Commands Used

Build and push a Docker image to the registry

```
docker build -t <proj name> <directory>
#add -q for less clutter on the screen
docker run -d -p <port>:<port on localhost> <name>
docker tag <name> <user/name:v>
docker push <user/name:v>
```

Set up a Kubernetes cluster inside Virtualbox (using Vagrant)

```
vagrant up
vagrant ssh
sudo su
curl -sfL https://get.k3s.io | sh -
k3s kubectl get node
cd /etc/rancher/k3s
vi k3s.yaml
kubectl get no -o wide
```

Create deployment

```
kubectl create deploy go-helloworld --image=nandiniproothi/go-helloworld:v1.0.0
```

If your pod gets an CreateContainerError, try running this command: `zypper install -t pattern apparmor` this uses SUSE's CLI `zypper` which can install, update, remove, and manage repositories `apparmor_parser` loads `apparmor` (a linux security module) profiles into the linux kernel

To check if the application is running in the container, we can forward the port

```
kubectl port-forward deployment/go-helloworld 6111:6111 --address 0.0.0.0
```

Now, check http://192.168.50.4:6111/ (check your Vagrantfile for the IP)

```
# st the static IP for the vagrant box
config.vm.network "private_network", ip: "192.168.50.4"
```

Edit the deployment's yaml file, change the version, pull that image from docker hub and run it on a different port. The RS is created on its own.

```
kubectl edit deploy go-helloworld -o yaml
/image
a
<change v1.0.0 to v2.0.0>
esc
:wq
kubectl get rs
kubectl get po
```

To expose the deployment to a service

```
kubectl expose deploy go-helloworld --port=6112 --target-port=6112
kubectl get svc
```

```
localhost:~ # kubectl get deploy
NAME            READY   UP-TO-DATE   AVAILABLE   AGE
go-helloworld   1/1     1            1           123m
localhost:~ # kubectl get po
NAME                             READY   STATUS    RESTARTS   AGE
go-helloworld-67ccd6486-jdrhm    1/1     Running   0          9m7s
localhost:~ # kubectl get svc
NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP   10.43.0.1    <none>        443/TCP   19h
localhost:~ # kubectl expose deploy go-helloworld --port=6112 --target-port=6112
service/go-helloworld exposed
localhost:~ # kubectl get svc
NAME            TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)    AGE
kubernetes      ClusterIP   10.43.0.1       <none>        443/TCP    19h
go-helloworld   ClusterIP   10.43.200.247   <none>        6112/TCP   102s
```

Note: ClusterIP is the default service type and TCP is the default network protocol for a service

To check if the service is accessible

```
kubectl run test-$RANDOM --namespace=default --rm -it --image=alpine -- sh
#creates a test pod and opens the command prompt. terminates the pod once the shell is
closed
wget -qO- 10.43.200.247:6112
```

To create a configmap, secret, and namespace

```
kubectl create cm test-cm --from-literal=color=blue
kubectl create secret generic test-sec --from-literal=color=blue
kubectl create ns test-udacity kubec
```

```
localhost:~ # kubectl create secret generic test-sec --from-literal=color=red
secret/test-sec created
localhost:~ # kubectl describe secret test-sec
Name:         test-sec
Namespace:    default
Labels:       <none>
Annotations:  <none>

Type:  Opaque

Data
====
color:  3 bytes
localhost:~ # kubectl get secret test-sec -o yaml
apiVersion: v1
data:
  color: cmVk
kind: Secret
metadata:
  creationTimestamp: "2021-07-26T08:00:07Z"
  name: test-sec
  namespace: default
  resourceVersion: "32104"
  uid: 8e4ed52c-0094-4e7a-8f97-06d782c70092
type: Opaque
```

For decoding the base64 value

```
echo "<string>" | base64 -d
#echo "cmVk" | base64 -d
```

To create a pod in a namespace

```
kubectl run test-$RANDOM --namespace=test-udacity --image=nginx
kubectl get po -n test-udacity
```

Get logs of a resource in a namespace

```
localhost:~ # kubectl logs po/test-7259 -n test-udacity
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2021/07/26 08:12:26 [notice] 1#1: using the "epoll" event method
2021/07/26 08:12:26 [notice] 1#1: nginx/1.21.1
2021/07/26 08:12:26 [notice] 1#1: built by gcc 8.3.0 (Debian 8.3.0-6)
2021/07/26 08:12:26 [notice] 1#1: OS: Linux 5.3.18-lp152.78-default
2021/07/26 08:12:26 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2021/07/26 08:12:26 [notice] 1#1: start worker processes
2021/07/26 08:12:26 [notice] 1#1: start worker process 32
2021/07/26 08:12:26 [notice] 1#1: start worker process 33
2021/07/26 08:12:26 [notice] 1#1: start worker process 34
2021/07/26 08:12:26 [notice] 1#1: start worker process 35
```

Exercise: Kubernetes Resources

```
kuebctl create ns demo
kubectl label ns demo tier=test
#verify by
#kubectl get ns demo --show-labels
kubectl create deploy nginx-alpine --image=nginx:alpine --namespace=demo --replicas=3
kubectl label deploy nginx-alpine app=nginx tag=alpine -n demo
kubectl expose deploy nginx-alpine --port=8111 --target-port=8111
kubectl create cm nginx-version --from-literal=version=alpine -n demo
```

```
localhost:~ # kubectl label deploy nginx-alpine app=nginx -n demo
error: 'app' already has a value (nginx-alpine), and --overwrite is false
localhost:~ # kubectl label deploy nginx-alpine tag=alpine -n demo
deployment.apps/nginx-alpine labeled
localhost:~ # kubectl get deploy nginx-alpine --show-labels -n demo
NAME           READY   UP-TO-DATE   AVAILABLE   AGE     LABELS
nginx-alpine   3/3     3            3           2m27s   app=nginx-alpine,tag=alpine
```

```
localhost:~ # kubectl describe cm nginx-version -n demo
Name:         nginx-version
Namespace:    demo
Labels:       <none>
Annotations:  <none>

Data
====
version:
----
alpine
Events:  <none>
```