1- Create ConfgMap or MongoDB EndPoint. (The MongoDB sevice name)

apple — vim config-deploy.yaml — 116×51

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: mongodb-configmap
data:
  mongo-url:mongo-service
~
~
~
~
~
```

```
Ehab_Ashraf ~ % vim config-deploy.yaml
Ehab_Ashraf ~ % kubectl apply -f config-deploy.yaml
configmap/mongodb-configmap created
```

2- Create A secret or MongoDB User & PWD

```
apiVersion: v1
kind: Secret
metadata:
  name: mongo-secret
type: Opaque
data:
  USER_NAME: bW9uZ291c2Vy
  PASSWORD: bW9uZ29wYXNzd29yZA==
~
~
```

```
Ehab_Ashraf ~ % echo -n mongouser | base64
bW9uZ291c2Vy
Ehab_Ashraf ~ % echo -n mongopassword | base64
bW9uZ29wYXNzd29yZA==
Ehab_Ashraf ~ % kubectl apply -f secret-deploy.yaml
Error from server (BadRequest): error when creating "secret-deploy.yaml": Secret in version "v
annot be handled as a Secret: illegal base64 data at input byte 13
Ehab_Ashraf ~ % vim secret-deploy.yaml
Ehab_Ashraf ~ % vim secret-deploy.yaml
Ehab_Ashraf ~ % kubectl apply -f secret-deploy.yaml
secret/mongo-secret created
Ehab_Ashraf ~ %
```

3- Create MongoDB Deployment Applicaton with Internal service (ClusterIp) Mongo DB needs username + password to operate
Vars needed in MongoDB:
MONGO_INITDB_ROOT_USERNAME: root
MONGO_INITDB_ROOT_PASSWORD: example

apple — vim mongo.yaml — 91×51

```
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mongodb_deploy
  labels:
    app: mongodb
spec:
  replicas: 3
  selector:
    matchLablels:
      app: mongodd
    templete:
      metadata:
        labels:
          app: mongodb
      spec:
        containers:
        - name: my-mongo-pod
          image: mongo:5.0
          ports:
          - containerPort: 3000
      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

---
apiVersion: v1
kind: Service
metadata:
 name: mongo_svc
spec:
 type: ClusterIp
 selector:
  matchLabels:
  app: mongodb
 ports:
 - port: 3000
 targetPort: 3000
 nodePort: 30007
"mongo.yaml" 52L, 933B
```

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
  template:
    metadata:
      labels:
        app: frontend
    spec:
      containers:
      - name: my-frontend-pod
        image: nanajanashia/k8s-demo-app:v1.0
        env:
        - name: MONGO_INITDB_ROOT_USERNAME
          valueFrom:
            secretKeyRef:
              name: mongo-secret
              key: USER_NAME
        - name: MONGO_INITDB_ROOT_PASSWORD
          valueFrom:
            secretKeyRef:
              name: mongo-secret
              key: PASSWORD
          envFrom:
            configMapRef:
                  mongodb-configmap

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

5- How many Nodes exist on the system?

```
Ehab_Ashraf ~ % kubectl get nodes
NAME       STATUS   ROLES           AGE   VERSION
minikube   Ready    control-plane   24d   v1.25.3
Ehab_Ashraf ~ %
```

6- Do you see any taints on master ?

```
Ehab_Ashraf ~ % kubectl describe nodes minikube
Name:               minikube
Roles:              control-plane
Labels:             beta.kubernetes.io/arch=arm64
                    beta.kubernetes.io/os=linux
                    color=blue
                    kubernetes.io/arch=arm64
                    kubernetes.io/hostname=minikube
                    kubernetes.io/os=linux
                    minikube.k8s.io/commit=986b1ebd987211ed16f8cc10aed7d2c42fc8392f
                    minikube.k8s.io/name=minikube
                    minikube.k8s.io/primary=true
                    minikube.k8s.io/updated_at=2023_01_18T02_17_00_0700
                    minikube.k8s.io/version=v1.28.0
                    node-role.kubernetes.io/control-plane=
                    node.kubernetes.io/exclude-from-external-load-balancers=
Annotations:        kubeadm.alpha.kubernetes.io/cri-socket: unix:///var/run/cri-dockerd.soc
k
                    node.alpha.kubernetes.io/ttl: 0
                    volumes.kubernetes.io/controller-managed-attach-detach: true
CreationTimestamp:  Wed, 18 Jan 2023 02:16:57 +0200
Taints:             <none>
```

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

8- Create a new deployment named blue with the nginx image and 3 replicas
Set Node Afnity to the deployment to place the pods on master only
NodeAfnity: requiredDuringSchedulingIgnoredDuringExecuton
Key: color
values: blue

```
Editor  Tab 1  Tab 2  +                                                    15 min
apiVersion: apps/v1
kind: Deployment
metadata:
  name: blue
  labels:
    app: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx
          image: nginx:1.14.2
          ports:
            - containerPort: 80
      affinity:
          nodeAffinity:
            requiredDuringSchedulingIgnoredDuringExecution:
              nodeSelectorTerms:
                - matchExpressions:
                  - key: color
                    operator: In
                    values:
                      - blue
      tolerations:
          - key: "node-role.kubernetes.io/control-plane"
            operator: "Exists"
            effect: "Noschedule"
~
```

9- Create a taint on node01 with key o spray, value o mortein and efect of NoSchedule

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

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

```
controlplane $ kubectl run mosquito --image nginx
pod/mosquito created
```

11- What is the state o the mosquito POD?

```
controlplane $ kubectl run mosquito --image nginx
pod/mosquito created
```

12- 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

```yaml
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    run: bee
  name: bee
spec:
  containers:
  - image: nginx
    name: bee
    ports:
    - containerPort: 80
  tolerations:
  - key: "spray"
    operator: "Equal"
    value: "mortein"
    effect: "NoSchedule"
```

```
controlplane $ vim pod.yml
controlplane $ kubectl apply -f pod.yml
pod/nginx created
controlplane $ kubectl get pods
NAME       READY   STATUS    RESTARTS   AGE
mosquito   0/1     Pending   0          7m53s
nginx      1/1     Running   0          5s
controlplane $
```