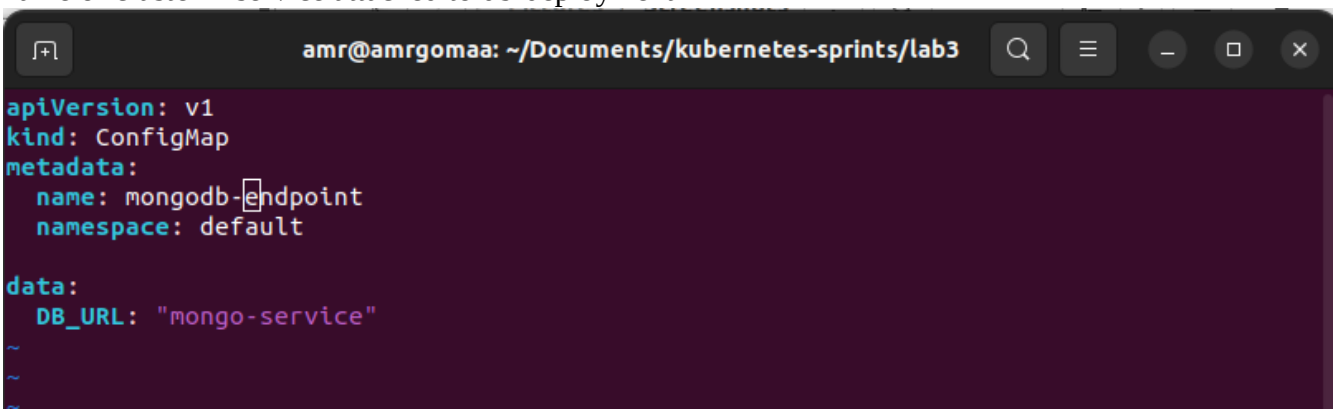
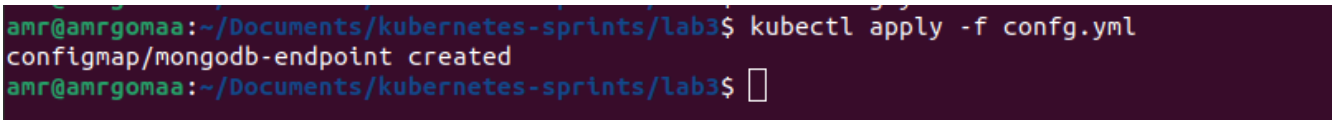
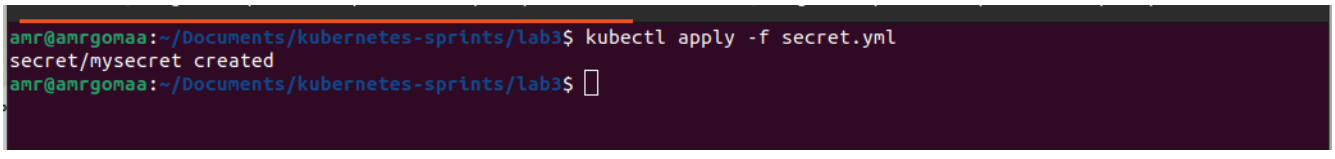
1
Create ConfgMap or MongoDB EndPoint. (The MondoDB sevice name)
DB_URL:mongo-service
name of clusterIP service attached to db-deployment





2
Create A secret or MongoDB User & PWD
USER_NAME: mongouser
USER_PWD: mongopassword



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

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  creationTimestamp: null
  labels:
    app: database
  name: mongodb-test
spec:
  replicas: 1
  selector:
    matchLabels:
      app: database
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: database
    spec:
      containers:
      - image: mongo:5.0
        name: mongo
        envFrom:
          - secretRef:
              name: mysecrete
        env:
          - name: MONGO_INITDB_ROOT_USERNAME
            value: "root"
          - name: MONGO_INITDB_ROOT_PASSWORD
            value: "example"

status: {}
~
~
```

```
amr@amrgomaa:~/Documents/kubernetes-sprints/lab3$ kubectl apply -f deploymentdb.yml
deployment.apps/mongodb-test created
amr@amrgomaa:~/Documents/kubernetes-sprints/lab3$
```

```yaml
apiVersion: v1
kind: Service
metadata:
  name: mongo-service
  labels:
    app: database
spec:
  selector:
    app: database
  type: ClusterIP
  ports:
    - protocol: TCP
      port: 3000
      targetPort: 3000
~
~
```

```
amr@amrgomaa:~/Documents/kubernetes-sprints/lab3$ kubectl apply -f service.yml
service/mongo-service created
```

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 30008- How many Nodes exist on the system?

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: webapp-deployment
  labels:
    app: webapp
spec:
  replicas: 1
  selector:
    matchLabels:
      app: webapp
  template:
    metadata:
      labels:
        app: webapp
    spec:
      containers:
      - image: nanajanashia/k8s-demo-app:v1.0
        name: webapp
        ports:
          - containerPort: 3000
        envFrom:
          - configMapRef:
              name: mongodb-endpoint
          - secretRef:
              name: mysecret
~
```

```
amr@amrgomaa:~/Documents/kubernetes-sprints/lab3$ kubectl apply -f deploymentweb.yml
\deployment.apps/webapp-deployment created
amr@amrgomaa:~/Documents/kubernetes-sprints/lab3$
```

```yaml
apiVersion: v1
kind: Service
metadata:
  name: web-app-service
  labels:
    app: database
spec:
  selector:
    app: webapp
  type: NodePort
  ports:
    - protocol: TCP
      port: 3000
      targetPort: 3000
      nodePort: 30010
~
```

```
amr@amrgomaa:~/Documents/kubernetes-sprints/lab3$ kubectl apply -f appservice.yml
service/web-app-service created
```

```
amr@amrgomaa:~/Documents/kubernetes-sprints/lab3$ kubectl get nodes -owide
NAME       STATUS   ROLES           AGE   VERSION   INTERNAL-IP      EXTERNAL-IP   OS-IMAGE              KERNEL-VERSION   CONTAINER-RUNTIME
minikube   Ready    control-plane   22d   v1.25.3   192.168.59.101   <none>        Buildroot 2021.02.12   5.10.57          docker://20.10.20
amr@amrgomaa:~/Documents/kubernetes-sprints/lab3$ curl 192.168.59.101:30010
```

```
amr@amrgomaa:~/Documents/kubernetes-sprints/lab3$ curl 192.168.59.101:30010
<html lang="en">
<style>
    .container {
        margin: 40px auto;
        width: 80%;
    }
    .button {
        width: 160px;
        height: 45px;
        border-radius: 6px;
        font-size: 15px;
        margin-top: 20px;
    }
    img {
        width: 328px;
        height: 287px;
        display: block;
        margin-bottom: 20px;
    }
    hr {
        width: 400px;
        margin-left: 0;
    }
    h3 {
        display: inline-block;
    }
    #container {
        display: none;
    }
    #container-edit {
        display: none;
    }
    #container-edit input {
        height: 32px;
    }
    #container-edit hr {
        margin: 25px 0;
    }
    #container-edit input {
        width: 195px;
        font-size: 15px;
    }
</style>
<script>
    (async function init() {
        const response = await fetch(`http://${window.location.host}/get-profile`);
        console.log("response", response);
        const user = await response.json();
        console.log(JSON.stringify(user));

        document.getElementById('name').textContent = user.name ? user.name : 'Anna Smith';
        document.getElementById('email').textContent = user.email ? user.email : 'anna.smith@example.com';
```

8- How many Nodes exist on the system?

```
Editor   Tab 1   +
Initialising Kubernetes... done

controlplane $ kubectl get nodes
NAME           STATUS   ROLES           AGE    VERSION
controlplane   Ready    control-plane   3d8h   v1.26.0
node01         Ready    <none>          3d7h   v1.26.0
controlplane $
```

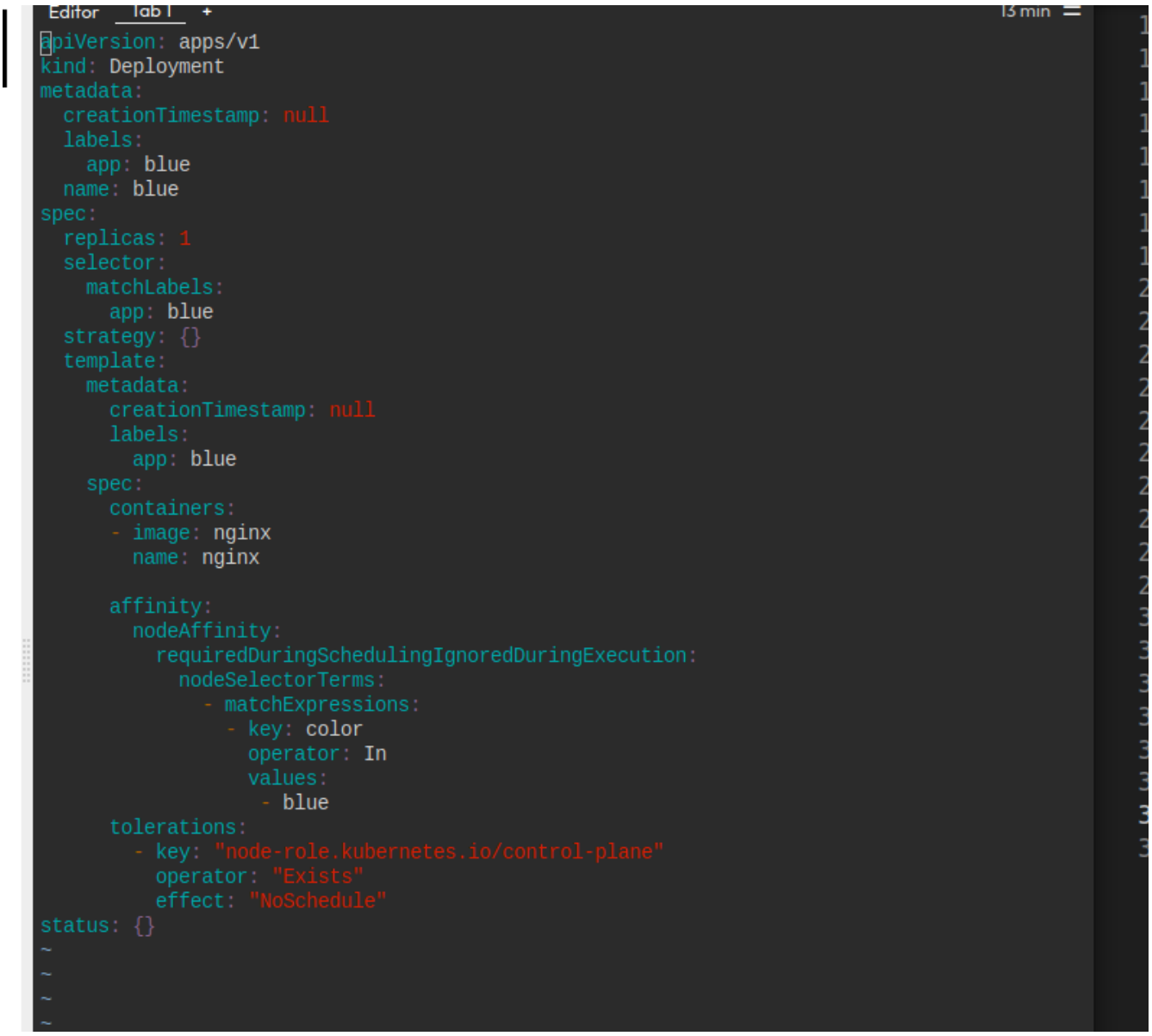
9- Do you see any taints on master ?

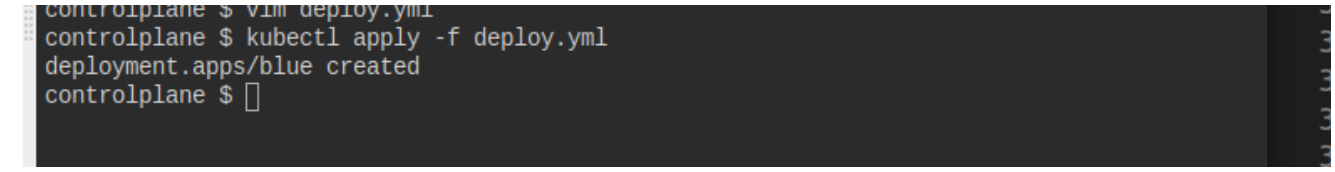
```
Editor   Tab 1   +
Initialising Kubernetes... done

controlplane $ kubectl get nodes
NAME           STATUS   ROLES           AGE    VERSION
controlplane   Ready    control-plane   3d8h   v1.26.0
node01         Ready    <none>          3d7h   v1.26.0
controlplane $ kubectl describe node controlplane | grep Taints
Taints:             node-role.kubernetes.io/control-plane:NoSchedule
controlplane $
```

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

```
Editor   Tab 1   +
controlplane $ kubectl label node controlplane color=blue
node/controlplane labeled
controlplane $
```

11- 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





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

```
controlplane $ kubectl taint nodes node01 spray=mortein:NoSchedule
node/node01 tainted
controlplane $ kubectl describe node node01 | grep Taints
Taints:             spray=mortein:NoSchedule
controlplane $
```

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

```
controlplane $ vim pod.yml
controlplane $ kubectl run mosquito --image nginx --port=80
```

14- What is the state o the mosquito POD?

```
controlplane $ kubectl get pod  mosquito
NAME       READY   STATUS    RESTARTS   AGE
mosquito   0/1     Pending   0          13m
```

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

```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 $ kubectl get pods
NAME       READY   STATUS    RESTARTS   AGE
bee        1/1     Running   0          26s
mosquito   0/1     Pending   0          7m34s
controlplane $
```