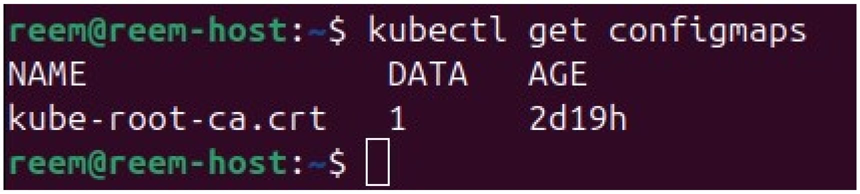
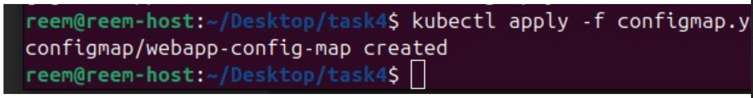
1- How many `ConfigMaps` exist in the environment?



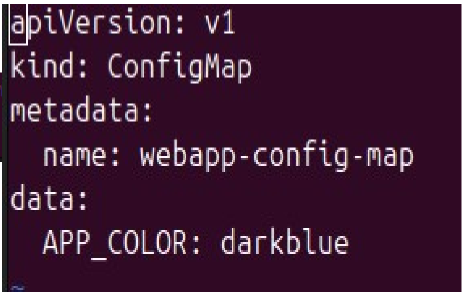
```
reem@reem-host:~$ kubectl get configmaps
NAME               DATA   AGE
kube-root-ca.crt   1      2d19h
reem@reem-host:~$ 
```

2- Create a new ConfigMap Use the spec given below.

ConfigName Name: webapp-config-map
Data: APP_COLOR=darkblue

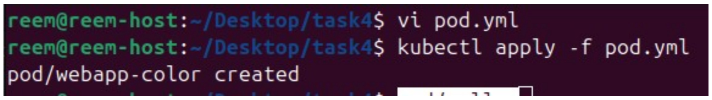


```
reem@reem-host:~/Desktop/task4$ kubectl apply -f configmap.y
configmap/webapp-config-map created
reem@reem-host:~/Desktop/task4$ 
```

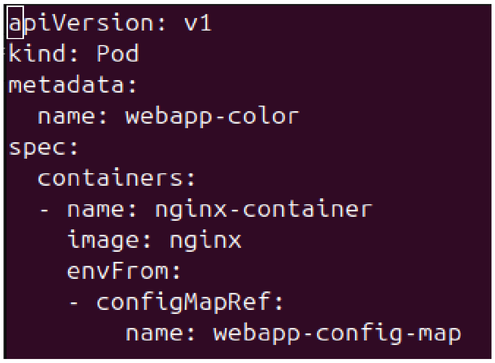


```
apiVersion: v1
kind: ConfigMap
metadata:
  name: webapp-config-map
data:
  APP_COLOR: darkblue
```

3- Create a webapp-color POD with nginx image and use the created ConfigMap

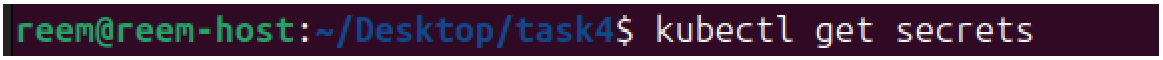


```
reem@reem-host:~/Desktop/task4$ vi pod.yml
reem@reem-host:~/Desktop/task4$ kubectl apply -f pod.yml
pod/webapp-color created
```



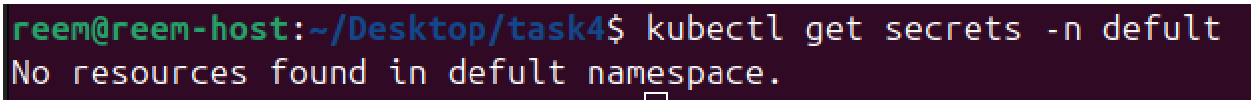
```
apiVersion: v1
kind: Pod
metadata:
  name: webapp-color
spec:
  containers:
  - name: nginx-container
    image: nginx
    envFrom:
    - configMapRef:
        name: webapp-config-map
```

4- How many `Secrets` exist on the system?



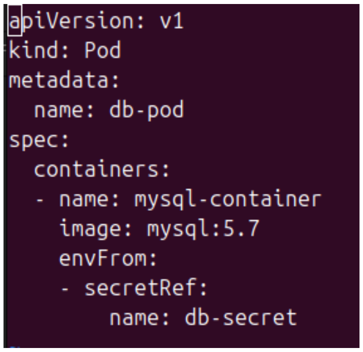
```
reem@reem-host:~/Desktop/task4$ kubectl get secrets
```

5- How many secrets are defined in the `default-token` secret?

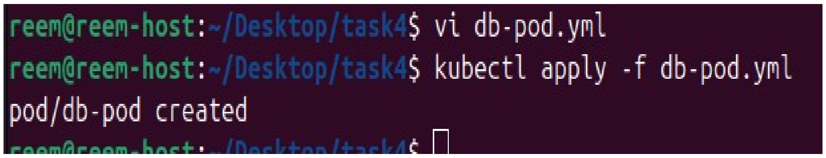


```
reem@reem-host:~/Desktop/task4$ kubectl get secrets -n defult
No resources found in defult namespace.
```

6- create a POD called db-pod with the image mysql:5.7 then check the POD status



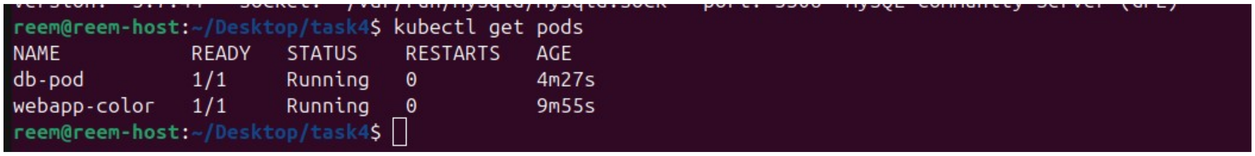
```
apiVersion: v1
kind: Pod
metadata:
  name: db-pod
spec:
  containers:
  - name: mysql-container
    image: mysql:5.7
    envFrom:
    - secretRef:
        name: db-secret
```



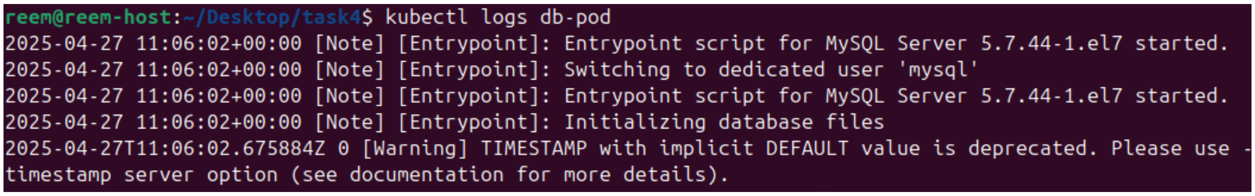
```
reem@reem-host:~/Desktop/task4$ vi db-pod.yml
reem@reem-host:~/Desktop/task4$ kubectl apply -f db-pod.yml
pod/db-pod created
reem@reem-host:~/Desktop/task4$ 
```

7- why the db-pod status not ready

Because it is missing required environment variables:



```
reem@reem-host:~/Desktop/task4$ kubectl get pods
NAME           READY   STATUS    RESTARTS   AGE
db-pod         1/1     Running   0          4m27s
webapp-color   1/1     Running   0          9m55s
reem@reem-host:~/Desktop/task4$ 
```



```
reem@reem-host:~/Desktop/task4$ kubectl logs db-pod
2025-04-27 11:06:02+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 5.7.44-1.el7 started.
2025-04-27 11:06:02+00:00 [Note] [Entrypoint]: Switching to dedicated user 'mysql'
2025-04-27 11:06:02+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 5.7.44-1.el7 started.
2025-04-27 11:06:02+00:00 [Note] [Entrypoint]: Initializing database files
2025-04-27T11:06:02.675884Z 0 [Warning] TIMESTAMP with implicit DEFAULT value is deprecated. Please use -
timestamp server option (see documentation for more details).
```

8- Create a new secret named `db-secret` with the data given below.

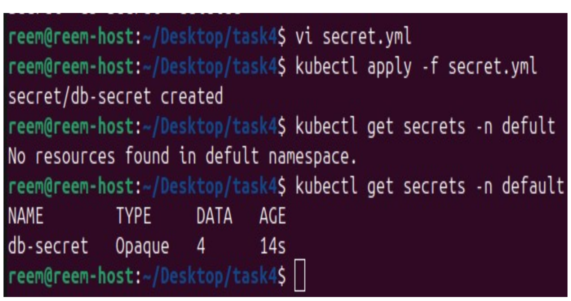
Secret Name: db-secret

Secret 1: MYSQL_DATABASE=sql01

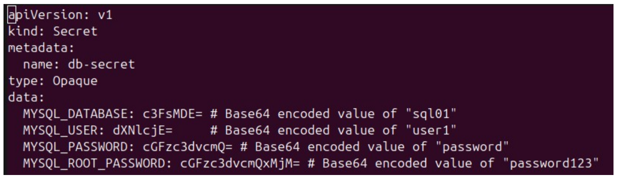
Secret 2: MYSQL_USER=user1

Secret3: MYSQL_PASSWORD=password

Secret 4: MYSQL_ROOT_PASSWORD=password123



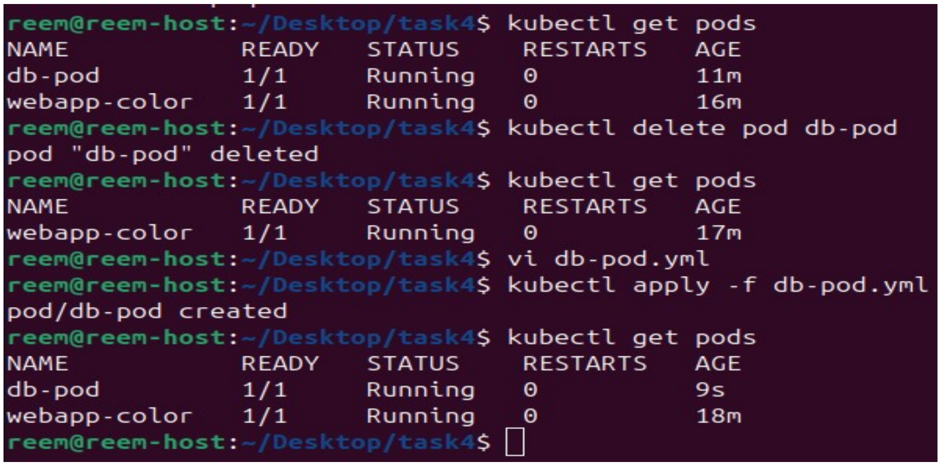
```
reem@reem-host:~/Desktop/task4$ vi secret.yml
reem@reem-host:~/Desktop/task4$ kubectl apply -f secret.yml
secret/db-secret created
reem@reem-host:~/Desktop/task4$ kubectl get secrets -n defult
No resources found in defult namespace.
reem@reem-host:~/Desktop/task4$ kubectl get secrets -n default
NAME        TYPE     DATA   AGE
db-secret   Opaque   4      14s
reem@reem-host:~/Desktop/task4$ 
```



```
apiVersion: v1
kind: Secret
metadata:
  name: db-secret
type: Opaque
data:
  MYSQL_DATABASE: c3FsMDE= # Base64 encoded value of "sql01"
  MYSQL_USER: dXNlcjE=     # Base64 encoded value of "user1"
  MYSQL_PASSWORD: cGFzc3dvcmQ= # Base64 encoded value of "password"
  MYSQL_ROOT_PASSWORD: cGFzc3dvcmQxMjM= # Base64 encoded value of "password123"
```

9- Configure `db-pod` to load environment variables from the newly created secret.

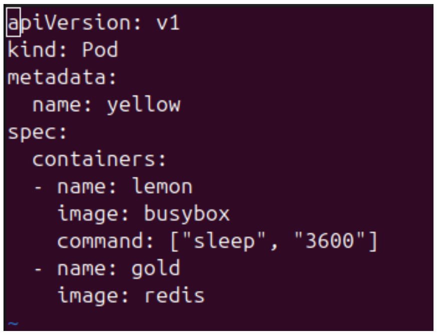
Delete and recreate the pod if required.



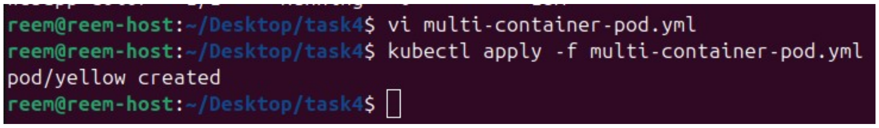
```
reem@reem-host:~/Desktop/task4$ kubectl get pods
NAME           READY   STATUS    RESTARTS   AGE
db-pod         1/1     Running   0          11m
webapp-color   1/1     Running   0          16m
reem@reem-host:~/Desktop/task4$ kubectl delete pod db-pod
pod "db-pod" deleted
reem@reem-host:~/Desktop/task4$ kubectl get pods
NAME           READY   STATUS    RESTARTS   AGE
webapp-color   1/1     Running   0          17m
reem@reem-host:~/Desktop/task4$ vi db-pod.yml
reem@reem-host:~/Desktop/task4$ kubectl apply -f db-pod.yml
pod/db-pod created
reem@reem-host:~/Desktop/task4$ kubectl get pods
NAME           READY   STATUS    RESTARTS   AGE
db-pod         1/1     Running   0          9s
webapp-color   1/1     Running   0          18m
reem@reem-host:~/Desktop/task4$ 
```

10- Create a multi-container pod with 2 containers. Name: yellow

Container 1 Name: lemon Container 1 Image: busybox
Container 2 Name: gold
Container 2 Image: redis

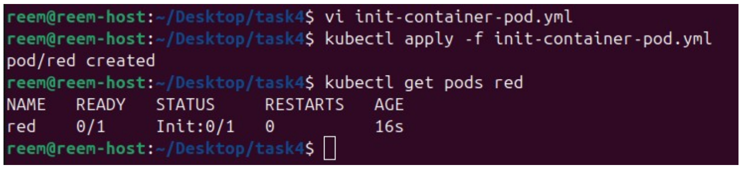


```
apiVersion: v1
kind: Pod
metadata:
  name: yellow
spec:
  containers:
  - name: lemon
    image: busybox
    command: ["sleep", "3600"]
  - name: gold
    image: redis
~
```

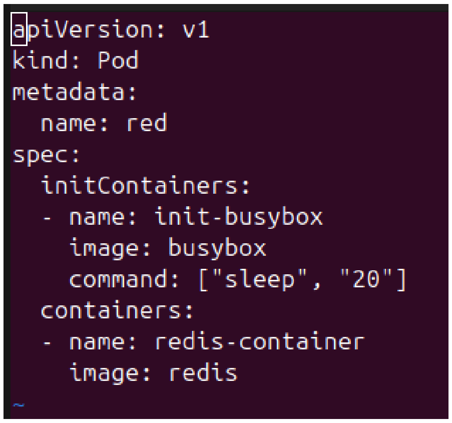


```
reem@reem-host:~/Desktop/task4$ vi multi-container-pod.yml
reem@reem-host:~/Desktop/task4$ kubectl apply -f multi-container-pod.yml
pod/yellow created
reem@reem-host:~/Desktop/task4$ 
```

11- Create a pod `red with redis image` and use an `initContainer` that uses the `busybox` image and `sleeps for 20` seconds



```
reem@reem-host:~/Desktop/task4$ vi init-container-pod.yml
reem@reem-host:~/Desktop/task4$ kubectl apply -f init-container-pod.yml
pod/red created
reem@reem-host:~/Desktop/task4$ kubectl get pods red
NAME    READY    STATUS      RESTARTS    AGE
red     0/1      Init:0/1    0           16s
reem@reem-host:~/Desktop/task4$ 
```



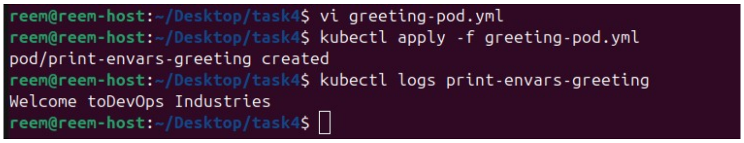
```
apiVersion: v1
kind: Pod
metadata:
  name: red
spec:
  initContainers:
  - name: init-busybox
    image: busybox
    command: ["sleep", "20"]
  containers:
  - name: redis-container
    image: redis
~
```

12- Create a pod named `print-envars-greeting`.

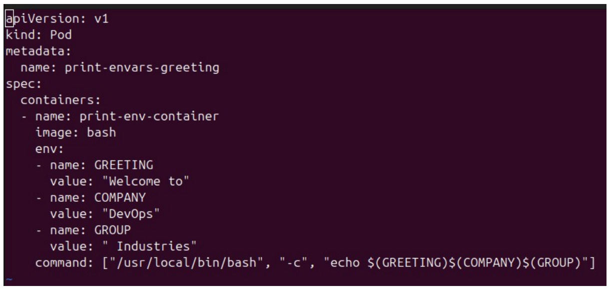
1. Configure spec as, the container name should be `print-env-container` and use `bash` image.
2. Create three environment variables:
   a. `GREETING` and its value should be “`Welcome to`”
   b. `COMPANY` and its value should be “`DevOps`”

c. `GROUP` and its value should be “`Industries`”

4. Use command to echo `["$(GREETING) $(COMPANY) $(GROUP)"]` message.
5. You can check the output using <`kubctl logs -f [ pod-name ]`> command.

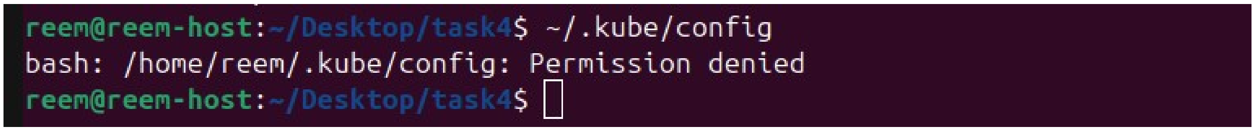


```
reem@reem-host:~/Desktop/task4$ vi greeting-pod.yml
reem@reem-host:~/Desktop/task4$ kubectl apply -f greeting-pod.yml
pod/print-envars-greeting created
reem@reem-host:~/Desktop/task4$ kubectl logs print-envars-greeting
Welcome toDevOps Industries
reem@reem-host:~/Desktop/task4$
```



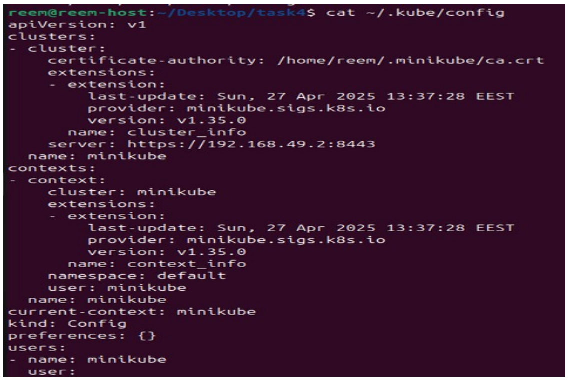
```
apiVersion: v1
kind: Pod
metadata:
  name: print-envars-greeting
spec:
  containers:
  - name: print-env-container
    image: bash
    env:
    - name: GREETING
      value: "Welcome to"
    - name: COMPANY
      value: "DevOps"
    - name: GROUP
      value: " Industries"
    command: ["/usr/local/bin/bash", "-c", "echo $(GREETING)$(COMPANY)$(GROUP)"]
~
```

13- Where is the default kubeconfig file located in the current environment?



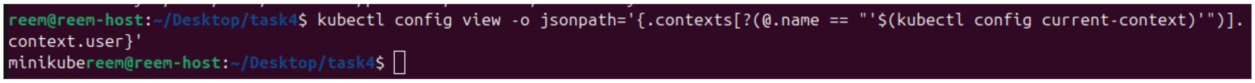
```
reem@reem-host:~/Desktop/task4$ ~/.kube/config
bash: /home/reem/.kube/config: Permission denied
reem@reem-host:~/Desktop/task4$
```

14- How many clusters are defined in the default kubeconfig file?



```
reem@reem-host:~/Desktop/task4$ cat ~/.kube/config
apiVersion: v1
clusters:
- cluster:
    certificate-authority: /home/reem/.minikube/ca.crt
    extensions:
    - extension:
        last-update: Sun, 27 Apr 2025 13:37:28 EEST
        provider: minikube.sigs.k8s.io
        version: v1.35.0
      name: cluster_info
    server: https://192.168.49.2:8443
  name: minikube
contexts:
- context:
    cluster: minikube
    extensions:
    - extension:
        last-update: Sun, 27 Apr 2025 13:37:28 EEST
        provider: minikube.sigs.k8s.io
        version: v1.35.0
      name: context_info
    namespace: default
    user: minikube
  name: minikube
current-context: minikube
kind: Config
preferences: {}
users:
- name: minikube
  user:
```

15- What is the user configured in the current context?



```
reem@reem-host:~/Desktop/task4$ kubectl config view -o jsonpath='{.contexts[?(@.name == "'$(kubectl config current-context)'")].
context.user}'
minikubereem@reem-host:~/Desktop/task4$
```

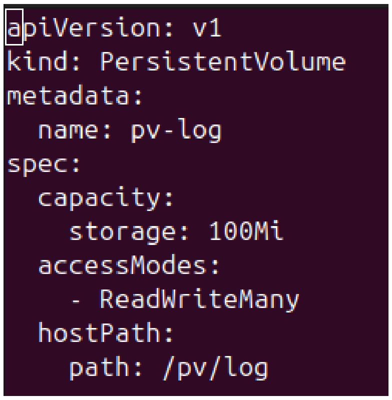
16- Create a `Persistent Volume` with the given specification.

Volume Name: pv-log

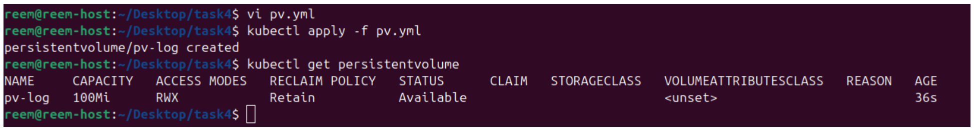
Storage: 100Mi

Access Modes: ReadWriteMany

Host Path: /pv/log



```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv-log
spec:
  capacity:
    storage: 100Mi
  accessModes:
    - ReadWriteMany
  hostPath:
    path: /pv/log
```



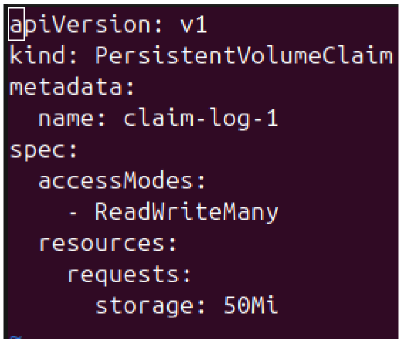
```
reem@reem-host:~/Desktop/task4$ vi pv.yml
reem@reem-host:~/Desktop/task4$ kubectl apply -f pv.yml
persistentvolume/pv-log created
reem@reem-host:~/Desktop/task4$ kubectl get persistentvolume
NAME     CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS      CLAIM   STORAGECLASS   VOLUMEATTRIBUTESCLASS   REASON   AGE
pv-log   100Mi      RWX            Retain           Available                          <unset>                          36s
reem@reem-host:~/Desktop/task4$ 
```

17- Create a `Persistent Volume Claim` with the given specification.

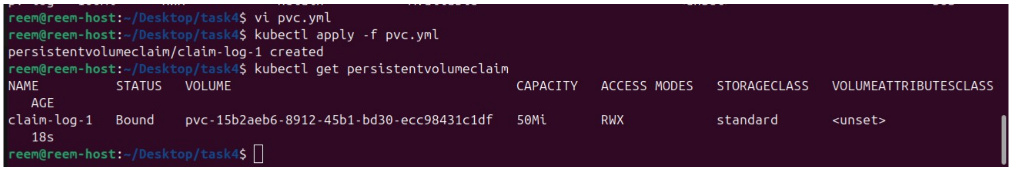
Volume Name: claim-log-1

Storage Request: 50Mi

Access Modes: ReadWriteMany



```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: claim-log-1
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 50Mi
```



```
reem@reem-host:~/Desktop/task4$ vi pvc.yml
reem@reem-host:~/Desktop/task4$ kubectl apply -f pvc.yml
persistentvolumeclaim/claim-log-1 created
reem@reem-host:~/Desktop/task4$ kubectl get persistentvolumeclaim
NAME          STATUS   VOLUME                                     CAPACITY   ACCESS MODES   STORAGECLASS   VOLUMEATTRIBUTESCLASS
   AGE
claim-log-1   Bound    pvc-15b2aeb6-8912-45b1-bd30-ecc98431c1df   50Mi       RWX            standard       <unset>
   18s
reem@reem-host:~/Desktop/task4$ 
```

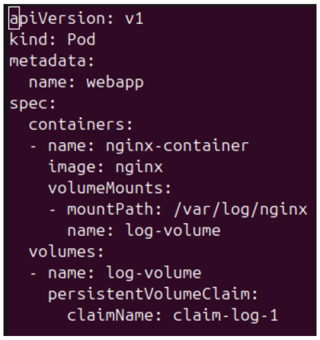
18- Create a `webapp` pod to use the persistent volume claim as its storage.

Name: webapp

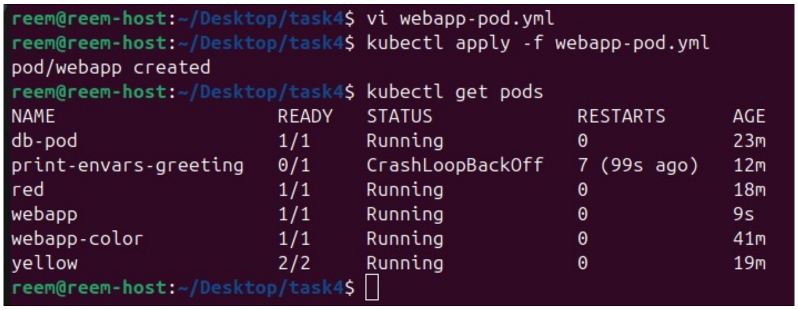
Image Name: nginx

Volume: PersistentVolumeClaim=claim-log-1

Volume Mount: /var/log/nginx



```
apiVersion: v1
kind: Pod
metadata:
  name: webapp
spec:
  containers:
  - name: nginx-container
    image: nginx
    volumeMounts:
    - mountPath: /var/log/nginx
      name: log-volume
  volumes:
  - name: log-volume
    persistentVolumeClaim:
      claimName: claim-log-1
```



```
reem@reem-host:~/Desktop/task4$ vi webapp-pod.yml
reem@reem-host:~/Desktop/task4$ kubectl apply -f webapp-pod.yml
pod/webapp created
reem@reem-host:~/Desktop/task4$ kubectl get pods
NAME                   READY   STATUS             RESTARTS      AGE
db-pod                 1/1     Running            0             23m
print-envars-greeting  0/1     CrashLoopBackOff   7 (99s ago)   12m
red                    1/1     Running            0             18m
webapp                 1/1     Running            0             9s
webapp-color           1/1     Running            0             41m
yellow                 2/2     Running            0             19m
reem@reem-host:~/Desktop/task4$ 
```