Lab 5 : Services

1. Create a Deployment as **my-deploy** using the image **paulbouwer/hello-kubernetes:1.10** on port **8080**.

Shell

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-deploy
spec:
  replicas: 2
  selector:
    matchLabels:
      app: my-deploy
  template:
    metadata:
      labels:
        app: my-deploy
    spec:
      containers:
      - name: my-deploy
        image: paulbouwer/hello-kubernetes:1.10
        ports:
        - containerPort: 8080
```

Shell

```
k apply -f deploy.yaml

k get deploy
```

2. Create a ClusterIP service to internally expose your deployment on port **80**.

Shell

```
apiVersion: v1
kind: Service
metadata:
  name: my-svc
```

```
spec:
  type: ClusterIP
  selector:
    app: my-deploy
  ports:
  - port: 80
    targetPort: 8080
```

Shell

```
k apply -f service.yaml

k get svc
```

3. Deploy a troubleshooting pod using the image **nicolaka/netshoot** to check the connectivity to the service.

Shell

```
kubectl get svc my-svc -o jsonpath='{.spec.clusterIP}'
```

🔥nicolaka/netshoot - Docker Image

Shell

```
kubectl run tmp-shell --rm -i --tty --image nicolaka/netshoot -- /bin/bash
tmp-shell:~# telnet <svc_ip> 80
Connected to <svc_ip>
```

4. Expose the Deployment externally using **NodePort** on port **30001**.

Shell

```
apiVersion: v1
kind: Service
metadata:
  name: my-svc
spec:
  type: NodePort
  selector:
    app: my-deploy
  ports:
  - port: 80
    targetPort: 8080
    nodePort: 30001
```

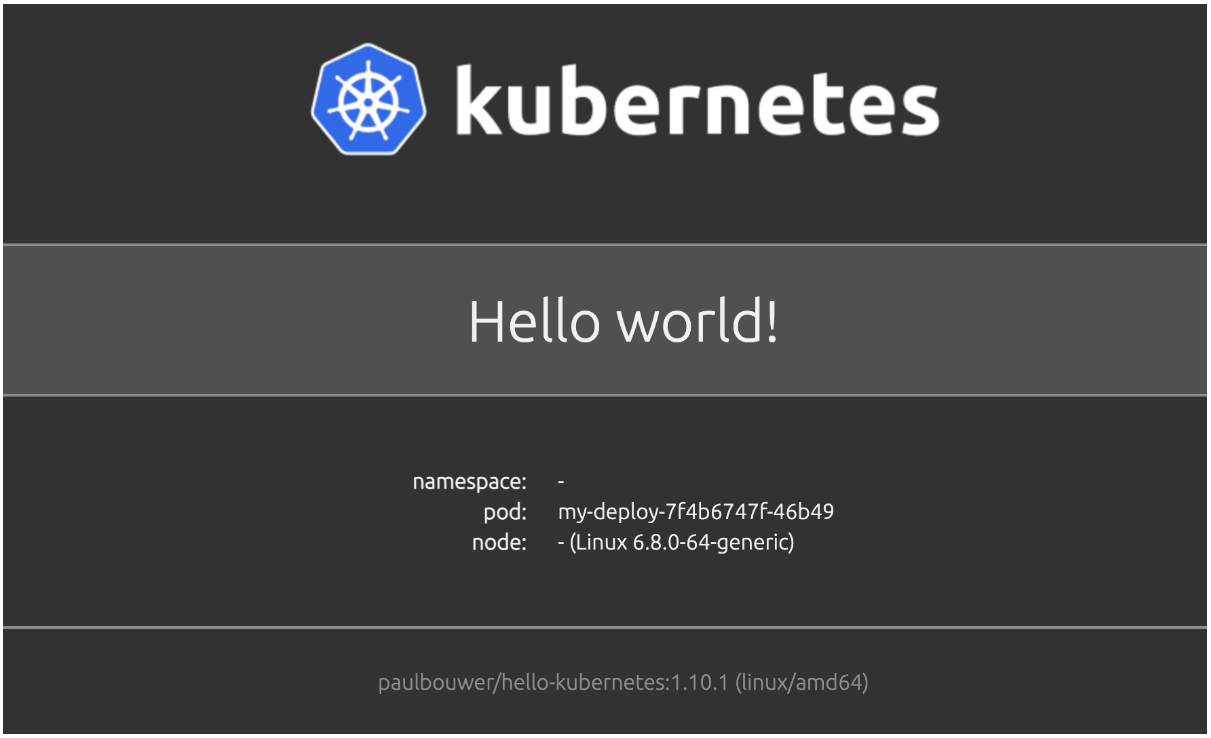
Shell

```
k apply -f nodeport.yaml

k get scv
```

5. Access the NodePort service using the minikube command for it.

None

```
minikube service my-svc --url

http://127.0.0.1:55616
```



Hello world!

```
namespace:   -
      pod:   my-deploy-7f4b6747f-46b49
     node:   - (Linux 6.8.0-64-generic)
```

paulbouwer/hello-kubernetes:1.10.1 (linux/amd64)