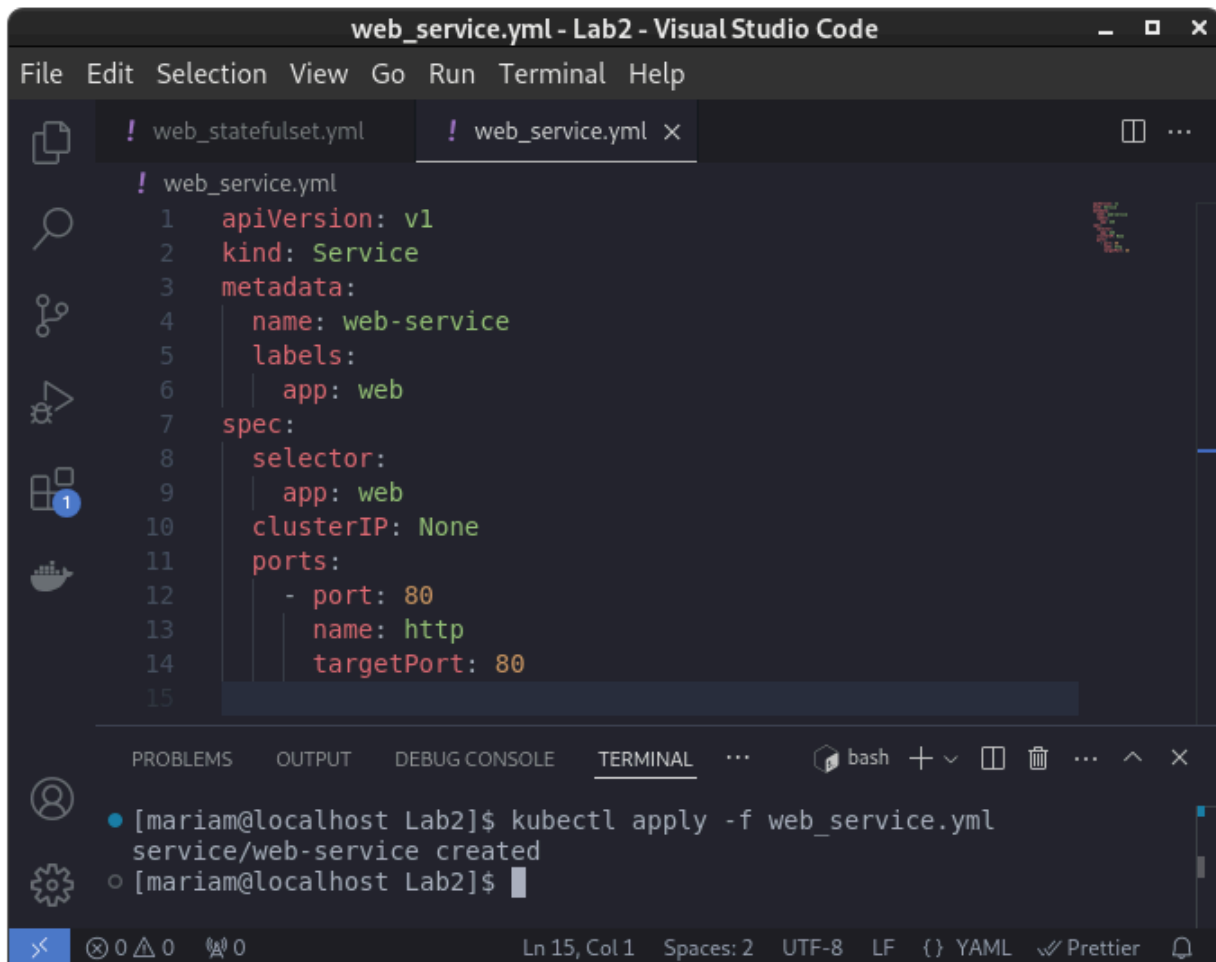


1. Create a StatefulSet named web-statefulset with 2 replicas using the nginx image.

The StatefulSet should have a Headless Service named web-service



The screenshot shows the Visual Studio Code editor with a file named `web_service.yml` open. The file contains a Kubernetes Service manifest. The terminal at the bottom shows the command `kubectl apply -f web_service.yml` being executed, resulting in the service being created.

```
web_service.yml - Lab2 - Visual Studio Code
File Edit Selection View Go Run Terminal Help

! web_statefulset.yml  ! web_service.yml x

! web_service.yml
1  apiVersion: v1
2  kind: Service
3  metadata:
4    name: web-service
5    labels:
6      app: web
7  spec:
8    selector:
9      app: web
10   clusterIP: None
11   ports:
12     - port: 80
13       name: http
14       targetPort: 80
15

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL ... bash + - [x] [trash] ... ^ x
• [mariam@localhost Lab2]$ kubectl apply -f web_service.yml
  service/web-service created
○ [mariam@localhost Lab2]$
```

Ln 15, Col 1 Spaces: 2 UTF-8 LF {} YAML ✓ Prettier

The screenshot shows the Visual Studio Code editor with a file named `web_statefulset.yml` open. The file contains a Kubernetes StatefulSet configuration. The terminal at the bottom shows the command `kubectl apply -f web_statefulset.yml` being executed, resulting in the message `statefulset.apps/web-statefulset created`.

```
web_statefulset.yml - Lab2 - Visual Studio Code
File Edit Selection View Go Run Terminal Help

! web_statefulset.yml x ! web_service.yml

! web_statefulset.yml
1  apiVersion: apps/v1
2  kind: StatefulSet
3  metadata:
4    name: web-statefulset
5  spec:
6    serviceName: web-service
7    replicas: 2
8    selector:
9      matchLabels:
10       app: web
11  template:
12    metadata:
13      labels:
14       app: web
15    spec:
16      containers:
17      - name: nginx
18        image: nginx:latest
19        ports:
20        - containerPort: 80

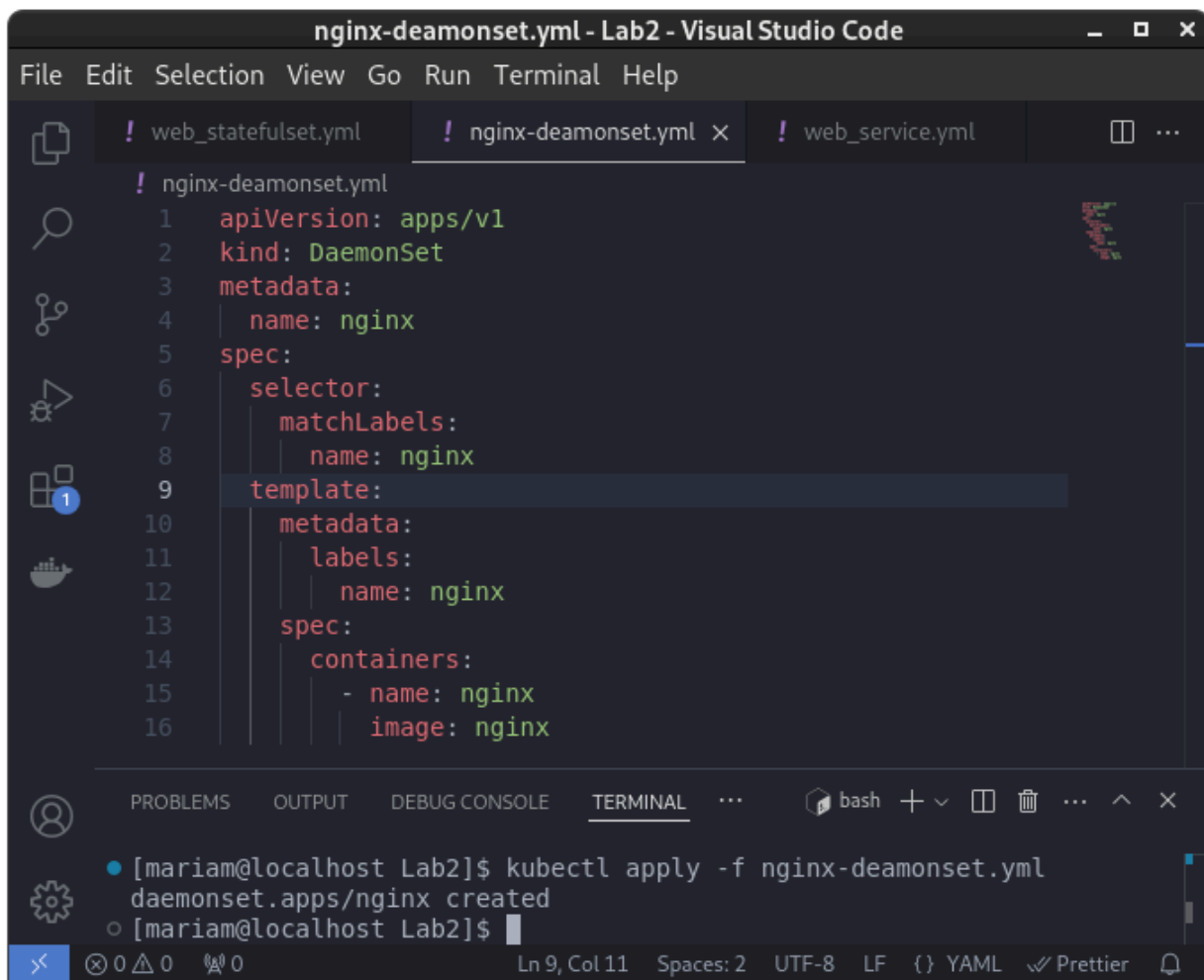
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL ... bash + - [x] [x] ... ^ x
• [mariam@localhost Lab2]$ kubectl apply -f web_statefulset.yml
statefulset.apps/web-statefulset created
○ [mariam@localhost Lab2]$
```

2. How many DaemonSets are created in the cluster in all namespaces?  
=> 0

The screenshot shows a terminal window with the command `kubectl get daemonsets` being executed. The output is `No resources found in default namespace.`

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL ... bash + - [x] [x] ... ^ x
• [mariam@localhost Lab2]$ kubectl get daemonsets
No resources found in default namespace.
○ [mariam@localhost Lab2]$
```

3. Create a DaemonSet named “nginx” with image “nginx”.



The screenshot shows the Visual Studio Code editor with a file named `nginx-deamonset.yml` open. The file contains a Kubernetes DaemonSet configuration for nginx. The configuration is as follows:

```
1  apiVersion: apps/v1
2  kind: DaemonSet
3  metadata:
4    name: nginx
5  spec:
6    selector:
7      matchLabels:
8        name: nginx
9    template:
10     metadata:
11       labels:
12         name: nginx
13     spec:
14       containers:
15         - name: nginx
16           image: nginx
```

The terminal at the bottom shows the command `kubectl apply -f nginx-deamonset.yml` being executed, resulting in the output `daemonset.apps/nginx created`.

4. How many pods have been created within the nginx DaemonSet and why?

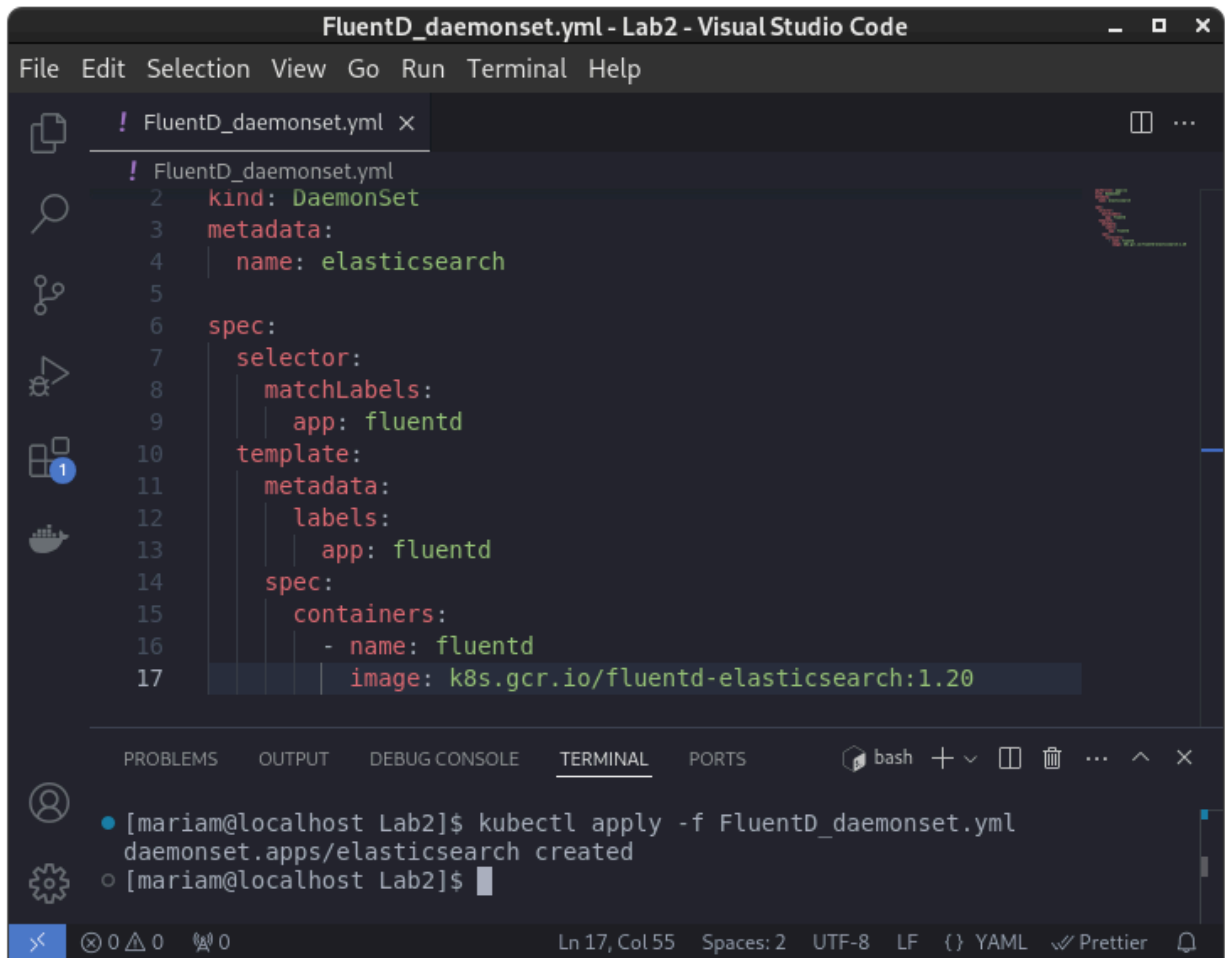
```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS bash + v [ ] [ ] ... ^ X
• [mariam@localhost Lab2]$ kubectl get daemonset nginx
NAME      DESIRED  CURRENT  READY  UP-TO-DATE  AVAILABLE  NODE SELECTOR
AGE
nginx     1        1        1      1            1          <none>
83s
○ [mariam@localhost Lab2]$
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS bash + v [ ] [ ] ... ^ X
• [mariam@localhost Lab2]$ kubectl get daemonset nginx
NAME      DESIRED  CURRENT  READY  UP-TO-DATE  AVAILABLE  NODE SELECTOR
AGE
nginx     1        1        1      1            1          <none>
83s
• [mariam@localhost Lab2]$ kubectl get nodes
NAME      STATUS  ROLES    AGE   VERSION
minikube  Ready   control-plane  24h   v1.32.0
○ [mariam@localhost Lab2]$
```

5. Create a pod named “ingot”

```
ingot_pod.yml - Lab2 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
! ingot_pod.yml x
! ingot_pod.yml
1  apiVersion: v1
2  kind: Pod
3  metadata:
4    name: ingot
5  spec:
6    containers:
7      - name: ingot-pod
8        image: alpine
9        command: ["sleep", "3600"]
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS bash + v [ ] [ ] ... ^ X
• [mariam@localhost Lab2]$ kubectl apply -f ingot_pod.yml
pod/ingot created
○ [mariam@localhost Lab2]$
```

6. Deploy a DaemonSet for FluentD Logging. Use the given specifications.  
→ Name: elasticsearch → Image: [k8s.gcr.io/fluentd-elasticsearch:1.20](https://k8s.gcr.io/fluentd-elasticsearch:1.20)



The image shows a Visual Studio Code editor window titled "FluentD\_daemonset.yml - Lab2 - Visual Studio Code". The editor displays a YAML file named "FluentD\_daemonset.yml" with the following content:

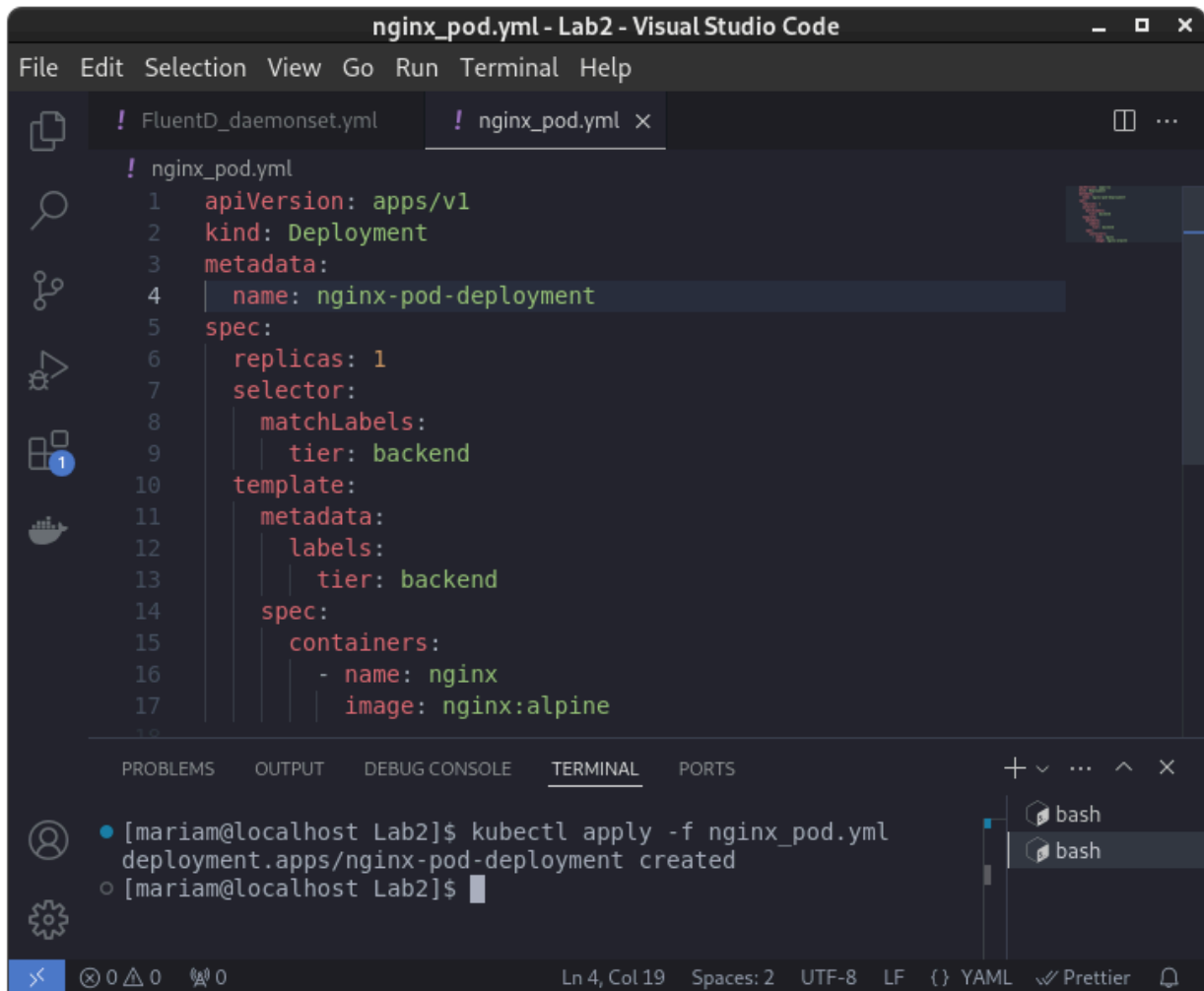
```
1 kind: DaemonSet
2 metadata:
3   name: elasticsearch
4 spec:
5   selector:
6     matchLabels:
7       app: fluentd
8   template:
9     metadata:
10       labels:
11         app: fluentd
12     spec:
13       containers:
14         - name: fluentd
15           image: k8s.gcr.io/fluentd-elasticsearch:1.20
```

The terminal at the bottom shows the command to apply the DaemonSet:

```
[mariam@localhost Lab2]$ kubectl apply -f FluentD_daemonset.yml
daemonset.apps/elasticsearch created
[mariam@localhost Lab2]$
```

The status bar at the bottom indicates the file is at line 17, column 55, with 2 spaces, UTF-8 encoding, LF line endings, and is a YAML file formatted by Prettier.

7. Deploy a pod named nginx-pod using the nginx:alpine image with the labels set to: tier=backend.



The screenshot shows the Visual Studio Code interface with a file named `nginx_pod.yml` open. The file contains a Kubernetes Deployment manifest for an nginx pod. The manifest is as follows:

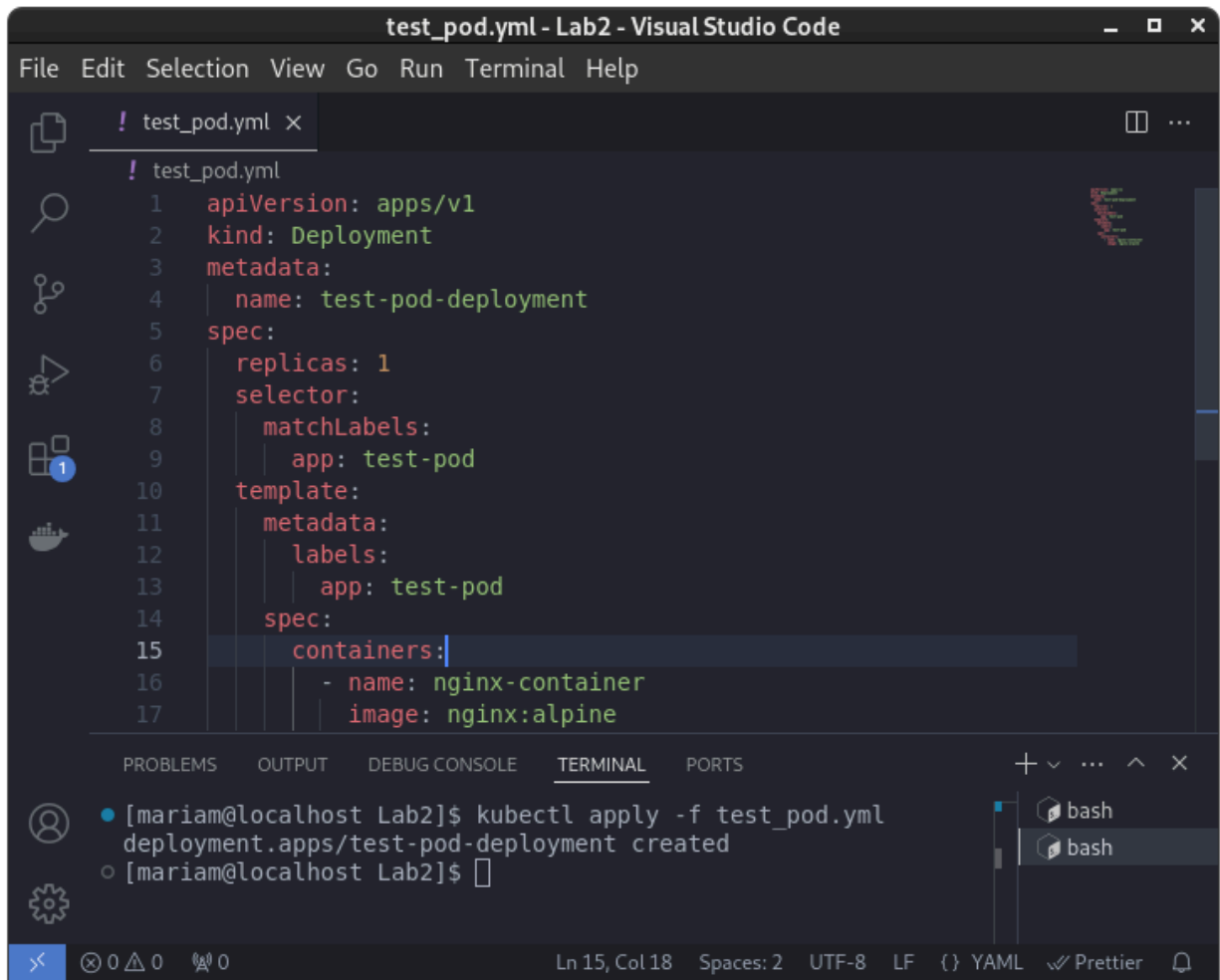
```
! nginx_pod.yml
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: nginx-pod-deployment
5  spec:
6    replicas: 1
7    selector:
8      matchLabels:
9        tier: backend
10   template:
11     metadata:
12       labels:
13         tier: backend
14     spec:
15       containers:
16       - name: nginx
17         image: nginx:alpine
```

The terminal at the bottom shows the command `kubectl apply -f nginx_pod.yml` being executed, resulting in the deployment being created. The terminal output is:

```
[mariam@localhost Lab2]$ kubectl apply -f nginx_pod.yml
deployment.apps/nginx-pod-deployment created
[mariam@localhost Lab2]$
```

The terminal also shows a `bash` prompt, indicating the user is in a shell.

8. Deploy a test pod using the nginx:alpine image.



The screenshot shows the Visual Studio Code interface with a file named `test_pod.yml` open. The file contains a Kubernetes Deployment manifest. The terminal at the bottom shows the command `kubectl apply -f test_pod.yml` being executed, resulting in the deployment being created. The status bar at the bottom indicates the file is a YAML document using Prettier formatting.

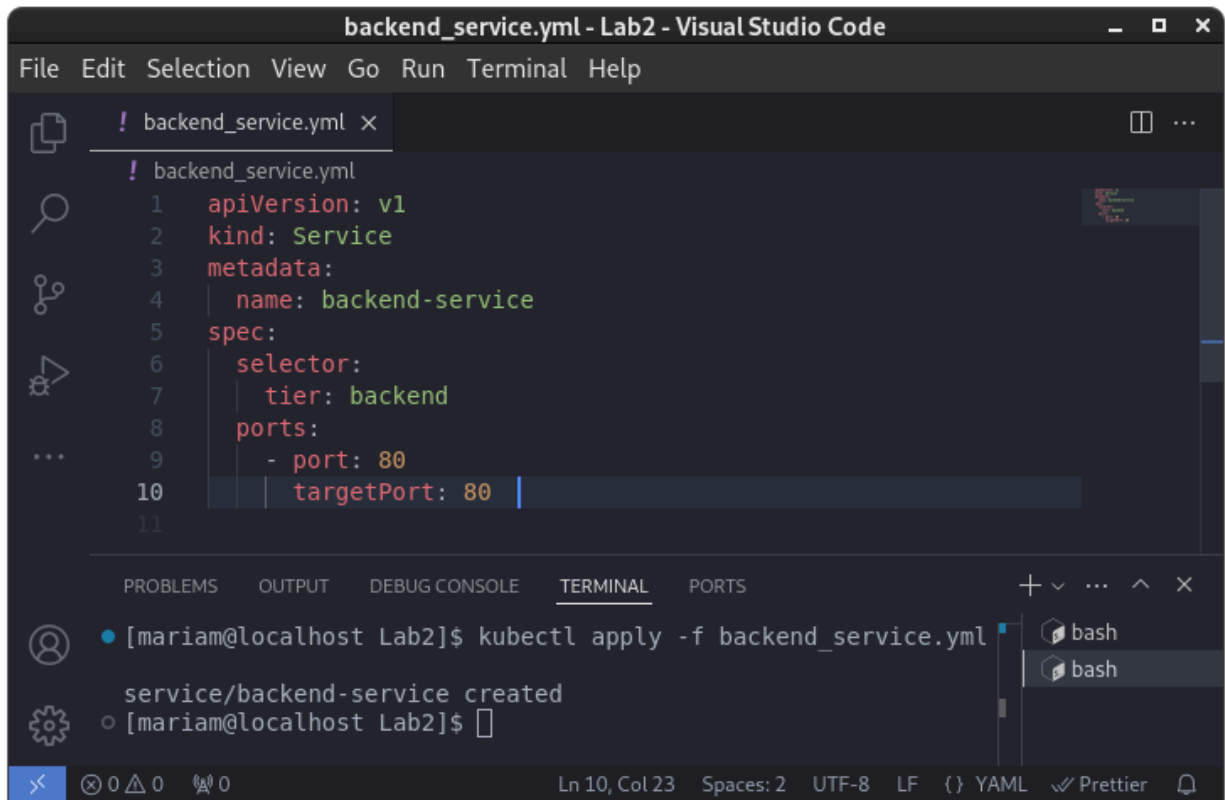
```
test_pod.yml - Lab2 - Visual Studio Code
File Edit Selection View Go Run Terminal Help

! test_pod.yml x
! test_pod.yml
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: test-pod-deployment
5  spec:
6    replicas: 1
7    selector:
8      matchLabels:
9        app: test-pod
10   template:
11     metadata:
12       labels:
13         app: test-pod
14     spec:
15       containers:
16       - name: nginx-container
17         image: nginx:alpine

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
• [mariam@localhost Lab2]$ kubectl apply -f test_pod.yml
deployment.apps/test-pod-deployment created
○ [mariam@localhost Lab2]$

Ln 15, Col 18 Spaces: 2 UTF-8 LF {} YAML Prettier
```

9. Create a service backend-service to expose the backend application within the cluster on port 80.



The screenshot shows the Visual Studio Code editor with a file named `backend_service.yml` open. The file contains a Kubernetes Service definition. The editor's interface includes a menu bar (File, Edit, Selection, View, Go, Run, Terminal, Help), a sidebar with icons for Explorer, Search, Source Control, and Run and Debug, and a bottom panel with tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and PORTS. The TERMINAL tab is active, showing the command `kubectl apply -f backend_service.yml` being executed, with the output `service/backend-service created`. The status bar at the bottom indicates the current position (Ln 10, Col 23), indentation (Spaces: 2), encoding (UTF-8), line ending (LF), language (YAML), and formatting (Prettier).

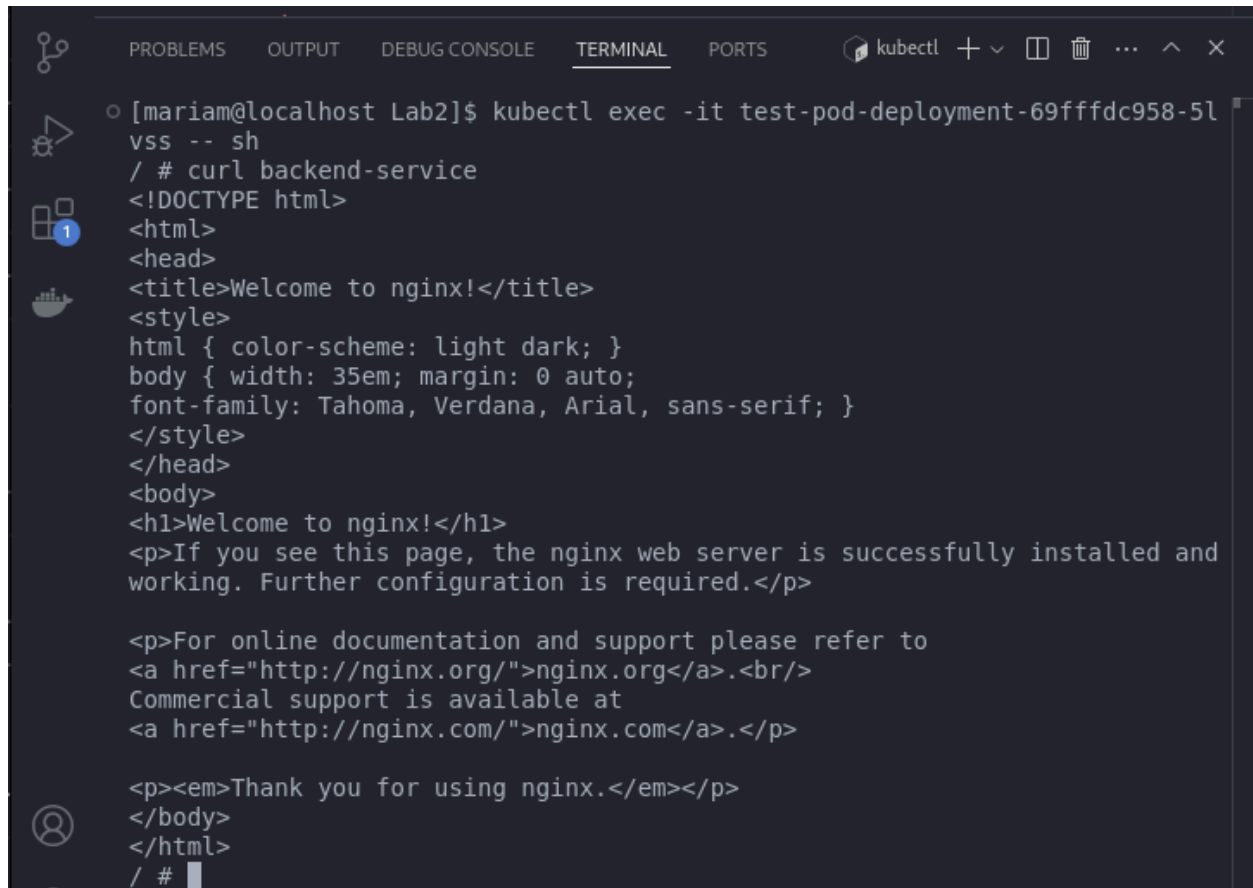
```
backend_service.yml - Lab2 - Visual Studio Code
File Edit Selection View Go Run Terminal Help

! backend_service.yml x
! backend_service.yml
1  apiVersion: v1
2  kind: Service
3  metadata:
4    name: backend-service
5  spec:
6    selector:
7      tier: backend
8    ports:
9      - port: 80
10     targetPort: 80
11

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
• [mariam@localhost Lab2]$ kubectl apply -f backend_service.yml
service/backend-service created
○ [mariam@localhost Lab2]$
```



10. Try to curl the backend-service from the test pod. What is the response?



```
[mariam@localhost Lab2]$ kubectl exec -it test-pod-deployment-69fffdc958-5l
vss -- sh
/ # curl backend-service
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
/ #
```

The response is an html page from nginx, which means the backend-service is correctly routing traffic to the nginx server that's running in the cluster.

11. Create a deployment named web-app using the image nginx with 2 replicas

The screenshot shows the Visual Studio Code editor with a file named `web_app_deployment.yml` open. The file contains a Kubernetes Deployment manifest for an application named `web-app`. The manifest specifies 2 replicas, uses the `nginx` image, and exposes port 80.

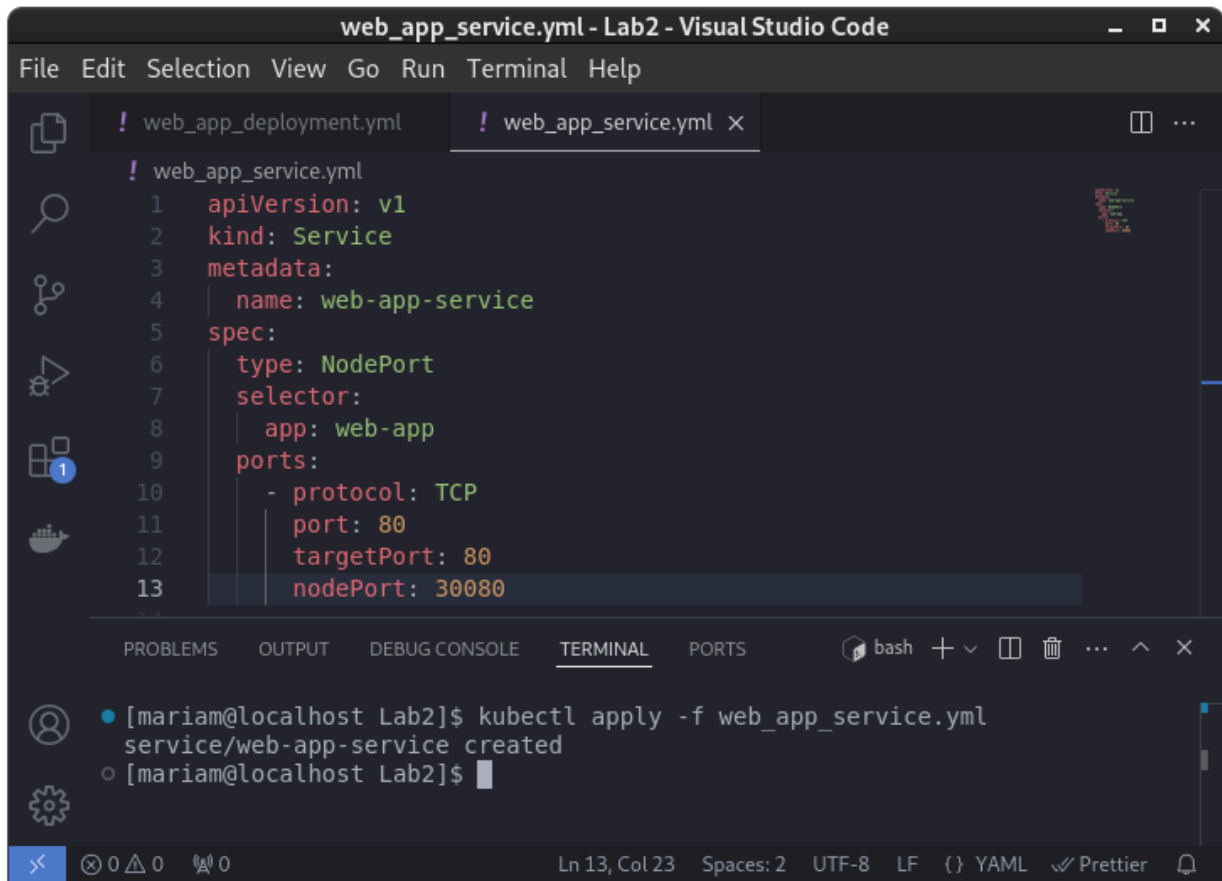
```
! web_app_deployment.yml
! web_app_deployment.yml
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: web-app
5  spec:
6    replicas: 2
7    selector:
8      matchLabels:
9        app: web-app
10   template:
11     metadata:
12       labels:
13         app: web-app
14     spec:
15       containers:
16       - name: nginx
17         image: nginx
18         ports:
19         - containerPort: 80
20
```

The bottom panel shows the **TERMINAL** view with the following commands and output:

```
[mariam@localhost Lab2]$ kubectl apply -f web_app_deployment.yml
deployment.apps/web-app created
[mariam@localhost Lab2]$
```

The status bar at the bottom indicates the current cursor position is at Line 20, Column 1, with 2 spaces, UTF-8 encoding, LF line endings, and the Prettier formatter is enabled.

12. Expose the web-app as service web-app-service application on port 80 and nodeport 30082 on the nodes on the cluster



The screenshot shows the Visual Studio Code editor with the file `web_app_service.yml` open. The file contains a Kubernetes Service definition for `web-app-service` of type `NodePort`, exposing port 80 on the node (30080) to port 80 of the pods. Below the editor, the terminal shows the command `kubectl apply -f web_app_service.yml` being executed, resulting in the service being created.

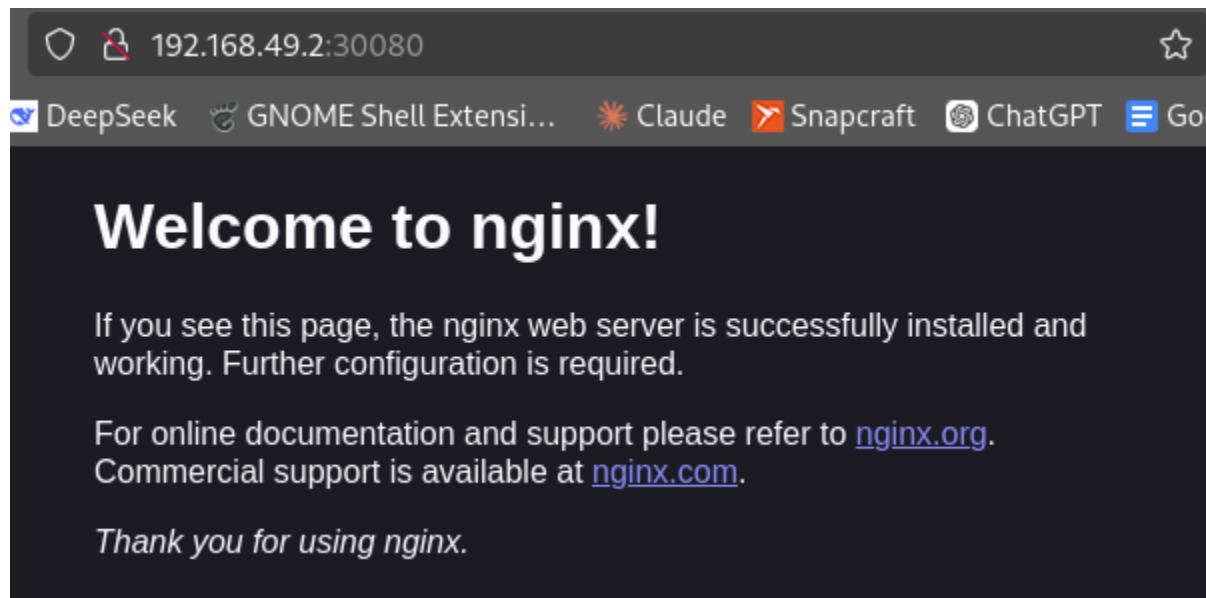
```
web_app_service.yml - Lab2 - Visual Studio Code
File Edit Selection View Go Run Terminal Help

! web_app_deployment.yml ! web_app_service.yml x

! web_app_service.yml
1  apiVersion: v1
2  kind: Service
3  metadata:
4    name: web-app-service
5  spec:
6    type: NodePort
7    selector:
8      app: web-app
9    ports:
10     - protocol: TCP
11       port: 80
12       targetPort: 80
13       nodePort: 30080

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
[mariam@localhost Lab2]$ kubectl apply -f web_app_service.yml
service/web-app-service created
[mariam@localhost Lab2]$
```

13. Access the web app from the node



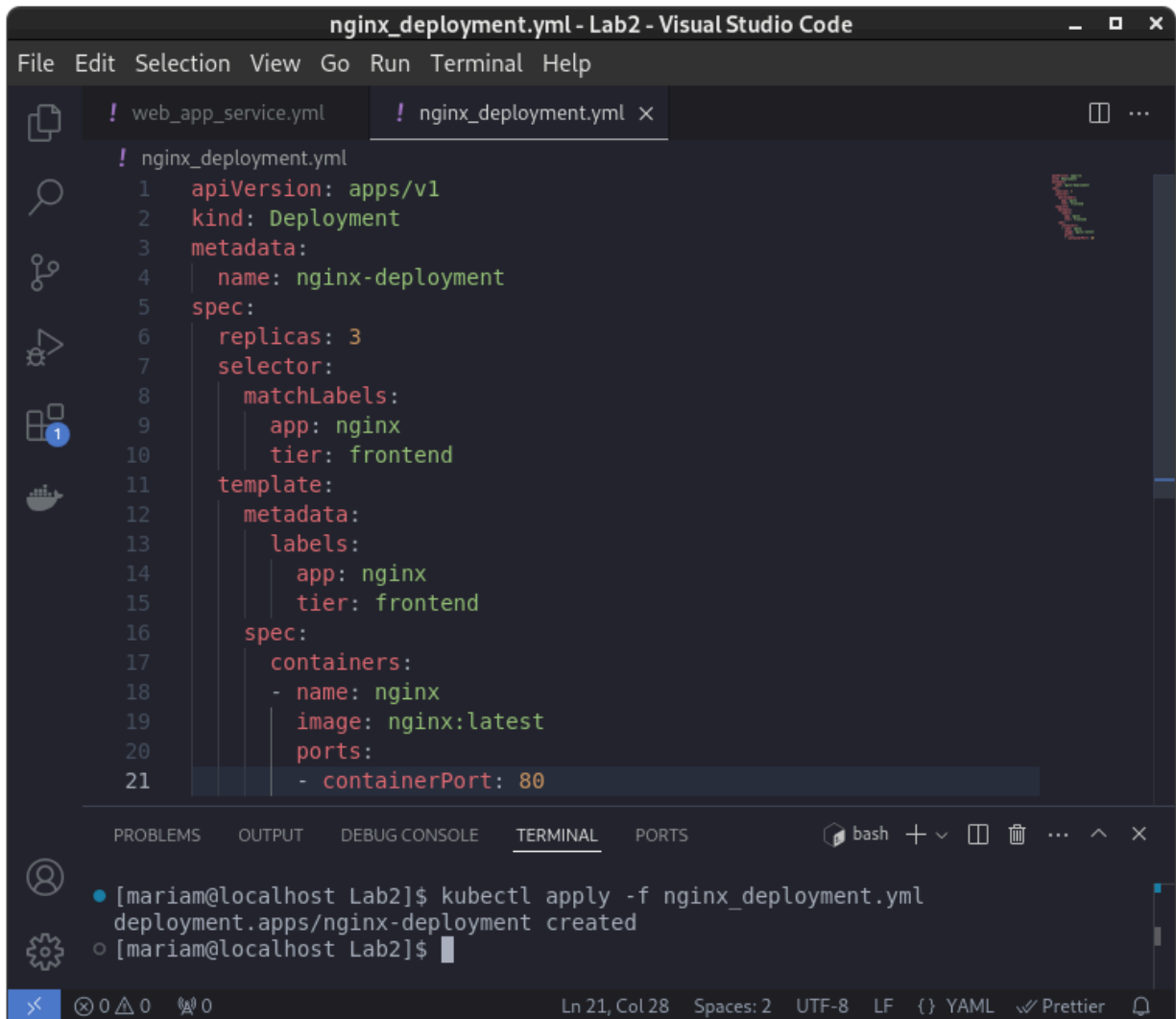
14. Create a deployment nginx with pod labels

→ app:nginx

→ tier:frontend

and set-based selectors on replicaset that allow filtering objects based on specific

conditions. Given the valid operators In, Exists.



```
nginx_deployment.yml - Lab2 - Visual Studio Code
File Edit Selection View Go Run Terminal Help

! web_app_service.yml ! nginx_deployment.yml x

! nginx_deployment.yml
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: nginx-deployment
5  spec:
6    replicas: 3
7    selector:
8      matchLabels:
9        app: nginx
10       tier: frontend
11  template:
12    metadata:
13      labels:
14        app: nginx
15        tier: frontend
16    spec:
17      containers:
18      - name: nginx
19        image: nginx:latest
20        ports:
21      - containerPort: 80

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
bash + v [] [] ... ^ x

• [mariaam@localhost Lab2]$ kubectl apply -f nginx_deployment.yml
deployment.apps/nginx-deployment created
○ [mariaam@localhost Lab2]$
```

15. When can we use the Loadbalancer service?

- This service in Kubernetes is used to expose applications externally, typically in cloud environments. It automatically provisions an external load balancer to distribute traffic to the Pods in cluster.