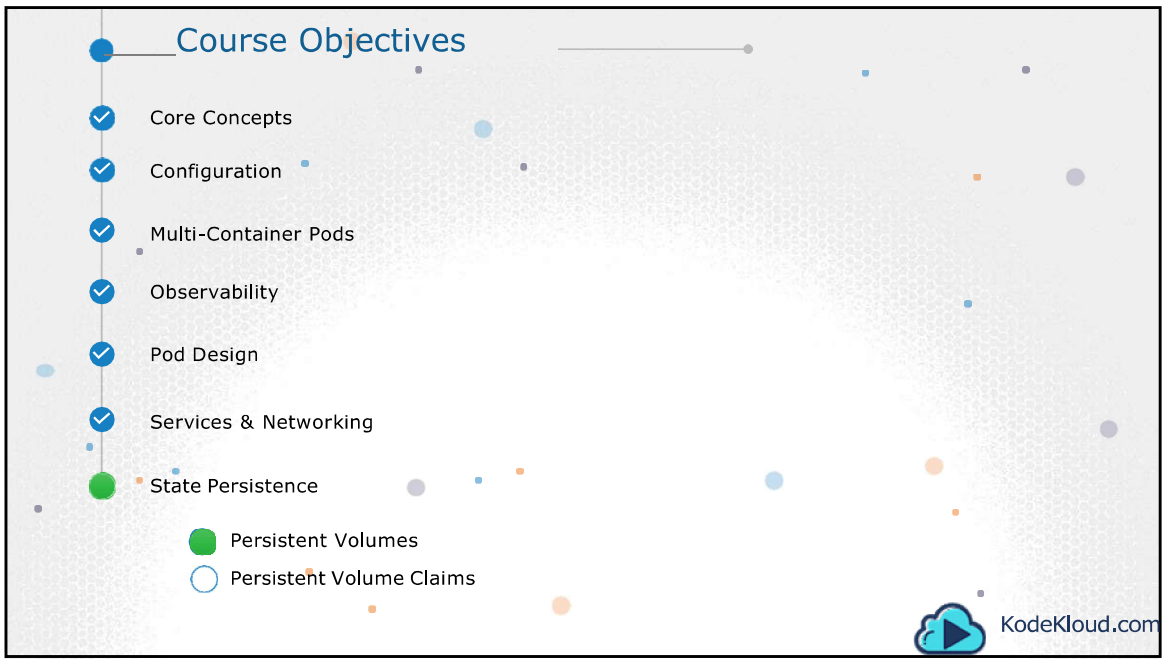
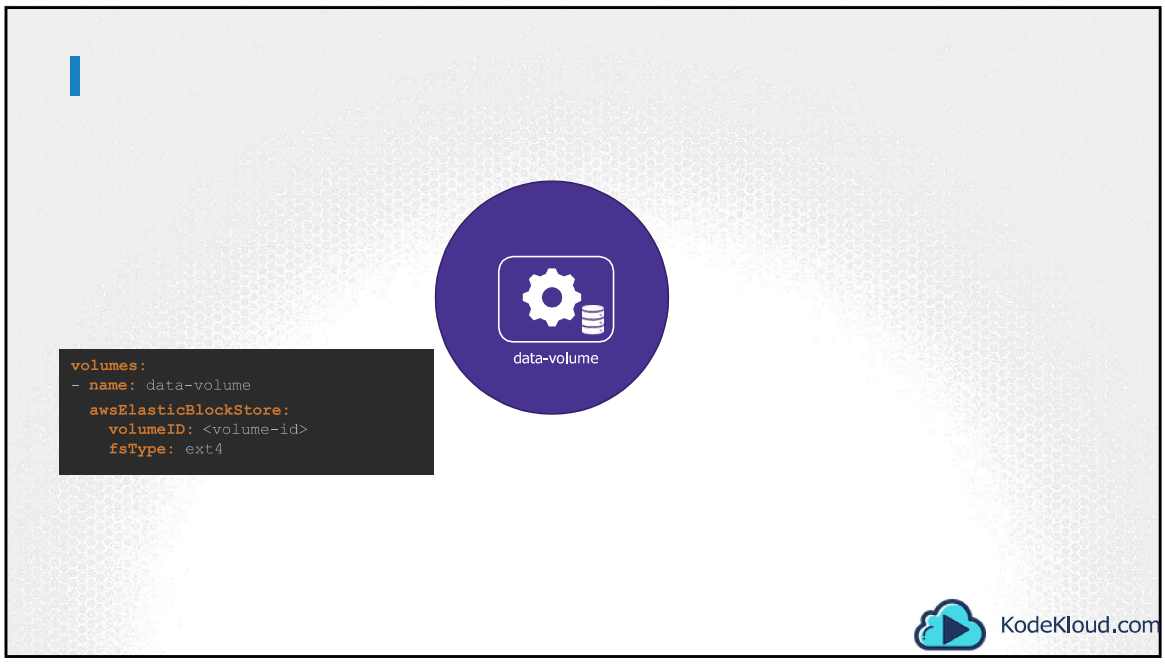
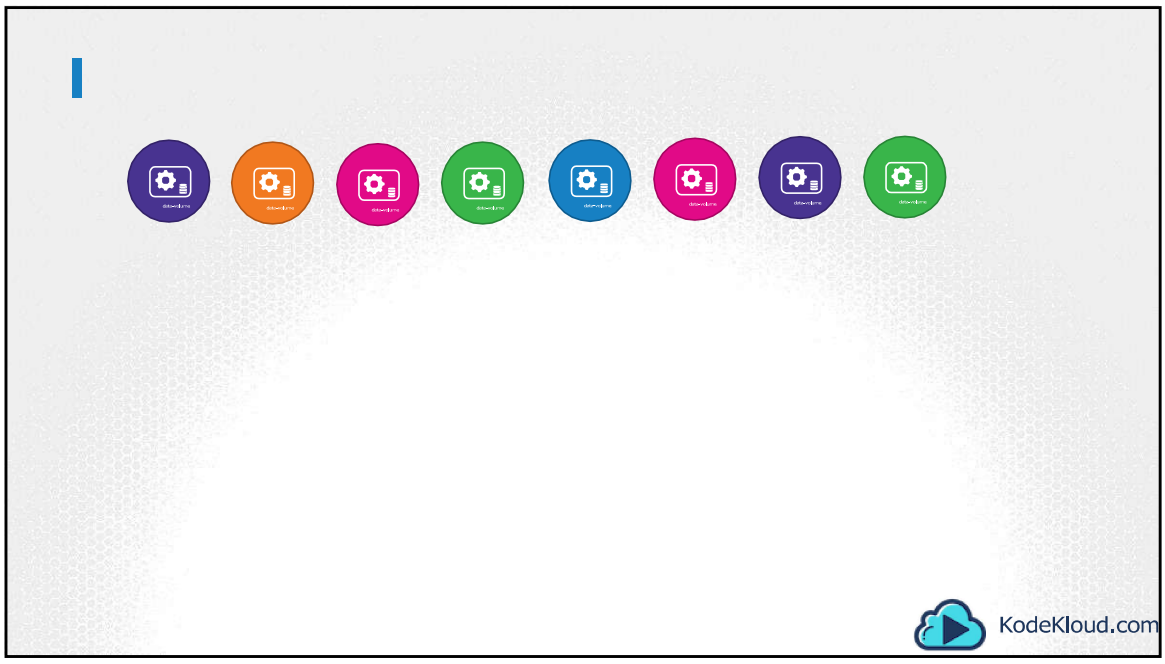


Hello and welcome to this lecture on Persistent Volumes in Kubernetes. My name is Mumshad Mannambeth and we are going through the Certified Kubernetes Application Developer's course.

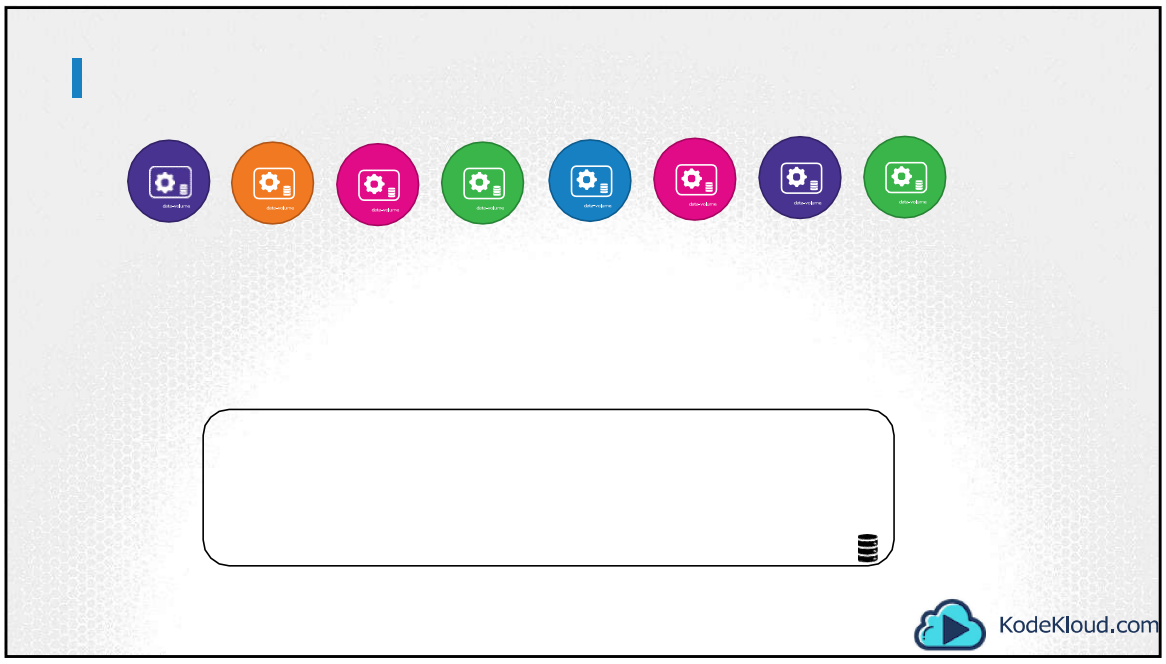
In the last lecture we learned about Volumes. Now we will discuss Persistent Volumes in Kubernetes.



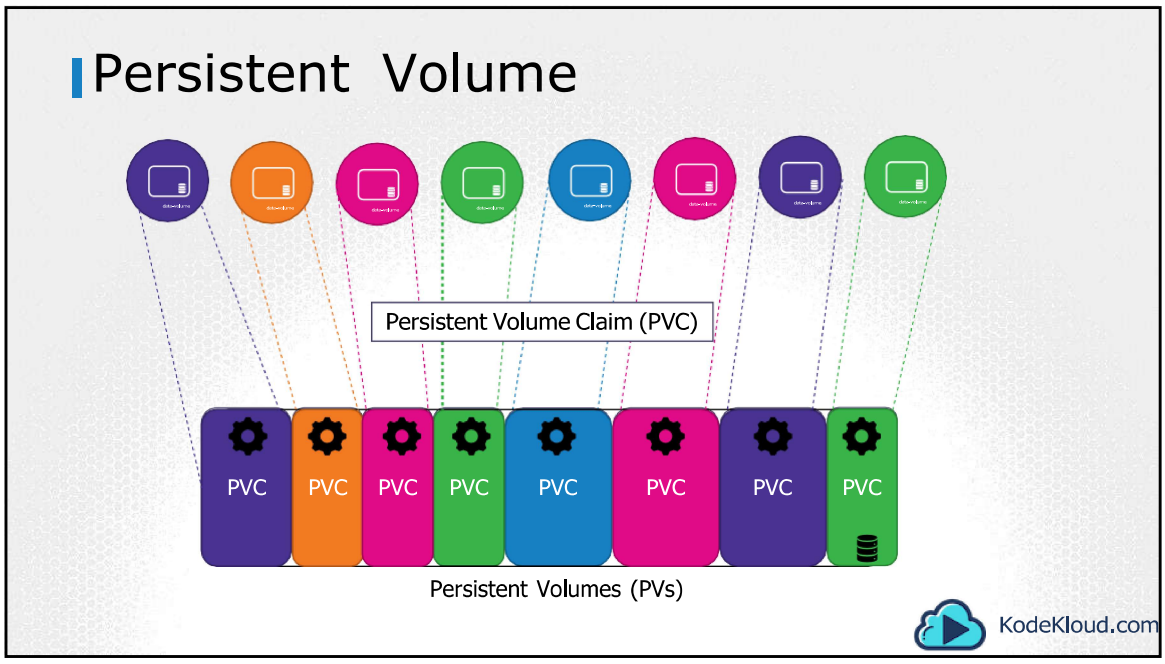
When we created volumes in the previous section we configured volumes within the POD definition file. So every configuration information required to configure storage for the volume goes within the pod definition file.



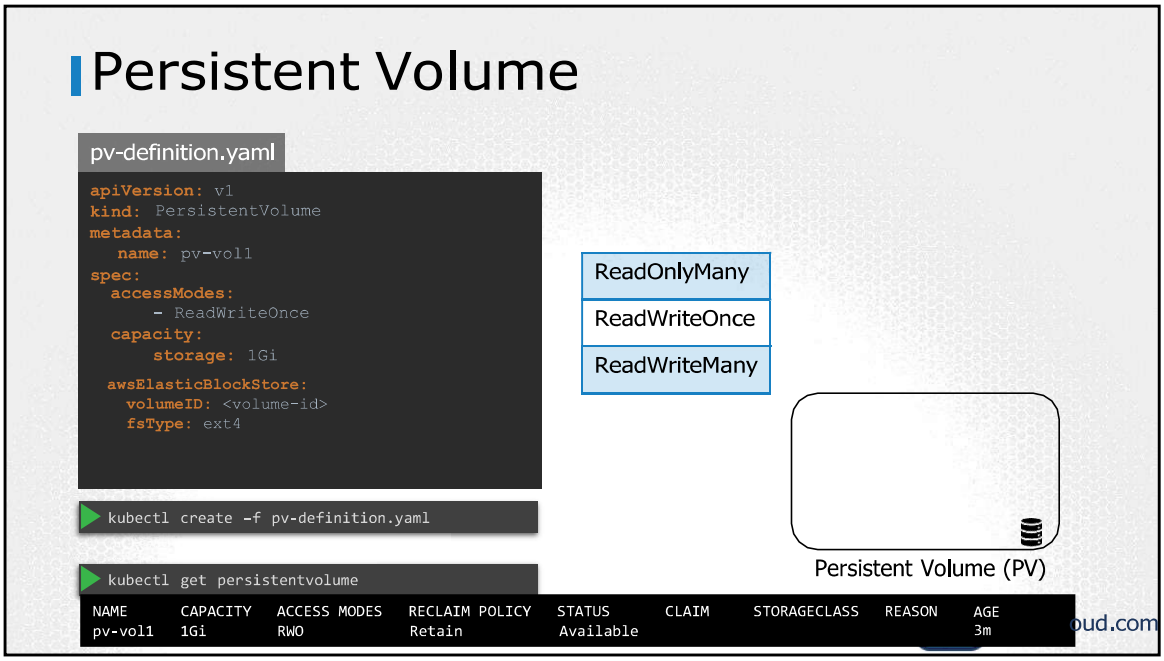
Now, when you have a large environment with a lot of users deploying a lot of PODs, the users would have to configure storage every time for each POD. Whatever storage solution is used, the user who deploys the PODs would have to configure that on all POD definition files in his environment. Every time a change is to be made, the user would have to make them on all of his PODs.



Instead, you would like to manage storage more centrally.



You would like it to be configured in a way that an administrator can create a large pool of storage, and then have users carve out pieces from it as required. That is where Persistent Volumes can help us. A Persistent Volume is a Cluster wide pool of storage volumes configured by an Administrator, to be used by users deploying applications on the cluster. The users can now select storage from this pool using Persistent Volume Claims.



Let us now create a Persistent Volume. We start with the base template and update the apiVersion, set the Kind to PersistentVolume, and name it pv-vol1. Under the spec section specify the accessModes.

Access Mode defines how the Volume should be mounted on the hosts. Weather in a ReadOnly mode, or ReadWrite mode. The supported values are ReadOnlyMany, ReadWriteOnce or ReadWriteMany mode.

Next, is the capacity. Specify the amount of storage to be reserved for this Persistent Volume. Which is set to 1GB here.

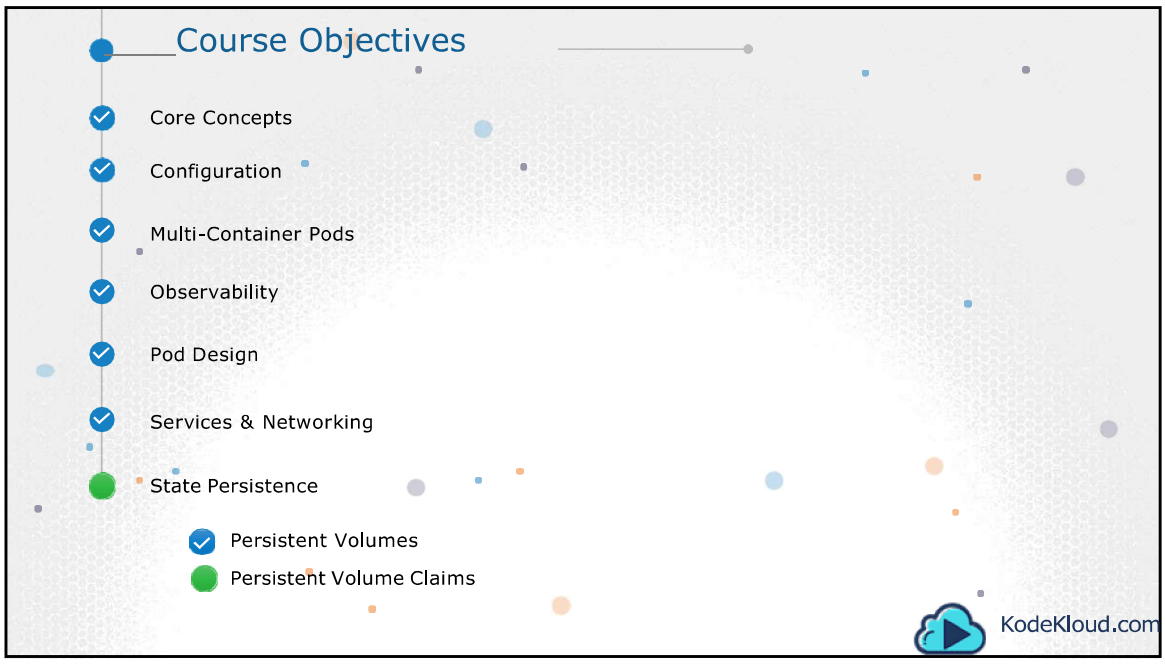
Next comes the volume type. We will start with the hostPath option that uses storage from the node's local directory. Remember this option is not to be used in a production environment.

To create the volume run the kubectl create command and to list the created volume run the kubectl get persistentvolume command.

Replace the hostPath option with one of the supported storage solutions as we saw

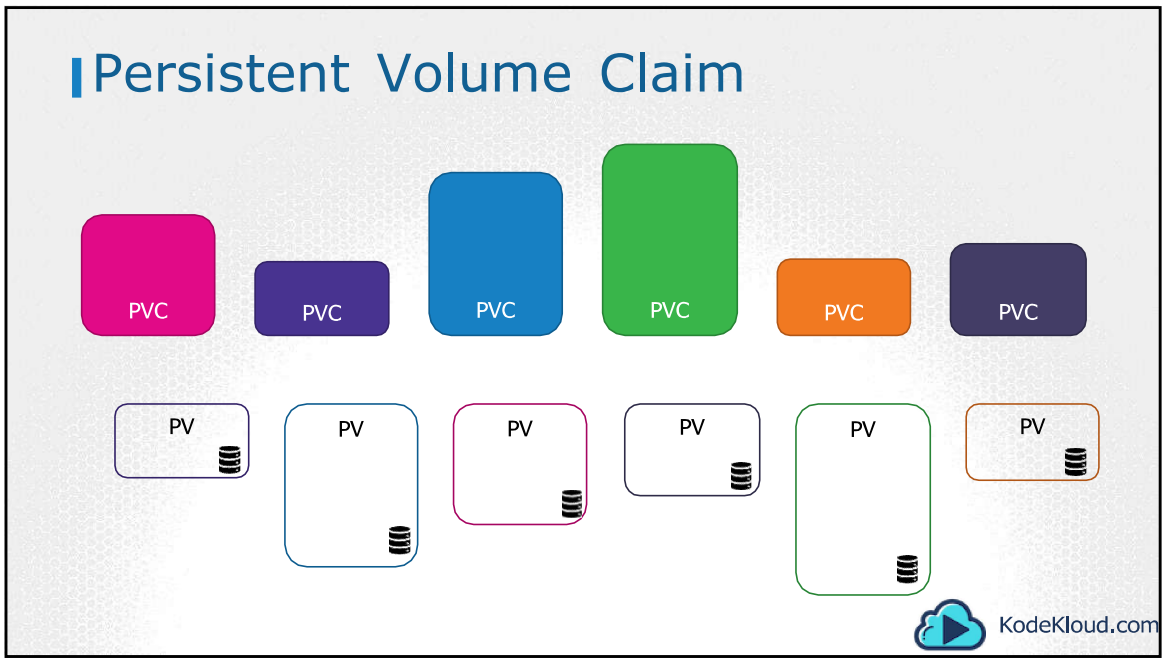
in the previous lecture like AWS Elastic Block Store.

Well that's it on Persistent Volumes in this lecture. Head over to coding challenges and practice configuring Persistent Volumes.



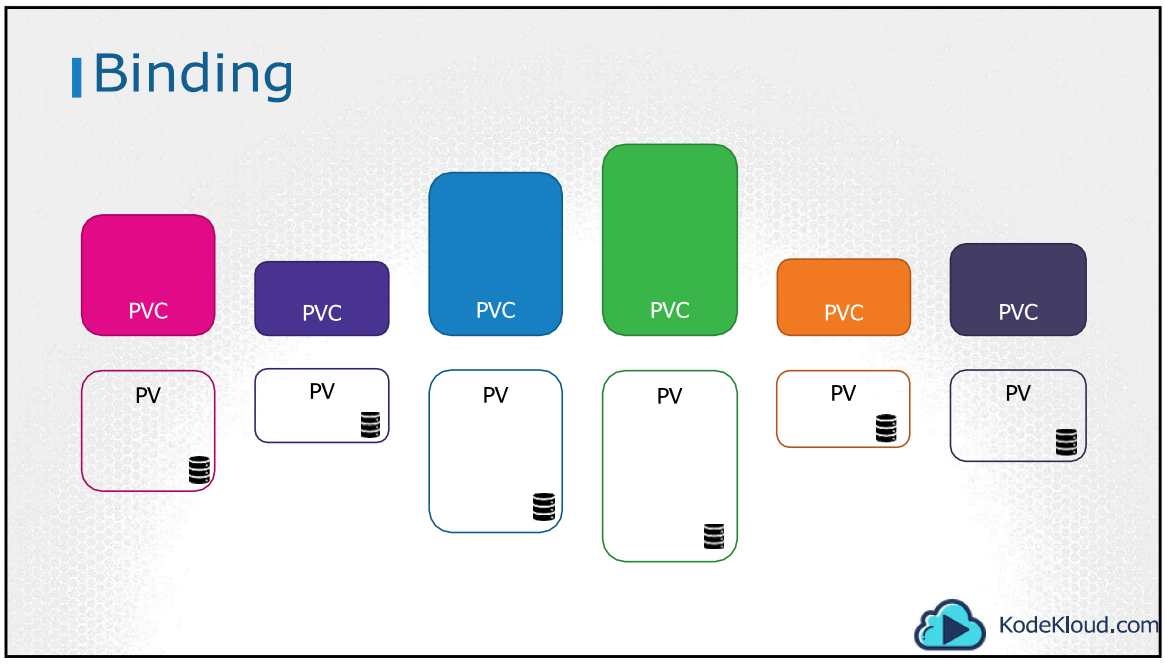
Hello and welcome to this lecture on Persistent Volumes in Kubernetes. My name is Mumshad Mannambeth and we are going through the Certified Kubernetes Application Developer's course.

In the last lecture we learned about Volumes. Now we will discuss Persistent Volumes in Kubernetes.

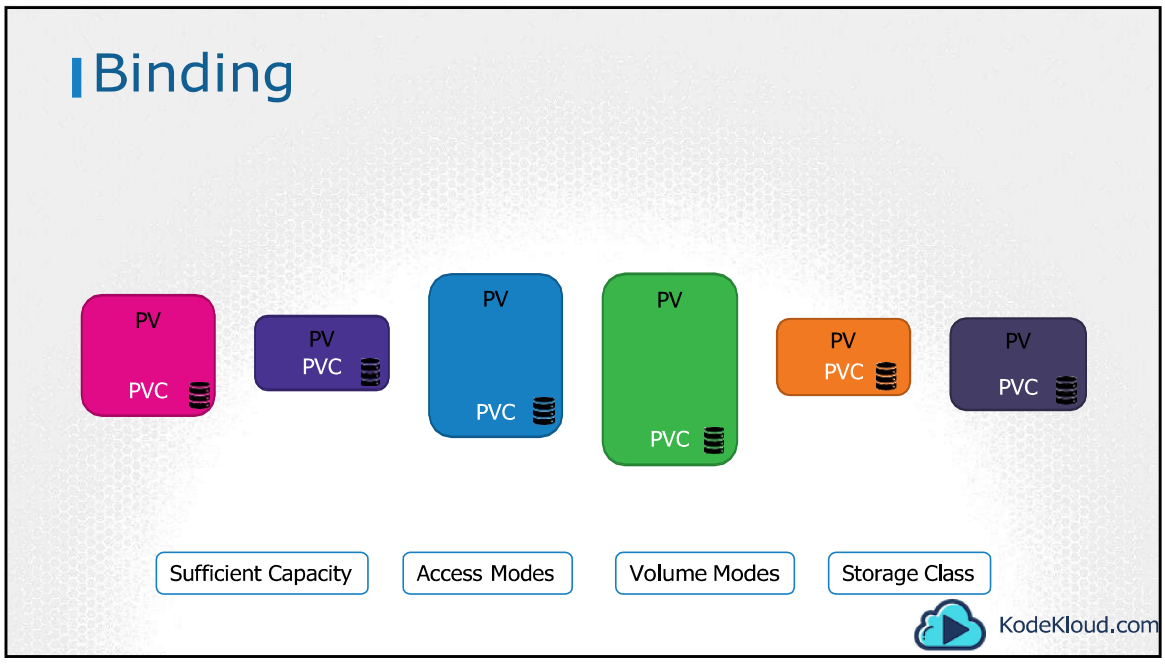


In the previous lecture we created a Persistent Volume. Now we will create a Persistent Volume Claim to make the storage available to a node.

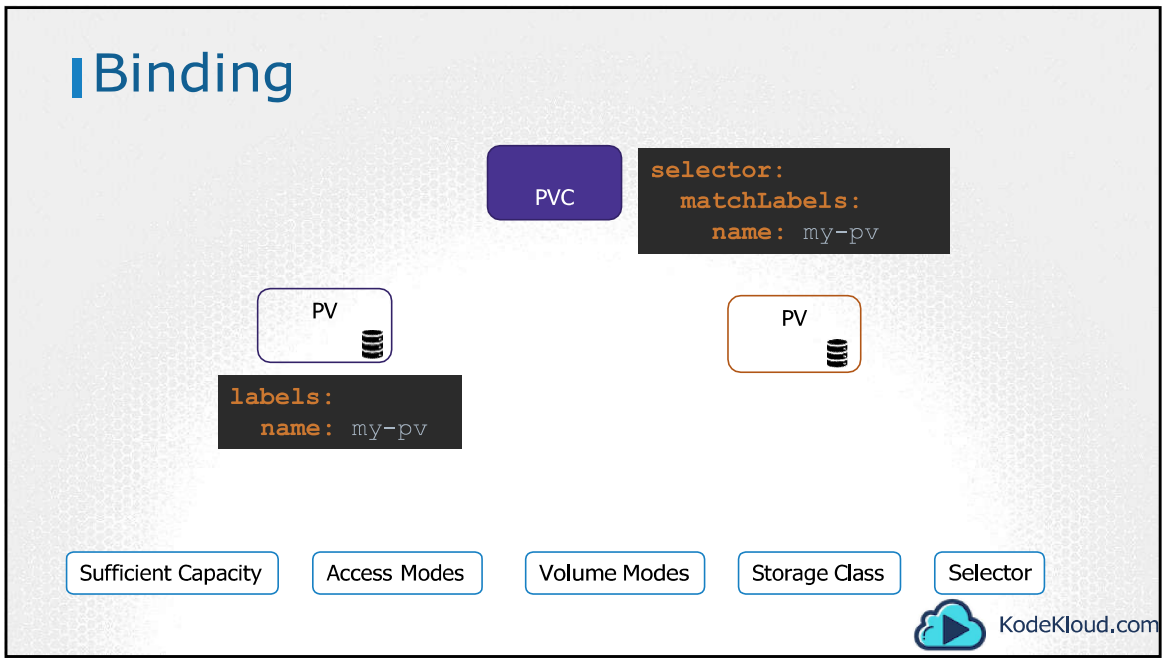
Persistent Volumes and Persistent Volume Claims are two separate objects in the Kubernetes namespace. An Administrator creates a set of Persistent Volumes and a user creates Persistent Volume Claims to use the storage. Once the Persistent Volume Claims are created, Kubernetes binds the Persistent Volumes to Claims based on the request and properties set on the volume.



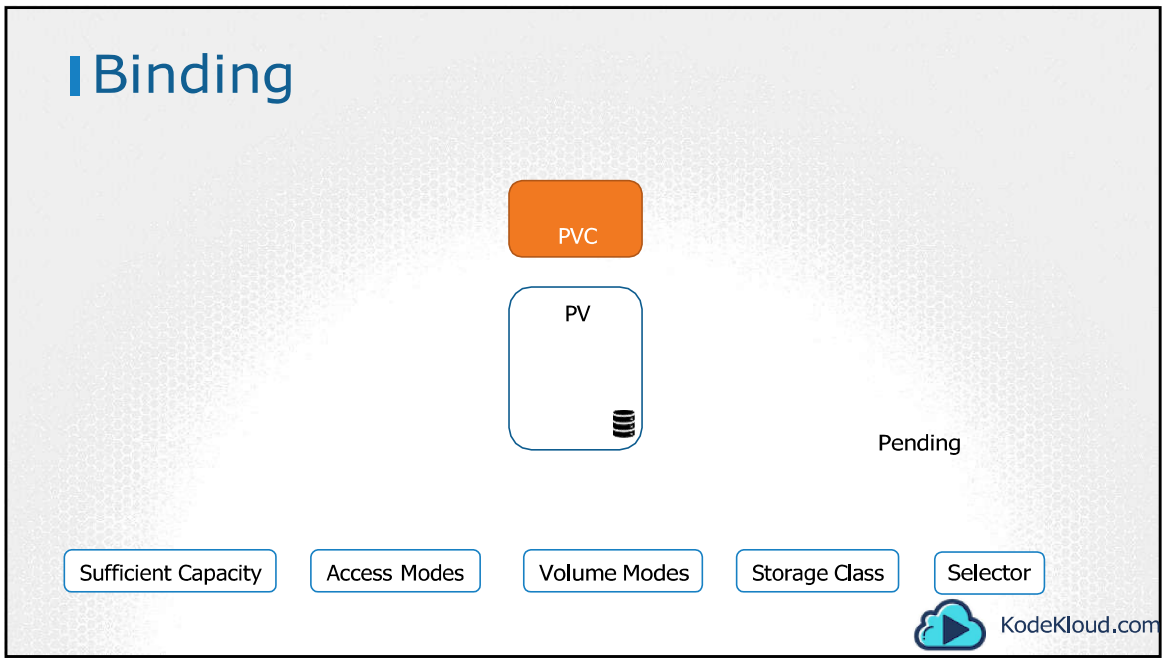
Once the Persistent Volume Claims are created, Kubernetes binds the Persistent Volumes to Claims based on the request and properties set on the volume.



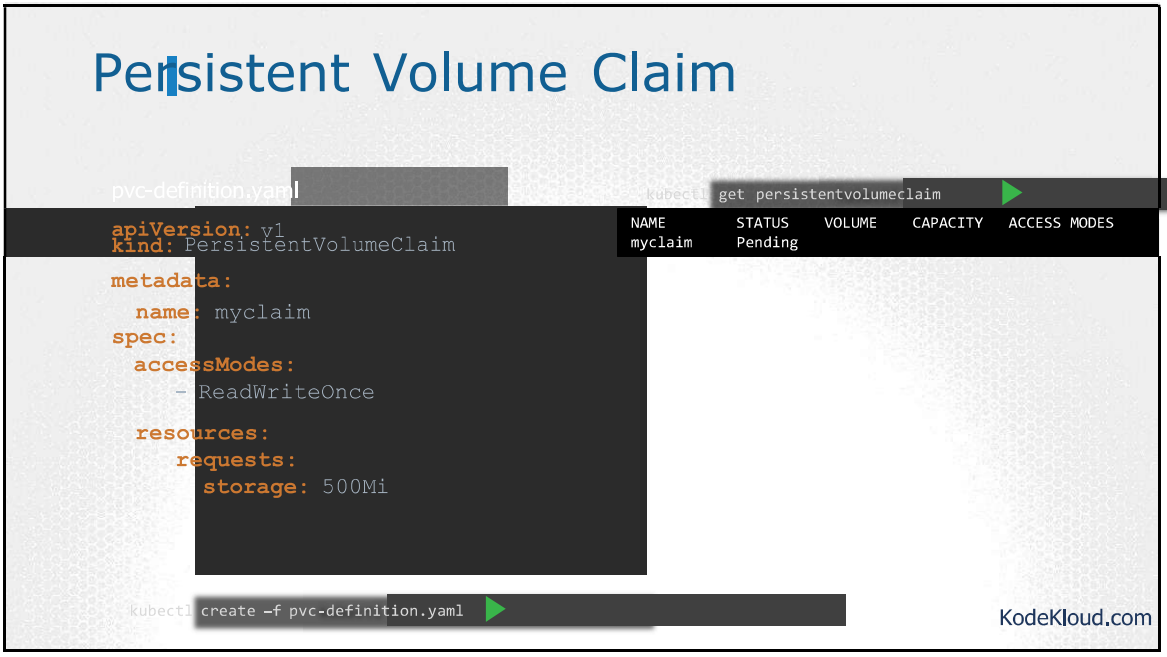
Every Persistent Volume Claim is bound to a single Persistent volume. During the binding process, kubernetes tries to find a Persistent Volume that has sufficient Capacity as requested by the Claim, and any other requested properties such as Access Modes, Volume Modes, Storage Class etc.



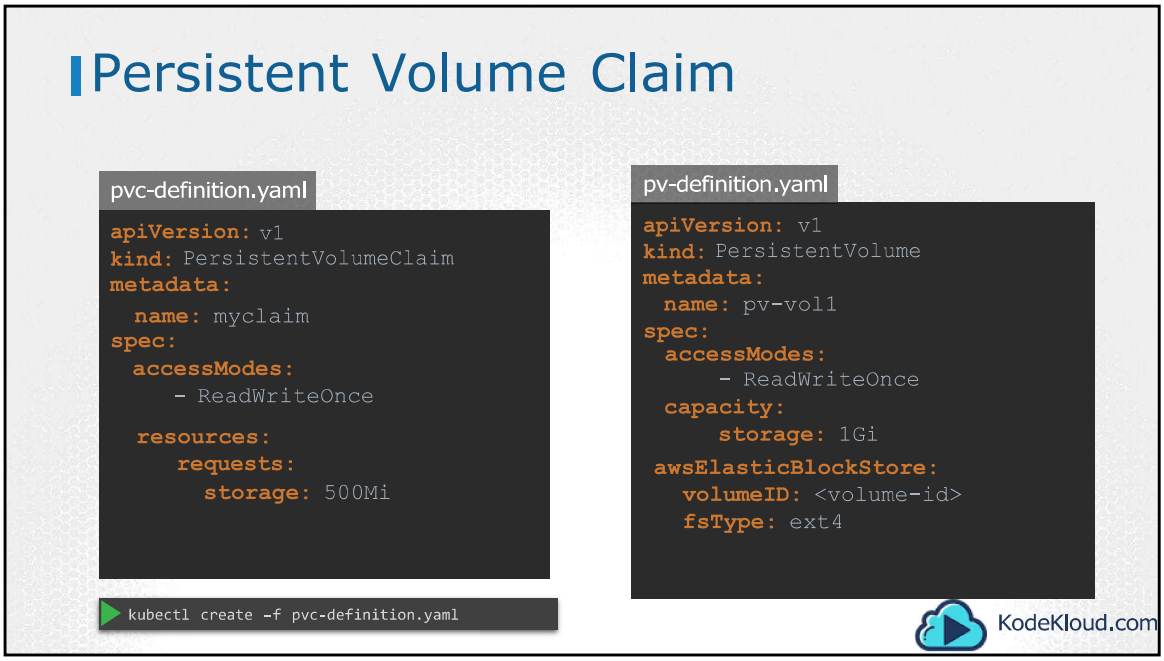
However, if there are multiple possible matches for a single claim, and you would like to specifically use a particular Volume, you could still use labels and selectors to bind to the right volumes.



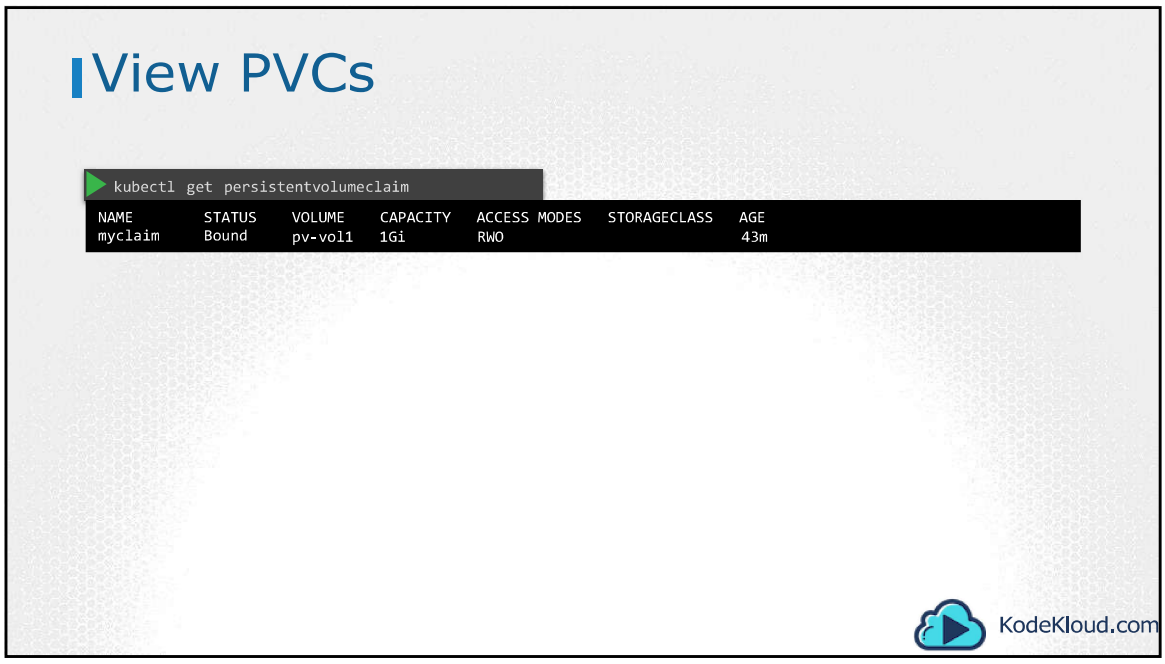
Finally, note that a smaller Claim may get bound to a larger volume if all the other criteria matches and there are no better options. There is a one-to-one relationship between Claims and Volumes, so no other claim can utilize the remaining capacity in the volume. If there are no volumes available the Persistent Volume Claim will remain in a pending state, until newer volumes are made available to the cluster. Once newer volumes are available the claim would automatically be bound to the newly available volume.



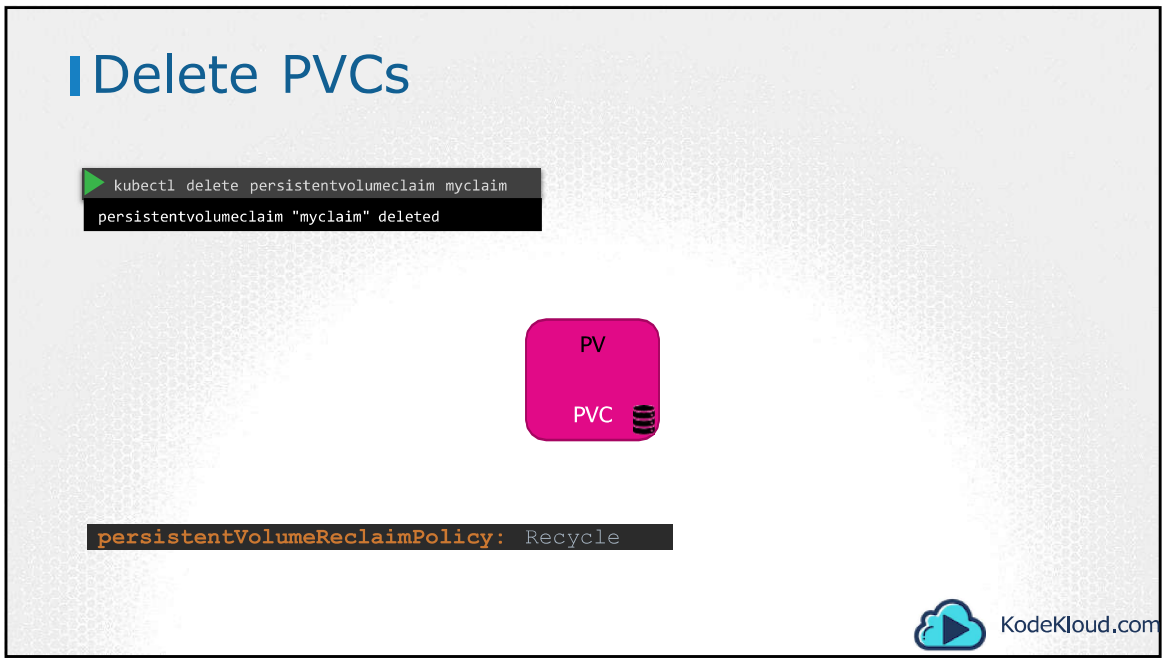
Let us now create a Persistent Volume Claim. We start with a blank template. Set the apiVersion to v1 and kind to PersistentVolumeClaim. We will name it myclaim. Under specification set the accessModes to ReadWriteOnce. And set resources to request a storage of 500 mega bytes. Create the claim using kubectl create command. To view the created claim run the kubectl get persistentvolumeclaim command. We see the claim in a pending state.



When the claim is created, kubernetes looks at the volume created previously. The access Modes match. The capacity requested is 500 Megabytes but the volume is configured with 1 GB of storage. Since there are no other volumes available, the PVC is bound to the PV.

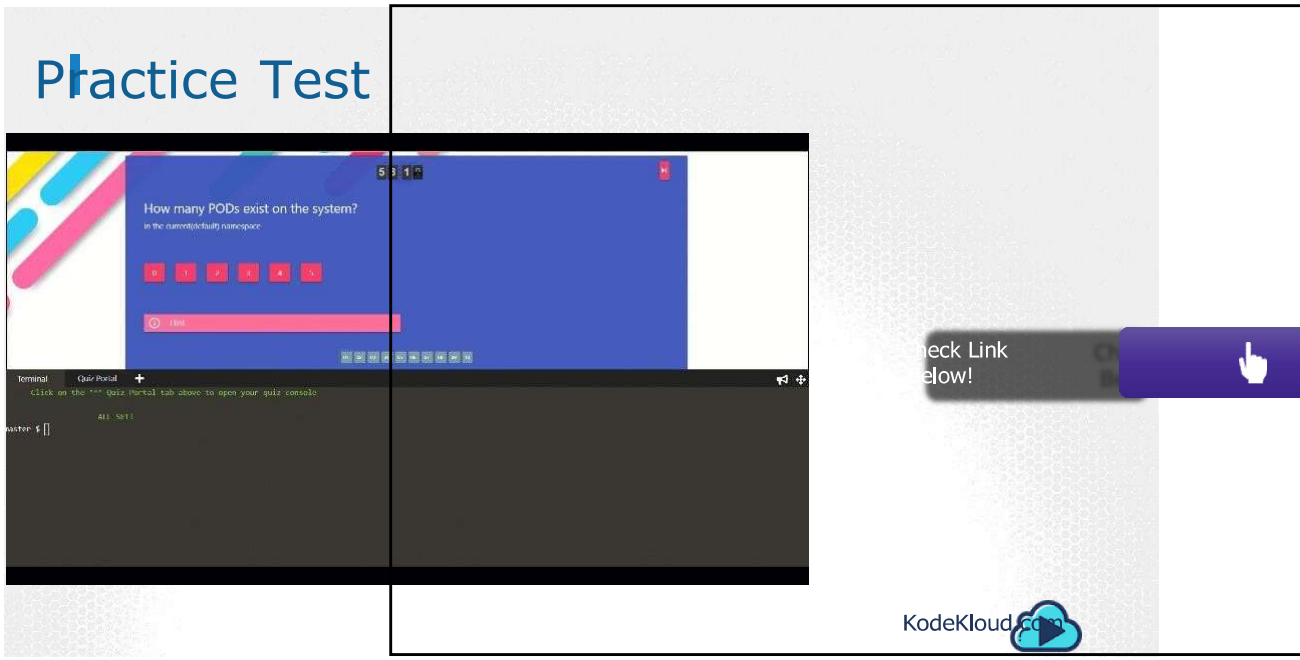


When we run the get volumes command again, we see the claim is bound to the persistent volume we created. Perfect!

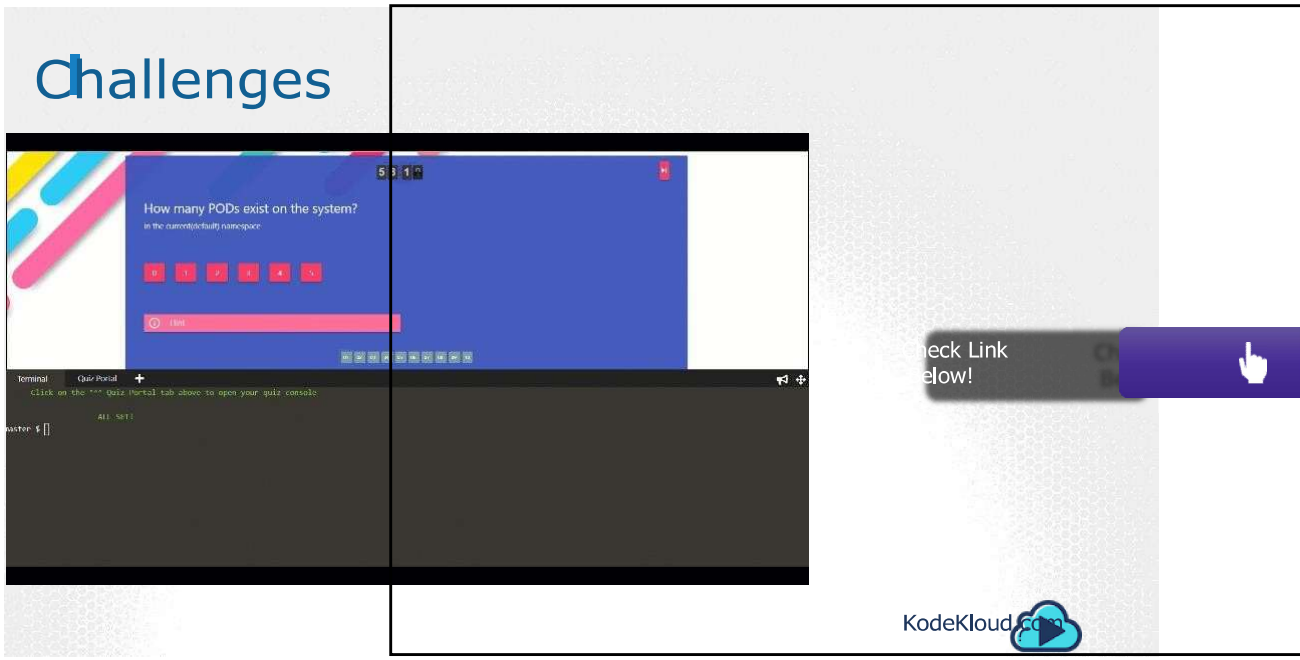


To delete a PVC run the kubectl delete persistentvolumeclaim command. But happens to the Underlying Persistent Volume when the claim is deleted? You can chose what is to happen to the volume. By default, it is set to Retain. Meaning the Persistent Volume will remain until it is manually deleted by the administrator. It is not available for re-use by any other claims. Or it can be Deleted automatically. This way as soon as the claim is deleted, the volume will be deleted as well. Or a third option is to recycle. In this case the data in the volume will be scrubbed before making it available to other claims.

Well that's it for this lecture. Head over to the coding exercises section and practice configuring and troubleshooting persistent volumes and volume claims in Kubernetes.



Access Test Here: https://kodekloud.com/courses/kubernetes-certification-course/lectures/6743707



Access the test here https://kodekloud.com/courses/kubernetes-certification-course/lectures/8414630