Z2dFS0FvSUJBUURBOE9xUFhqa
kt5QkZBQVFEVEhuUWVoNVZERXFsSjBhVCtwVUV6dnZLSkh6NTNleFJvCjJTb2lCY2lzQ1dsTWZwNmNCbHF6W
FQ0MnUwSU03ZFlzUTB0T3liU1hrV2ZScThhWHE0NXVrV3VjTWhBNkN1VFgKNTJhUzZONGxJZG5MSTYrdzdaa
npsOUw5bVNqSGRTL3l2b2hiR0g5RGpEaFpEODNhbDVQR3kzRFQ1a00ySUtQUgp4KzJMQW1GTndYNytMNzla
U0tPeUpnQmdySnFJS0ZLb1JxcTBaUGZGR0J0TVhTMk9JbGVwd1RPMHZoUVNOaDluCkJYSDNLZEp1OUxCR0c
yaC82empISzg0SHNXSUhyaGVOWDd1Y1dPYXRSS01nREkxajU3Y0NMbllDSTJoVUkwYlEKUXliVll5Qi9lUXU4Mj
ErTnhlditObjNMdEVEcmhuYmNaZXM3QWdNQkFBR2dBREFOQmdrcWhraUc5dzBCQVFzRgpBQU9DQVFFQW
FwaE1IbHdPWTdJTUNwcmM1R1JPN0lDZWJ1dnF1QjhlYmRCcDVXT3FuQUwrU0pRUUREL0FveEp1CjJiamkrVzJr
ZUFWdXVrc2pGYmROVld2ZmVzZkxYUkM5bEhiVE9nVGswU0EybC9oRXJWZHNwYzBHcmp3dWtFMUlKNkZYd
kZKNWZWOVZYYnpHREozRWc5SWp5R3RFWHQrZ01DOGZnbFhaUFVYTm91ZEQ0c1N3Y2hVeEFGdDBldWp
HeApmS2o3bDZBL2k4QjZDRGEvNlpGWGVMZ0RwL01mbGNqc2pUTFRQTWdxTk1jbGg0bTlSMFRIZ1YwQnQ
4dVpFQmlMCm9heUpvUmxkdUlVQzQzb0pEclhaSlg0UUZpRGI1U0t3b1g0RFBDR2lEZDlzSVJwNjNaWHZWNjh
BY1BMb0lSNFQKV25haXVBSFpVb24veFVnRHgrYk5MNmRtSjFlK1JBPT0KLS0tLS1FTkQgQ0VSVElGSUNBVEUg
UkVRVUVTVC0tLS0tCg==


#Manual Steps:-
root@control-plane:/home/ubuntu/two_tier_app_k8# k get csr
NAME        AGE   SIGNERNAME                                    REQUESTOR                 REQUESTEDDURATION
CONDITION
csr-5fffp   34m   kubernetes.io/kube-apiserver-client-kubelet   system:bootstrap:ew7rfc   <none>
Approved,Issued
csr-for-gk  63s   kubernetes.io/kube-apiserver-client          kubernetes-admin          <none>
Pending
csr-l4fvl   38m   kubernetes.io/kube-apiserver-client-kubelet   system:node:control-plane  <none>
Approved,Issued
csr-nmqd4   31m   kubernetes.io/kube-apiserver-client-kubelet   system:bootstrap:ew7rfc   <none>
Approved,Issued
root@control-plane:/home/ubuntu/two_tier_app_k8# k certificate approve csr-for-gk
certificatesigningrequest.certificates.k8s.io/csr-for-gk approved
root@control-plane:/home/ubuntu/two_tier_app_k8# k get csr
NAME        AGE   SIGNERNAME                                    REQUESTOR                 REQUESTEDDURATION
CONDITION
csr-5fffp   35m   kubernetes.io/kube-apiserver-client-kubelet   system:bootstrap:ew7rfc   <none>
Approved,Issued
csr-for-gk  105s  kubernetes.io/kube-apiserver-client          kubernetes-admin          <none>
Approved,Issued
csr-l4fvl   39m   kubernetes.io/kube-apiserver-client-kubelet   system:node:control-plane  <none>
Approved,Issued
csr-nmqd4   32m   kubernetes.io/kube-apiserver-client-kubelet   system:bootstrap:ew7rfc   <none>
Approved,Issued
root@control-plane:/home/ubuntu/two_tier_app_k8# cd user_role/
root@control-plane:/home/ubuntu/two_tier_app_k8/user_role#
root@control-plane:/home/ubuntu/two_tier_app_k8/user_role# kubectl get csr csr-for-gk -o
jsonpath='{.status.certificate}' | base64 --decode > gk.cer

root@control-plane:/home/ubuntu/two_tier_app_k8/user_role# kubectl config set-credentials gk --client-
key /home/ubuntu/two_tier_app_k8/user_role/gk.key  --client-certificate
/home/ubuntu/two_tier_app_k8/user_role/gk.cer
User "gk" set.
root@control-plane:/home/ubuntu/two_tier_app_k8/user_role# kubectl config set-context gk@kubernetes
--cluster kubernetes --user gk
Context "gk@kubernetes" created.
root@control-plane:/home/ubuntu/two_tier_app_k8/user_role# kubectl config view
apiVersion: v1
```

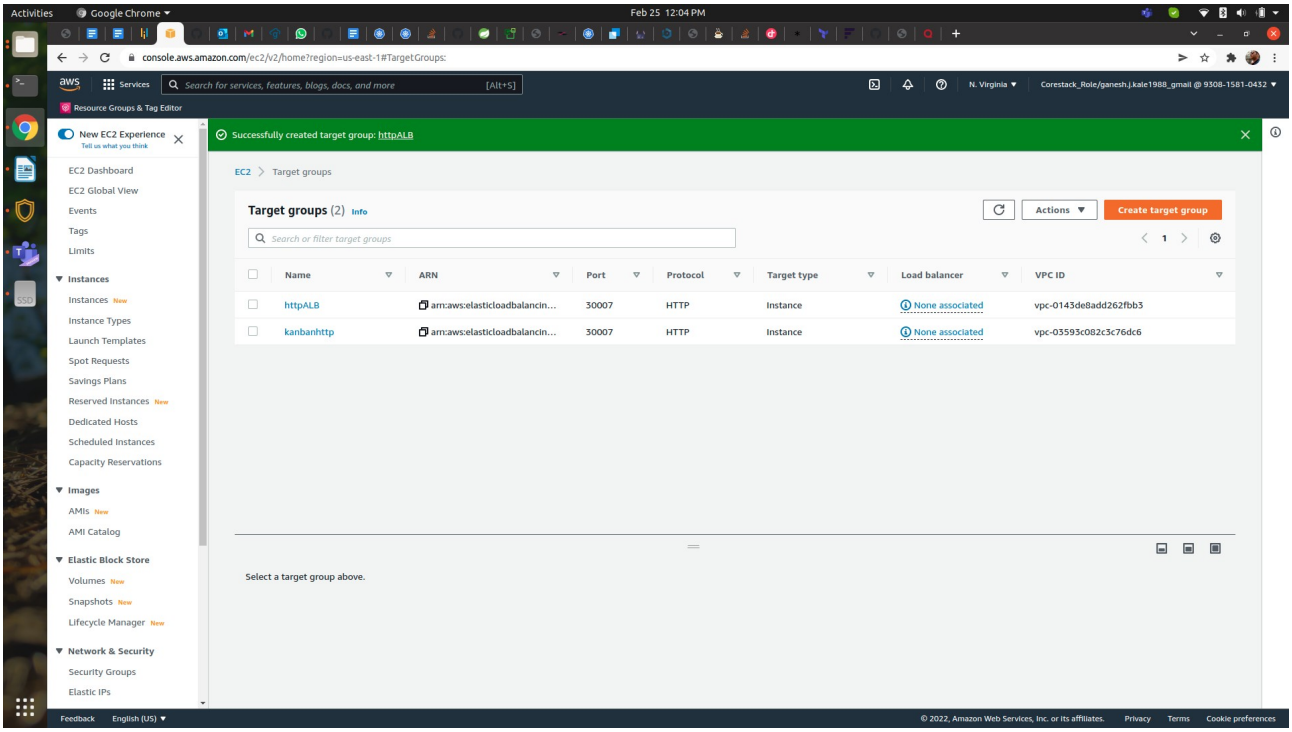
```
clusters:
- cluster:
    certificate-authority-data: DATA+OMITTED
    server: https://172.31.2.133:6443
  name: kubernetes
contexts:
- context:
    cluster: kubernetes
    user: gk
  name: gk@kubernetes
- context:
    cluster: kubernetes
    user: kubernetes-admin
  name: kubernetes-admin@kubernetes
current-context: kubernetes-admin@kubernetes
kind: Config
preferences: {}
users:
- name: gk
  user:
    client-certificate: /home/ubuntu/two_tier_app_k8/user_role/gk.cer
    client-key: /home/ubuntu/two_tier_app_k8/user_role/gk.key
- name: kubernetes-admin
  user:
    client-certificate-data: REDACTED
    client-key-data: REDACTED
root@control-plane:/home/ubuntu/two_tier_app_k8/user_role# kubectl config use-context gk@kubernetes
Switched to context "gk@kubernetes".
root@control-plane:/home/ubuntu/two_tier_app_k8/user_role# kubectl auth can-i --as gk get pods
Error from server (Forbidden): users "gk" is forbidden: User "gk" cannot impersonate resource "users" in
API group "" at the cluster scope
root@control-plane:/home/ubuntu/two_tier_app_k8/user_role# k get po
NAME                           READY   STATUS    RESTARTS   AGE
redis-primary-58c7df987d-28tnl   1/1     Running   0          4m36s
redis-replica-8c6c65b47-gxt4x    1/1     Running   0          4m21s
redis-replica-8c6c65b47-r9qh7    1/1     Running   0          4m21s
redis-replica-8c6c65b47-swvhp    1/1     Running   0          4m36s
webapp-556bbb797c-6x98z          1/1     Running   0          4m21s
webapp-556bbb797c-9jwhk          1/1     Running   0          4m21s
webapp-556bbb797c-bbmwt          1/1     Running   0          4m21s
webapp-556bbb797c-bfkgd          1/1     Running   0          4m21s
webapp-556bbb797c-bsjlm          1/1     Running   0          4m21s
webapp-556bbb797c-dcrcp          1/1     Running   0          4m36s
webapp-556bbb797c-h2td9          1/1     Running   0          4m21s
webapp-556bbb797c-r74c5          1/1     Running   0          4m21s
webapp-556bbb797c-rg9w5          1/1     Running   0          4m21s
webapp-556bbb797c-vq8lq          1/1     Running   0          4m21s
root@control-plane:/home/ubuntu/two_tier_app_k8/user_role# k get nodes
Error from server (Forbidden): nodes is forbidden: User "gk" cannot list resource "nodes" in API group "" at
the cluster scope
root@control-plane:/home/ubuntu/two_tier_app_k8/user_role# k get svc
Error from server (Forbidden): services is forbidden: User "gk" cannot list resource "services" in API group
"" in the namespace "default"
```

################Stage-B End####################

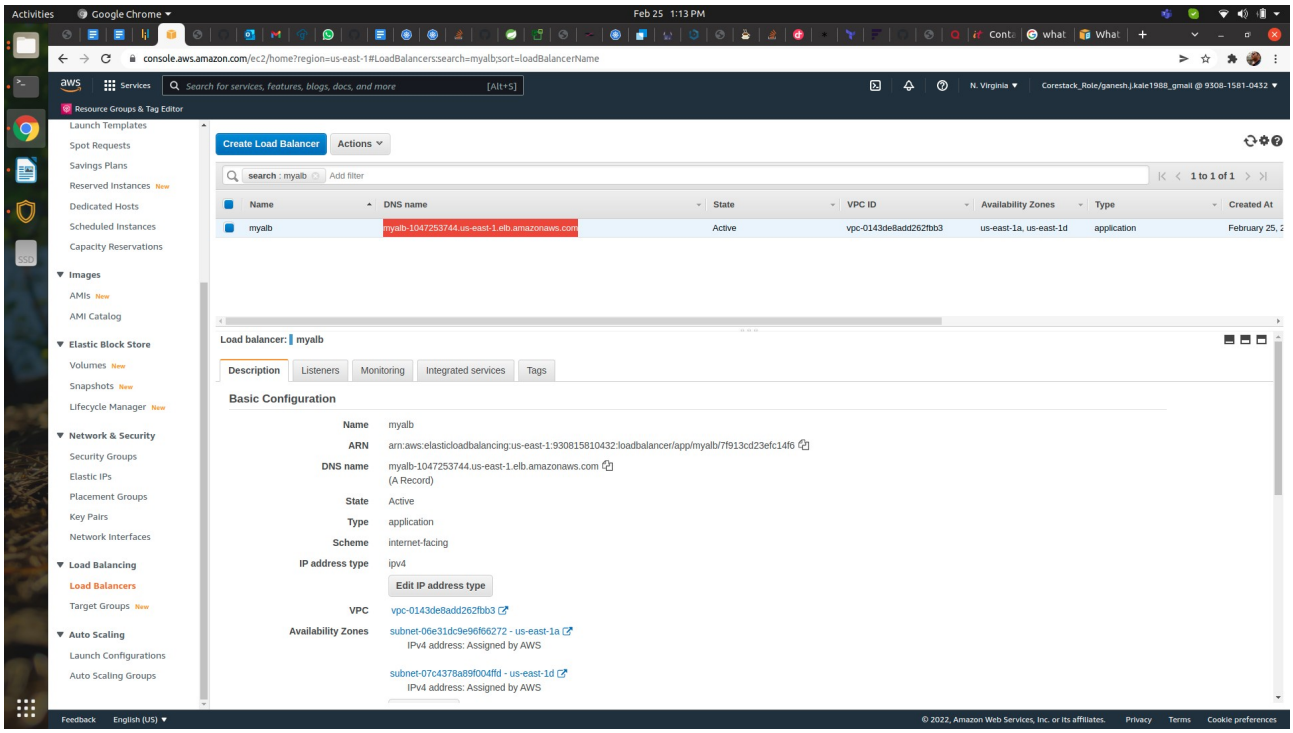
Devops Infra Optimization Capstone Project

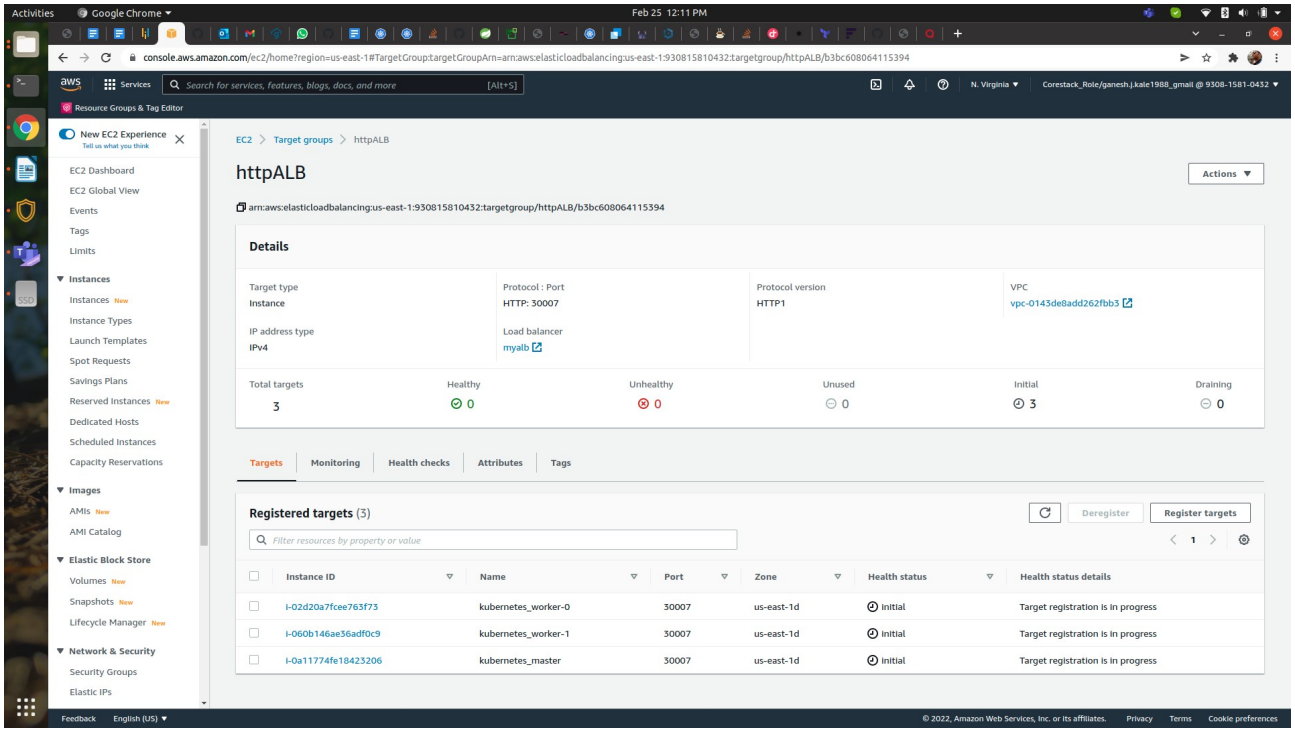
Annexure C



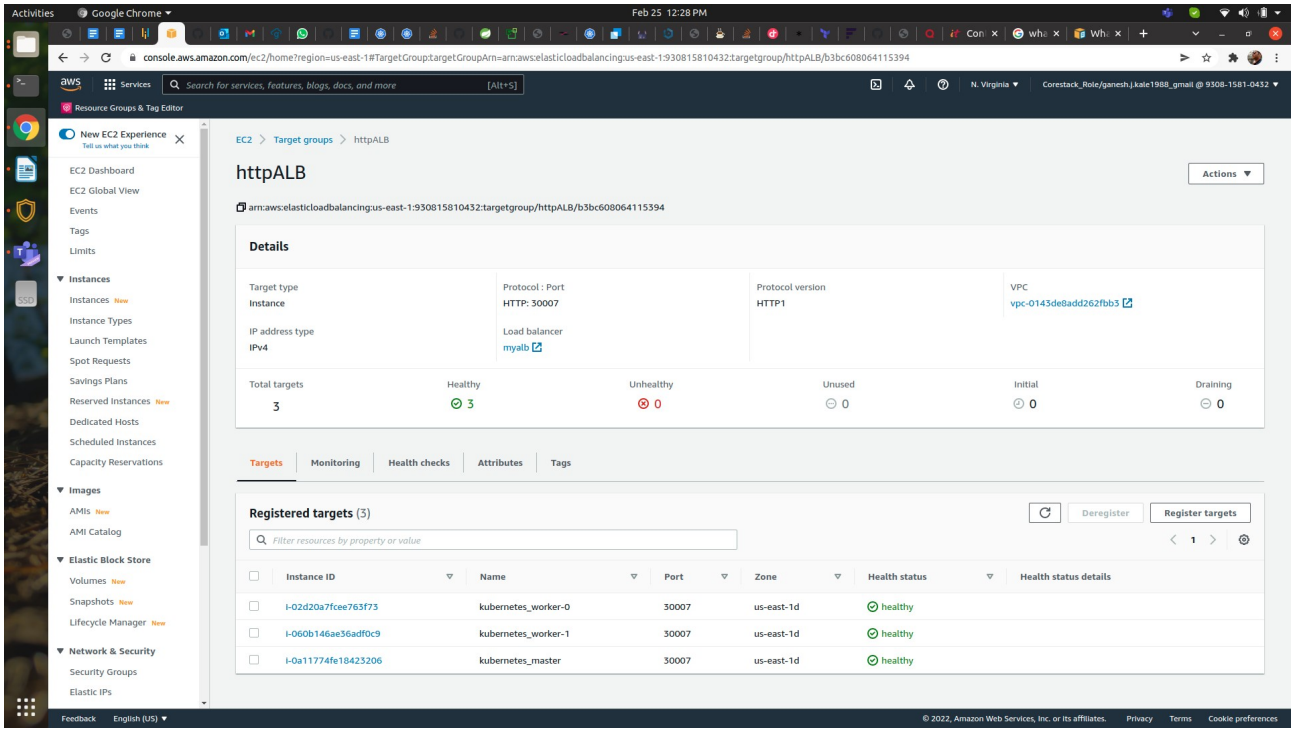


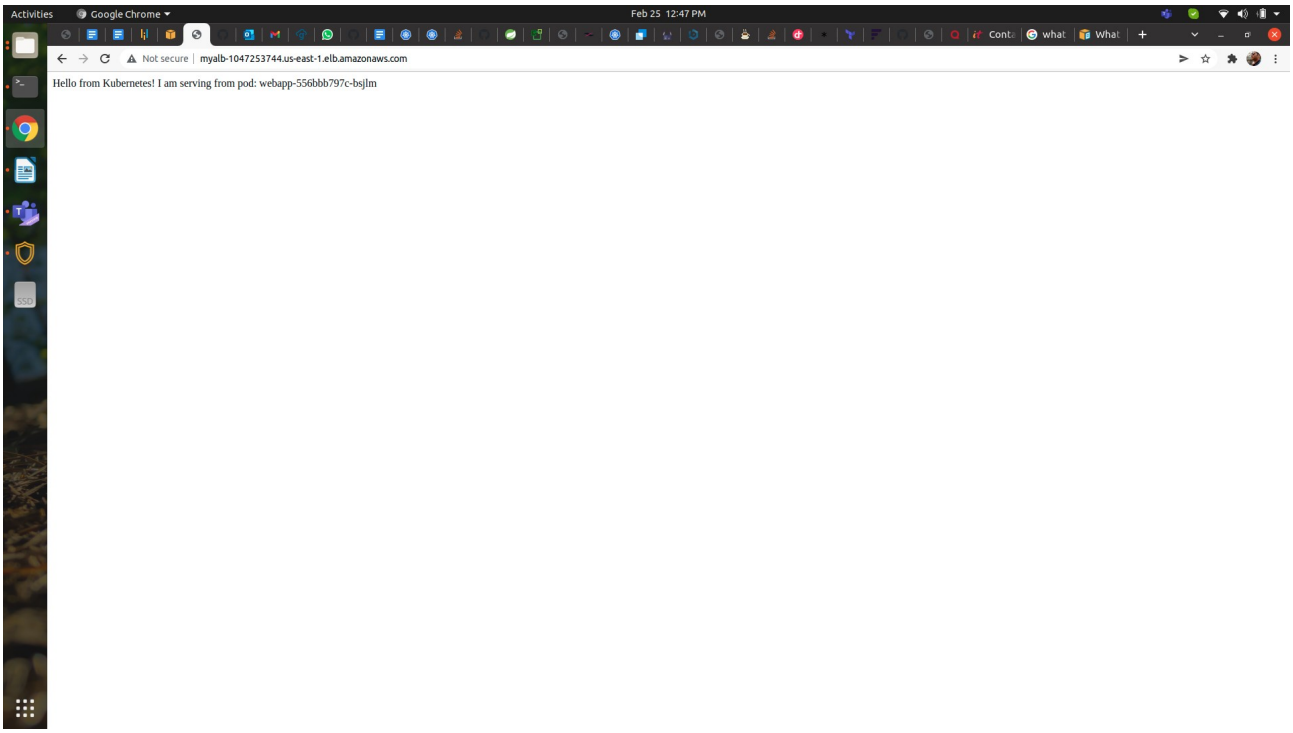
Devops Infra Optimization Capstone Project



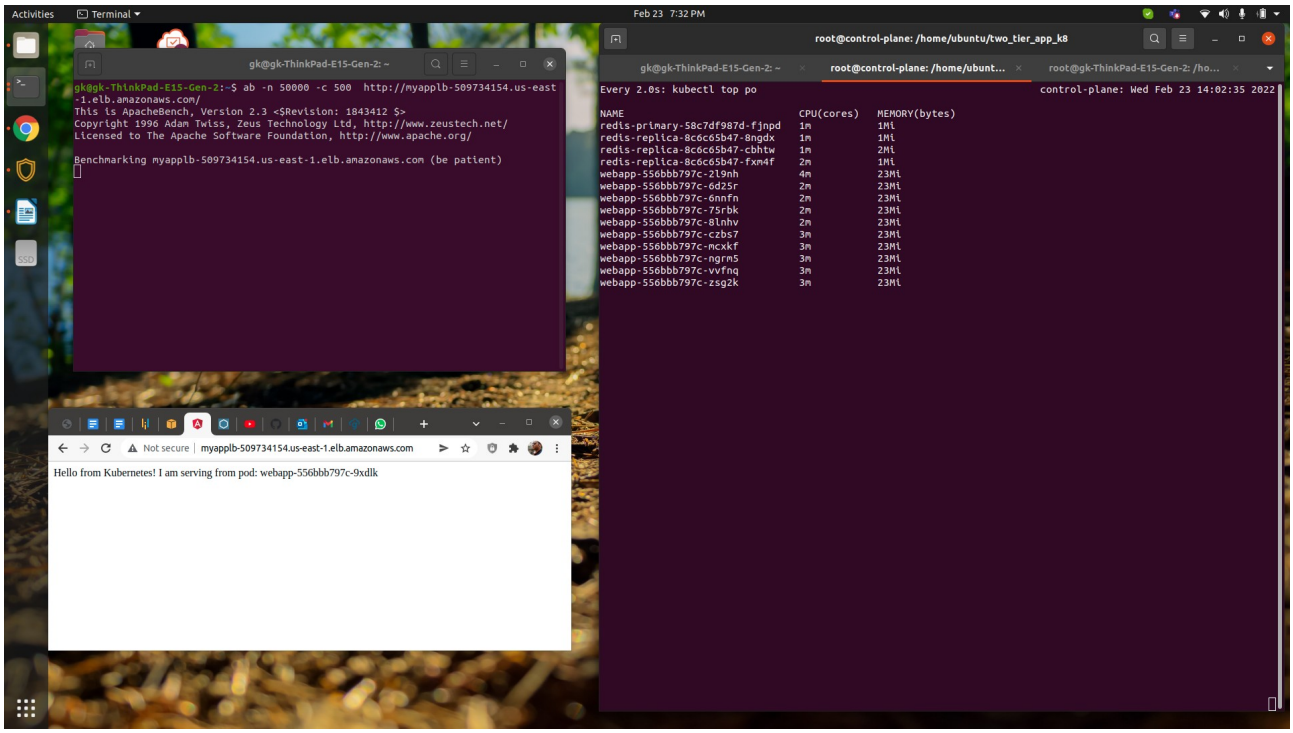


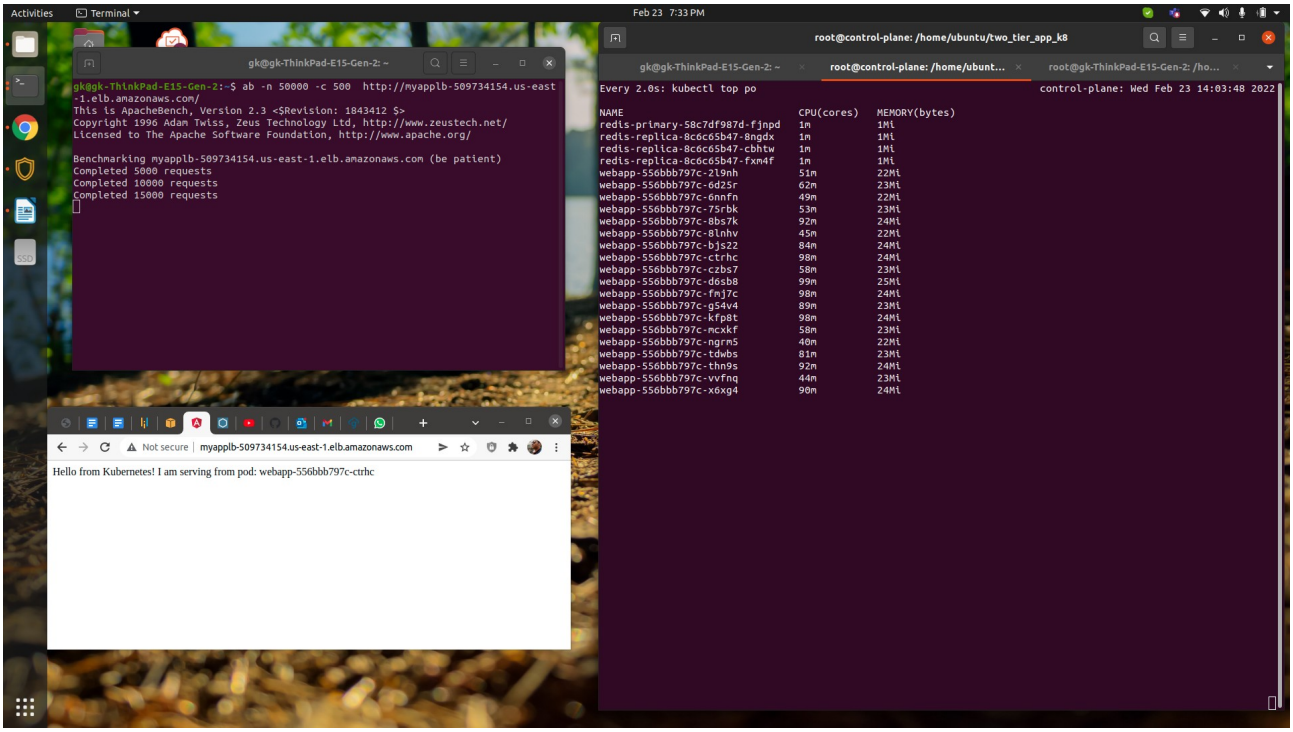
Devops Infra Optimization Capstone Project





Annexure D


Activities
Terminal
Feb 23 7:32 PM
gk@gk-ThinkPad-E15-Gen-2: ~
gk@gk-ThinkPad-E15-Gen-2:~$ ab -n 50000 -c 500 http://myapplb-509734154.us-east-1.elb.amazonaws.com/
This is ApacheBench, Version 2.3 <$Revision: 1843412 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/
Benchmarking myapplb-509734154.us-east-1.elb.amazonaws.com (be patient)
root@control-plane: /home/ubuntu/two_tier_app_k8
gk@gk-ThinkPad-E15-Gen-2: ~
root@control-plane: /home/ubunt...
root@gk-ThinkPad-E15-Gen-2: /ho...
Every 2.0s: kubectl top po
control-plane: Wed Feb 23 14:02:35 2022
NAME CPU(cores) MEMORY(bytes)
redis-primary-58c7df987d-fjnpd 1m 1Mi
redis-replica-8c6c65b47-8ngdx 1m 1Mi
redis-replica-8c6c65b47-cbhtw 1m 2Mi
redis-replica-8c6c65b47-fxm4f 2m 1Mi
webapp-556bbb797c-2l9nh 4m 23Mi
webapp-556bbb797c-6d25r 2m 23Mi
webapp-556bbb797c-6nnfn 2m 23Mi
webapp-556bbb797c-75rbk 2m 23Mi
webapp-556bbb797c-8lnhv 2m 23Mi
webapp-556bbb797c-czbs7 3m 23Mi
webapp-556bbb797c-mcxkf 3m 23Mi
webapp-556bbb797c-ngrn5 3m 23Mi
webapp-556bbb797c-vvfnq 3m 23Mi
webapp-556bbb797c-zsg2k 3m 23Mi
Not secure
myapplb-509734154.us-east-1.elb.amazonaws.com
Hello from Kubernetes! I am serving from pod: webapp-556bbb797c-9xdlk


Activities
Terminal
Feb 23 7:33 PM
gk@gk-ThinkPad-E15-Gen-2: ~
gk@gk-ThinkPad-E15-Gen-2:~$ ab -n 50000 -c 500 http://myapplb-509734154.us-east-1.elb.amazonaws.com/
This is ApacheBench, Version 2.3 <$Revision: 1843412 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/
Benchmarking myapplb-509734154.us-east-1.elb.amazonaws.com (be patient)
Completed 5000 requests
Completed 10000 requests
Completed 15000 requests
root@control-plane: /home/ubuntu/two_tier_app_k8
gk@gk-ThinkPad-E15-Gen-2: ~
root@control-plane: /home/ubunt...
root@gk-ThinkPad-E15-Gen-2: /ho...
Every 2.0s: kubectl top po
control-plane: Wed Feb 23 14:03:48 2022
NAME CPU(cores) MEMORY(bytes)
redis-primary-58c7df987d-fjnpd 1m 1Mi
redis-replica-8c6c65b47-8ngdx 1m 1Mi
redis-replica-8c6c65b47-cbhtw 1m 1Mi
redis-replica-8c6c65b47-fxm4f 1m 1Mi
webapp-556bbb797c-2l9nh 51m 22Mi
webapp-556bbb797c-6d25r 62m 23Mi
webapp-556bbb797c-6nnfn 49m 22Mi
webapp-556bbb797c-75rbk 53m 23Mi
webapp-556bbb797c-8bs7k 92m 24Mi
webapp-556bbb797c-8lnhv 45m 22Mi
webapp-556bbb797c-bjs22 84m 24Mi
webapp-556bbb797c-ctrhc 98m 24Mi
webapp-556bbb797c-czbs7 58m 23Mi
webapp-556bbb797c-d6sb8 99m 25Mi
webapp-556bbb797c-fnj7c 98m 24Mi
webapp-556bbb797c-g54v4 89m 23Mi
webapp-556bbb797c-kfp8t 98m 24Mi
webapp-556bbb797c-mcxkf 58m 23Mi
webapp-556bbb797c-ngrn5 40m 22Mi
webapp-556bbb797c-tdwbs 81m 23Mi
webapp-556bbb797c-thn9s 92m 24Mi
webapp-556bbb797c-vvfnq 44m 23Mi
webapp-556bbb797c-x6xg4 90m 24Mi
Not secure
myapplb-509734154.us-east-1.elb.amazonaws.com
Hello from Kubernetes! I am serving from pod: webapp-556bbb797c-ctrhc