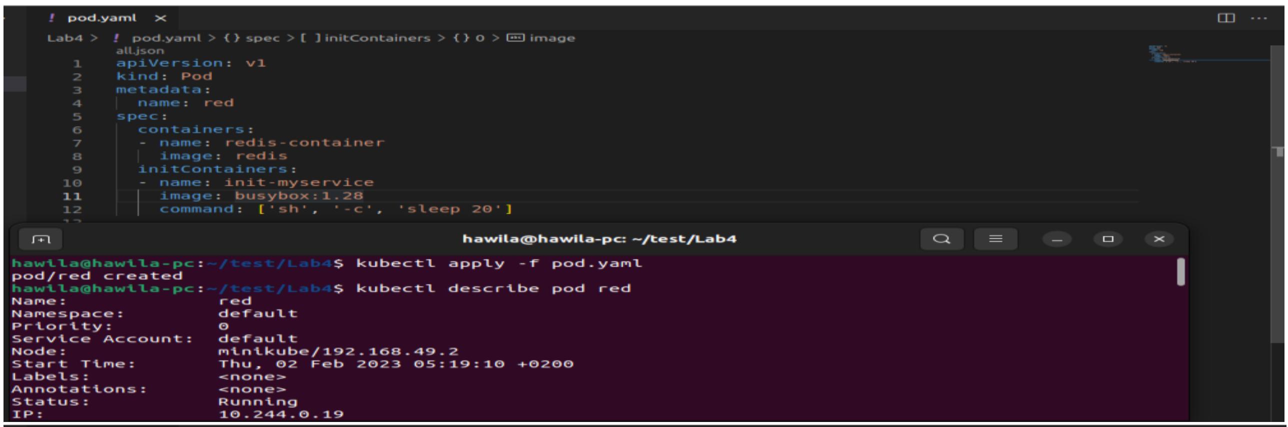
1- Create a pod red with redis image and use an initContainer that uses the busybox image and sleeps for 20 seconds

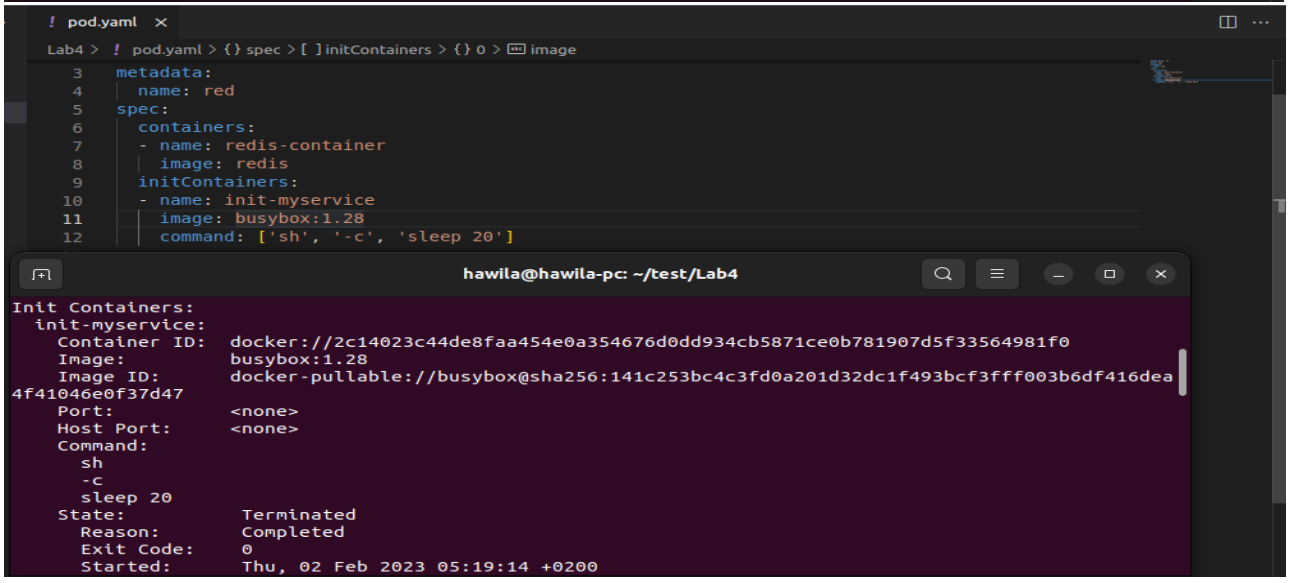


pod.yaml
Lab4 > pod.yaml > {} spec > [ ] initContainers > {} 0 > image
all.json

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: red
spec:
  containers:
  - name: redis-container
    image: redis
  initContainers:
  - name: init-myservice
    image: busybox:1.28
    command: ['sh', '-c', 'sleep 20']
```

hawila@hawila-pc: ~/test/Lab4

```
hawila@hawila-pc:~/test/Lab4$ kubectl apply -f pod.yaml
pod/red created
hawila@hawila-pc:~/test/Lab4$ kubectl describe pod red
Name:             red
Namespace:        default
Priority:         0
Service Account:  default
Node:             minikube/192.168.49.2
Start Time:       Thu, 02 Feb 2023 05:19:10 +0200
Labels:           <none>
Annotations:      <none>
Status:           Running
IP:               10.244.0.19
```



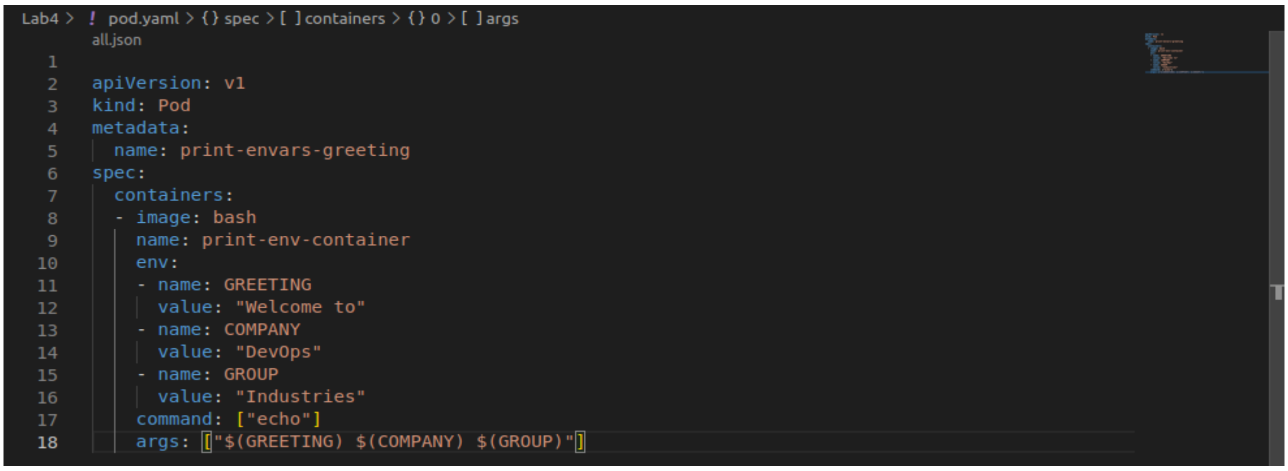
pod.yaml
Lab4 > pod.yaml > {} spec > [ ] initContainers > {} 0 > image

```yaml
metadata:
  name: red
spec:
  containers:
  - name: redis-container
    image: redis
  initContainers:
  - name: init-myservice
    image: busybox:1.28
    command: ['sh', '-c', 'sleep 20']
```

hawila@hawila-pc: ~/test/Lab4

```
Init Containers:
  init-myservice:
    Container ID:  docker://2c14023c44de8faa454e0a354676d0dd934cb5871ce0b781907d5f33564981f0
    Image:         busybox:1.28
    Image ID:      docker-pullable://busybox@sha256:141c253bc4c3fd0a201d32dc1f493bcf3fff003b6df416dea
4f41046e0f37d47
    Port:          <none>
    Host Port:     <none>
    Command:
      sh
      -c
      sleep 20
    State:          Terminated
      Reason:       Completed
      Exit Code:    0
      Started:      Thu, 02 Feb 2023 05:19:14 +0200
```

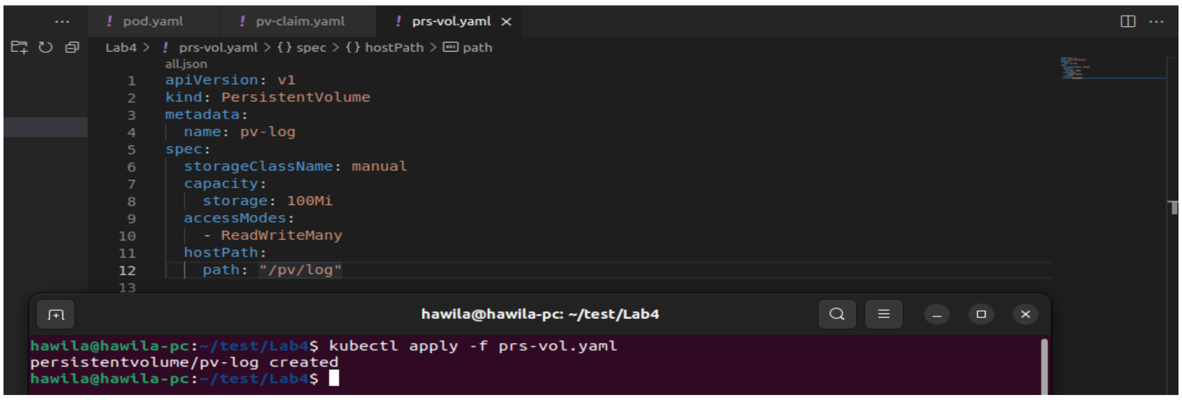
2- Create a pod named print-envars-greeting.



Lab4 > pod.yaml > {} spec > [ ] containers > {} 0 > [ ] args
all.json

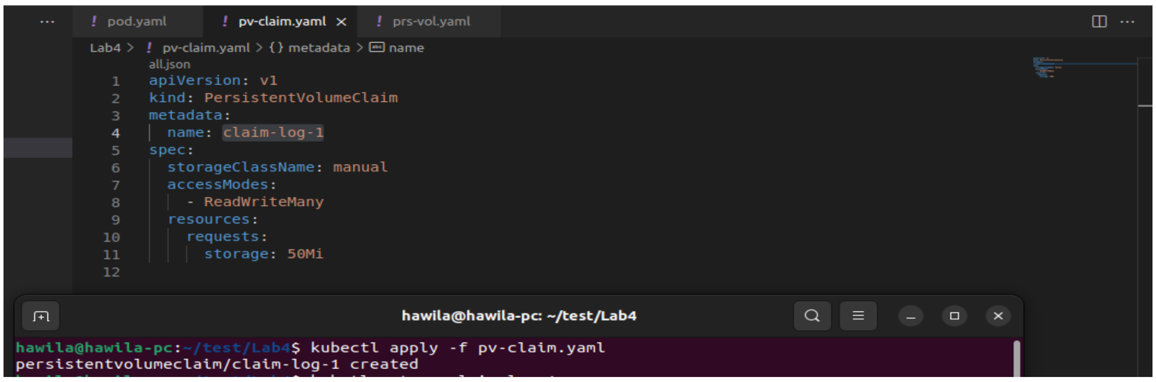
```yaml

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

3- Create a Persistent Volume with the given specification.

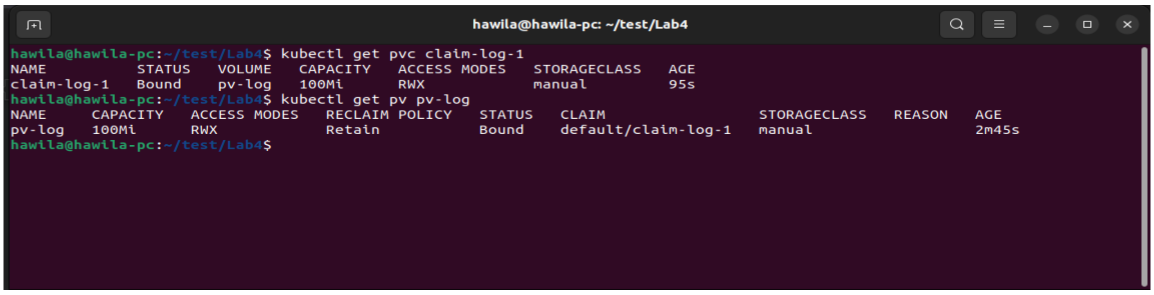

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv-log
spec:
  storageClassName: manual
  capacity:
    storage: 100Mi
  accessModes:
    - ReadWriteMany
  hostPath:
    path: "/pv/log"
```

```
hawila@hawila-pc:~/test/Lab4$ kubectl apply -f prs-vol.yaml
persistentvolume/pv-log created
hawila@hawila-pc:~/test/Lab4$
```

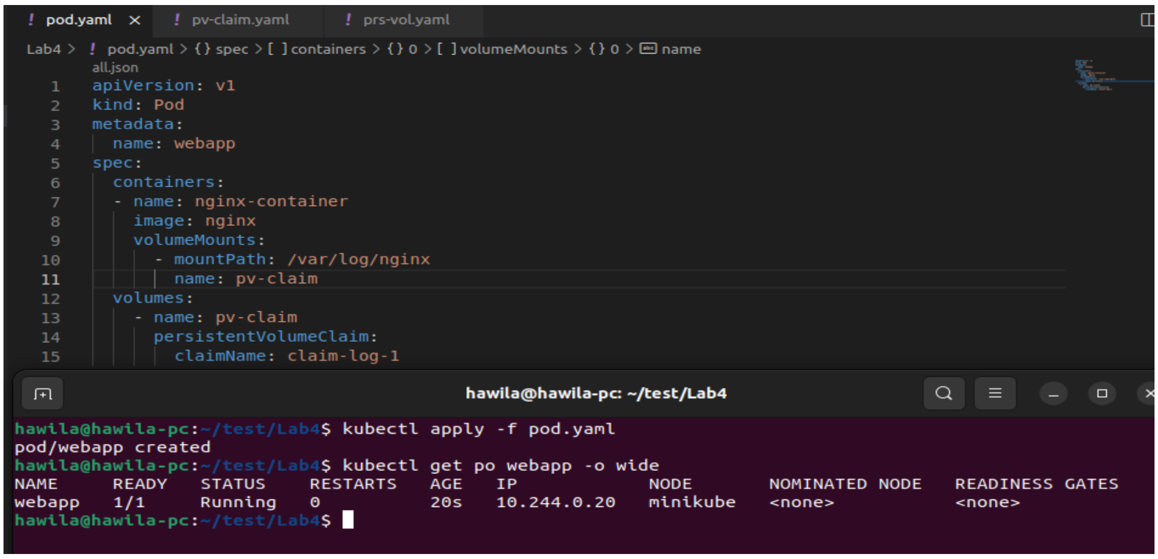
4- Create a Persistent Volume Claim with the given specification.


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

```
hawila@hawila-pc:~/test/Lab4$ kubectl apply -f pv-claim.yaml
persistentvolumeclaim/claim-log-1 created
```


```
hawila@hawila-pc:~/test/Lab4$ kubectl get pvc claim-log-1
NAME          STATUS   VOLUME   CAPACITY   ACCESS MODES   STORAGECLASS   AGE
claim-log-1   Bound    pv-log   100Mi      RWX            manual         95s
hawila@hawila-pc:~/test/Lab4$ kubectl get pv pv-log
NAME     CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS   CLAIM                 STORAGECLASS   REASON   AGE
pv-log   100Mi      RWX            Retain           Bound    default/claim-log-1   manual                  2m45s
hawila@hawila-pc:~/test/Lab4$
```

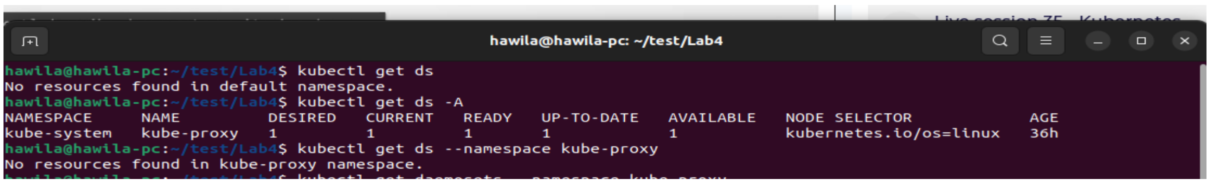
5- Create a webapp pod to use the persistent volume claim as its storage.


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

```
hawila@hawila-pc:~/test/Lab4$ kubectl apply -f pod.yaml
pod/webapp created
hawila@hawila-pc:~/test/Lab4$ kubectl get po webapp -o wide
NAME     READY   STATUS    RESTARTS   AGE   IP            NODE       NOMINATED NODE   READINESS GATES
webapp   1/1     Running   0          20s   10.244.0.20   minikube   <none>           <none>
hawila@hawila-pc:~/test/Lab4$
```

6- How many DaemonSets are created in the cluster in all namespaces?

7- what DaemonSets exist on the kube-system namespace?

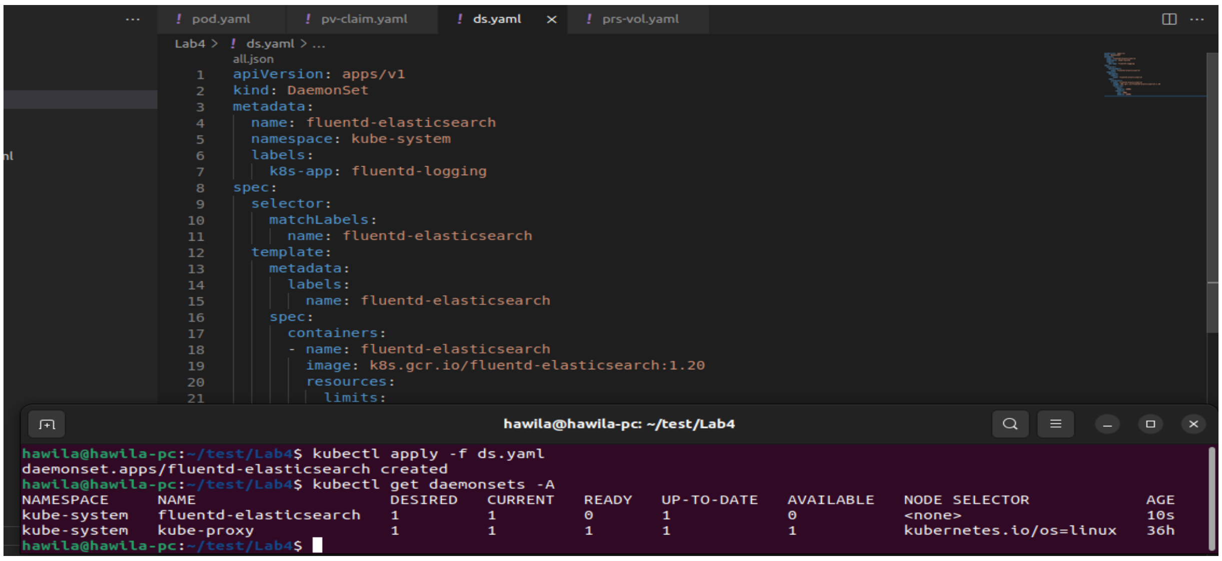


```
hawila@hawila-pc:~/test/Lab4$ kubectl get ds
No resources found in default namespace.
hawila@hawila-pc:~/test/Lab4$ kubectl get ds -A
NAMESPACE     NAME         DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR            AGE
kube-system   kube-proxy   1         1         1       1            1           kubernetes.io/os=linux   36h
hawila@hawila-pc:~/test/Lab4$ kubectl get ds --namespace kube-proxy
No resources found in kube-proxy namespace.
```

8- What is the image used by the POD deployed by the kube-proxy

DaemonSet

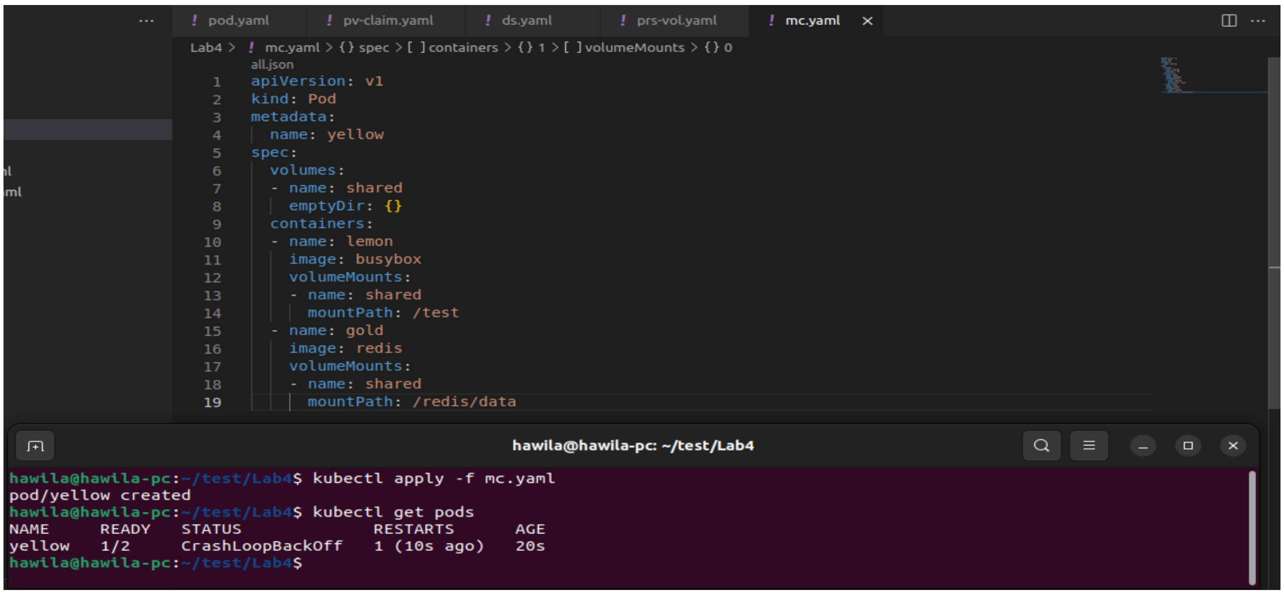
9- Deploy a DaemonSet for FluentD Logging.



```yaml
apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: fluentd-elasticsearch
  namespace: kube-system
  labels:
    k8s-app: fluentd-logging
spec:
  selector:
    matchLabels:
      name: fluentd-elasticsearch
  template:
    metadata:
      labels:
        name: fluentd-elasticsearch
    spec:
      containers:
      - name: fluentd-elasticsearch
        image: k8s.gcr.io/fluentd-elasticsearch:1.20
        resources:
          limits:
```

```
hawila@hawila-pc:~/test/Lab4$ kubectl apply -f ds.yaml
daemonset.apps/fluentd-elasticsearch created
hawila@hawila-pc:~/test/Lab4$ kubectl get daemonsets -A
NAMESPACE     NAME                    DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR            AGE
kube-system   fluentd-elasticsearch   1         1         0       1            0           <none>                   10s
kube-system   kube-proxy              1         1         1       1            1           kubernetes.io/os=linux   36h
hawila@hawila-pc:~/test/Lab4$
```

10- Create a multi-container pod with 2 containers.

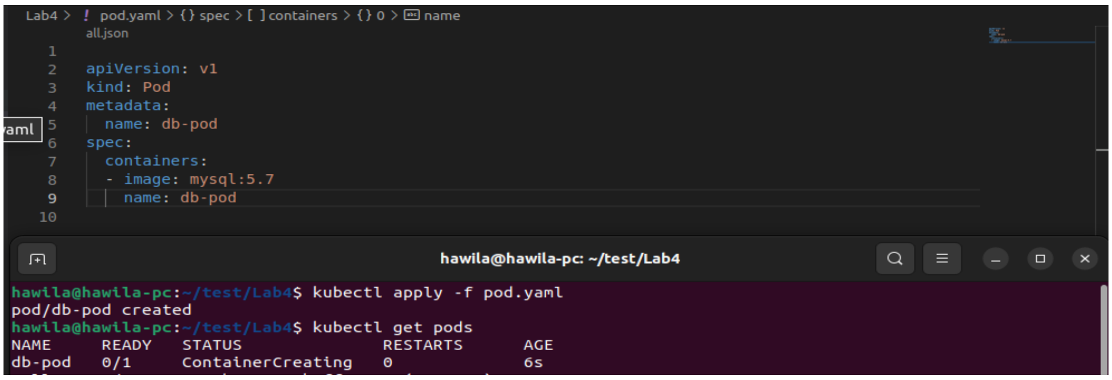

pod.yaml
pv-claim.yaml
ds.yaml
prs-vol.yaml
mc.yaml
Lab4 > mc.yaml > {} spec > [ ] containers > {} 1 > [ ] volumeMounts > {} 0
hawila@hawila-pc: ~/test/Lab4

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: yellow
spec:
  volumes:
  - name: shared
    emptyDir: {}
  containers:
  - name: lemon
    image: busybox
    volumeMounts:
    - name: shared
      mountPath: /test
  - name: gold
    image: redis
    volumeMounts:
    - name: shared
      mountPath: /redis/data
```

```
hawila@hawila-pc:~/test/Lab4$ kubectl apply -f mc.yaml
pod/yellow created
hawila@hawila-pc:~/test/Lab4$ kubectl get pods
NAME     READY   STATUS             RESTARTS      AGE
yellow   1/2     CrashLoopBackOff   1 (10s ago)   20s
hawila@hawila-pc:~/test/Lab4$
```

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

POD status

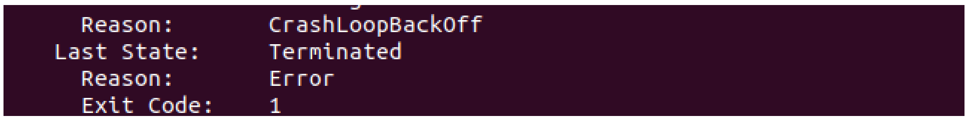

Lab4 > ! pod.yaml > {} spec > [ ] containers > {} 0 > name
all.json

```yaml

apiVersion: v1
kind: Pod
metadata:
  name: db-pod
spec:
  containers:
  - image: mysql:5.7
    name: db-pod

```

hawila@hawila-pc: ~/test/Lab4

```
hawila@hawila-pc:~/test/Lab4$ kubectl apply -f pod.yaml
pod/db-pod created
hawila@hawila-pc:~/test/Lab4$ kubectl get pods
NAME      READY   STATUS              RESTARTS     AGE
db-pod    0/1     ContainerCreating   0            6s
```

12- why the db-pod status not ready



```
      Reason:       CrashLoopBackOff
    Last State:     Terminated
      Reason:       Error
      Exit Code:    1
```

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

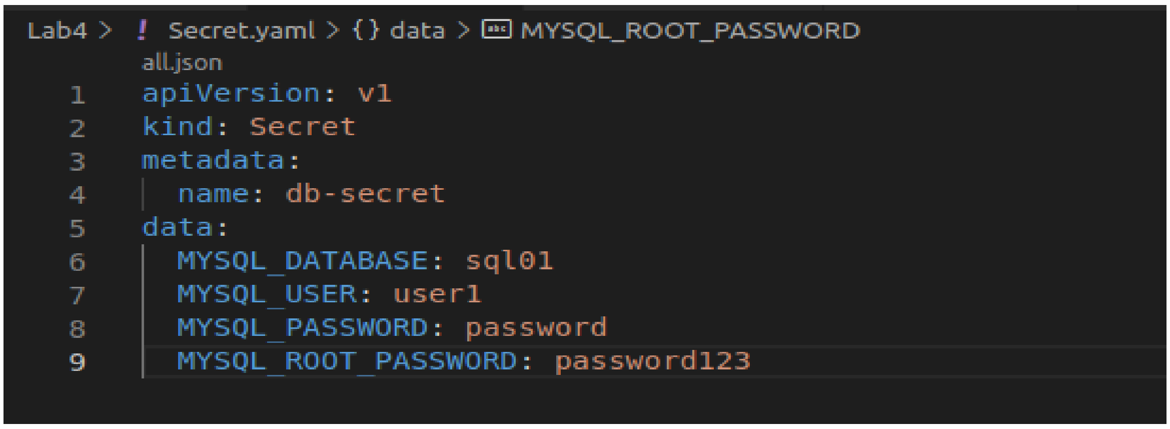
Secret Name: db-secret

Secret 1: MYSQL_DATABASE=sql01

Secret 2: MYSQL_USER=user1

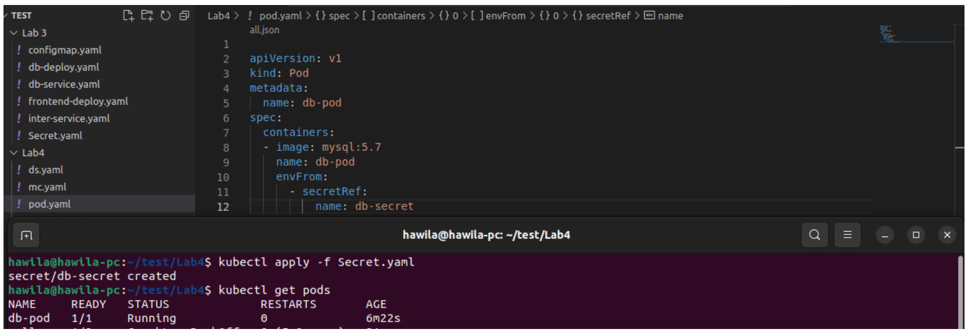
Secret3: MYSQL_PASSWORD=password

Secret 4: MYSQL_ROOT_PASSWORD=password123



```yaml
apiVersion: v1
kind: Secret
metadata:
  name: db-secret
data:
  MYSQL_DATABASE: sql01
  MYSQL_USER: user1
  MYSQL_PASSWORD: password
  MYSQL_ROOT_PASSWORD: password123
```

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



```yaml

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

```
hawila@hawila-pc:~/test/Lab4$ kubectl apply -f Secret.yaml
secret/db-secret created
hawila@hawila-pc:~/test/Lab4$ kubectl get pods
NAME      READY   STATUS              RESTARTS        AGE
db-pod    1/1     Running             0               6m22s
```