

## Kubernetes (Lab\_3)

1 Create ConfigMap or MongoDB EndPoint.

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: mongodb-configmap
data:
  DB_URL: mongo-service
  clusterIP_name: mongo-svc
```

2 Create A secret or MongoDB User & PWD

```
apiVersion: v1
kind: Secret
metadata:
  name: mango-secret
data:
  USER_NAME: Alsafa_Wagdy
  PASSWORD: mypassword
```

3 Create MongoDB Deployment Application with Internal service (ClusterIp) Mongo DB needs username + password to operate

```
apiVersion: v1
kind: Service
metadata:
  name: mongo-svc
spec:
  type: ClusterIp
  selector:
    matchLabels:
      app: mongo-db
  ports:
    - port: 80
      targetPort: 80
      nodePort: 30007
```

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mongodb_deploy
  labels:
    app: mongodb
spec:
  replicas: 3
  selector:
    matchLabels:
      app: mongodb_pod
  template:
    metadata:
      labels:
        app: mongodb_pod
    spec:
      containers:
        - name: my-mongo-pod
          image: mongo:5.0
          env:
            - name: MONGO_INITDB_ROOT_USERNAME
              valueFrom:
                secretKeyRef:
                  name: mango-secret
                  key: USER_NAME
            - name: MONGO_INITDB_ROOT_PASSWORD
              valueFrom:
                secretKeyRef:
                  name: mango-secret
                  key: PASSWORD
          envFrom:
            configMapRef:
              mongodb-configmap
```

- 4 Create webApp Deployment(FrontEnd( with external service) and it needs to access MongoDB, so it needs username+ password + mongodb endpoint (mongodb service) container runs on 3000

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: frontend_deploy
  labels:
    app: frontend
spec:
  replicas: 3
  selector:
    matchLabels:
      app: frontend_pod
  template:
    metadata:
      labels:
        app: frontend_pod
    spec:
      containers:
        - name: my-frontend-pod
          image: nanajanashia/k8s-demo-app:v1.0
          env:
            - name: MONGO_INITDB_ROOT_USERNAME
              valueFrom:
                secretKeyRef:
                  name: mango-secret
                  key: USER_NAME
            - name: MONGO_INITDB_ROOT_PASSWORD
              valueFrom:
                secretKeyRef:
                  name: mango-secret
                  key: PASSWORD
          envFrom:
            configMapRef:
              name: mongodb-configmap
```

```
apiVersion: v1
kind: Service
metadata:
  name: NodePort-svc
spec:
  type: NodePort
  ports:
    - port: 3000
      targetPort: 3000
      nodePort: 30007
```

- 8- How many Nodes exist on the system?

```
controlplane $ k get nodes
NAME           STATUS    ROLES    AGE   VERSION
controlplane   Ready     control-plane  34d   v1.26.0
node01         Ready     <none>      34d   v1.26.0
```

- 9- Do you see any taints on master? *No, there isn't any taint*

```
controlplane $ kubectl describe nodes controlplane | grep Taint
Taints:                <none>
```

10- Apply a label color=blue to the master node

```
controlplane $ k taint node controlplane color=blue:NoSchedule
node/controlplane tainted
```

11- Create a new deployment named blue with the nginx image and 3 replicas. Set Node Affinity to the deployment to place the pods on master only.

NodeAffinity: requiredDuringSchedulingIgnoredDuringExecution. Key: color values: blue

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: blue
  labels:
    app: nginx
spec:
  selector:
    matchLabels:
      app: nginx
  replicas: 3
  template:
    metadata:
      labels:
        app: nginx
    spec:
      affinity:
        nodeAffinity:
          requiredDuringSchedulingIgnoredDuringExecution:
            nodeSelectorTerms:
              - matchExpressions:
                  - key: color
                    operator: In
                    values:
                      - blue
      containers:
        - name: nginx
          image: nginx
```

```
controlplane $ k apply -f blue.yaml
deployment.apps/blue created
```

12-Create a taint on node01 with key of spray, value of mortein and effect of NoSchedule

```
controlplane $ kubectl taint node node01 spray=mortein:NoSchedule
node/node01 tainted
```

13-Create a new pod with the NGINX image, and Pod name as mosquito

```
controlplane $ k run mosquito --image=nginx
pod/mosquito created
```

14- What is the state of mosquito POD? *pending*

```
controlplane $ k get po mosquito
```

NAME	READY	STATUS	RESTARTS	AGE
mosquito	0/1	Pending	0	2m8s

15-Create another pod named bee with the NGINX image, which has a toleraton set to the taint Mortein Image name: nginx -- Key: spray -- Value: mortein -- Efect: NoSchedule --Status: Running

```
apiVersion: v1
kind: Pod
metadata:
  name: bee
spec:
  containers:
  - name: nginx
    image: nginx
  tolerations:
  - key: "spray"
    operator: "Equal"
    value: "mortein"
    effect: "NoSchedule"
```

```
controlplane $ k get po
```

NAME	READY	STATUS	RESTARTS	AGE
bee	1/1	Running	0	11s