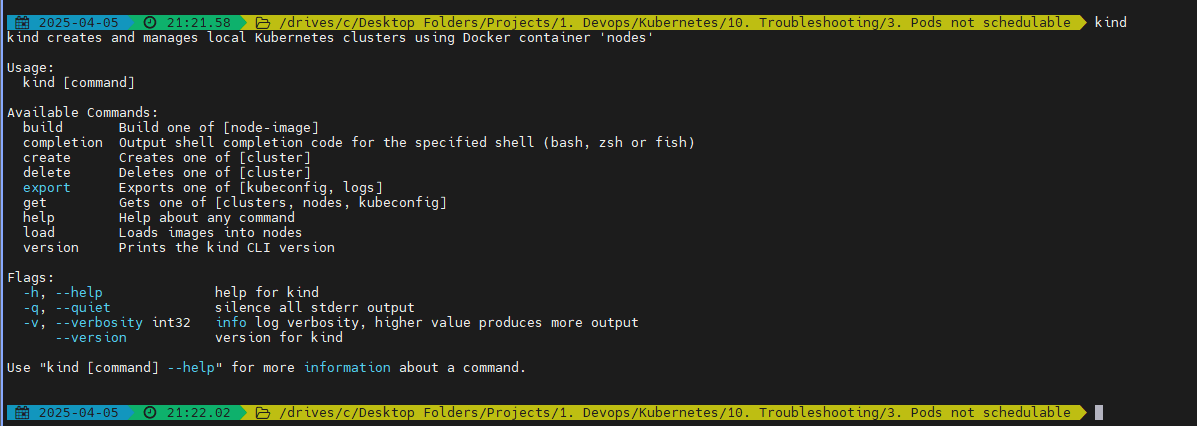
To test multimode concepts use kind(Kubernetes in Docker):

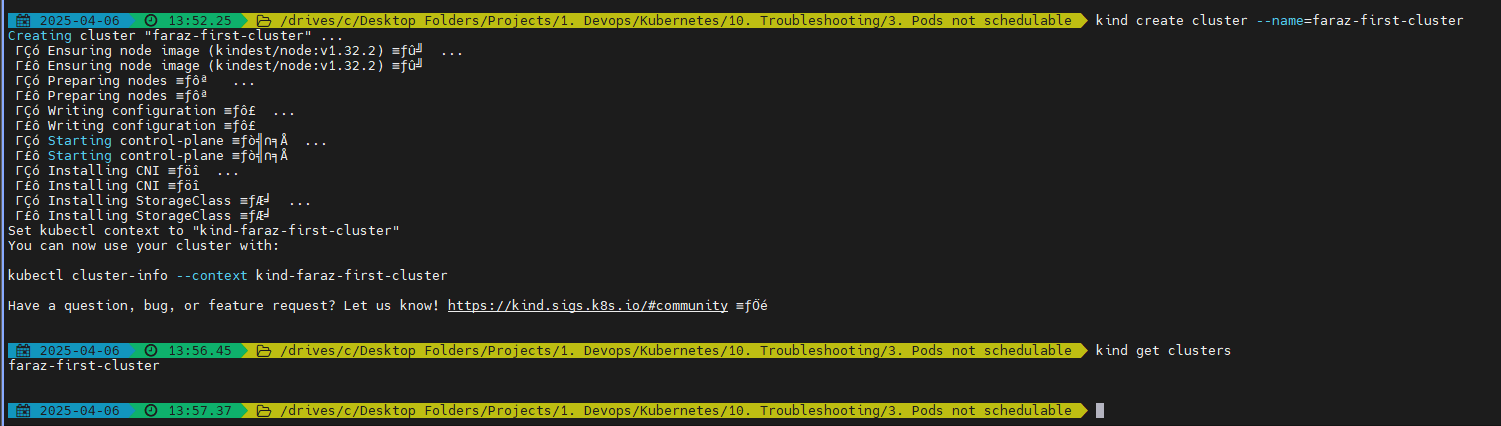
* choco install kind

After installing KinD test if Kind is properly installed or not:

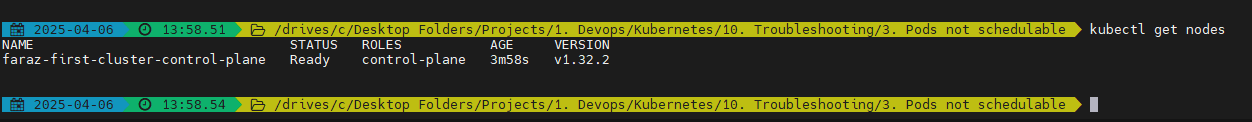


Create your cluster:

* kind create cluster --name=faraz-first-cluster
* kind get clusters



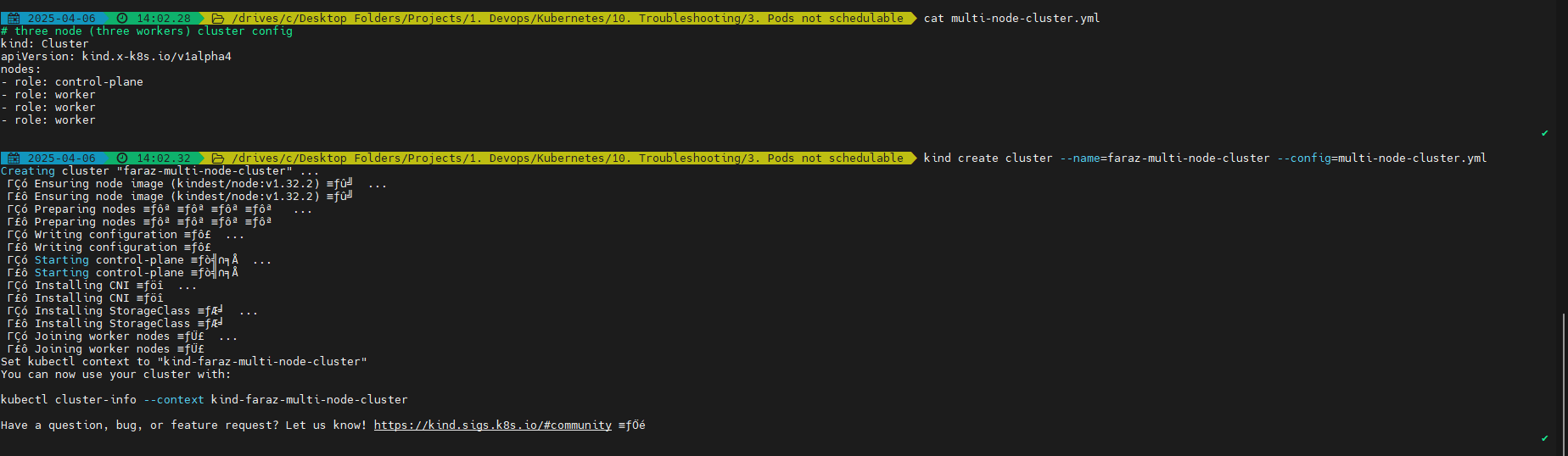
* kubectl get nodes

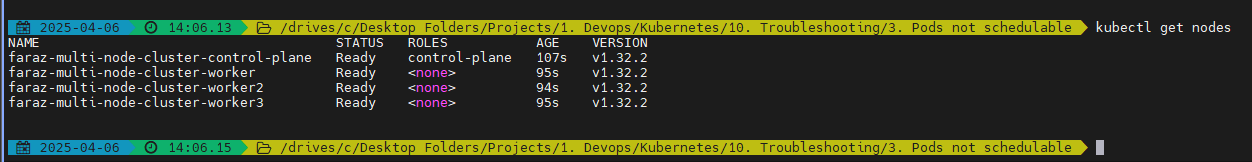


By default it will create a single node cluster

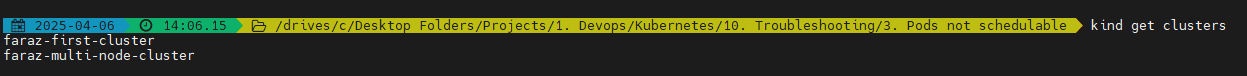
To create multi-node:

* kind create cluster --name=faraz-multi-node-cluster --config=multi-node-cluster.yml

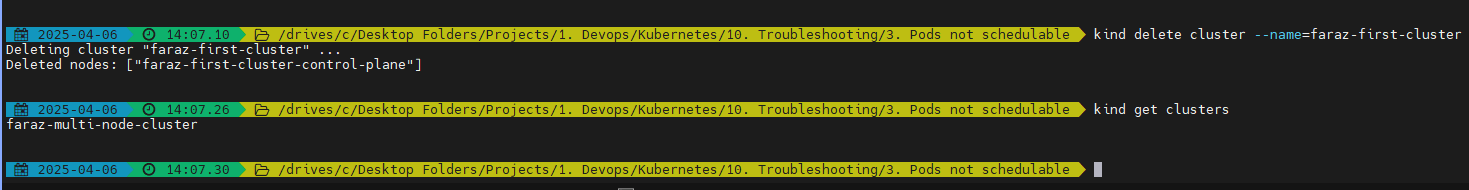




You can delete clusters:



* kind delete cluster --name=faraz-first-cluster

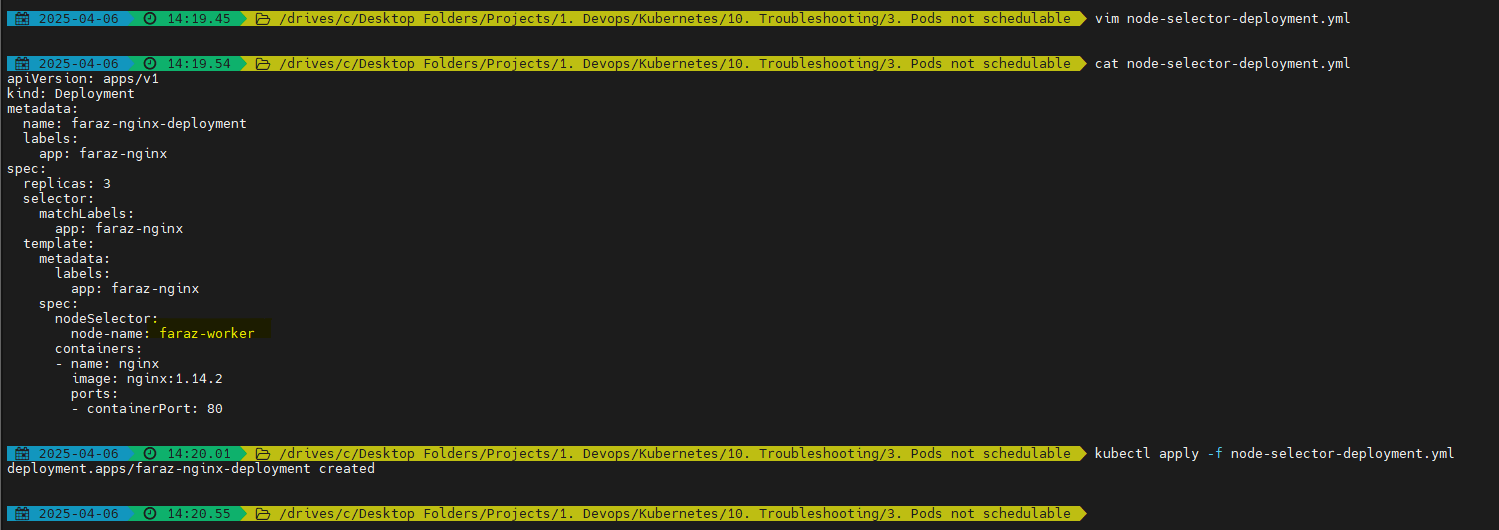


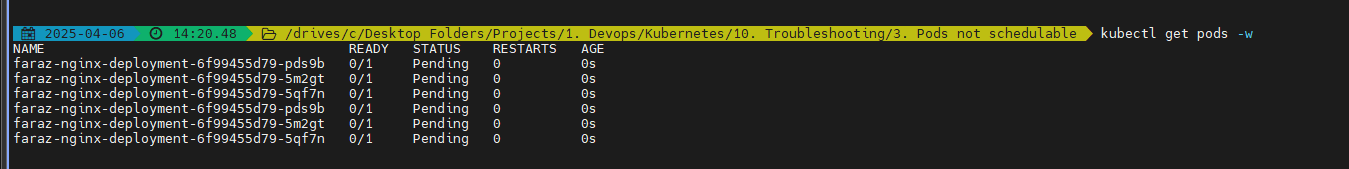
You can always switch context using:

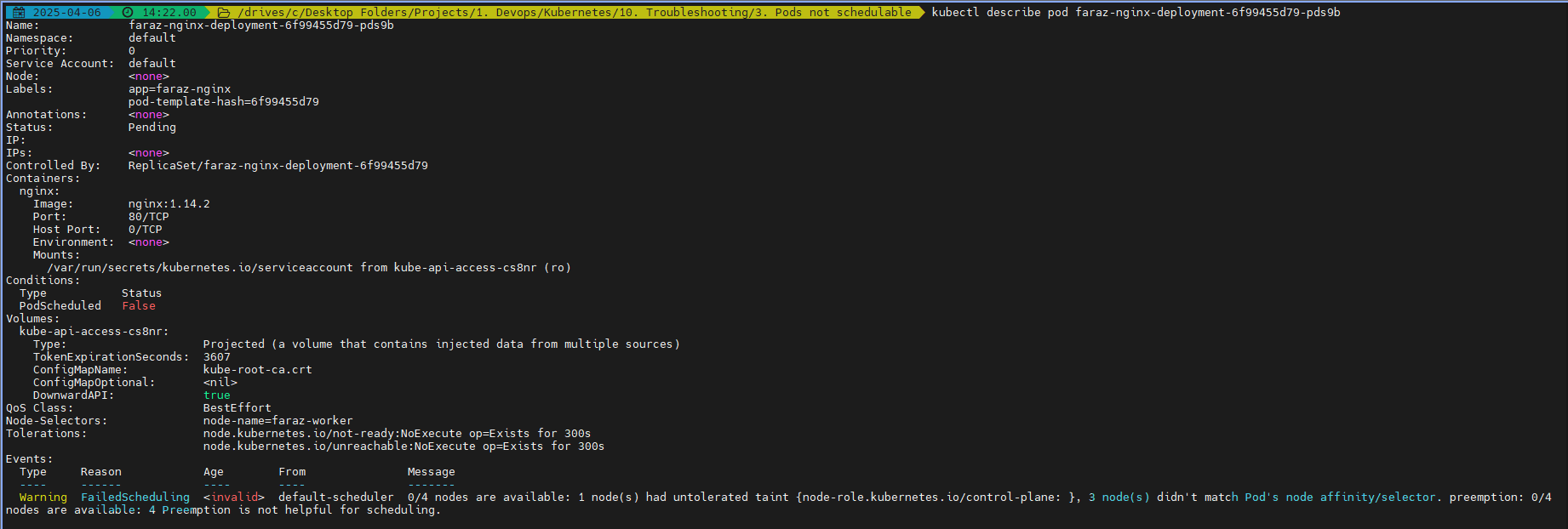
* kubectl config view | grep kind
* kubectl config get-contexts
* kubectl config current-context
* kubectl config use-context <context-name>

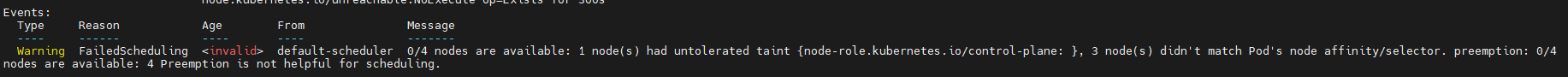
**Node Selector:**

Now create a deployment on your cluster:



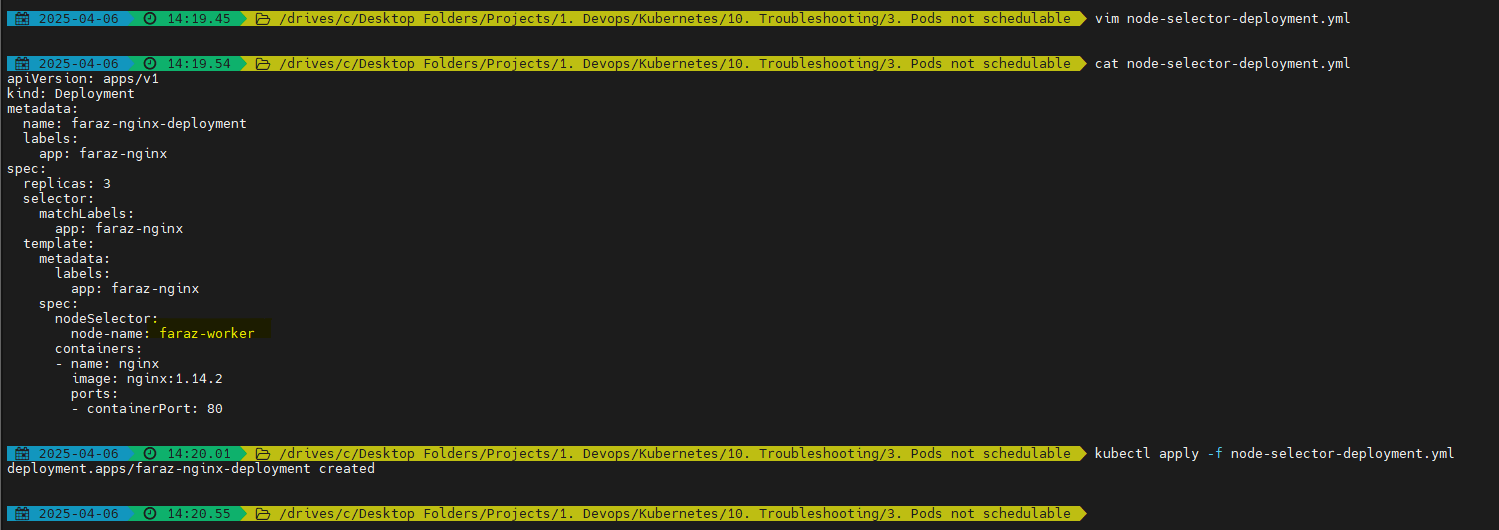


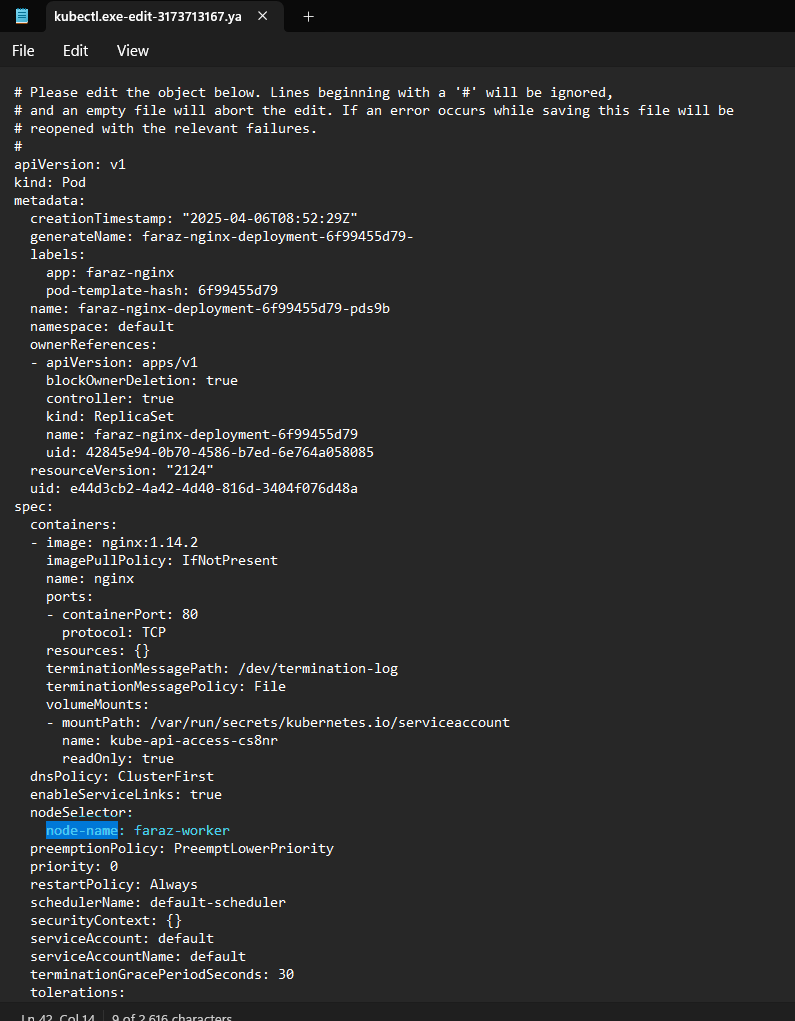




We are getting this error because we don’t have any node with that label:

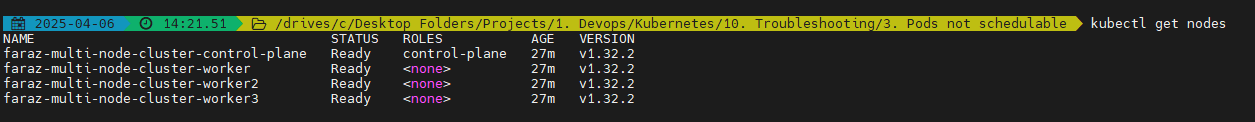
* KUBE\_EDITOR=notepad kubectl edit pod faraz-nginx-deployment-6f99455d79-pds9b



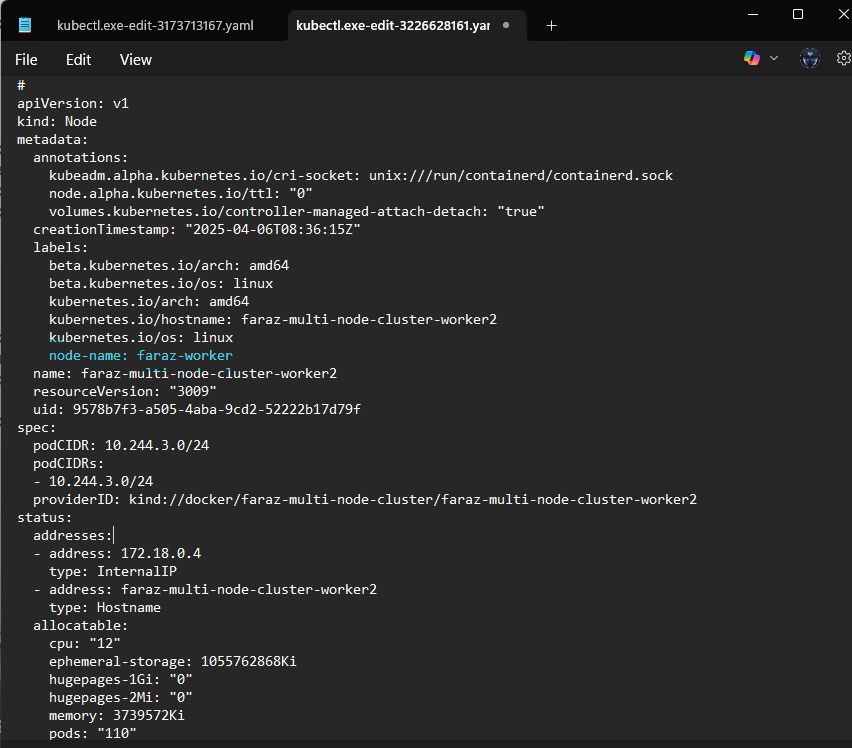


This is used when we want to run a pod only on ARM/AMD processor. Hence using nodeSelector we can use that

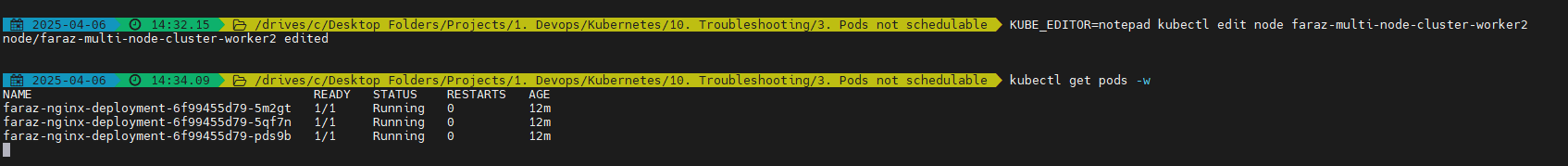
You can solve this issue by:

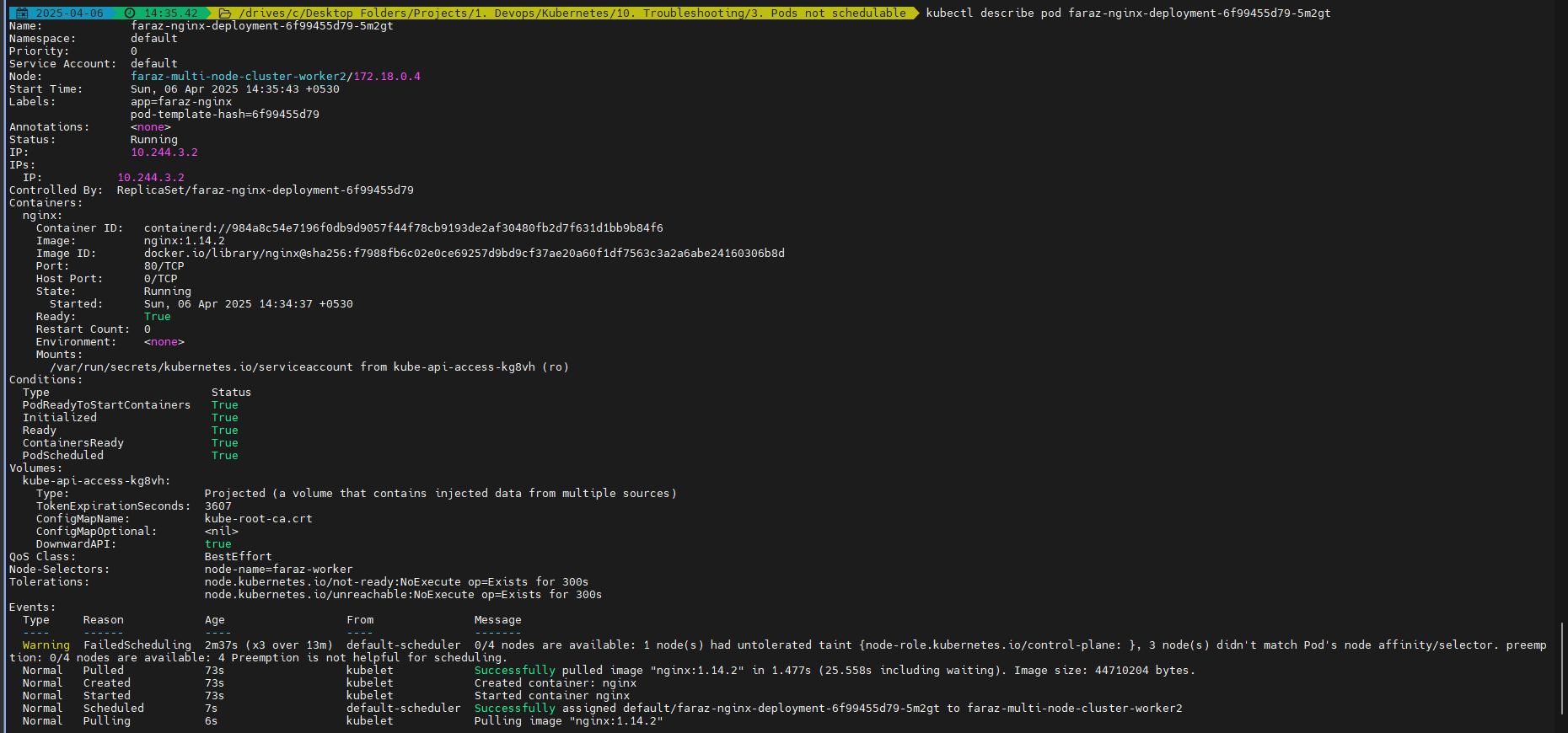


* KUBE\_EDITOR=notepad kubectl edit node faraz-multi-node-cluster-worker2



Now you will be able to see that the pods are scheduled on faraz-multi-node-cluster-worker2:





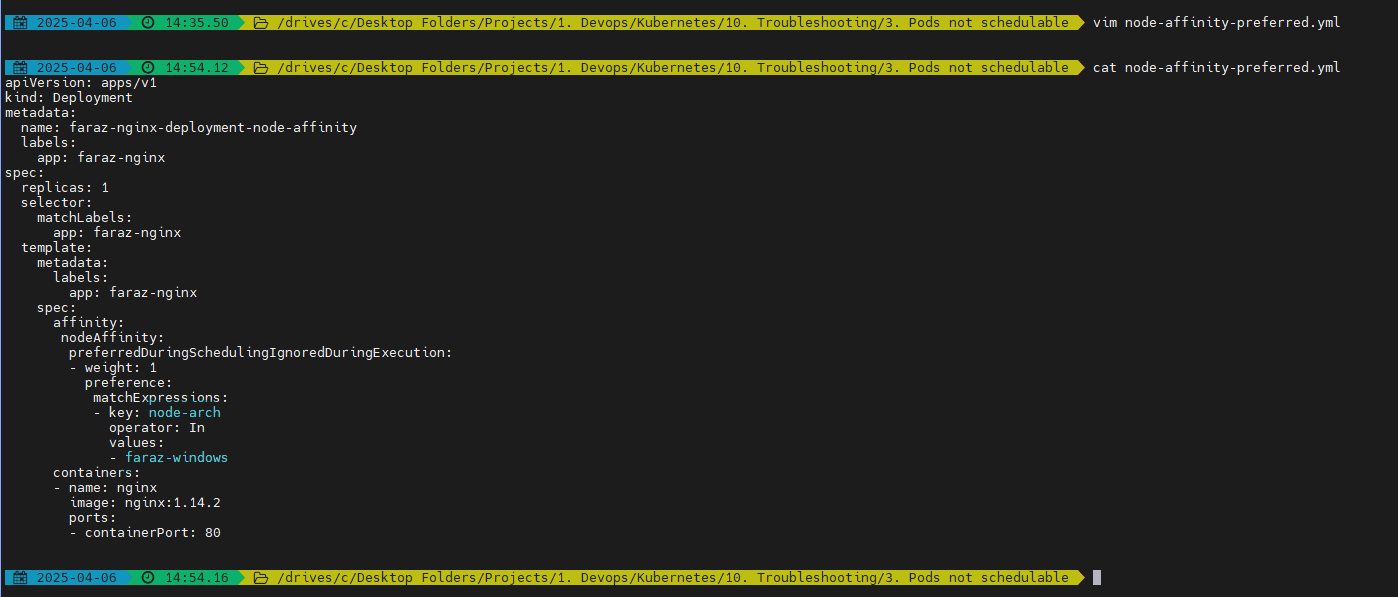
* Step 1: Add label to the node
* Step 2: Add nodeSelector to the pod/deployment

Node selector will not schedule if no match is found

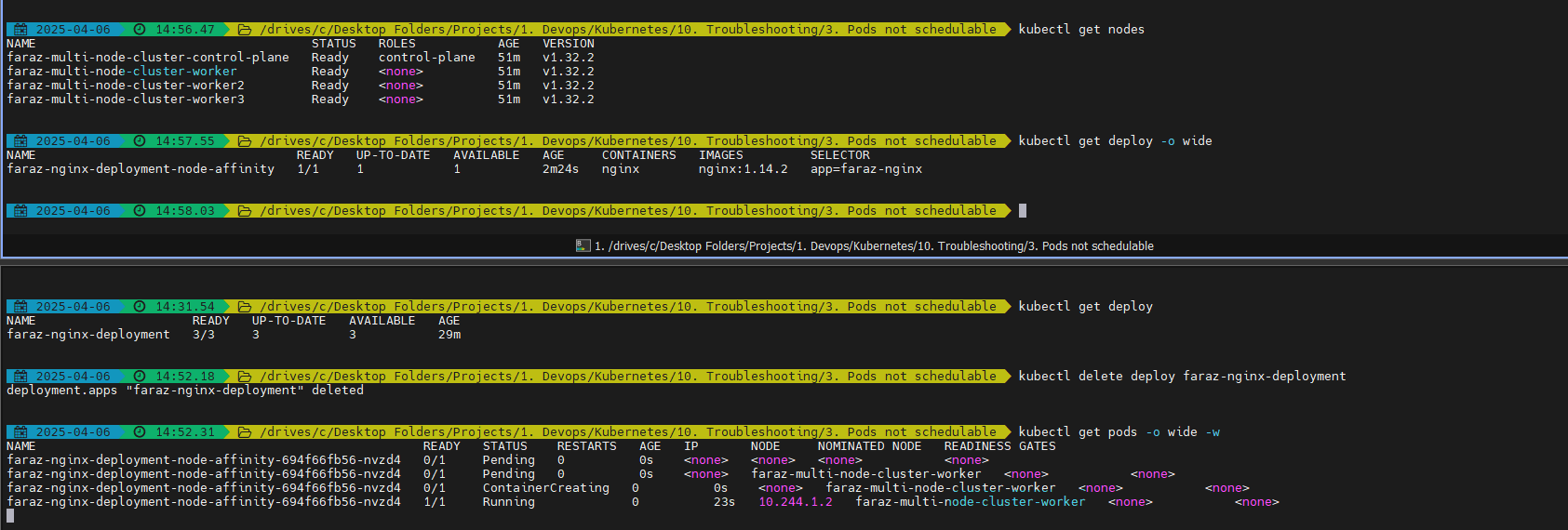
**Node Affinity:**

* Preferred: If scheduler finds the preferred label schedule it, if not then schedule it anywhere
* Required: Similar to Node Selector where we are telling the scheduler that that schedule the pod on the node if 100% match

Create your deployment for preferred:



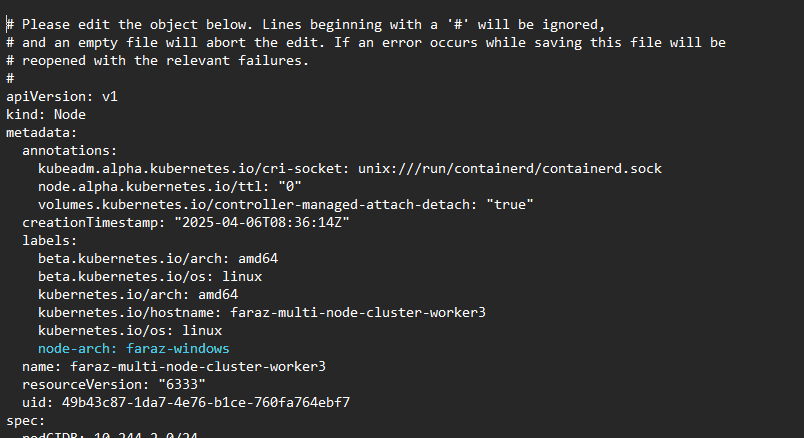
It randomly got scheduled on:



Now if we want to schedule it on any preferred node:

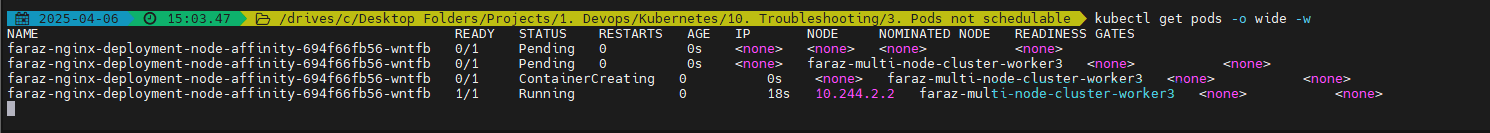
Edit the node:

* KUBE\_EDITOR=notepad kubectl edit node faraz-multi-node-cluster-worker3



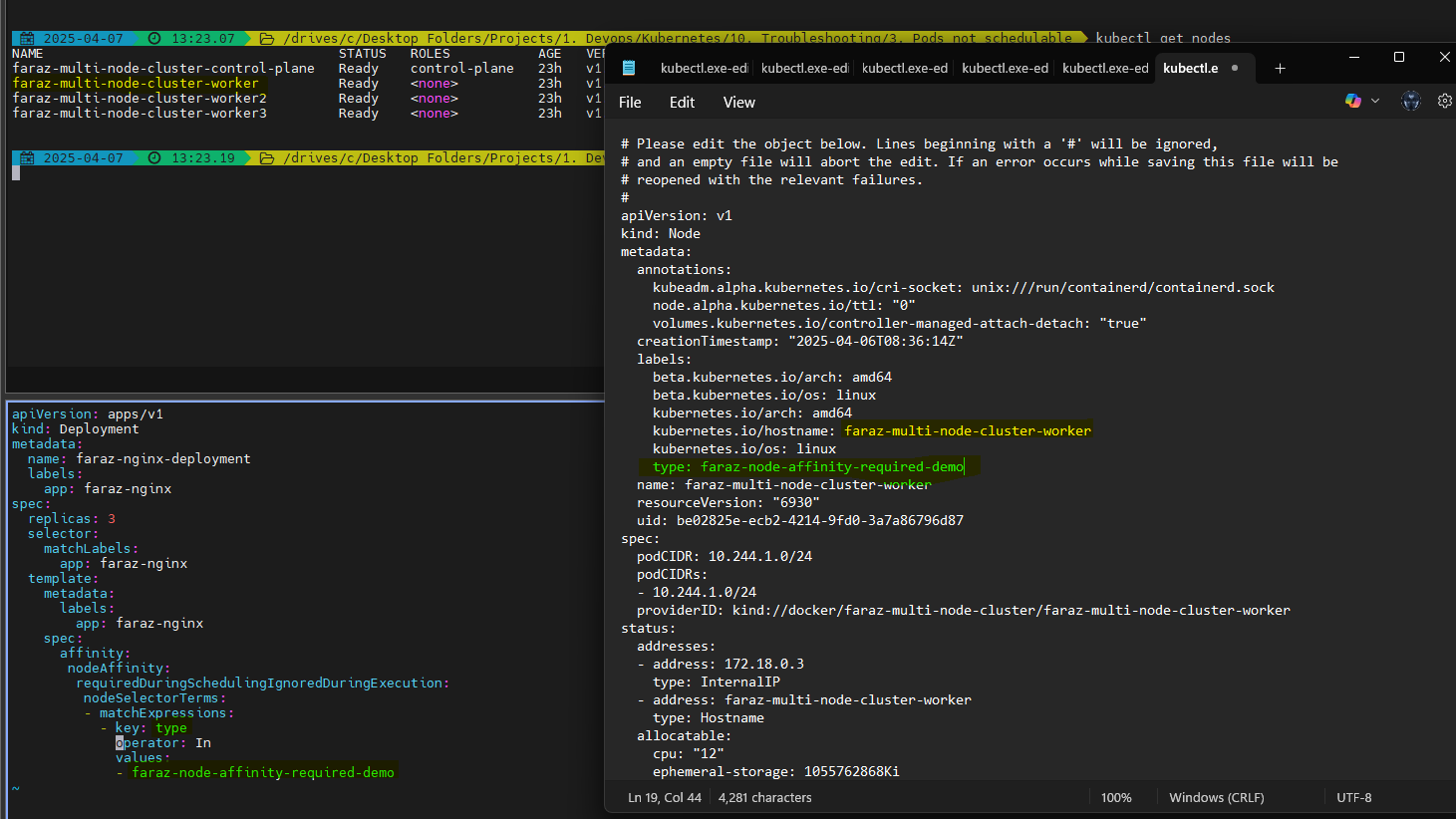


You will be able to see that the node is scheduled on preferred node:



Node Affinity ->Required:

* Edit deployment nodeAffinity
* Edit node labels



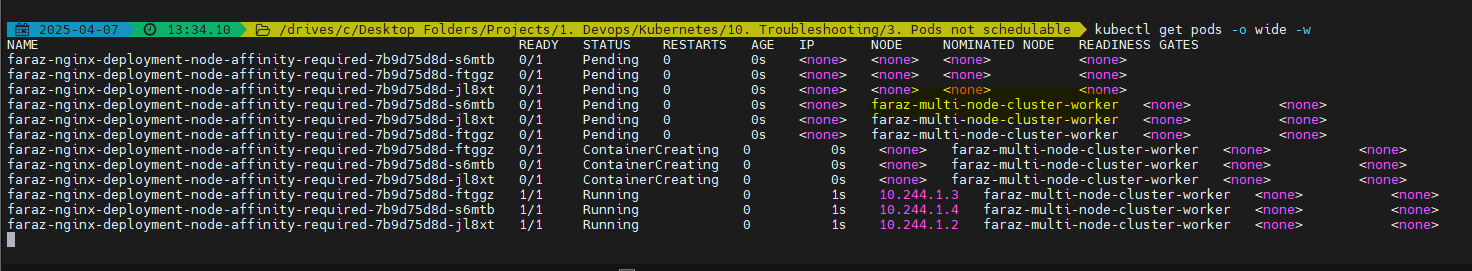
Deployment:



Now apply the deployment:



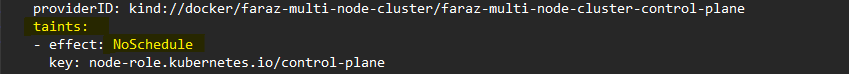
You will notice that the pods are scheduled on the selected nodes from nodeAffinity:



**Taints:**

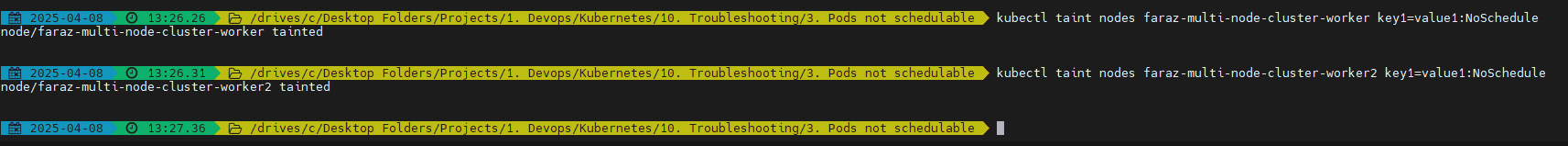
By default your control plane will be tainted as no pods should be scheduled on it

* KUBE\_EDITOR=notepad kubectl edit node faraz-multi-node-cluster-control-plane

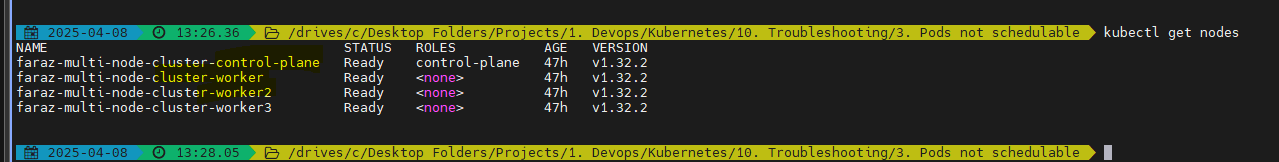


You can taint any node by doing:

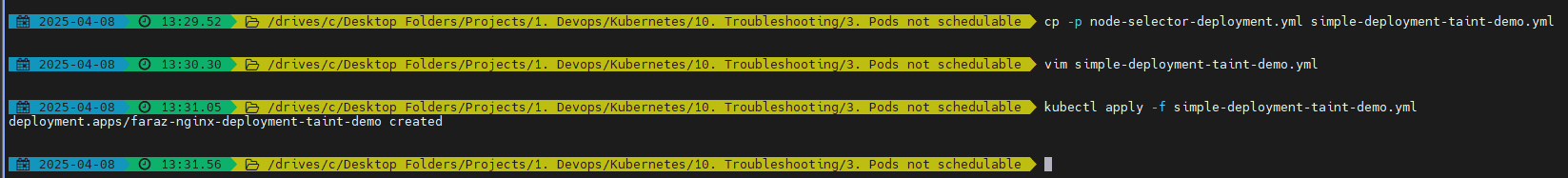
* kubectl taint nodes <nodename>key1=value1:NoSchedule

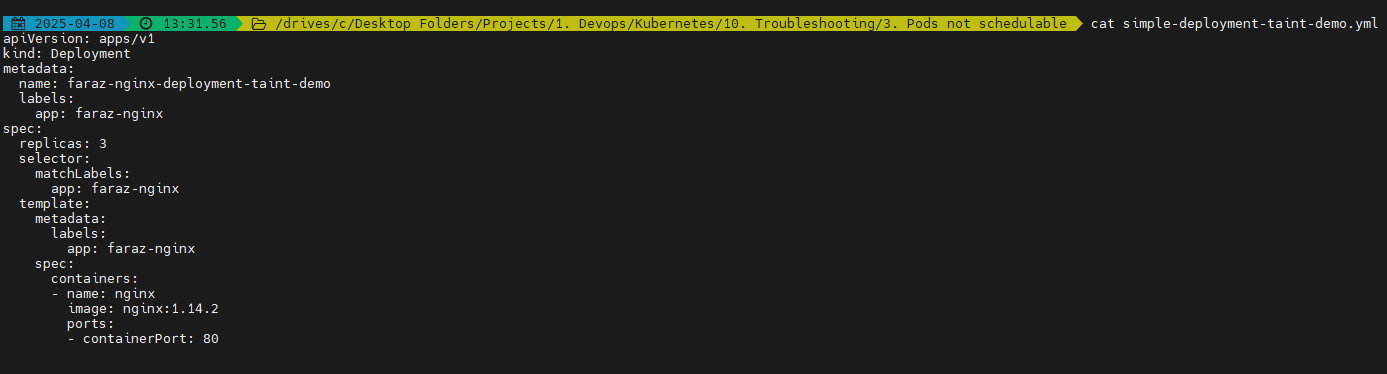


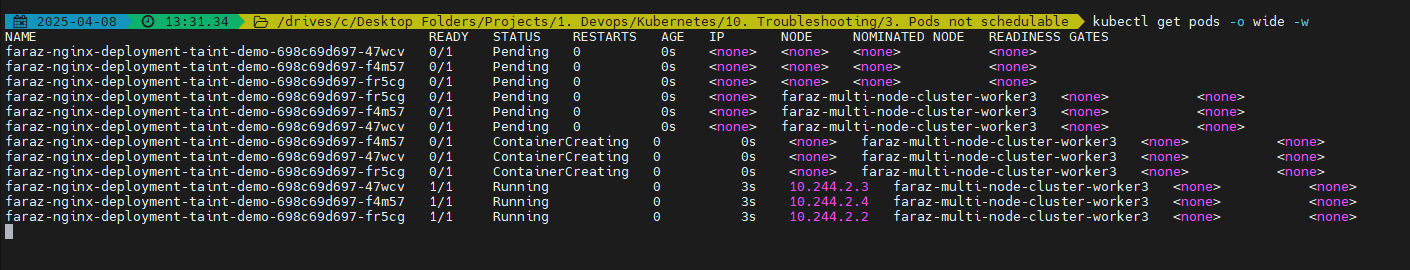
Now we have tainted the above nodes:



Let’s demonstrate if the pods get scheduled on the worker 3 or not:



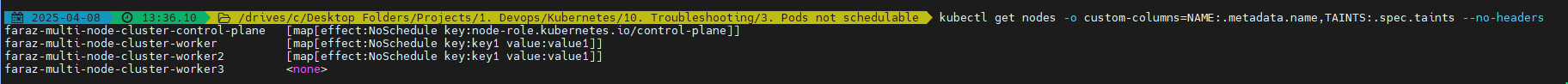




Hence, the pods were scheduled on worker 3

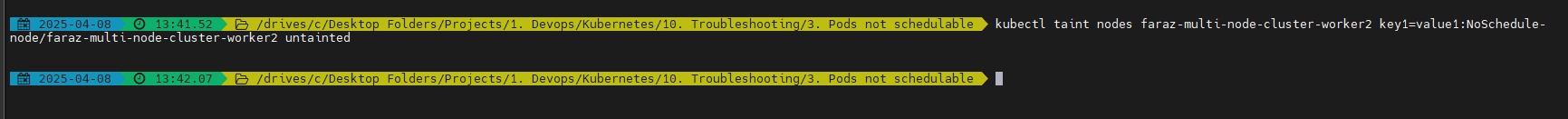
Get the tainted nodes:

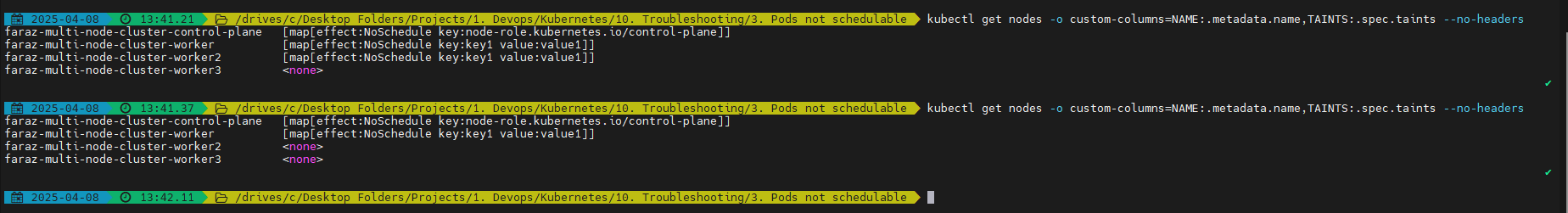
* kubectl get nodes -o custom-columns=NAME:.metadata.name,TAINTS:.spec.taints --no-headers



To remove the taint from the node:

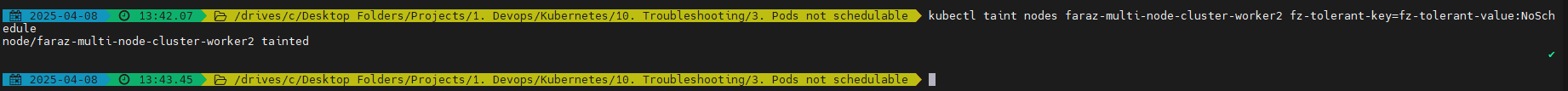
* kubectl taint nodes faraz-multi-node-cluster-worker2 key1=value1:NoSchedule-

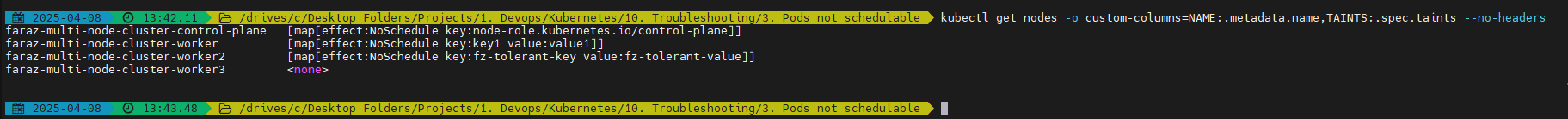


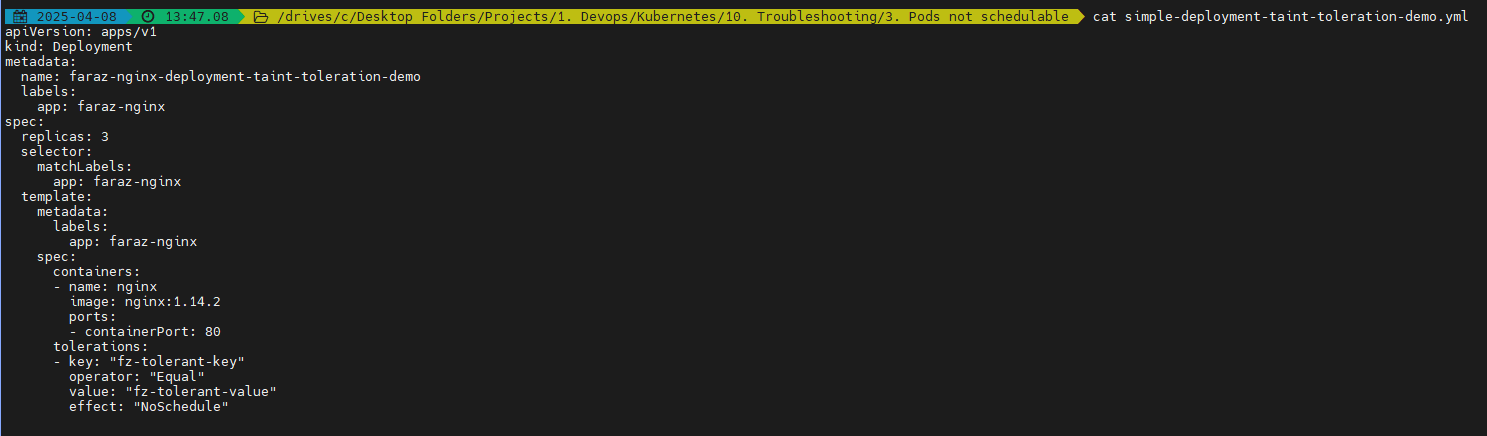


Taint a Node:

* kubectl taint nodes faraz-multi-node-cluster-worker2 fz-tolerant-key=fz-tolerant-value:NoSchedule







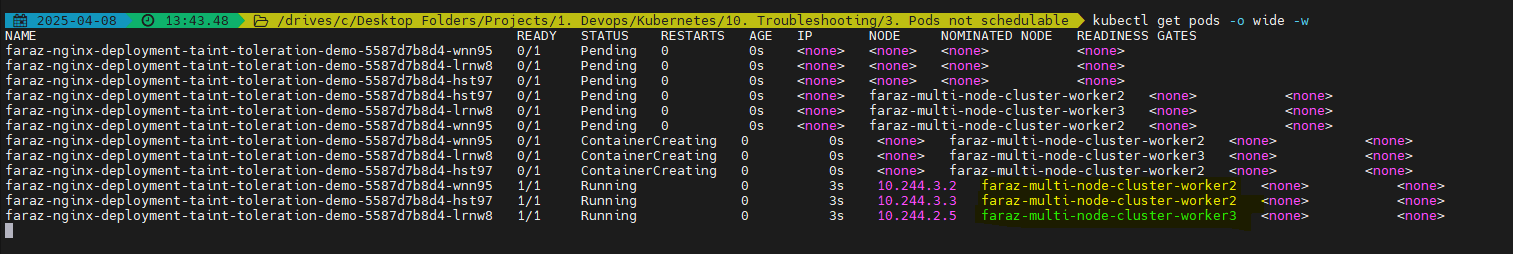
Here's why your pod might have scheduled on faraz-multi-node-cluster-worker3:

Node 3 (faraz-multi-node-cluster-worker3) Doesn't Have Any Taints: As per the output of kubectl get nodes, faraz-multi-node-cluster-worker3 has <none> listed under TAINTS. This means it has no taints to restrict scheduling. Kubernetes considers nodes without taints as fair game for pods unless there are specific nodeSelectors, affinity rules, or similar constraints defined in your pod spec.

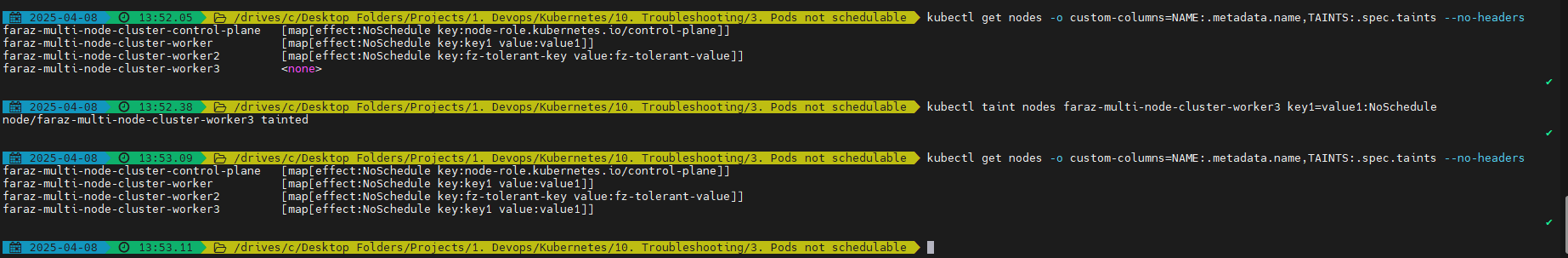
Toleration Not Exclusive to Node 2: While your deployment defines a toleration for the taint fz-tolerant-key=fz-tolerant-value:NoSchedule (present on faraz-multi-node-cluster-worker2), tolerations do not force pods to schedule on specific nodes. They simply allow the pods to tolerate those nodes' taints. Kubernetes scheduler then evaluates other factors like resource availability to determine where to place the pod.

Pod Scheduling Preferences: If faraz-multi-node-cluster-worker3 had sufficient resources and fewer competing workloads compared to other nodes, Kubernetes may have chosen it, even though it has no specific taints or tolerations.

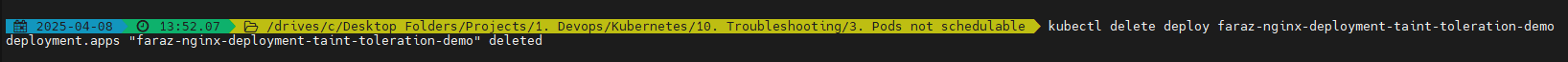
To enforce scheduling on faraz-multi-node-cluster-worker2 specifically, you can use node affinity or nodeSelector in your pod spec.



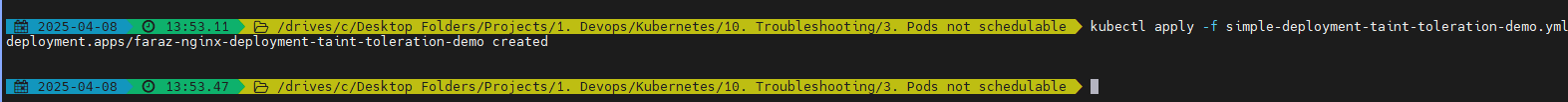
Now that we have tainted all the nodes:



You can delete the deployment:



Apply the deployment:



You will be able to see that all the pods are tolerated on worker node 2 which has matching toleration:

