1- Create a pod named volume-share-datacenter.

For first container, use image centos:latest, container should be named as volume-container-datacenter-1, and run a command '/bin/bash', '-c' and 'sleep 10000'. Volume volume-share should be mounted at path /tmp/news.

For second container, use image centos:latest, container should be named as volume-container-datacenter-2, and run a command '/bin/bash', '-c' and 'sleep 10000'. Volume volume-share should be mounted at path /tmp/cluster.

Volumes to be named as volume-share and use emptyDir: { }.

After creating the pod, exec into the first container volume-container-datacenter-1, and create a file news.txt with content Welcome from datacenter-1! under the mount path of first container /tmp/news.

The file news.txt should be present under the mounted path /tmp/cluster of second container volume-container-datacenter-2 as they are using shared volumes.

2- Create a pod named webserver.

Create an emptyDir volume name: shared-logs.

Create two containers from nginx and ubuntu images with latest tag only and remember to mention tag i.e nginx:latest, nginx container name should be nginx-container and ubuntu container name should be sidecar-container on webserver pod.

Add command on sidecar-container "sh","-c","while true; do cat /var/log/nginx/access.log /var/log/nginx/error.log; sleep 30; done"

Mount volume /var/log/nginx on both containers, all containers should be up and running.

3- Create a new service account with the name `pvviewer`. Grant this Service account access to `list` all PersistentVolumes in the cluster by creating an appropriate cluster role called `pvviewer-role` and ClusterRoleBinding called `pvviewer-role-binding`.

Deploy HaProxy

1- Create a namespace haproxy-controller-devops.

2- Create a ServiceAccount haproxy-service-account-devops under the same namespace.

3- Create a ClusterRole which should be named as haproxy-cluster-role-devops, to grant permissions "get", "list", "watch", "create", "patch", "update" to "Configmaps",”secrets”,"endpoints","nodes","pods","services", "namespaces","events","serviceaccounts".

4- Create a ClusterRoleBinding which should be named as haproxy-cluster-role-binding-devops under the same namespace. Define roleRef apiGroup should be rbac.authorization.k8s.io, kind should be ClusterRole, name should be haproxy-cluster-role-devops and subjects kind should be ServiceAccount, name should be haproxy-service-account-devops and namespace should be haproxy-controller-devops.

5- Create a backend deployment which should be named as backend-deployment-devops under the same namespace,
labels “run=ingress-default-backend”,
replicas=1
container name=backend-container-devops,
image gcr.io/google_containers/defaultbackend:1.0
containerPort=8080.

6- Create a service for backend which should be named as “service-backend-devops” under the same namespace, port should be named as port-backend.

7- Create a deployment for frontend which should be named haproxy-ingress-devops under the same namespace.
replicas=1,
labels “run=haproxy-ingress”.
container name=ingress-container-devops
service account=haproxy-service-account-devops
image=haproxytech/kubernetes-ingress, give

args as --default-backend-service=haproxy-controller-devops/service-backend-devops,
resources
  requests
    cpu=500m
    memory=50Mi
livenessProbe
   httpGet path should be /healthz its port should be 1024.
The first port name should be http and its containerPort should be 80,
second port name should be https and its containerPort should be 443
third port name should be stat its containerPort should be 1024.
Define environment as TZ=Etc/UTC,
POD_NAME its
valueFrom:
 fieldRef:
    fieldPath: metadata.name
POD_NAMESPACE its
 valueFrom:
   fieldRef:
     fieldPath: metadata.namespace.

8- Create a service for frontend which should be named as ingress-service-devops under same namespace, type should be NodePort. The first port name should be http, its port should be 80, protocol should be TCP, targetPort should be 80 and nodePort should be 32456. The second port name should be https, its port should be 443, protocol should be TCP, targetPort should be 443 and nodePort should be 32567. The third port name should be stat, its port should be 1024, protocol should be TCP, targetPort should be 1024 and nodePort should be 32678.

9- Access the proxy states on port “1024” via the nodePort service “ingress-service-devops” via browser