*Picture yourself as an indomitable power filled with positive attitude and faith that you are achieving your goals.*

– Napolean Hill

**Weight: 10**

Take a backup of the etcd cluster and save it to /opt/etcd-backup.db.

* Backup Completed

**Weight: 10**

Create a Pod called redis-storage with image: redis:alpine with a Volume of type emptyDir that lasts for the life of the Pod.

Specs on the below.

* Pod named 'redis-storage' created
* Pod 'redis-storage' uses Volume type of emptyDir
* Pod 'redis-storage' uses volumeMount with mountPath = /data/redis

**Weight: 8**

Create a new pod called super-user-pod with image busybox:1.28. Allow the pod to be able to set system\_time.

The container should sleep for 4800 seconds.

* Pod: super-user-pod
* Container Image: busybox:1.28
* Is SYS\_TIME capability set for the container?

**Weight: 12**

A pod definition file is created at /root/CKA/use-pv.yaml. Make use of this manifest file and mount the persistent volume called pv-1. Ensure the pod is running and the PV is bound.

mountPath: /data  
  
persistentVolumeClaim Name: my-pvc

* persistentVolume Claim configured correctly
* pod using the correct mountPath
* pod using the persistent volume claim?

**Weight: 15**

Create a new deployment called nginx-deploy, with image nginx:1.16 and 1 replica. Next upgrade the deployment to version 1.17 using rolling update.

* Deployment : nginx-deploy. Image: nginx:1.16
* Image: nginx:1.16
* Task: Upgrade the version of the deployment to 1:17
* Task: Record the changes for the image upgrade

**Weight: 15**

Create a new user called john. Grant him access to the cluster. John should have permission to create, list, get, update and delete pods in the development namespace . The private key exists in the location: /root/CKA/john.key and csr at /root/CKA/john.csr.

Important Note: As of kubernetes 1.19, the CertificateSigningRequest object expects a signerName.  
  
Please refer the documentation to see an example. The documentation tab is available at the top right of terminal.

* CSR: john-developer Status:Approved
* Role Name: developer, namespace: development, Resource: Pods
* Access: User 'john' has appropriate permissions

**Weight: 15**

Create a nginx pod called nginx-resolver using image nginx, expose it internally with a service called nginx-resolver-service. Test that you are able to look up the service and pod names from within the cluster. Use the image: busybox:1.28 for dns lookup. Record results in /root/CKA/nginx.svc and /root/CKA/nginx.pod

* Pod: nginx-resolver created
* Service DNS Resolution recorded correctly
* Pod DNS resolution recorded correctly

Use the command kubectl run and create a nginx pod and busybox pod. Resolve it, nginx service and its pod name from busybox pod.

To create a pod nginx-resolver and expose it internally:

kubectl run nginx-resolver --image=nginx

kubectl expose pod nginx-resolver --name=nginx-resolver-service --port=80 --target-port=80 --type=ClusterIP

To create a pod test-nslookup. Test that you are able to look up the service and pod names from within the cluster:

kubectl run test-nslookup --image=busybox:1.28 --rm -it --restart=Never -- nslookup nginx-resolver-service

kubectl run test-nslookup --image=busybox:1.28 --rm -it --restart=Never -- nslookup nginx-resolver-service > /root/CKA/nginx.svc

Get the IP of the nginx-resolver pod and replace the dots(.) with hyphon(-) which will be used below.

kubectl get pod nginx-resolver -o wide

kubectl run test-nslookup --image=busybox:1.28 --rm -it --restart=Never -- nslookup <P-O-D-I-P.default.pod> > /root/CKA/nginx.pod

**Weight: 15**

Create a static pod on node01 called nginx-critical with image nginx and make sure that it is recreated/restarted automatically in case of a failure.

Use /etc/kubernetes/manifests as the Static Pod path for example.

* static pod configured under /etc/kubernetes/manifests ?
* Pod nginx-critical-node01 is up and running