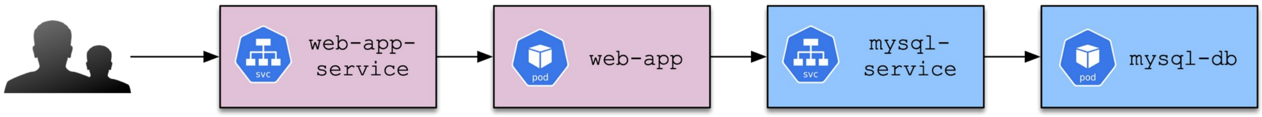
Lab15 – Troubleshooting an issue for an application

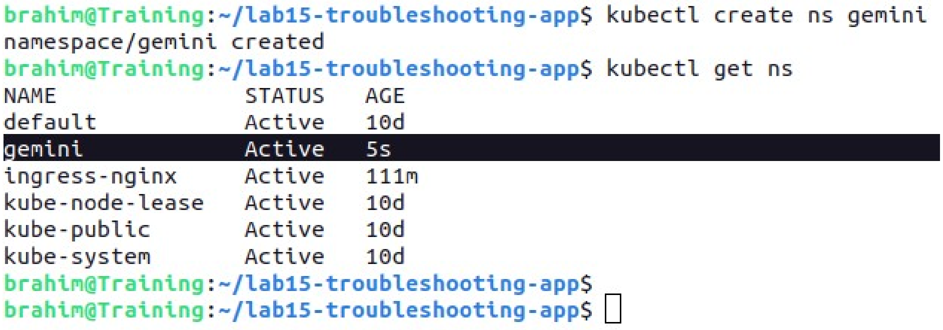
In this exercise, you will troubleshoot a misconfigured application stack. The application stack consists of a web application implemented using node.js, and a MySQL database. The web application connects to the database upon requesting its endpoint. Web application and MySQL database run in a Pod. Both Pods have been exposed by a Service. The Service for the web application Pod is of type `NodePort`. The Service for the MySQL database is of type `ClusterIP`.

The following image shows the high-level architecture.



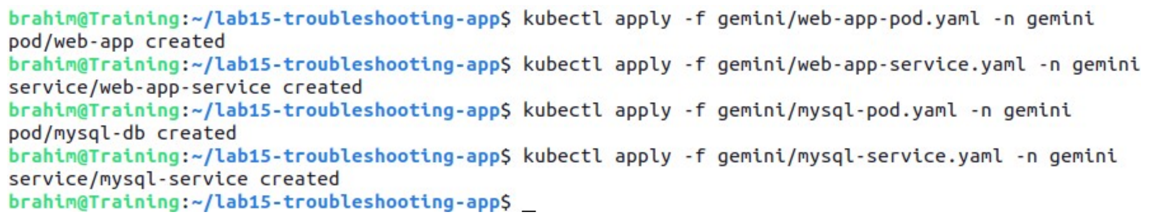
Fixing the issue in namespace "gemini"

1. Create a new namespace named `gemini`.



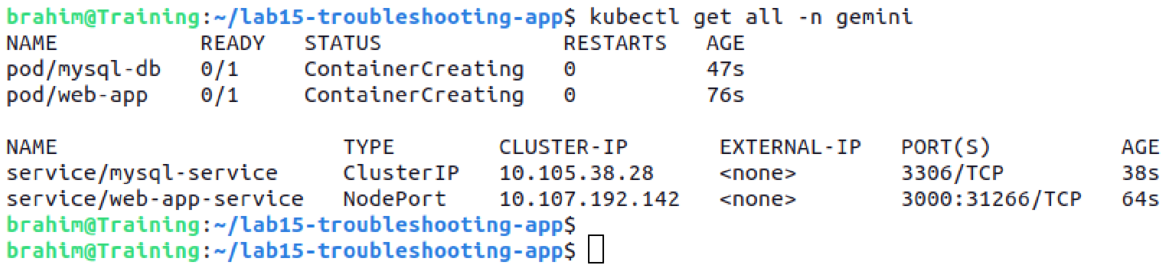
```
brahim@Training:~/lab15-troubleshooting-app$ kubectl create ns gemini
namespace/gemini created
brahim@Training:~/lab15-troubleshooting-app$ kubectl get ns
NAME              STATUS   AGE
default           Active   10d
gemini            Active   5s
ingress-nginx     Active   111m
kube-node-lease   Active   10d
kube-public       Active   10d
kube-system       Active   10d
brahim@Training:~/lab15-troubleshooting-app$
brahim@Training:~/lab15-troubleshooting-app$ 
```

2. Within the namespace, create the following objects in the given order from the YAML files: `gemini/mysql-pod.yaml`, `gemini/mysql-service.yaml`, `gemini/web-app-pod.yaml`, `gemini/web-app-service.yaml`.



```
brahim@Training:~/lab15-troubleshooting-app$ kubectl apply -f gemini/web-app-pod.yaml -n gemini
pod/web-app created
brahim@Training:~/lab15-troubleshooting-app$ kubectl apply -f gemini/web-app-service.yaml -n gemini
service/web-app-service created
brahim@Training:~/lab15-troubleshooting-app$ kubectl apply -f gemini/mysql-pod.yaml -n gemini
pod/mysql-db created
brahim@Training:~/lab15-troubleshooting-app$ kubectl apply -f gemini/mysql-service.yaml -n gemini
service/mysql-service created
brahim@Training:~/lab15-troubleshooting-app$ _
```

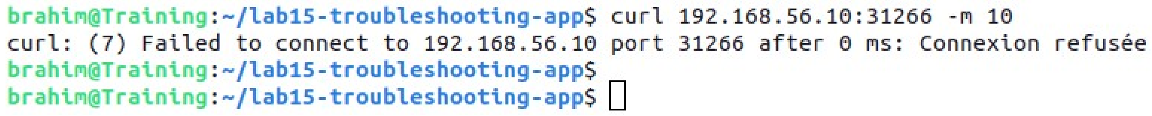
3. List all the objects and ensure that their status shows `Running`.



```
brahim@Training:~/lab15-troubleshooting-app$ kubectl get all -n gemini
NAME             READY   STATUS              RESTARTS   AGE
pod/mysql-db     0/1     ContainerCreating   0          47s
pod/web-app      0/1     ContainerCreating   0          76s

NAME                      TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
service/mysql-service     ClusterIP   10.105.38.28     <none>        3306/TCP         38s
service/web-app-service   NodePort    10.107.192.142   <none>        3000:31266/TCP   64s
brahim@Training:~/lab15-troubleshooting-app$
brahim@Training:~/lab15-troubleshooting-app$
```

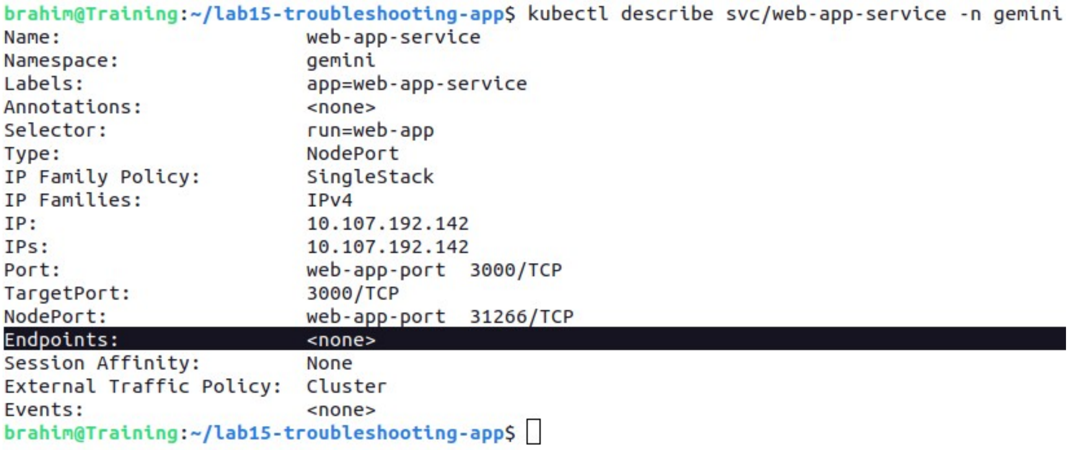
4. The Pod running web application exposes the container port 3000. From your machine, use your browser or execute `curl` or `wget` to access the application through the Service endpoint from outside of the cluster. A successful response should render `Successfully connected to database!`, a failure response should render `Failed to connect to database: <error message>`.



```
brahim@Training:~/lab15-troubleshooting-app$ curl 192.168.56.10:31266 -m 10
curl: (7) Failed to connect to 192.168.56.10 port 31266 after 0 ms: Connexion refusée
brahim@Training:~/lab15-troubleshooting-app$
brahim@Training:~/lab15-troubleshooting-app$
```

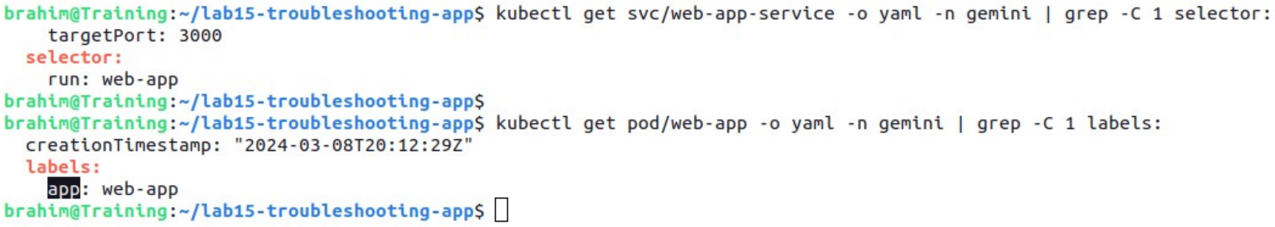
5. Identify the underlying issue and fix it.

   Have a look at the details of the `web-app-service`. You will see that no endpoint is listed so something's wrong.



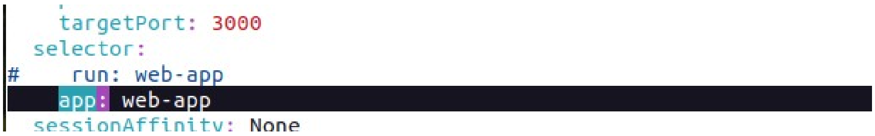
```
brahim@Training:~/lab15-troubleshooting-app$ kubectl describe svc/web-app-service -n gemini
Name:                     web-app-service
Namespace:                gemini
Labels:                   app=web-app-service
Annotations:              <none>
Selector:                 run=web-app
Type:                     NodePort
IP Family Policy:         SingleStack
IP Families:              IPv4
IP:                       10.107.192.142
IPs:                      10.107.192.142
Port:                     web-app-port  3000/TCP
TargetPort:               3000/TCP
NodePort:                 web-app-port  31266/TCP
Endpoints:                <none>
Session Affinity:         None
External Traffic Policy:  Cluster
Events:                   <none>
brahim@Training:~/lab15-troubleshooting-app$ 
```

Upon further inspection, you will find that the Service is using the label selector `run: web-app`, however, the assigned label to the Pod is `app: web-app`.

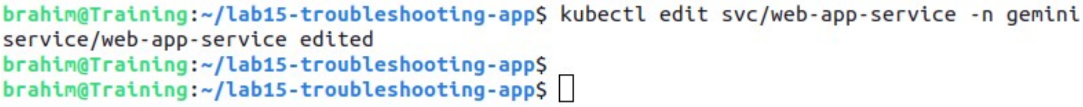


```
brahim@Training:~/lab15-troubleshooting-app$ kubectl get svc/web-app-service -o yaml -n gemini | grep -C 1 selector:
    targetPort: 3000
  selector:
    run: web-app
brahim@Training:~/lab15-troubleshooting-app$
brahim@Training:~/lab15-troubleshooting-app$ kubectl get pod/web-app -o yaml -n gemini | grep -C 1 labels:
  creationTimestamp: "2024-03-08T20:12:29Z"
  labels:
    app: web-app
brahim@Training:~/lab15-troubleshooting-app$ 
```

Change the label selector by editing the live objects.

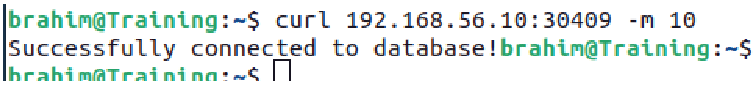


```
    targetPort: 3000
  selector:
#    run: web-app
    app: web-app
  sessionAffinity: None
```

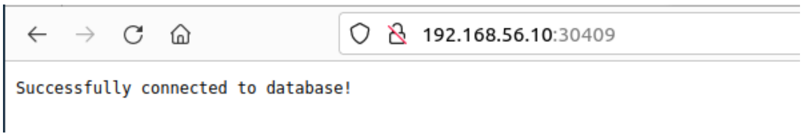


```
brahim@Training:~/lab15-troubleshooting-app$ kubectl edit svc/web-app-service -n gemini
service/web-app-service edited
brahim@Training:~/lab15-troubleshooting-app$
brahim@Training:~/lab15-troubleshooting-app$ 
```

You can now connect to the web application via the Service.



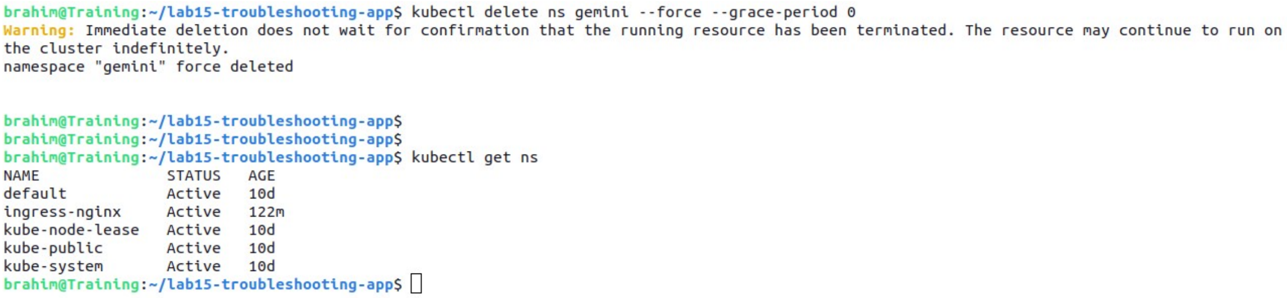
```
brahim@Training:~$ curl 192.168.56.10:30409 -m 10
Successfully connected to database!brahim@Training:~$
brahim@Training:~$ 
```



192.168.56.10:30409

Successfully connected to database!

6. Delete the namespace `gemini`.

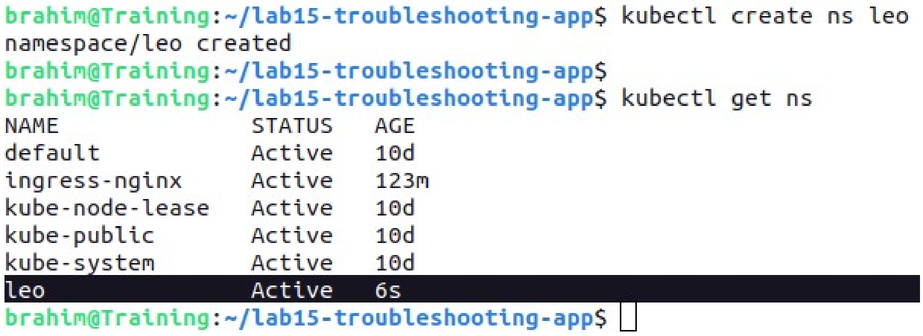


```
brahim@Training:~/lab15-troubleshooting-app$ kubectl delete ns gemini --force --grace-period 0
Warning: Immediate deletion does not wait for confirmation that the running resource has been terminated. The resource may continue to run on
the cluster indefinitely.
namespace "gemini" force deleted


brahim@Training:~/lab15-troubleshooting-app$
brahim@Training:~/lab15-troubleshooting-app$
brahim@Training:~/lab15-troubleshooting-app$ kubectl get ns
NAME              STATUS   AGE
default           Active   10d
ingress-nginx     Active   122m
kube-node-lease   Active   10d
kube-public       Active   10d
kube-system       Active   10d
brahim@Training:~/lab15-troubleshooting-app$
```

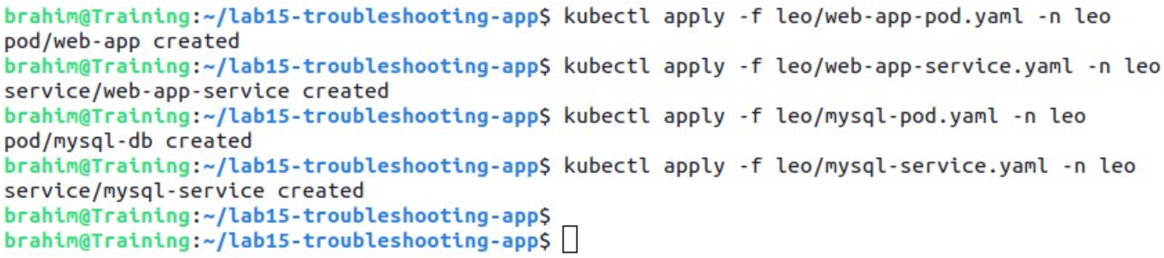
Fixing the issue in namespace "leo"

7. Create a new namespace named `leo`.



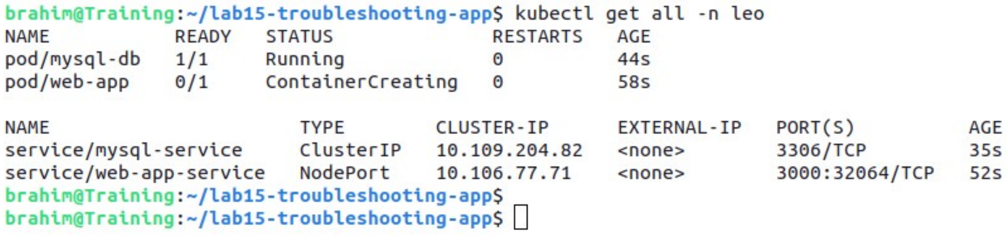
```
brahim@Training:~/lab15-troubleshooting-app$ kubectl create ns leo
namespace/leo created
brahim@Training:~/lab15-troubleshooting-app$
brahim@Training:~/lab15-troubleshooting-app$ kubectl get ns
NAME              STATUS   AGE
default           Active   10d
ingress-nginx     Active   123m
kube-node-lease   Active   10d
kube-public       Active   10d
kube-system       Active   10d
leo               Active   6s
brahim@Training:~/lab15-troubleshooting-app$
```

8. Within the namespace, create the following objects in the given order from the YAML files: `leo/mysql-pod.yaml`, `leo/mysql-service.yaml`, `leo/web-app-pod.yaml`, `leo/web-app-service.yaml`.



```
brahim@Training:~/lab15-troubleshooting-app$ kubectl apply -f leo/web-app-pod.yaml -n leo
pod/web-app created
brahim@Training:~/lab15-troubleshooting-app$ kubectl apply -f leo/web-app-service.yaml -n leo
service/web-app-service created
brahim@Training:~/lab15-troubleshooting-app$ kubectl apply -f leo/mysql-pod.yaml -n leo
pod/mysql-db created
brahim@Training:~/lab15-troubleshooting-app$ kubectl apply -f leo/mysql-service.yaml -n leo
service/mysql-service created
brahim@Training:~/lab15-troubleshooting-app$
brahim@Training:~/lab15-troubleshooting-app$
```

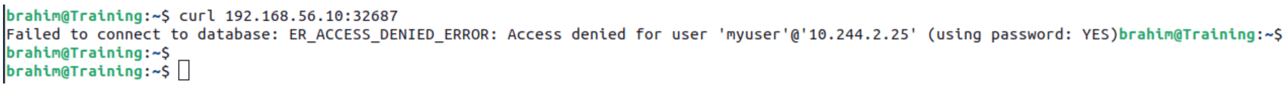
9. List all the objects and ensure that their status shows `Running`.



```
brahim@Training:~/lab15-troubleshooting-app$ kubectl get all -n leo
NAME           READY   STATUS              RESTARTS   AGE
pod/mysql-db   1/1     Running             0          44s
pod/web-app    0/1     ContainerCreating   0          58s

NAME                       TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
service/mysql-service      ClusterIP   10.109.204.82   <none>        3306/TCP         35s
service/web-app-service    NodePort    10.106.77.71    <none>        3000:32064/TCP   52s
brahim@Training:~/lab15-troubleshooting-app$
brahim@Training:~/lab15-troubleshooting-app$ 
```

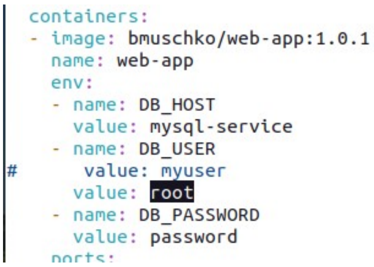
10. The Pod running web application exposes the container port 3000. From your machine, use your browser or execute `curl` or `wget` to access the application through the Service endpoint from outside of the cluster. A successful response should render `Successfully connected to database!`, a failure response should render `Failed to connect to database: <error message>`.



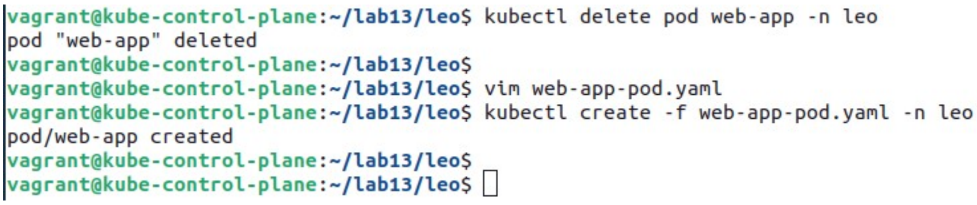
```
brahim@Training:~$ curl 192.168.56.10:32687
Failed to connect to database: ER_ACCESS_DENIED_ERROR: Access denied for user 'myuser'@'10.244.2.25' (using password: YES)brahim@Training:~$
brahim@Training:~$
brahim@Training:~$ 
```

11. Identify the underlying issue and fix it.

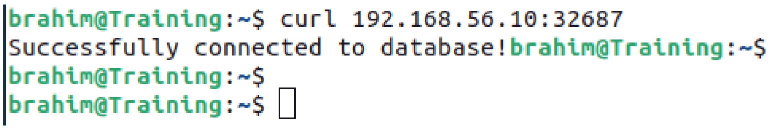
The MySQL Pod does not define a user named `myuser`. The only user that's available is the user named `root`. Therefore, we'll need to change the value of the environment variable `DB_USER` in the `web-app` Pod. Environment variables cannot be changed for a live object. Therefore, the Pod needs to be deleted and recreated.



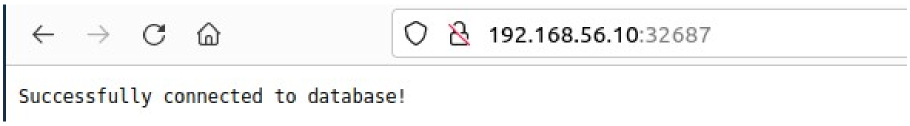
```
  containers:
  - image: bmuschko/web-app:1.0.1
    name: web-app
    env:
    - name: DB_HOST
      value: mysql-service
    - name: DB_USER
#      value: myuser
      value: root
    - name: DB_PASSWORD
      value: password
    ports:
```



```
vagrant@kube-control-plane:~/lab13/leo$ kubectl delete pod web-app -n leo
pod "web-app" deleted
vagrant@kube-control-plane:~/lab13/leo$
vagrant@kube-control-plane:~/lab13/leo$ vim web-app-pod.yaml
vagrant@kube-control-plane:~/lab13/leo$ kubectl create -f web-app-pod.yaml -n leo
pod/web-app created
vagrant@kube-control-plane:~/lab13/leo$
vagrant@kube-control-plane:~/lab13/leo$ 
```

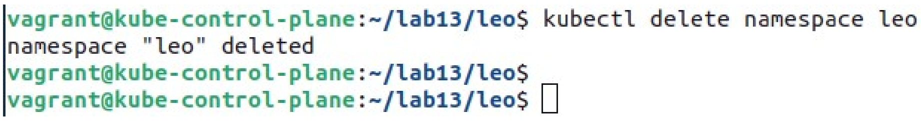


```
brahim@Training:~$ curl 192.168.56.10:32687
Successfully connected to database!brahim@Training:~$
brahim@Training:~$
brahim@Training:~$ 
```



```
Successfully connected to database!
```

12. Delete the namespace `leo`.



```
vagrant@kube-control-plane:~/lab13/leo$ kubectl delete namespace leo
namespace "leo" deleted
vagrant@kube-control-plane:~/lab13/leo$
vagrant@kube-control-plane:~/lab13/leo$ 
```