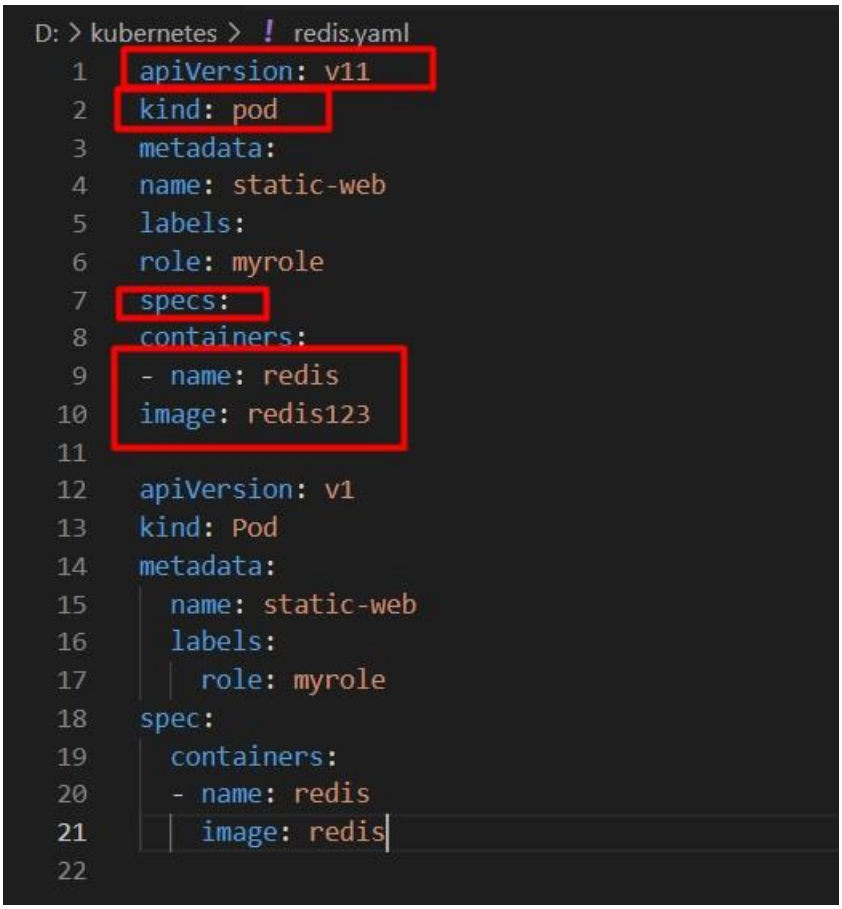
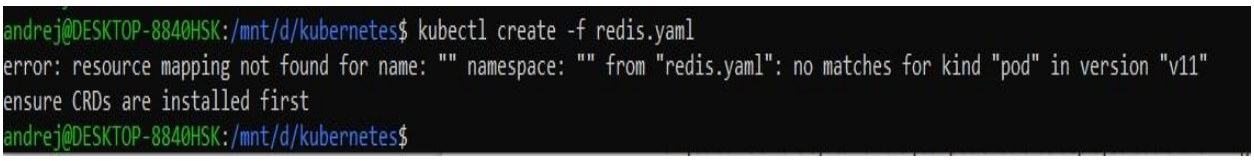
Practice2: Working with pod manifest files

1. Now it is time to deploy pod using manifest file (declarative approach). Copy the following code block on your local computer in a file called redis.yaml:



- The code block from the labs is not correct. From the picture **highlighted code blocks are not correct. The version below is correct.**



4. When you solve all the syntax errors your pod should be deployed but is it running? What is the status of your pod?

```
andrej@DESKTOP-8840HSK:/mnt/d/kubernetes$ kubectl get pods
NAME          READY   STATUS         RESTARTS   AGE
static-web    0/1     ErrImagePull   0          11s
andrej@DESKTOP-8840HSK:/mnt/d/kubernetes$
```

---The pod status is Error Image Pull

5. Check the events associated with this pod. Run the **kubectl describe pod static-web** command. What are the events showing? Why your pod is not running?

```
Events:
  Type     Reason     Age                From               Message
  ----     ------     ----               ----               -------
  Normal   Scheduled  106s               default-scheduler  Successfully assigned default/static-web to aks-agentpool-37021570-vmss000002
  Normal   Pulling    14s (x4 over 106s) kubelet            Pulling image "redis123"
  Warning  Failed     14s (x4 over 106s) kubelet            Failed to pull image "redis123": rpc error: code = Unknown desc = failed to pull and unpack image "docker.io/library/redis123:latest": failed
to resolve reference "docker.io/library/redis123:latest": pull access denied, repository does not exist or may require authorization: server message: insufficient_scope: authorization failed
  Warning  Failed     14s (x4 over 106s) kubelet            Error: ErrImagePull
  Normal   BackOff    3s (x6 over 105s)  kubelet            Back-off pulling image "redis123"
  Warning  Failed     3s (x6 over 105s)  kubelet            Error: ImagePullBackOff
```

--- The pod cannot find the image redis123, cannot find in the docker library, we can test to see if there is redis123 image.

```
andrej@DESKTOP-8840HSK:/mnt/d/kubernetes$ docker pull redis123
Using default tag: latest
Error response from daemon: pull access denied for redis123, repository does not exist or may require 'docker login': denied: requested access to the resource is denied
andrej@DESKTOP-8840HSK:/mnt/d/kubernetes$
```

7. Locate the image information and put the correct image name. Redeploy the pod (fist run kubectl delete pod static-web to delete the pod, then run kubectl create once again).

--- From the task 1, we can see that value for the image is "**redis**" and it pulling default version 6.2.4 of redis.

```
andrej@DESKTOP-8840HSK:/mnt/d/kubernetes$ kubectl delete pod static-web
pod "static-web" deleted
andrej@DESKTOP-8840HSK:/mnt/d/kubernetes$ kubectl create -f redis.yaml
pod/static-web created
andrej@DESKTOP-8840HSK:/mnt/d/kubernetes$ kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
static-web    1/1     Running   0          7s
```

9. Now you can delete the pod. Try to delete it using the **kubectl delete –f redis.yaml**

```
andrej@DESKTOP-8840HSK:/mnt/d/kubernetes$ kubectl delete -f redis.yaml
pod "static-web" deleted
andrej@DESKTOP-8840HSK:/mnt/d/kubernetes$
```

10. Your next task is to create and test nginx pod definition. Your definition should use the nginx official image, should use label named app with value frontend and should publish port 80. Make sure you complete this task because we will use this template in our next Labs. Your nginx pod should be running without any issues

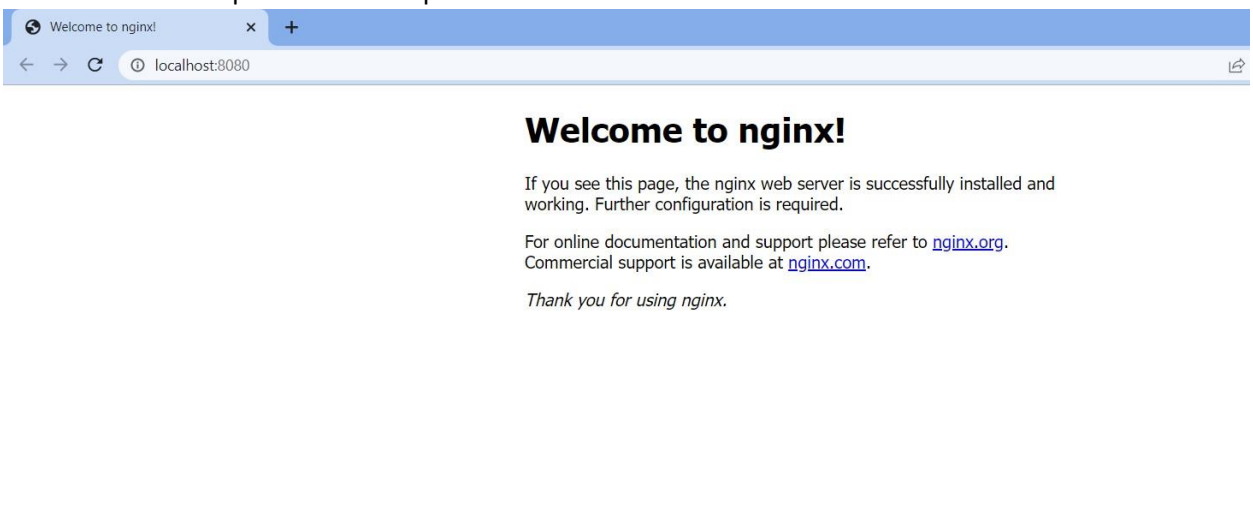
D: > kubernetes > ! nginx-pod.yaml

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: nginx-pod
  labels:
    app: frontend
spec:
  containers:
  - name: nginx
    image: nginx:latest
    ports:
    - containerPort: 80

```

```
andrej@DESKTOP-8840HSK:/mnt/d/kubernetes$ kubectl create -f nginx-pod.yaml
pod/nginx-pod created
andrej@DESKTOP-8840HSK:/mnt/d/kubernetes$ kubectl get pods
NAME        READY   STATUS    RESTARTS   AGE
nginx-pod   1/1     Running   0          10s
andrej@DESKTOP-8840HSK:/mnt/d/kubernetes$
```

We can forward to port 8080 and open in localhost:8080




```
andrej@DESKTOP-8840HSK:/mnt/d/kubernetes$ kubectl port-forward nginx-pod 8080:80
Forwarding from 127.0.0.1:8080 -> 80
Forwarding from [::1]:8080 -> 80
```

11. Final task of this practice will be to define pod definition with following details: - Image=memcached - Port= 11211 - Label app=web - CPU request=0.35 cores - RAM request=0.15 GB - CPU limit=0.5 cores - Ram limit=0.25 GB - Restart policy=Never

12. Don’t forget to try your pod definition.

```
): > kubernetes > ! memcached-pod.yaml
 1  apiVersion: v1
 2  kind: Pod
 3  metadata:
 4    name: memcached-pod
 5    labels:
 6      app: web
 7  spec:
 8    restartPolicy: Never
 9    containers:
10    - name: memcached
11      image: memcached:latest
12      ports:
13      - containerPort: 11211
14      resources:
15        requests:
16          cpu: 350m
17          memory: 150Mi
18        limits:
19          cpu: 500m
20          memory: 250Mi
21
```

```
andrej@DESKTOP-8840HSK:/mnt/d/kubernetes$ kubectl get pods
NAME            READY   STATUS    RESTARTS   AGE
memcached-pod   1/1     Running   0          6s
nginx-pod       1/1     Running   0          12m
```