**ConfigMap:**

Generally we use ConfigMap when we want to pass parameters to an Image. Some applications/images need parameters to run them on different environments like testing, staging, UAT, production etc, during that time it is not a good idea to add this parameters directly into an Image file because in that case it will be difficult to manage it when we want to push those images/PODs on to different environments.

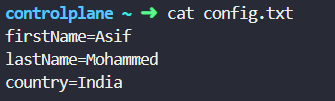
Every environment have different parameters, settings, API’s. Hence, it’s a good idea to maintain these settings/parameters on a different file or configuration so that we can attach these configurations to the image/PODs depending on the environment it is being used.

We can create a ConfigMap using Imperative method or Declarative method.

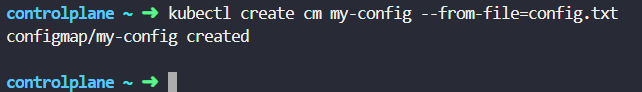
**There are two ways by which we can create a configMap**

**Imperative method One:**

First we are adding all the settings/parameters into a file called config.txt as shown below



Now, we used the above file to create a configMap as my-config as shown below.

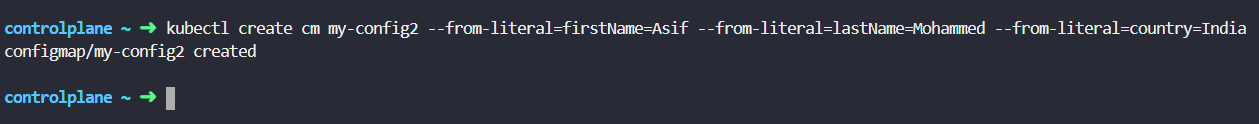


You can see the configuration below:

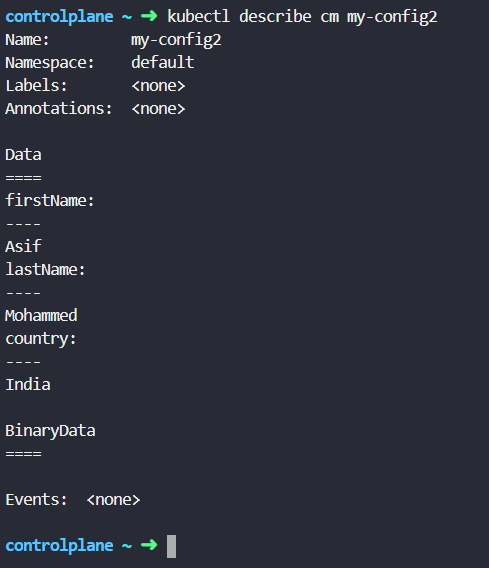


**Imperative method two:**

You can use –from-literal to create parameter and their values as shown below



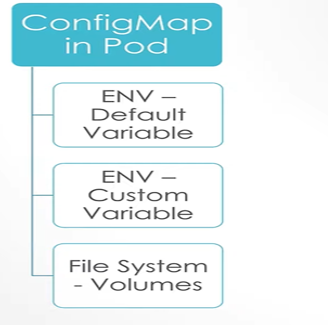
You can see how the configurations are created



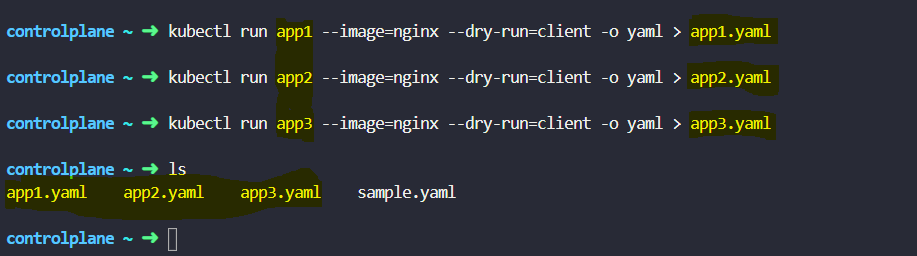
=================================================================================

Now, we have to attach these ConfigMaps to PODs. Let’s see how we can do this.

We can attach ConfigMaps to PODs in 3 ways.



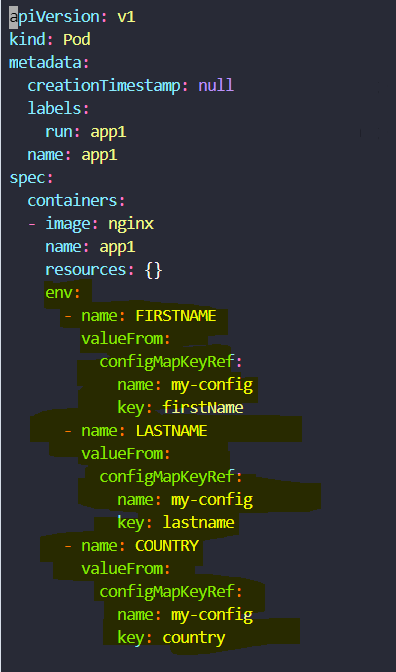
Let’s first create a YAML file to create a 3 PODs to show the example



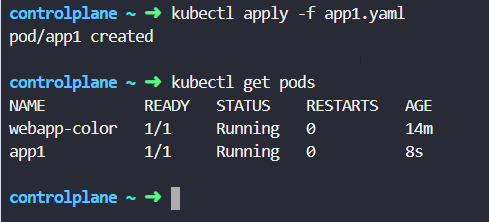
Now, we created 3 PODs, let’s begin with the First method:

**1st Method -**

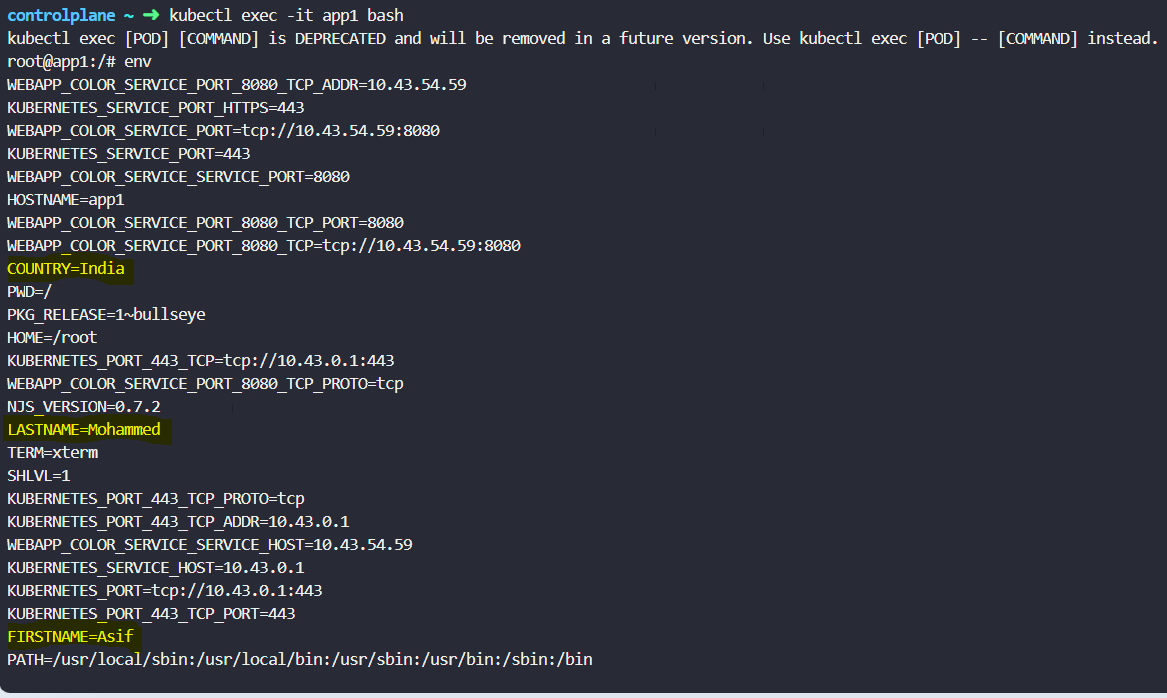
Here, we are editing the *app1.yaml* file for attaching the **ConfigMap** details which we created ( *firstName*, *lastname* and *country* which are part of *my-config* **ConfigMap**)



After attaching the **ConfigMap** details to the **POD** app1, we can create the POD **app1** which will be attached with **ConfigMap** details of *my-config* which we created earlier.



The POD app1 is in running state as we can see above. Now, let’s get into the POD app1 to see if the ConfigMap details are attached to the POD or not.

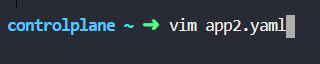


After seeing the above screen, it is confirmed that the configuration of my-config is attached to the POD app1 (Highlighted in Yellow).

=====================================================================================

**2nd Method – How to pass custom variables of ConfigMap to a POD**

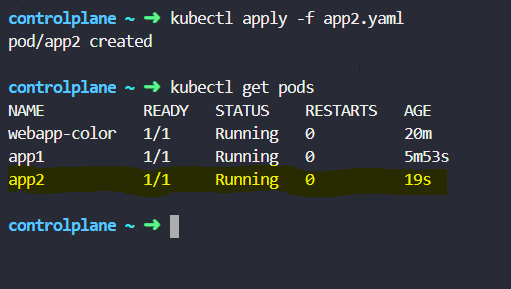
Now, let’s update the app2.yaml file to attach the ConfigMap changes.



Here, we are just adding the ConfigMap (my-config) to the POD (app2) as highlighted below. So that it can take all the configurations which are part of my-config to the POD app2.



After updating the changes, we can see that the POD app2 is in Running state.

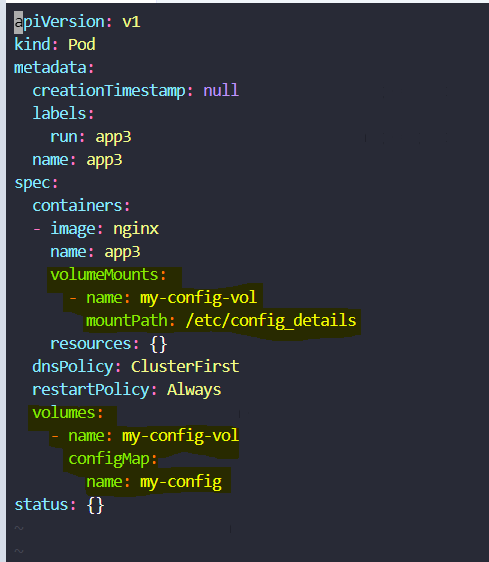


Lets get into POD app2 in a interactive mode and check our configuration changes are updated to POD or not.

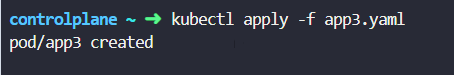


After seeing the above screen, we can confirm that the ConfigMap (my-config) changes are applied to the POD (app2) which are highlighted.

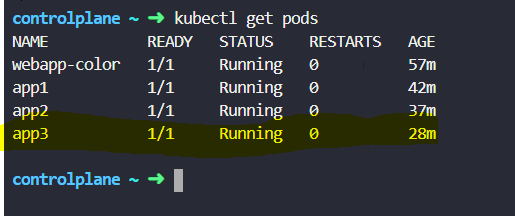
3rd Method – In this method we will mount the ConfigMap my-config configurations to the POD using the volumeMount. Here, we are adding the configurations of my-config to the POD at the path “/etc/config-details” as highlighted below.



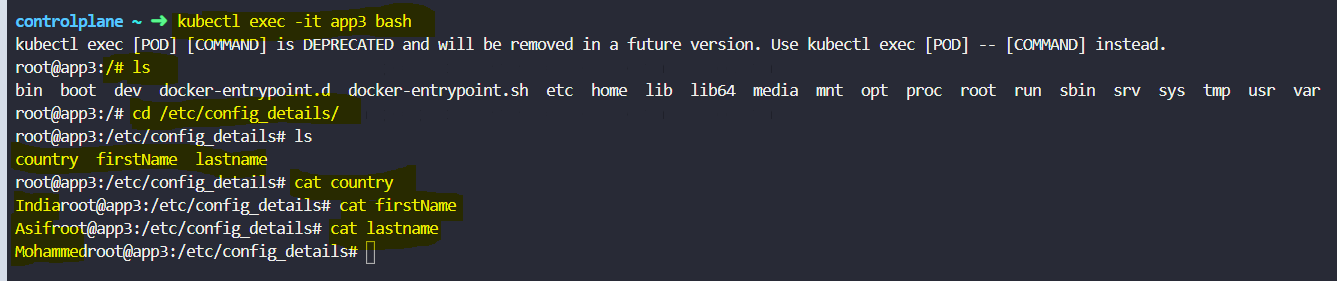
Lets apply the changes which we made to the POD app3



After applying the changes we can see that the POD app3 is in Running state



Now, let’s confirm if the changes of my-config is attached with the POD app3 or not.



After seeing the above screen, it is confirmed that the POD app3 is updated with the ConfigMap configuration of my-config.

**NOTE: Here, in this document I demonstrated**

1. **How to create ConfigMap.**
2. **How to create PODs.**
3. **How to attach ConfigMap to a POD (Shown 3 different methods for this)**