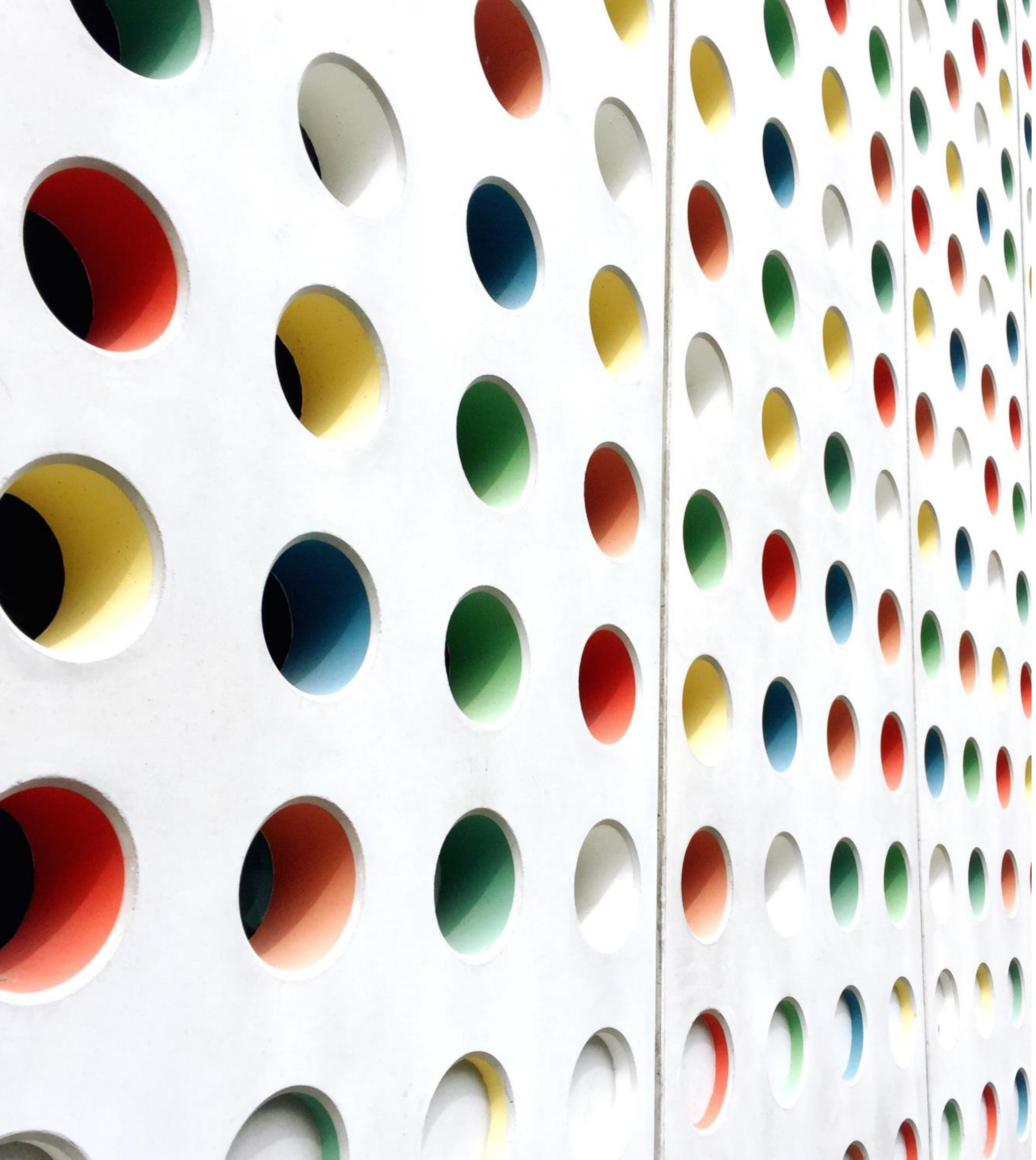


Cloud Architecture

AD@N1_K@MAL

Challenge scenario

You have started a new role as a Cloud Architect for Jooli Inc. You are expected to help design and manage the infrastructure at Jooli. Common tasks revolve around designing environments for the various projects within the Jooli Inc. family but also include provisioning resources for projects.

You are expected to have the skills and knowledge for these tasks, so don't expect step-by-step guides.

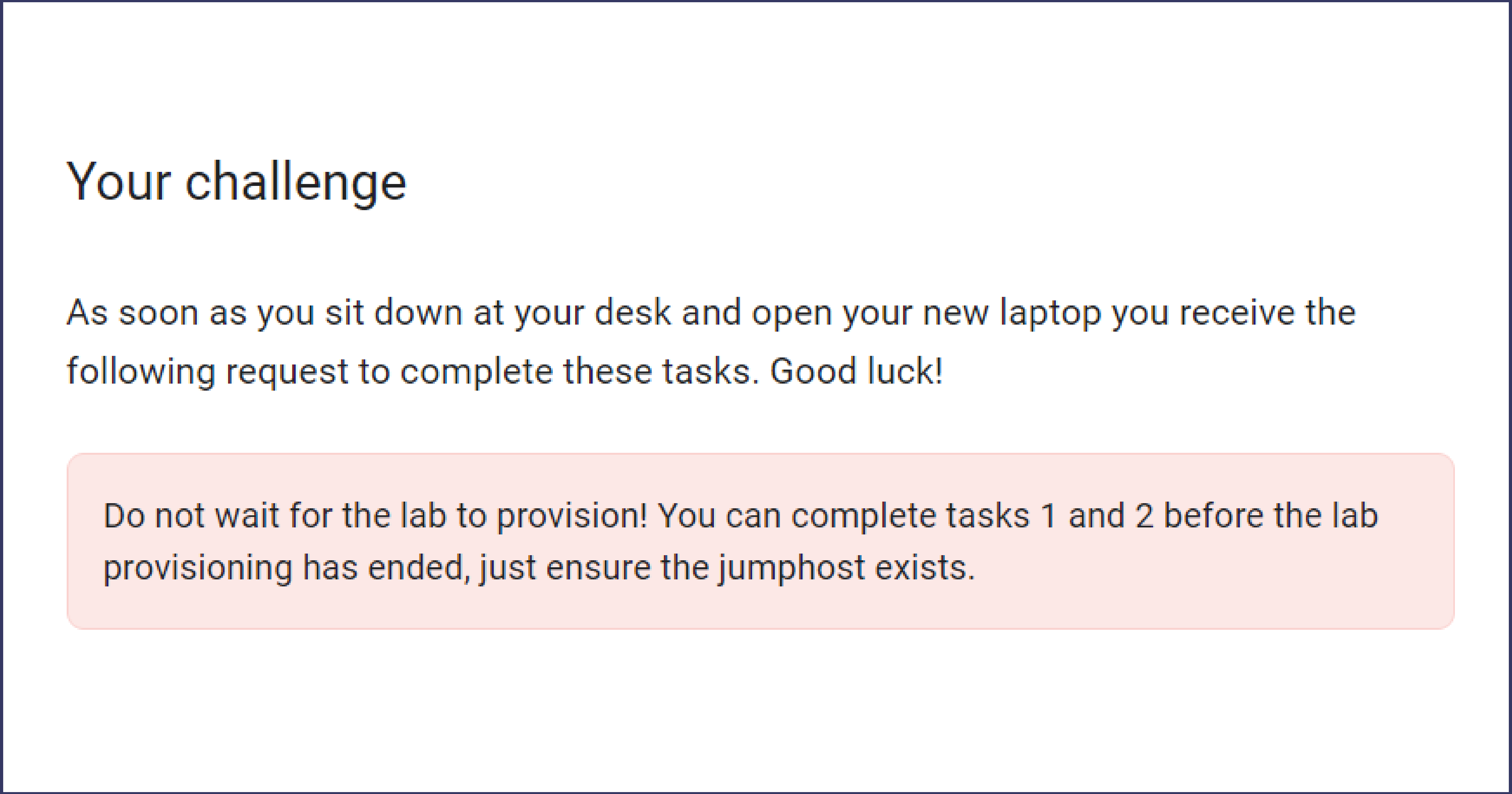
You have been asked to assist the kraken team complete setting up their product development environment. The previous Cloud Architect working with the kraken team was unfortunately too curious about if krakens were real or not, and has gone missing after venturing out into the open sea last weekend in search of such a beast.

Jooli Inc. management has supreme faith in your abilities, don't let them down! (Seriously, they don't need the dates to slip further).

The kraken team are building a next generation tool and they will host the application on Kubernetes. The project source code is stored in Cloud Source Repositories, with Spinnaker building and deploying any changes into the build Kubernetes environment.

Some Jooli Inc. standards you should follow:

- Create all resources in the `us-east1` region and `us-east1-b` zone, unless otherwise directed.
- Use the project VPCs.
- Naming is normally *team-resource*, e.g. an instance could be named **kraken-webserver1**.
- Allocate cost effective resource sizes. Projects are monitored and excessive resource use will result in the containing project's termination (and possibly yours), so beware. This is the guidance the monitoring team is willing to share; unless directed, use `n1-standard-1`.



Your challenge

As soon as you sit down at your desk and open your new laptop you receive the following request to complete these tasks. Good luck!

Do not wait for the lab to provision! You can complete tasks 1 and 2 before the lab provisioning has ended, just ensure the jumphost exists.

Task 1: Create the production environment

The previous Cloud Architect had written the **Deployment Manager** configuration to build the network for kraken's production environment. You can find the DM configuration on your **jumphost** in `/work/dm`. Create the network using the Deployment Manager configuration (`/work/dm/prod-network.yaml` and `/work/dm/prod-network.jinja`). Make sure you review the configuration before deploying it.

You will also need to create the Kubernetes environment. The application is already created and in the Container Repository.

- Create a two (2) node cluster called `kraken-prod` in the `kraken-prod-vpc` (remember to use `--num-nodes` to create 2 nodes only).
- Use `kubectl` with the files in `/work/k8s` to create the frontend and backend deployments and services (which will expose the frontend service via a load balancer).

The architecture is simple, the diagram below describes the environment. The new work you need to complete (VPC and Kubernetes cluster with services) is in the red box.

Compte engine > VM Instance.

Name: kraken-jumphost

us-east1-b

Login to jumphost via SSH button OR

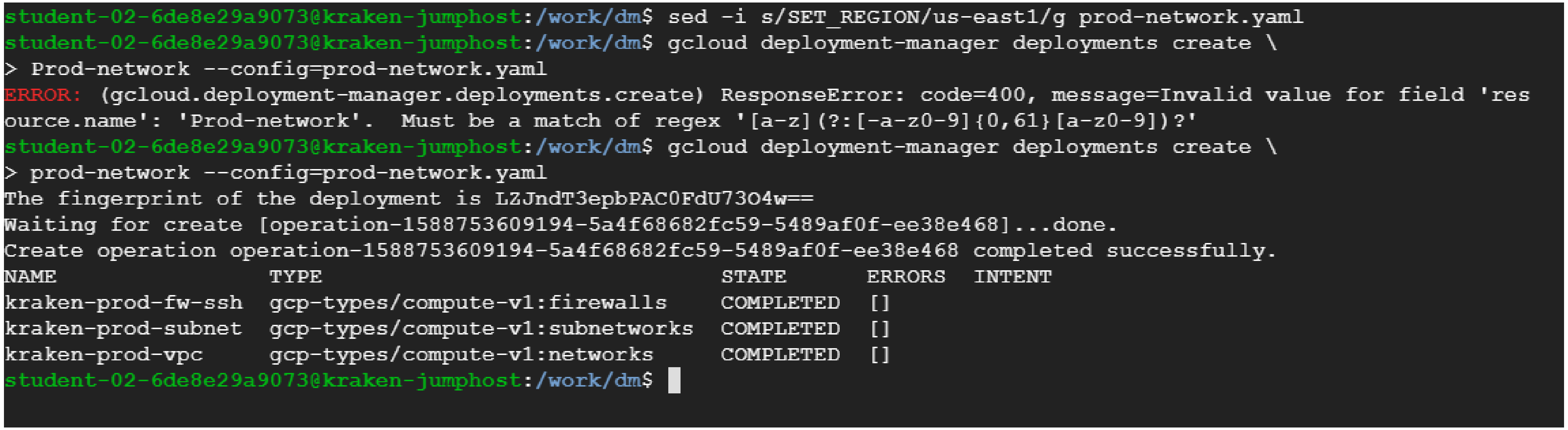
```
cloudshell gcloud compute ssh kraken-jumphost --internal-I --zone  us-east1-b
```

```
cd /work/dm

sed -i s/SET_REGION/us-east1/g prod-network.yaml


gcloud deployment-manager deployments create \

> prod-network --config=prod-network.yaml
```



```
student-02-6de8e29a9073@kraken-jumphost:/work/dm$ sed -i s/SET_REGION/us-east1/g prod-network.yaml
student-02-6de8e29a9073@kraken-jumphost:/work/dm$ gcloud deployment-manager deployments create \
> Prod-network --config=prod-network.yaml
ERROR: (gcloud.deployment-manager.deployments.create) ResponseError: code=400, message=Invalid value for field 'res
ource.name': 'Prod-network'.  Must be a match of regex '[a-z](?:[-a-z0-9]{0,61}[a-z0-9])?'
student-02-6de8e29a9073@kraken-jumphost:/work/dm$ gcloud deployment-manager deployments create \
> prod-network --config=prod-network.yaml
The fingerprint of the deployment is LZJndT3epbPAC0FdU73O4w==
Waiting for create [operation-1588753609194-5a4f68682fc59-5489af0f-ee38e468]...done.
Create operation operation-1588753609194-5a4f68682fc59-5489af0f-ee38e468 completed successfully.
NAME                TYPE                                STATE      ERRORS  INTENT
kraken-prod-fw-ssh  gcp-types/compute-v1:firewalls      COMPLETED  []
kraken-prod-subnet  gcp-types/compute-v1:subnetworks    COMPLETED  []
kraken-prod-vpc     gcp-types/compute-v1:networks       COMPLETED  []
student-02-6de8e29a9073@kraken-jumphost:/work/dm$
```

Creating the production Kubernetes cluster and launch the application

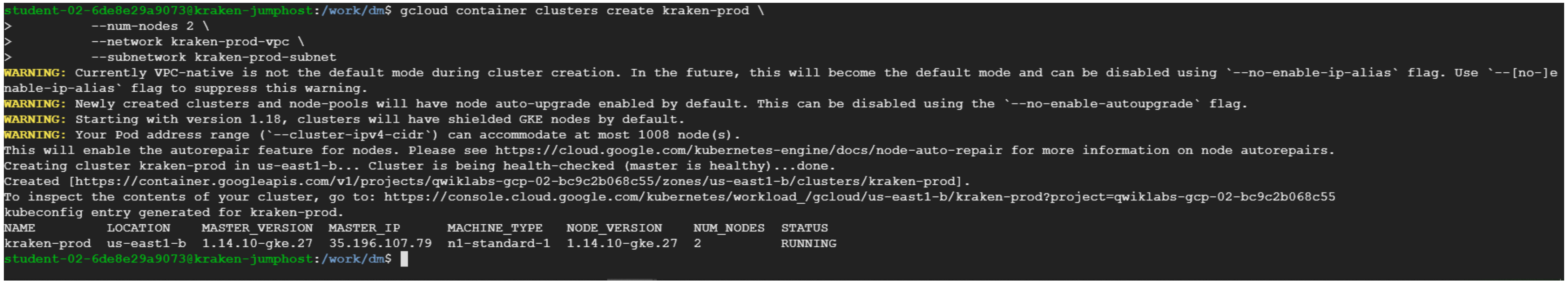
```
gcloud config set compute/zone us-east1-b

gcloud container clusters create kraken-pod\

--num-nodes 2\

--network kraken-prod-vpc\

--subnetwork kraken-prod-subnet
```

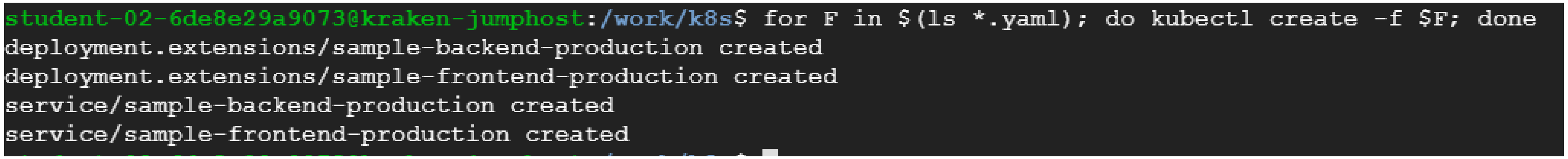


```
student-02-6de8e29a9073@kraken-jumphost:/work/dm$ gcloud container clusters create kraken-prod \
>          --num-nodes 2 \
>          --network kraken-prod-vpc \
>          --subnetwork kraken-prod-subnet
WARNING: Currently VPC-native is not the default mode during cluster creation. In the future, this will become the default mode and can be disabled using `--no-enable-ip-alias` flag. Use `--[no-]e
nable-ip-alias` flag to suppress this warning.
WARNING: Newly created clusters and node-pools will have node auto-upgrade enabled by default. This can be disabled using the `--no-enable-autoupgrade` flag.
WARNING: Starting with version 1.18, clusters will have shielded GKE nodes by default.
WARNING: Your Pod address range (`--cluster-ipv4-cidr`) can accommodate at most 1008 node(s).
This will enable the autorepair feature for nodes. Please see https://cloud.google.com/kubernetes-engine/docs/node-auto-repair for more information on node autorepairs.
Creating cluster kraken-prod in us-east1-b... Cluster is being health-checked (master is healthy)...done.
Created [https://container.googleapis.com/v1/projects/qwiklabs-gcp-02-bc9c2b068c55/zones/us-east1-b/clusters/kraken-prod].
To inspect the contents of your cluster, go to: https://console.cloud.google.com/kubernetes/workload_/gcloud/us-east1-b/kraken-prod?project=qwiklabs-gcp-02-bc9c2b068c55
kubeconfig entry generated for kraken-prod.
NAME         LOCATION    MASTER_VERSION  MASTER_IP      MACHINE_TYPE   NODE_VERSION    NUM_NODES  STATUS
kraken-prod  us-east1-b  1.14.10-gke.27  35.196.107.79  n1-standard-1  1.14.10-gke.27  2          RUNNING
student-02-6de8e29a9073@kraken-jumphost:/work/dm$
```

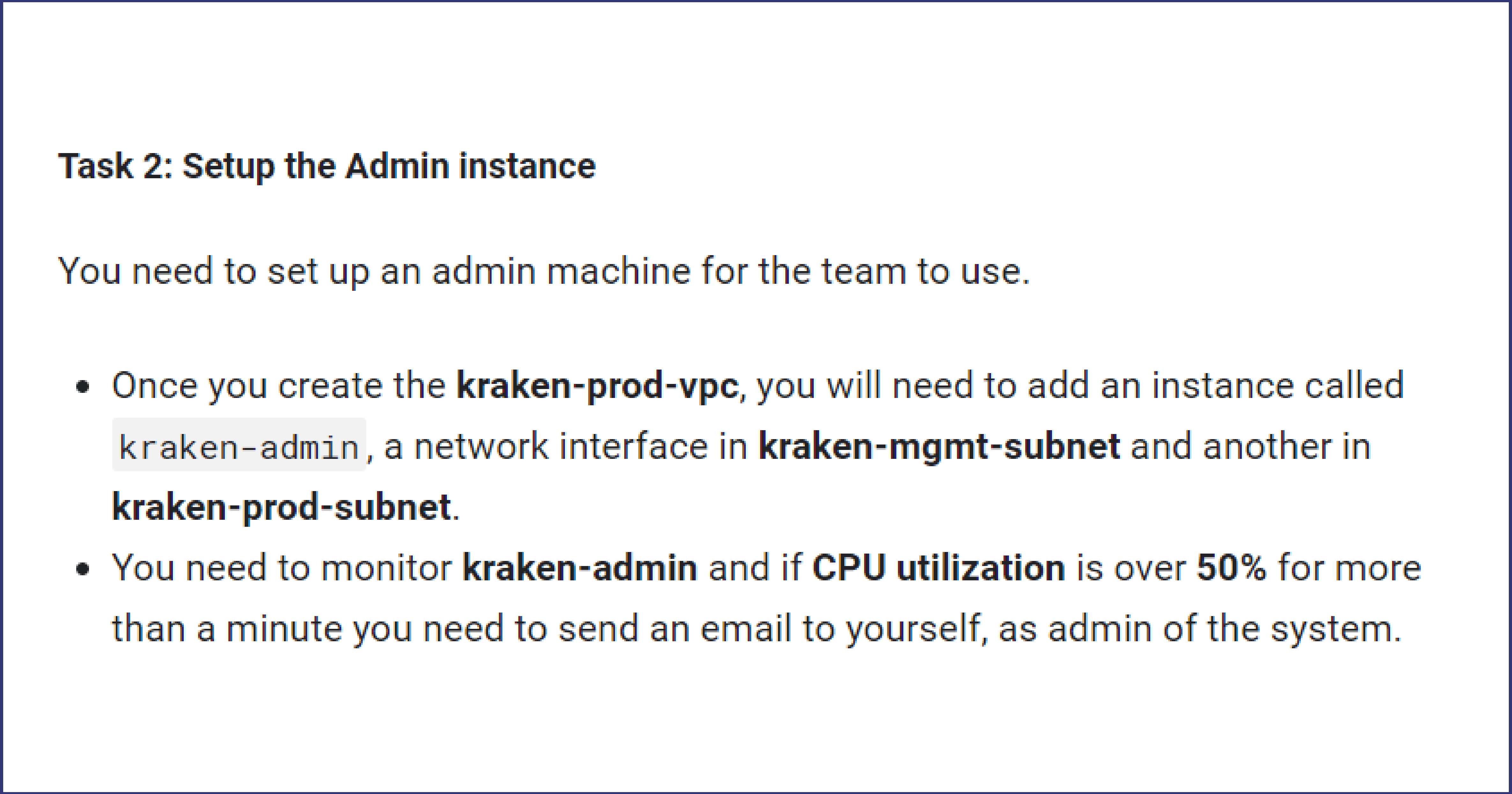
```
gcloud container clusters get-credentials kraken-prod

cd /work/k8s
```

for F in $(ls *.yaml); do kubectl create -f $F; done



```
student-02-6de8e29a9073@kraken-jumphost:/work/k8s$ for F in $(ls *.yaml); do kubectl create -f $F; done
deployment.extensions/sample-backend-production created
deployment.extensions/sample-frontend-production created
service/sample-backend-production created
service/sample-frontend-production created
```



Task 2: Setup the Admin instance

You need to set up an admin machine for the team to use.

- Once you create the **kraken-prod-vpc**, you will need to add an instance called `kraken-admin`, a network interface in **kraken-mgmt-subnet** and another in **kraken-prod-subnet**.
- You need to monitor **kraken-admin** and if **CPU utilization** is over **50%** for more than a minute you need to send an email to yourself, as admin of the system.

Create kraken-admin

gcloud config set compute/zone us-east1-b

gcloud compute instances create kraken-admin --network-interfaces="subnet=kraken-mgmt.-subnet" --network-interfaces="subnet=kraken-prod-subnet"

Create alert:

Open monitoring

Create policy

Name: Utilize_jumphost

Add Condition:

GCE VM Instance

CPU Utilization

Filter:

Instance_name

Value: kraken-admin

Configuration:

Threshold: 50

For: 1 minute

ADD

Add notification channel

Cp username qwik acc

Save

Task 3: Verify the Spinnaker deployment

Do not start this task until the lab has completed provisioning the lab resources.

The previous architect set up Spinnaker in **kraken-build-vpc**. Please connect to the Spinnaker console and verify that the built resources are working.

To access the Spinnaker console use Cloud Shell and `kubectl` to **port forward** the **spin-deck** pod from port **9000** to **8080** and then use Cloud Shell's **web preview**.

You must test that a change to the source code will result in the automated deployment of the new build. You should pull the **sample-app** repository to make the changes. Make sure you push a **new, updated, tag**.

WAIT TO END PROVISION IN LAB

Use cloudshell & run

```
gcloud config set compute/zone us-east1-b

gcloud container clusters get-credentials spinnaker-tutorial


DECK_POD=$(kubectl get pods --namespace default \

>            -l "cluster=spin-deck" -o jsonpath="{.items[0].metadata.name}")


kubectl port-forward --namespace default $DECK_POD 8080:9000 >> /dev/null &
```

Go to cloudshell webpreview & go to application -> sample

Open pipelines and manually run the pipeline if it has not already running.

Approve the deployment to production.

Check the production frontend endpoint.

Back in cloudshell run these commands to push a change

```
gcloud config set compute/zone us-east1-b

gcloud source repos clone sample-app

cd sample-app

touch a
```

```
git config --global user.email "$(gcloud config get-value core/account)“

git config --global user.name “Student”

git commit –a -m “change"

git tag v1.0.1

git push --tags
```