LÀM VIỆC VỚI K8S WORKLOAD

1. Mở 1 terminal riêng và theo dõi các sự kiện diễn ra trong cluster

```
kubectl get events --watch
LAST SEEN   TYPE     REASON           OBJECT                         MESSAGE
53m         Normal   Scheduled        pod/hello-world-5b76c5697b-g7wnj
Successfully assigned default/hello-world-5b76c5697b-g7wnj to gke-toka02-default-
pool-ae3df2a0-1l3v
53m         Normal   Pulling          pod/hello-world-5b76c5697b-g7wnj    Pulling
image "gcr.io/google-samples/hello-app:1.0"
53m         Normal   Pulled           pod/hello-world-5b76c5697b-g7wnj
Successfully pulled image "gcr.io/google-samples/hello-app:1.0"
```

2. Tạo pod hello-world sử dụng câu lệnh và sử dụng manifest

a. Tạo pod bằng câu lệnh kubectl nhưng chưa triển khai lên cluster

```
kubectl run hello-world-pod \
    --image gcr.io/google-samples/hello-app:1.0 \
    --restart Never \
    --port 8080 \
    --dry-run=client -o yaml


apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    run: hello-world-pod
  name: hello-world-pod
spec:
  containers:
  - image: gcr.io/google-samples/hello-app:1.0
    name: hello-world-pod
    ports:
    - containerPort: 8080
    resources: {}
  dnsPolicy: ClusterFirst
  restartPolicy: Never
status: {}
```

b. Tạo pod với cấu hình như trên

```
kubectl run hello-world-pod \
    --image gcr.io/google-samples/hello-app:1.0 \
    --restart Never \
    --port 8080

pod/hello-world-pod created
```

c. Kiểm tra trạng thái của pod mới được khởi tạo

```
kubectl get pod hello-world-pod

NAME              READY   STATUS    RESTARTS   AGE
hello-world-pod   1/1     Running   0          2m26s
```

d. Xoá pod và kiểm tra kết quả

```
kubectl delete pod hello-world-pod
pod "hello-world-pod" deleted

kubectl get pod hello-world-pod
Error from server (NotFound): pods "hello-world-pod" not found
```

e. Tạo pod tương tự từ manifest file

```
cat <<'EOF' > hello-world-pod.yaml
apiVersion: v1
kind: Pod
metadata:
 name: hello-world-pod
spec:
 containers:
 - name: hello-world
   image: gcr.io/google-samples/hello-app:1.0
   ports:
   - containerPort: 8080
EOF
https://github.com/hungtran84/k8s-cka/blob/master/d2_workloads/01_workloads_fundamental/01_pod_fundamental/hello-world-pod.yaml
```

```
kubectl apply -f hello-world-pod.yaml
pod/hello-world-pod created
```

f. Truy cập hello-world web application

```
kubectl run bb -it --rm \

  --image radial/busyboxplus:curl \
  --restart Never \
  -- curl <POD_IP>:8080

Hello, world!
Version: 1.0.0
Hostname: hello-world-pod
pod "bb" deleted

!!!Find the pod ip address yourself
```

3. Tạo deployment cho app hello-world

a. Tạo deployment bằng câu lệnh kubectl nhưng chưa triển khai lên cluster

```
kubectl create deployment hello-world \
   --image gcr.io/google-samples/hello-app:1.0 \
   --dry-run=client -oyaml

apiVersion: apps/v1
kind: Deployment
metadata:
  creationTimestamp: null
  labels:
    app: hello-world
  name: hello-world
spec:
  replicas: 1
  selector:
    matchLabels:
      app: hello-world
  strategy: {}
  template:
    metadata:
```

```
    creationTimestamp: null
    labels:
      app: hello-world
  spec:
    containers:
    - image: gcr.io/google-samples/hello-app:1.0
      name: hello-app
      resources: {}
status: {}
```

b. Tạo pod tương tự từ manifest file

```
cat <<'EOF' > hello-world-deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
 name: hello-world
spec:
 replicas: 1
 selector:
   matchLabels:
     app: hello-world
 template:
   metadata:
     labels:
       app: hello-world
   spec:
     containers:
     - name: hello-world
       image: gcr.io/google-samples/hello-app:1.0
       ports:
       - containerPort: 8080
EOF
```

https://github.com/hungtran84/k8s-cka/blob/master/d2_workloads/01_workloads_fundamental/01_pod_fundamental/deployment.yaml

```
kubectl apply -f hello-world-deployment.yaml
deployment.apps/hello-world created
```

c. Kiểm tra trạng thái của deployment và pod

```
kubectl get deploy hello-world
NAME        READY   UP-TO-DATE   AVAILABLE   AGE
hello-world   1/1     1            1           72s

kubectl get pods
NAME                     READY   STATUS    RESTARTS   AGE
hello-world-5b76c5697b-g7wnj   1/1     Running   0          3m23s
hello-world-pod          1/1     Running   0          161m


# list the labels of pods
kubectl get pods --show-labels
NAME                     READY   STATUS    RESTARTS   AGE   LABELS
hello-world-5b76c5697b-g7wnj   1/1     Running   0          12m   app=hello-
world,pod-template-hash=5b76c5697b
hello-world-pod          1/1     Running   0          170m  <none>

# list pods matched specific labels
kubectl get pods --selector app=hello-world
NAME                     READY   STATUS    RESTARTS   AGE
hello-world-5b76c5697b-g7wnj   1/1     Running   0          12m
```

d. Kiểm tra trạng thái của replicaset

```
kubectl get replicaset
NAME                DESIRED   CURRENT   READY   AGE
hello-world-5b76c5697b   1         1         1       22m


kubectl get replicasets -oyaml

apiVersion: v1
items:
- apiVersion: apps/v1
  kind: ReplicaSet
  metadata:
    annotations:
      deployment.kubernetes.io/desired-replicas: "1"
```

```
    deployment.kubernetes.io/max-replicas: "2"
    deployment.kubernetes.io/revision: "1"
  creationTimestamp: "2020-07-12T13:15:56Z"
  generation: 1
  labels:
    app: hello-world
    pod-template-hash: 5b76c5697b
  name: hello-world-5b76c5697b
  namespace: default
  ownerReferences:
  - apiVersion: apps/v1
    blockOwnerDeletion: true
    controller: true
    kind: Deployment
    name: hello-world
    uid: 0c00f985-973b-4b5e-bc08-da42d852ba4e
  resourceVersion: "51923"
  selfLink: /apis/apps/v1/namespaces/default/replicasets/hello-world-5b76c5697b
  uid: 356405c7-1070-42c3-b13e-50c87921519b
spec:
  replicas: 1
  selector:
    matchLabels:
      app: hello-world
      pod-template-hash: 5b76c5697b
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: hello-world
        pod-template-hash: 5b76c5697b
    spec:
      containers:
      - image: gcr.io/google-samples/hello-app:1.0
        imagePullPolicy: IfNotPresent
        name: hello-world
        ports:
        - containerPort: 8080
```

```
        protocol: TCP
      resources: {}
      terminationMessagePath: /dev/termination-log
      terminationMessagePolicy: File
    dnsPolicy: ClusterFirst
    restartPolicy: Always
    schedulerName: default-scheduler
    securityContext: {}
    terminationGracePeriodSeconds: 30
 status:
   availableReplicas: 1
   fullyLabeledReplicas: 1
   observedGeneration: 1
   readyReplicas: 1
   replicas: 1
kind: List
metadata:
 resourceVersion: ""
 selfLink: ""

!!! We don't need to deal with the replicaset but the deployment/statefulset instead
```

e. Xoá pod của deployment và theo dõi kết quả:

```
kubectl delete pods hello-world-5b76c5697b-g7wnj
pod "hello-world-5b76c5697b-g7wnj" deleted

kubectl get pods --selector app=hello-world
NAME                          READY   STATUS    RESTARTS   AGE
hello-world-5b76c5697b-rn4gb   1/1     Running   0          2m16s
```

4. Tăng/giảm kích thước deployment

a. Tăng số lượng pod của deployment từ 1 (default) lên 3 và quan sát kết quả:

```
kubectl scale deployment hello-world --replicas=3
deployment.apps/hello-world scaled


kubectl get pods --selector app=hello-world -owide \
    | awk {'print $1" " $7'} \
```

```
    | column -t

NAME                     NODE
hello-world-5b76c5697b-h6hss  gke-toka02-default-pool-ae3df2a0-1l3v
hello-world-5b76c5697b-jpdz5  gke-toka02-default-pool-ae3df2a0-5f3v
hello-world-5b76c5697b-rn4gb  gke-toka02-default-pool-ae3df2a0-3zgr
```

b. Giảm pod replica của deployment về 1:

```
kubectl scale deployment hello-world --replicas=1
deployment.apps/hello-world scaled

kubectl get pods --selector app=hello-world -owide \
    | awk {'print $1" " $7'} \
    | column -t
NAME                         NODE
hello-world-5b76c5697b-rn4gb  gke-toka02-default-pool-ae3df2a0-3zgr
```

5. Sử dụng port-forwarding để truy cập vào pod/deployment:

```
kubectl port-forward hello-world-5b76c5697b-rn4gb 8080:8080 &
deployment.apps/hello-world scaled

curl http://localhost:8080
Hello, world!
Version: 1.0.0
Hostname: hello-world-5b76c5697b-rn4gb
```

6. Sử dụng multi-containers pod

a. Tạo manifest cho 1 multi-container pod:

```
cat <<'EOF' > multicontainer-pod.yaml
apiVersion: v1
kind: Pod
metadata:
 name: multicontainer-pod
spec:
 containers:
 - name: producer
   image: ubuntu
   command: ["/bin/bash"]
   args: ["-c", "while true; do echo $(hostname) $(date) >> /var/log/index.html; sleep
10; done"]
   volumeMounts:
   - name: webcontent
     mountPath: /var/log
 - name: consumer
   image: nginx
```

```
  ports:
    - containerPort: 80
  volumeMounts:
  - name: webcontent
    mountPath: /usr/share/nginx/html
 volumes:
 - name: webcontent
   emptyDir: {}
EOF
https://github.com/hungtran84/k8s-cka/blob/master/d2_workloads/01_workloads_fundamental/01_pod_fundamental/deployment.yaml
```

Tham khảo thêm về emptyDir volume tại đây:
https://kubernetes.io/docs/concepts/storage/volumes/#emptydir

b. Deploy multi-container pod cluster:

```
kubectl apply -f multicontainer-pod.yaml
pod/multicontainer-pod created

kubectl get pods multicontainer-pod \
 -o jsonpath='{.spec.containers[*].name}'
producer consumer
```

c. Truy cập vào container tên là producer và kiểm tra log file:

```
kubectl exec -it multicontainer-pod --container producer -- /bin/sh
# ls -la /var/log
total 16
drwxrwxrwx 2 root root 4096 Jul 13 04:51 .
drwxr-xr-x 1 root root 4096 Jul  3 02:00 ..
-rw-r--r-- 1 root root 4368 Jul 13 05:06 index.html
# tail /var/log/index.html
multicontainer-pod Mon Jul 13 05:05:26 UTC 2020
multicontainer-pod Mon Jul 13 05:05:36 UTC 2020
multicontainer-pod Mon Jul 13 05:05:46 UTC 2020
multicontainer-pod Mon Jul 13 05:05:56 UTC 2020
multicontainer-pod Mon Jul 13 05:06:06 UTC 2020
multicontainer-pod Mon Jul 13 05:06:16 UTC 2020
multicontainer-pod Mon Jul 13 05:06:26 UTC 2020
multicontainer-pod Mon Jul 13 05:06:36 UTC 2020
```

```
multicontainer-pod Mon Jul 13 05:06:46 UTC 2020
multicontainer-pod Mon Jul 13 05:06:56 UTC 2020
# exit
```

d. Truy cập vào container tên là consumer và kiểm tra web content:

```
kubectl exec -it multicontainer-pod --container consumer -- /bin/sh
# ls -la /usr/share/nginx/html
total 36
drwxrwxrwx 2 root root  4096 Jul 13 04:51 .
drwxr-xr-x 3 root root  4096 Jul 10 20:26 ..
-rw-r--r-- 1 root root 22368 Jul 13 06:09 index.html
# tail /usr/share/nginx/html/index.html
multicontainer-pod Mon Jul 13 06:07:57 UTC 2020
multicontainer-pod Mon Jul 13 06:08:07 UTC 2020
multicontainer-pod Mon Jul 13 06:08:17 UTC 2020
multicontainer-pod Mon Jul 13 06:08:27 UTC 2020
multicontainer-pod Mon Jul 13 06:08:37 UTC 2020
multicontainer-pod Mon Jul 13 06:08:47 UTC 2020
multicontainer-pod Mon Jul 13 06:08:57 UTC 2020
multicontainer-pod Mon Jul 13 06:09:07 UTC 2020
multicontainer-pod Mon Jul 13 06:09:17 UTC 2020
multicontainer-pod Mon Jul 13 06:09:27 UTC 2020
# exit
```

e. Sử dụng port-forward để truy cập vào ứng dụng:

```
kubectl port-forward multicontainer-pod 8080:80 &
[1] 10350

curl http://localhost:8080
Handling connection for 8080
multicontainer-pod Mon Jul 13 04:51:56 UTC 2020
multicontainer-pod Mon Jul 13 04:52:06 UTC 2020
multicontainer-pod Mon Jul 13 04:52:16 UTC 2020
multicontainer-pod Mon Jul 13 04:52:26 UTC 2020
. . . .
# Kill port-forwarding process
fg
ctrl + c
```

```
# Cleanup pod after use
kubectl delete pod multicontainer-pod
pod "multicontainer-pod" deleted
```

7. Pod lifecycle

a. Truy cập vào container để kiểm tra process

```
kubectl exec -it hello-world-pod -- ps
PID   USER     TIME   COMMAND
    1 root       0:00 ./hello-app
    9 root       0:00 ps
```

b. Kill container process và theo dõi kết quả

```
# Kill container process and see the result
kubectl exec -it hello-world-pod -- /usr/bin/killall hello-app

kubectl get po hello-world-pod
NAME              READY   STATUS    RESTARTS   AGE
hello-world-pod   1/1     Running   1          6m32s

# Look at Containers->State, Last State, Reason, Exit Code, Restart Count and
Events
kubectl describe po hello-world-pod

Name:         hello-world-pod
Namespace:    default
Priority:     0
Node:         gke-toka02-default-pool-ae3df2a0-1l3v/10.148.0.6
Start Time:   Mon, 13 Jul 2020 16:46:36 +0700
Labels:       <none>
Annotations:  kubernetes.io/limit-ranger: LimitRanger plugin set: cpu request for
container hello-world
Status:       Running
IP:           10.60.1.13
IPs:
  IP:  10.60.1.13
Containers:
```

```
  hello-world:
    Container ID:
docker://c6e6f2886936f50e05a181d045b177e3f63e82e40493f9881268757baa995352
    Image:          gcr.io/google-samples/hello-app:1.0
    Image ID:       docker-pullable://gcr.io/google-samples/hello-app@sha256:c62ead5b8c15c231f9e786250b07909daf6c266d0fcddd93fea882eb722c3be4
    Port:           80/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Mon, 13 Jul 2020 16:51:04 +0700
    Last State:     Terminated
      Reason:       Error
      Exit Code:    2
      Started:      Mon, 13 Jul 2020 16:46:38 +0700
      Finished:     Mon, 13 Jul 2020 16:51:03 +0700
    Ready:          True
    Restart Count:  1
    Requests:
      cpu:        100m
    Environment:  <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from default-token-2mdc5 (ro)
Conditions:
  Type              Status
  Initialized       True
  Ready             True
  ContainersReady   True
  PodScheduled      True
Volumes:
  default-token-2mdc5:
    Type:        Secret (a volume populated by a Secret)
    SecretName:  default-token-2mdc5
    Optional:    false
QoS Class:       Burstable
Node-Selectors:  <none>
Tolerations:     node.kubernetes.io/not-ready:NoExecute for 300s
```

```
                node.kubernetes.io/unreachable:NoExecute for 300s
Events:
  Type    Reason     Age              From                                Message
  ----    ------     ----             ----                                -------
  Normal  Scheduled  11m              default-scheduler
Successfully assigned default/hello-world-pod to gke-toka02-default-pool-ae3df2a0-1l3v
  Normal  Pulled     7m2s (x2 over 11m)  kubelet, gke-toka02-default-pool-ae3df2a0-1l3v  Container image "gcr.io/google-samples/hello-app:1.0" already present on machine
  Normal  Created    7m2s (x2 over 11m)  kubelet, gke-toka02-default-pool-ae3df2a0-1l3v  Created container hello-world
  Normal  Started    7m2s (x2 over 11m)  kubelet, gke-toka02-default-pool-ae3df2a0-1l3v  Started container hello-world
# Cleanup
kubectl delete po hello-world-pod
pod "hello-world-pod" deleted
```

c. Tạo manifest

```
https://github.com/hungtran84/k8s-cka/blob/master/d2_workloads/01_workloads_fundamental/03_pod_lifecycle/pod-restart-policy.yaml
cat <<'EOF' > pod-restart-policy.yaml
apiVersion: v1
kind: Pod
metadata:
 name: hello-world-onfailure-pod
spec:
 containers:
 - name: hello-world
   image: gcr.io/google-samples/hello-app:1.0
 restartPolicy: OnFailure
---
apiVersion: v1
kind: Pod
metadata:
 name: hello-world-never-pod
spec:
```

```
 containers:
 - name: hello-world
   image: gcr.io/google-samples/hello-app:1.0
 restartPolicy: Never
EOF
```

d. Triển khai 2 pod mới lên cluster

```
kubectl apply -f pod-restart-policy.yaml
pod/hello-world-onfailure-pod created
pod/hello-world-never-pod created
```

e. Huỷ tiến trình trên pod có restartPolicy là Never và kiểm tra kết quả

```
kubectl exec -it hello-world-never-pod -- /usr/bin/killall hello-app
command terminated with exit code 137

kubectl get po hello-world-never-pod
NAME                 READY   STATUS   RESTARTS   AGE
hello-world-never-pod   0/1     Error    0          7m34s
```

f. Huỷ tiến trình trên pod có restartPolicy là OnFailure và kiểm tra kết quả

```
kubectl exec -it hello-world-onfailure-pod -- /usr/bin/killall hello-app
command terminated with exit code 137

kubectl get pods hello-world-onfailure-pod
NAME                    READY   STATUS    RESTARTS   AGE
hello-world-onfailure-pod   1/1     Running   1          12m
```

8. Container probes

a. Tạo manifest

```
https://github.com/hungtran84/k8s-
cka/blob/master/d2_workloads/01_workloads_fundamental/04_container_probes/con
tainer-probes.yaml
cat <<'EOF' > container-probes.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
 name: hello-world
spec:
```

```
  replicas: 1
  selector:
    matchLabels:
      app: hello-world
  template:
    metadata:
      labels:
        app: hello-world
    spec:
      containers:
      - name: hello-world
        image: gcr.io/google-samples/hello-app:1.0
        ports:
        - containerPort: 8080
        livenessProbe:
          tcpSocket:
            port: 8081
          initialDelaySeconds: 10
          periodSeconds: 5
        readinessProbe:
          httpGet:
            path: /
            port: 8081
          initialDelaySeconds: 10
          periodSeconds: 5
EOF
```

b. Triển khai 2 pod mới lên cluster

```
kubectl apply -f container-probes.yaml
deployment.apps/hello-world created

kubectl get po --selector app=hello-world -w
NAME                     READY   STATUS    RESTARTS   AGE
hello-world-79bcb87954-z5l4v   0/1     Running   2          46s
hello-world-79bcb87954-z5l4v   0/1     Running   3          63s
```

c. Sửa livenessProbe, readinessProbe và cập nhật deployment:

```
kubectl apply -f s05/container-probes.yaml
```

```
deployment.apps/hello-world configured

kubectl get po --selector app=hello-world
NAME                     READY   STATUS    RESTARTS   AGE
hello-world-74c9c5c5d-mfbnx   1/1     Running   0          69s
```