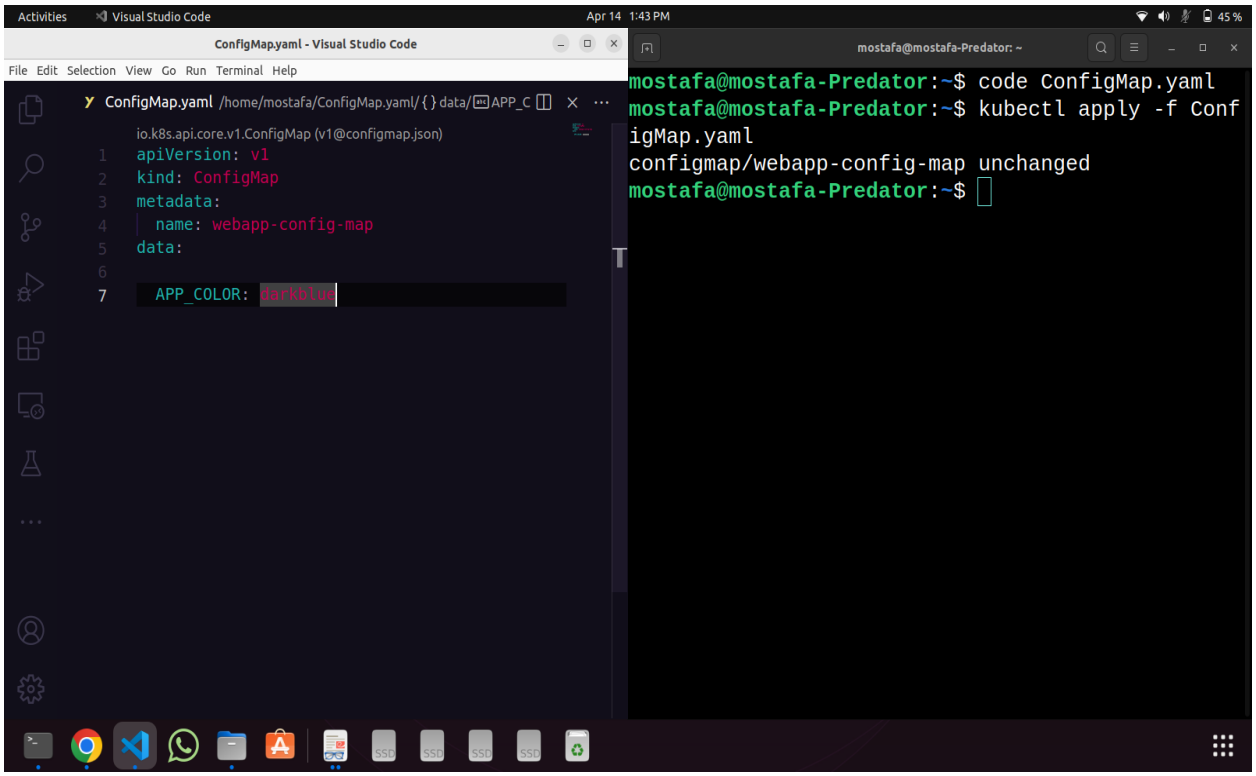
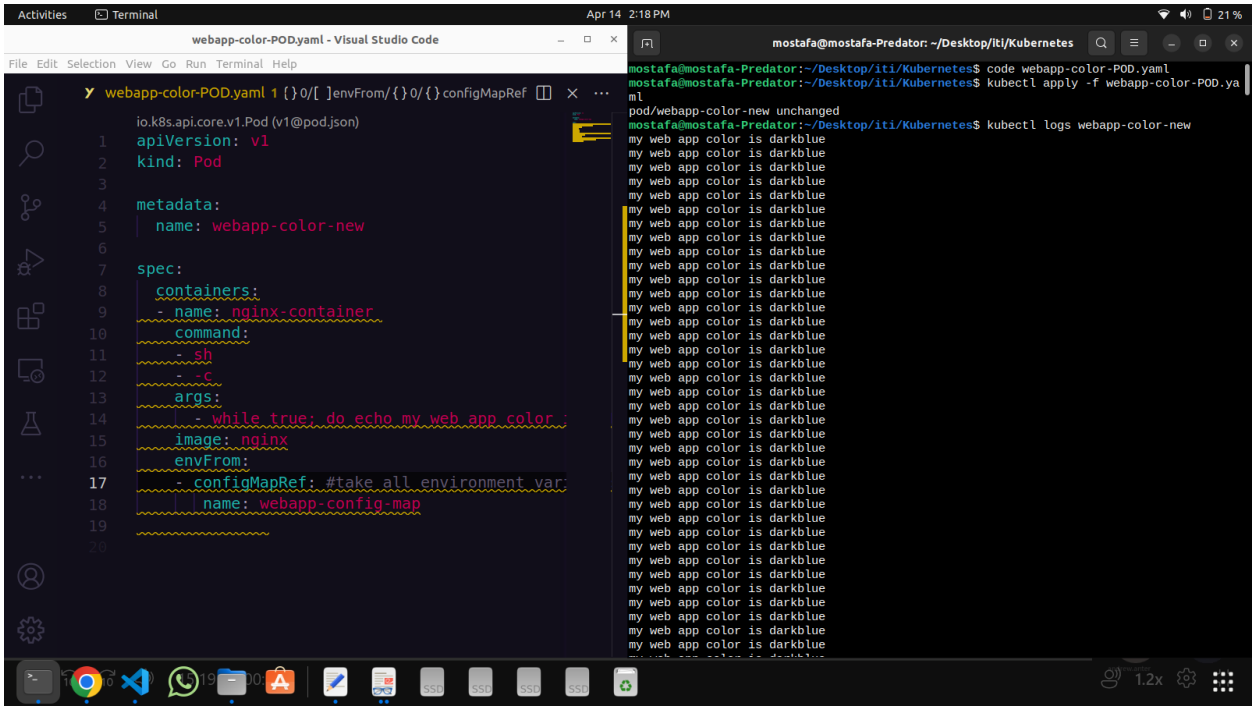
Q1) How many ConfigMaps exist in the environment?

```
mostafa@mostafa-Predator:~$ kubectl get configmap
NAME                  DATA   AGE
kube-root-ca.crt      1      6d21h
mostafa@mostafa-Predator:~$
```

Q2) Create a new ConfigMap Use the spec given : ConfigName Name: webapp-config-map, Data: APP_COLOR=darkblue



Q3) Create a webapp-color POD with nginx image and use the created ConfigMap



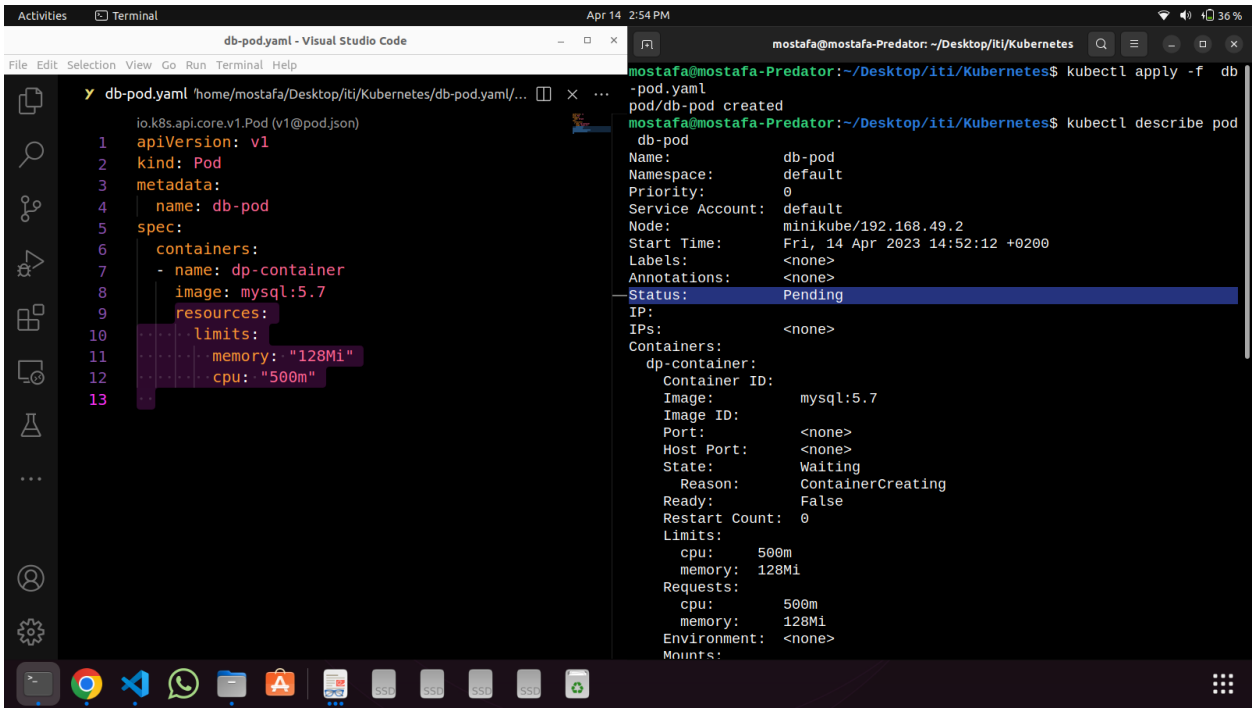
ConfigMapRef: all env variable of configmap

Q4) How many Secrets exist on the system?

, Q5) How many secrets are defined in the default-token secret?

```
mostafa@mostafa-Predator:~/Desktop/iti/Kubernetes$ kubectl get secret
No resources found in default namespace.
mostafa@mostafa-Predator:~/Desktop/iti/Kubernetes$ kubectl get secret -A
NAMESPACE              NAME                            TYPE     DATA   AGE
kubernetes-dashboard   kubernetes-dashboard-certs      Opaque   0      6d22h
kubernetes-dashboard   kubernetes-dashboard-csrf       Opaque   1      6d22h
kubernetes-dashboard   kubernetes-dashboard-key-holder Opaque   2      6d22h
```

Q6) Create a POD called db-pod with the image mysql:5.7 then check the POD status.



Q7) why is the db-pod status not ready ?

Pulling mysql images requires a username and password which aren't provided yet.

Q8) 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

Use -o yaml > newfile.yaml to redirect command to yaml file, Specially with secrets

```
mostafa@mostafa-Predator:~/Desktop/iti/Kubernetes$ kubectl create secret generic db-secret --from-literal MYSQL_DATABASE=sql01 --from-literal MYSQL_USER=user1  --from-literal MYSQL_PASSWORD=1242 --from-literal MYSQL_ROOT_PASSWORD=1242123 -o yaml > db-secret.yaml
mostafa@mostafa-Predator:~/Desktop/iti/Kubernetes$ kubectl get secret
NAME        TYPE     DATA   AGE
db-secret   Opaque   4      3s
mostafa@mostafa-Predator:~/Desktop/iti/Kubernetes$
```

db-secret.yaml /home/mostafa/Desktop/iti/Kubernetes/db-secret.yaml/ apiVersion

io.k8s.api.core.v1.Secret (v1@secret.json)

```yaml
apiVersion: v1
data:
  MYSQL_DATABASE: c3FsMDE=
  MYSQL_PASSWORD: MTI0Mg==
  MYSQL_ROOT_PASSWORD: MTI0MjEyMw==
  MYSQL_USER: dXNlcjE=
kind: Secret
metadata:
  creationTimestamp: "2023-04-14T13:16:34Z"
  name: db-secret
  namespace: default
  resourceVersion: "74562"
  uid: 9bec0627-9dcf-4d66-b223-d93f6327669c
type: Opaque
```

```
mostafa@mostafa-Predator:~/Desktop/iti/Kubernetes$ kubectl describe secret db-secret
Name:         db-secret
Namespace:    default
Labels:       <none>
Annotations:  <none>

Type:  Opaque

Data
====
MYSQL_ROOT_PASSWORD:  7 bytes
MYSQL_USER:           5 bytes
MYSQL_DATABASE:       5 bytes
MYSQL_PASSWORD:       4 bytes
```

Q9) Configure db-pod to load environment variables from the newly created secret

db-pod.yaml Kubernetes • db-pod.yaml/{}spec/[]containers/{}0/[]env/{}2/{}valueFrom

io.k8s.api.core.v1.Pod (v1@pod.json)

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: db-pod
spec:
  containers:
    - name: db-container
      image: migs/mysql-5.7
      env:
        - name: mysql-DB-name
          valueFrom:
            secretKeyRef:
              name: db-secret
              key: MYSQL_DATABASE

        - name: username
          valueFrom:
            secretKeyRef:
              name: db-secret
              key: MYSQL_USER

        - name: password
          valueFrom:
            secretKeyRef:
              name: db-secret
              key: MYSQL_PASSWORD

        - name: root-password
          valueFrom:
            secretKeyRef:
              name: db-secret
              key: MYSQL_ROOT_PASSWORD


      resources:
        limits:
          memory: "128Mi"
          cpu: "500m"
```

0 1 Live Share minikube default gcr.io/k8s-minikube/kicbase : minikube | my-react-app : adoring_chebyshev | my-react-app : pedantic_ishizak Quokka

Q10) 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

multiContainer.yaml 1 Kubernetes • multiContainer.yaml/{}spec/[]containers/{}1/image

io.k8s.api.core.v1.Pod (v1@pod.json)

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: multi-container-pod
spec:
  containers:
    - name: lemon
      image: busybox
      tty: true
    - name: gold
      image: redis
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL

```
mostafa@mostafa-Predator:~/Desktop/iti/Kubernetes$ kubectl apply -f multiContainer.yaml
pod/multi-container-pod created
mostafa@mostafa-Predator:~/Desktop/iti/Kubernetes$
```

0 1 Live Share minikube default gcr.io/k8s-minikube/kicbase : minikube | my-react-app : adoring_chebyshev Quokka

Q11) Create a pod red with redis image and use an initContainer that uses the busybox image and sleeps for 20 seconds

redPod.yaml 1 Kubernetes • redPod.yaml/ { } spec/[ ]initContainers/ { } 0/[ ]command/ 1

io.k8s.api.core.v1.Pod (v1@pod.json)

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: red
spec:
  containers:
    - image: redis
      name: redis-containerss
  initContainers:
    - name: init-bbx
      image: busybox
      command:
        - sleep
        - "20"
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL

```
mostafa@mostafa-Predator:~/Desktop/iti/Kubernetes$ kubectl logs red -c init-bbx
mostafa@mostafa-Predator:~/Desktop/iti/Kubernetes$
```

Q12) 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.

print-envars-greeting-pod.yaml Kubernetes • print-envars-greeting-pod.yaml/{}spec/[ ]containers/{}0/[ ]env/{}0/value

io.k8s.api.core.v1.Pod (v1@pod.json)

```yaml
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: ["echo"]
    args: ["$(GREETING) $(COMPANY) $(GROUP)"]

    resources:
      limits:
        memory: "128Mi"
        cpu: "500m"
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL bash

```
mostafa@mostafa-Predator:~/Desktop/iti/Kubernetes$ kubectl apply -f print-envars-greeting-pod.yaml
pod/print-envars-greeting created
mostafa@mostafa-Predator:~/Desktop/iti/Kubernetes$ kubectl logs -f print-envars-greeting
Welcome to DevOps Industries
mostafa@mostafa-Predator:~/Desktop/iti/Kubernetes$
```

0 ⚠ 1 Live Share minikube default gcr.io/k8s-minikube/kicbase : minikube | my-react-app : adoring_chebyshev | my-react-a Quokka UTF-8 LF YAML kubernetes://schema/v1@pod

Q13) Where is the default kubeconfig file located in the current environment?

Q14) How many clusters are defined in the default kubeconfig file? 1

Q15) What is the user configured in the current context? minikube

```
PROBLEMS 1   OUTPUT   DEBUG CONSOLE   TERMINAL                                   bash
mostafa@mostafa-Predator:~/Desktop/iti/Kubernetes$ cat ~/.kube/config
apiVersion: v1
clusters:
- cluster:
    certificate-authority: /home/mostafa/.minikube/ca.crt
    extensions:
    - extension:
        last-update: Mon, 17 Apr 2023 13:45:23 EET
        provider: minikube.sigs.k8s.io
        version: v1.30.1
      name: cluster_info
    server: https://192.168.49.2:8443
  name: minikube
contexts:
- context:
    cluster: minikube
    extensions:
    - extension:
        last-update: Mon, 17 Apr 2023 13:45:23 EET
        provider: minikube.sigs.k8s.io
        version: v1.30.1
      name: context_info
    namespace: default
    user: minikube
  name: minikube
current-context: minikube
kind: Config
preferences: {}
users:
- name: minikube
  user:
    client-certificate: /home/mostafa/.minikube/profiles/minikube/client.crt
    client-key: /home/mostafa/.minikube/profiles/minikube/client.key
mostafa@mostafa-Predator:~/Desktop/iti/Kubernetes$
0 ⚠ 1   Live Share   minikube   default   gcr.io/k8s-minikube/kicbase : minikube | my-react-app : adoring_chebyshev | my-react-a  Quokka   UTF-8   LF   YAML   kubernetes://schema/v1@pod
```

Q16) Create a Persistent Volume with the given specification.

Volume Name: pv-log

Storage: 100Mi

Access Modes: ReadWriteMany

Host Path: /pv/log

```
pv-log-volume.yaml   Kubernetes • pv-log-volume.yaml/ { } spec/ { } hostPath/ path
io.k8s.api.core.v1.PersistentVolume (v1@persistentvolume.json)
1   apiVersion: v1
2   kind: PersistentVolume
3   metadata:
4     name: pv-log
5   spec:
6     capacity:
7       storage: 100Mi
8
9     accessModes:
10      - ReadWriteMany
11    persistentVolumeReclaimPolicy: Delete
12    storageClassName: ""
13    hostPath:
14      path: /pv/log
```

```
PROBLEMS 1   OUTPUT   DEBUG CONSOLE   TERMINAL                                   bash
mostafa@mostafa-Predator:~/Desktop/iti/Kubernetes$ code pv-log-volume.yaml
mostafa@mostafa-Predator:~/Desktop/iti/Kubernetes$ kubectl apply -f pv-log-volume.yaml
persistentvolume/pv-log created
mostafa@mostafa-Predator:~/Desktop/iti/Kubernetes$
0 ⚠ 1   Live Share   minikube   default   gcr.io/k8s-minikube/kicbase : minikube | my-react-app : adoring_chebyshev | my-react-a  Quokka   LF   YAML   kubernetes://schema/v1@persistentvolume
```

Q17) Create a Persistent Volume Claim with the given specification.
Volume Name: claim-log-1
Storage Request: 50Mi
Access Modes: ReadWriteMany

pv-claim-log-1.yaml Kubernetes • pv-claim-log-1.yaml/ { } spec/[ ]accessModes/ 0

io.k8s.api.core.v1.PersistentVolumeClaim (v1@persistentvolumeclaim.json)

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

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL

```
mostafa@mostafa-Predator:~/Desktop/iti/Kubernetes$ code pv-claim-log-1.yaml
mostafa@mostafa-Predator:~/Desktop/iti/Kubernetes$ kubectl apply -f pv-claim-log-1.yaml
persistentvolumeclaim/claim-log-1 created
mostafa@mostafa-Predator:~/Desktop/iti/Kubernetes$
```

Q18 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

webAppPod.yaml Kubernetes • webAppPod.yaml/ { } metadata/ name

io.k8s.api.core.v1.Pod (v1@pod.json)

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: webapp
spec:
  containers:
  - name: webapp-containers
    image: nginx
    volumeMounts:
      - mountPath: /var/log/nginx
        name: pv-storage
    resources:
      limits:
        memory: "128Mi"
        cpu: "500m"

  volumes:
    - name: pv-storage
      persistentVolumeClaim:
        claimName: claim-log-1
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL

```
mostafa@mostafa-Predator:~/Desktop/iti/Kubernetes$ kubectl apply -f  webAppPod.yaml
pod/webapp created
mostafa@mostafa-Predator:~/Desktop/iti/Kubernetes$
```