Exercise 7.3: Rolling Updates and Rollbacks

One of the advantages of micro-services is the ability to replace and upgrade a container while continuing to respond to client requests. We will use the `recreate` setting that upgrades a container when the predecessor is deleted, then the use the `RollingUpdate` feature as well, which begins a rolling update immediately.

nginx versions

The **nginx** software updates on a distinct timeline from Kubernetes. If the lab shows an older version please use the current default, and then a newer version. Versions can be verified on the repositories on the registry

1. Begin by viewing the current `strategy` setting for the `Deployment` created in the previous section.

```
student@cp:~$ kubectl get deploy webserver -o yaml | grep -A 4 strategy
```

```
        strategy:
        rollingUpdate:
          maxSurge: 25%
          maxUnavailable: 25%
        type: RollingUpdate
```

2. Edit the object to use the `Recreate` update strategy. This would allow the manual termination of some of the pods, resulting in an updated image when they are recreated.

```
student@cp:~$ kubectl edit deploy webserver
```

```
....
strategy:
rollingUpdate:              # <-- remove this line
  maxSurge: 25%             # <-- remove this line
  maxUnavailable: 25%       # <-- remove this line
type: Recreate          # <-- Edit this line
:q....
```

3. Update the `Deployment` to use a newer version of the **nginx** server. This time use the **set** command instead of **edit**. Set the version to be `1.23.1-alpine`.

```
student@cp:~$ kubectl set image deploy webserver nginx=nginx:1.23.1-alpine
```

```
  deployment.apps/webserver image updated
```

4. Verify that the `Image:` parameter for the Pod checked in the previous section is unchanged.

```
        student@cp:~$ kubectl get pod
```

```
        NAME                         READY   STATUS    RESTARTS   AGE
  webserver-6cf9cd5c74-qjph4   1/1     Running   0          35s
  webserver-6cf9cd5c74-zc6x9   1/1     Running   0          35s
```

```
student@cp:~$ kubectl describe po webserver-6cf9cd5c74-qjph4 |grep Image:
```

```
  Image:          nginx:1.23.1-alpine
```

5. View the history of changes for the `Deployment`. You should see two revisions listed. As we did not add the the `change-cause annotation` we didn't see why the object updated.

```
student@cp:~$ kubectl rollout history deploy webserver
```

```
deployment.apps/webserver
REVISION  CHANGE-CAUSE
1         <none>
2         <none>
```

6. View the settings for the various versions of the `Deployment`. The `Image:` line should be the only difference between the two outputs.

```
student@cp:~$ kubectl rollout history deploy webserver --revision=1
```

```
deployment.apps/webserver with revision #1
Pod Template:
  Labels:       app=webserver
        pod-template-hash=6cbc654ddc
  Containers:
   nginx:
    Image:      nginx:1.22.1
    Port:       <none>
    Host Port:  <none>
    Environment:        <none>
    Mounts:     <none>
  Volumes:      <none>
```

```
student@cp:~$ kubectl rollout history deploy webserver --revision=2
```

```
....
  Image:      nginx:1.23.1-alpine
.....
```

7. Use `kubectl rollout undo` to change the `Deployment` back to previous version.

```
student@cp:~$ kubectl rollout undo deploy webserver
```

```
deployment.apps/webserver rolled back
```

```
student@cp:~$ kubectl get pod
```

```
 NAME                         READY   STATUS    RESTARTS   AGE
webserver-6cbc654ddc-7wb5q   1/1     Running   0          37s
webserver-6cbc654ddc-svbtj   1/1     Running   0          37s
```

```
student@cp:~$ kubectl describe pod webserver-6cbc654ddc-7wb5q |grep Image:
```

```
Image:              nginx:1.22.1
```

8. Clean up the system by removing the `Deployment`.

```
student@cp:~$ kubectl delete deploy webserver
```

```
deployment.apps "webserver" deleted
```