In Kubernetes, annotations are declared similarly to labels, but they serve a different purpose. They represent key-value pairs for providing descriptive metadata. The most important differentiator is that annotations cannot be used for querying or selecting objects. Typical examples of annotations may include SCM commit hash IDs, release information, or contact details for teams operating the object. Make sure to put the value of an annotation into single- or double-quotes if it contains special characters or spaces.

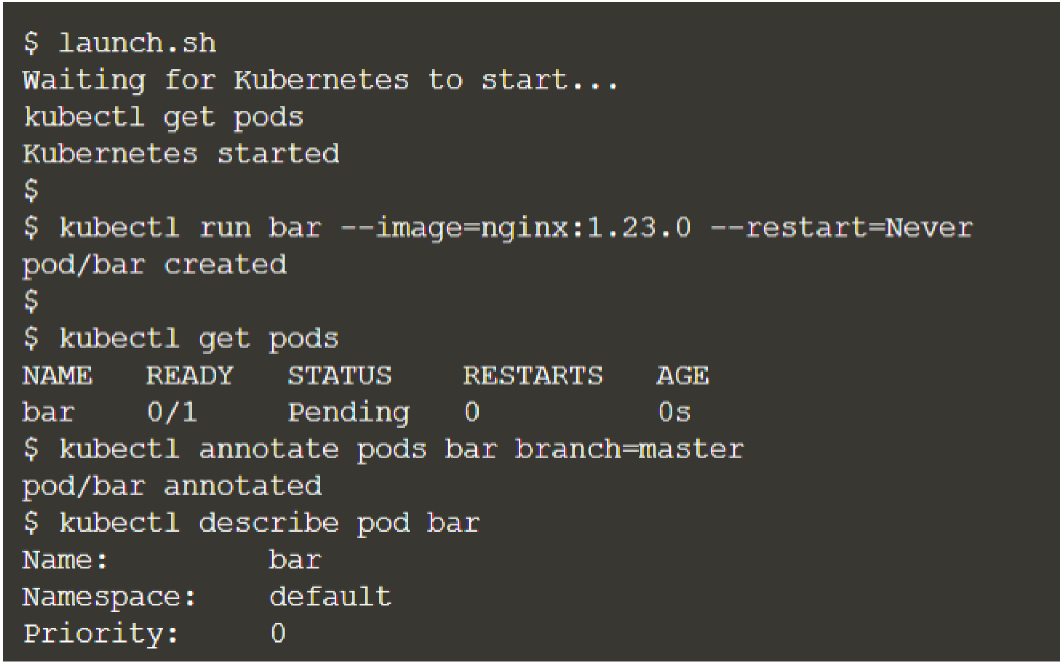
You can use the annotate command to add an annotation to an existing object. For example, to add the branch=master annotation to a Pod named bar, you can run:

kubectl annotate pods bar branch=master

By default, the annotate command will not let you overwrite an existing label. To do this, you need to add the --overwrite flag.

For annotations, there's no corresponding command-line option similar to --show-labels. To inspect assigned annotation for an object, you have to describe the object:

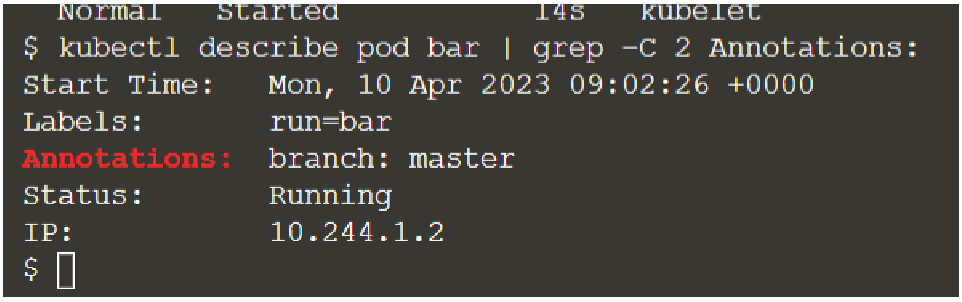
kubectl describe pod bar



```
$ launch.sh
Waiting for Kubernetes to start...
kubectl get pods
Kubernetes started
$
$ kubectl run bar --image=nginx:1.23.0 --restart=Never
pod/bar created
$
$ kubectl get pods
NAME   READY   STATUS    RESTARTS   AGE
bar    0/1     Pending   0          0s
$ kubectl annotate pods bar branch=master
pod/bar annotated
$ kubectl describe pod bar
Name:         bar
Namespace:    default
Priority:     0
```

You can combine the command with the Unix grep command to render a more condensed view of the assigned annotation.

kubectl describe pod bar | grep -C 2 Annotations:



```
  Normal   Started          14s   kubelet
$ kubectl describe pod bar | grep -C 2 Annotations:
Start Time:    Mon, 10 Apr 2023 09:02:26 +0000
Labels:        run=bar
Annotations:   branch: master
Status:        Running
IP:            10.244.1.2
$ 
```

The kubectl describe pod command is used to display detailed information about a specific Kubernetes pod, including its current state, events, and metadata. The grep command is then used to filter the output of the kubectl describe pod command, specifically searching for the line that contains the Annotations field.

The -C 2 option used with grep is used to display 2 lines of context around the match, so you can see the context of the Annotations field within the output of the kubectl describe pod command.

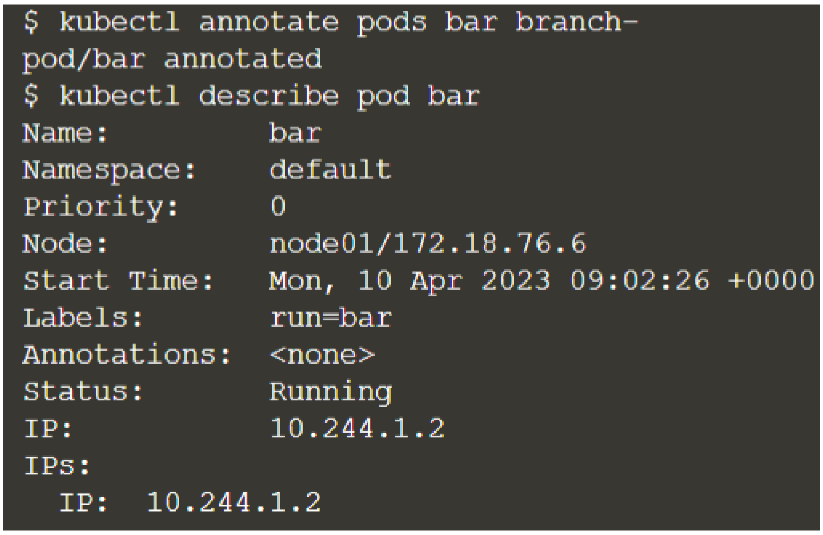
So the command kubectl describe pod bar | grep -C 2 Annotations: will display the Annotations field of the pod named bar along with two lines of context above and below it. This can be useful for understanding the annotations that are associated with the pod and how they may be affecting its behavior.

Removing Annotations
If you want to remove an annotation, you can use the <annotation-name>- syntax:

kubectl annotate pods bar branch-

This will remove the branch annotation from the Pod named bar. The output of the following command shows that the annotation key-value pair branch=master has been removed:

kubectl describe pod bar



```
$ kubectl annotate pods bar branch-
pod/bar annotated
$ kubectl describe pod bar
Name:           bar
Namespace:      default
Priority:       0
Node:           node01/172.18.76.6
Start Time:     Mon, 10 Apr 2023 09:02:26 +0000
Labels:         run=bar
Annotations:    <none>
Status:         Running
IP:             10.244.1.2
IPs:
  IP:  10.244.1.2
```

Annotations assign human-readable metadata to existing object. For example, you may want to assign a contact number for the person on call to a Pod. kubectl is a powerful tool for managing annotations for existing objects. You can add annotations to objects, or remove them using the labels command.