Lab 4: Deployments

Training Goals Covered:

- Use Deployments to manage application scaling and updates.
- Feel free to use either imperative commands or declarative configuration.

Steps:

1. Create a deployment as **my-deploy** with **4** replicas in the default namespace using the image **nginx:1.28**.

```Shell
k create deployment my-deploy --image=nginx:1.28 --replicas=4
```

2. Review the deployment location (nodes) of the pods.

```Shell
k get pod -o wide
```

3. Increase the number of replicas to **5**.

```Shell
k scale --replicas=5 deployment/my-deploy
```

4. Monitor the rollout and check the revision content.

```Shell
k rollout status deployment/my-deploy
k rollout history deployment/my-deploy
k rollout history deployment/my-deploy --revision=1
```

5. Describe the deployment and check the name of the container.

Shell
```shell
k describe deploy my-deploy | grep -A2 "Containers:"
```

6. Pause the deployment rollout.

Shell
```shell
k rollout pause deploy/my-deploy
```

7. Change the image of the deployment to **nginx:1.29**.

Shell
```shell
k set image deployment my-deploy nginx=nginx:1.29
```

8. Confirm that the pods of the deployment are running under the new version.

Shell
```shell

k describe deploy my-deploy | grep -A2 "Containers:"

k describe po <pod-name> | grep -A5 "Containers:"

k rollout resume deploy/my-deploy

k rollout history deployment/my-deploy

k rollout history deployment/my-deploy --revision=2
```

9. Rollback the deployment to the first revision.

Shell
```shell
k rollout undo deployment my-deploy --to-revision=1
```

```
k describe deploy my-deploy | grep -A2 "Containers:"

k describe pod <pod-name> | grep -A5 "Containers:"
```