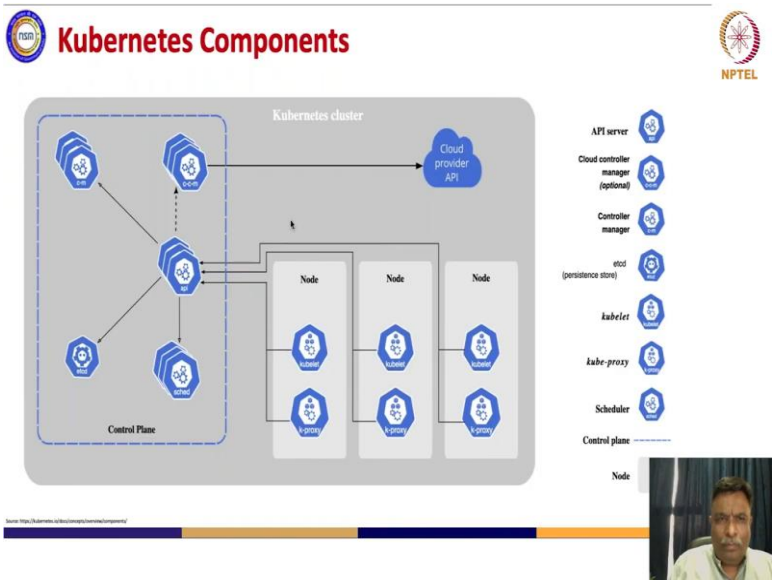
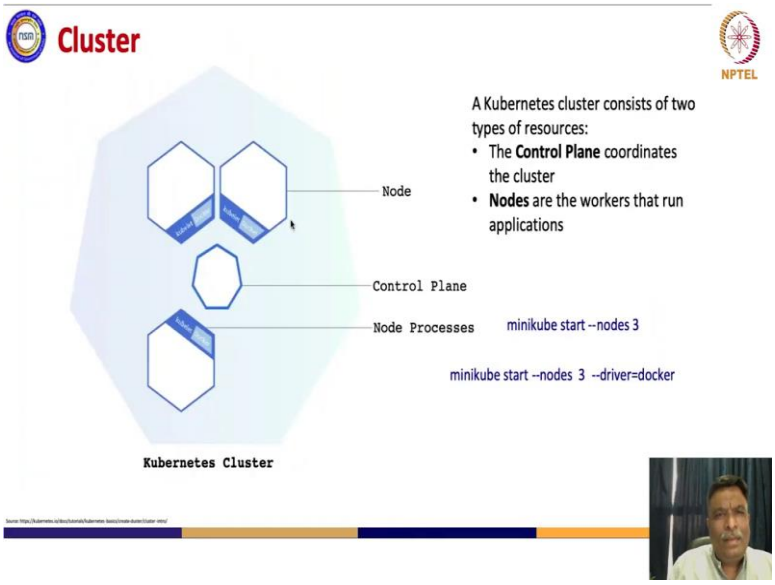
Applied Accelerated Artificial Intelligence
Prof. Satyadhyan Chickerur
School of Computer Science and Engineering
Indian Institute of Technology, Madras

Lecture - 11
DeepOps: Deep Dive into Kubernetes with deployment of various AI based Services
Session I - Kubernetes Part - 2

(Refer Slide Time: 00:32)



(Refer Slide Time: 00:34)



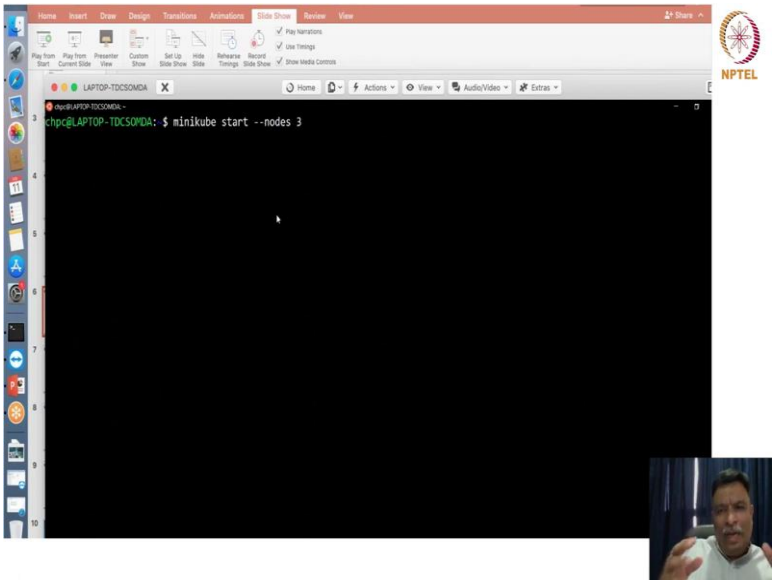
So, let us try to go into a bit of a detail of what a cluster is. If you see this diagram, ok if you see this diagram, this is let us say a Kubernetes Cluster. What we are trying to show here is there is a control plane here; this control plane is responsible for managing this cluster. These are various nodes. Each of these nodes are running certain docker containers and then there is a kubelet, ok.

So, this is a control plane. These are various nodes, this another way of representing a cluster, but this makes things a bit easier. So, I thought we will discuss this. So, this is a control plane. These are various nodes. Each of these nodes, ok will have certain containers.

So, if you technically connect to the previous slide we can assume a Kubernetes cluster will consist of two types of resources, control plane coordinates the cluster and nodes are the workers that run the application. So, this is a very simplistic definition of what a Kubernetes cluster is.

So, now, let us try to actually see, ok a Kubernetes cluster. Do a hands on thing and start understanding that, ok.

(Refer Slide Time: 02:10)

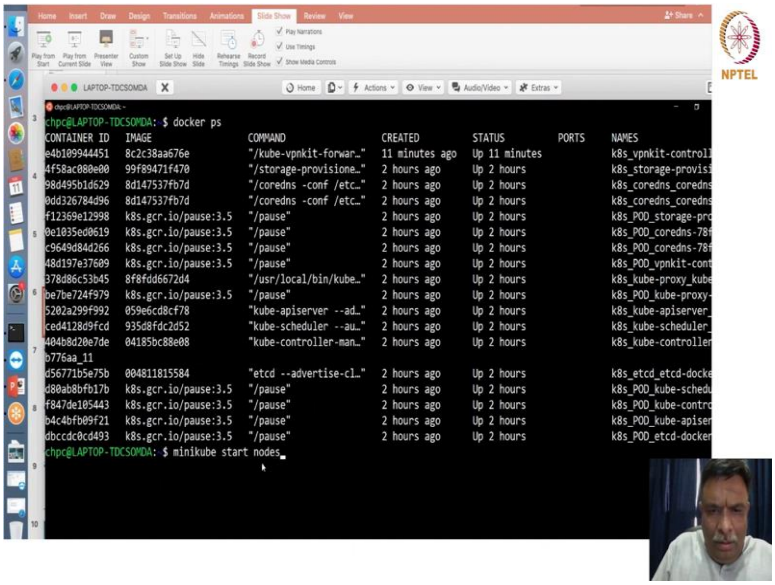


So, the first thing which we do is; so, I am using something which is called as minikube here. A minikube basically is a stripped down version of Kubernetes and it is a single node cluster, right for learning Kubernetes.

So, I am using that to make you people familiar with the concept of Kubernetes. And tomorrow we will see, the actual dashboard with certain specific things, right and then do DeepOps on it. And, then do certain things like the dashboard analysis and something like that, ok.

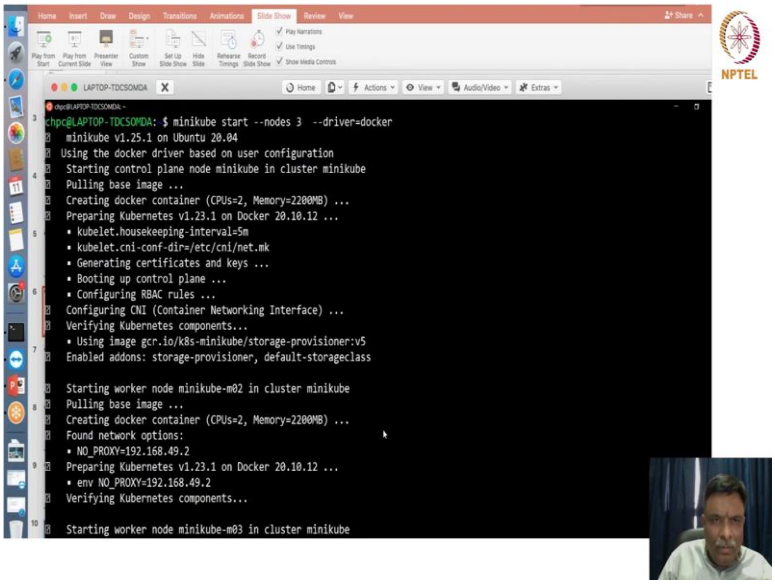
So, now, let us try to do this minikube is a stripped down version of Kubernetes, ok. So, we will start minikube start nodes 3. So, I hope this is visible; we will increase the size of this a bit, yes. So, when I say minikube start nodes 3 before that let me just go and check whether do I have any dockers running, ok.

(Refer Slide Time: 04:03)



So, these are certain dockers which are running, ok which are related to some other issues, but for the time being assume that whatever we are thinking of in terms of minikube node, right those are not running. We will see that. So, I just wanted to show you that when we start our cluster; so let me clear this and then let me start, minikube start nodes 3.

(Refer Slide Time: 04:50)

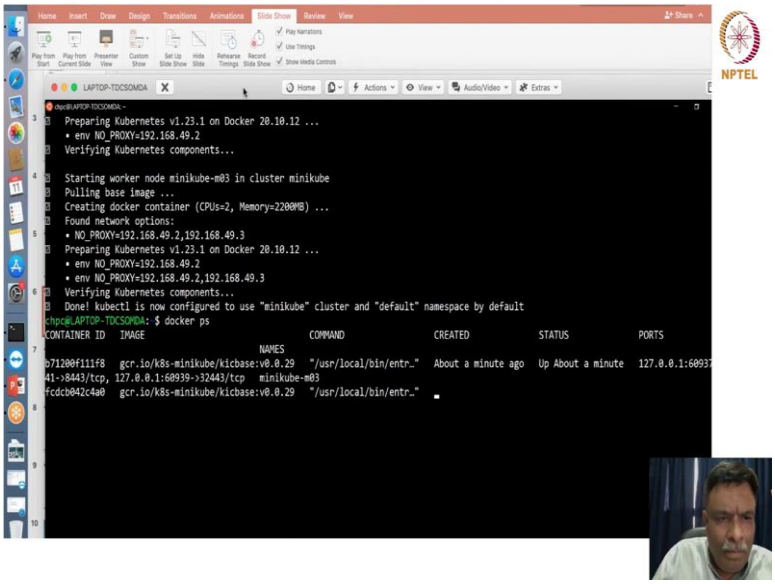


So, what are we going to do here just try to understand this when this image is getting created. We are creating, right docker containers now, right. And what is that going to do? We are creating 3 nodes, ok. Each of these nodes are supposed to run, ok some particular workload as a docker. So, we are creating 3 nodes. Out of these 3 nodes we will have, 1 node as a master node and another 2 nodes as worker nodes.

So, if you see here we have created a minikube now, the name is minikube, the cluster is having one control plane node or a master node. We are creating a docker container which has got 2 CPUs with some memory, and then we are trying to start a worker node which is named as minikube-m02 which basically is the second node technically or the first worker node, ok.

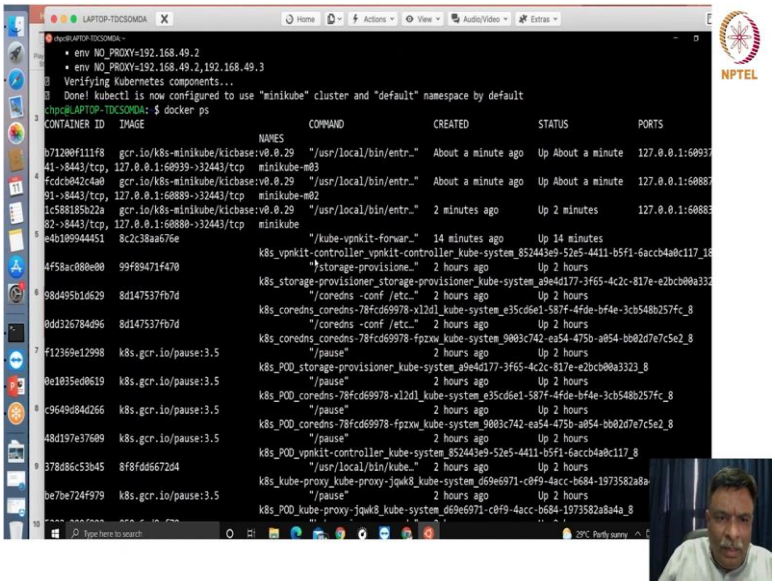
And if you see here, sorry; if you see here; what is this? Ok.

(Refer Slide Time: 06:59)



If you see here, we have another worker node which is minikube-m03, which basically is again a worker node in that particular cluster which is treated as minikube, right. So, I hope this is clear.

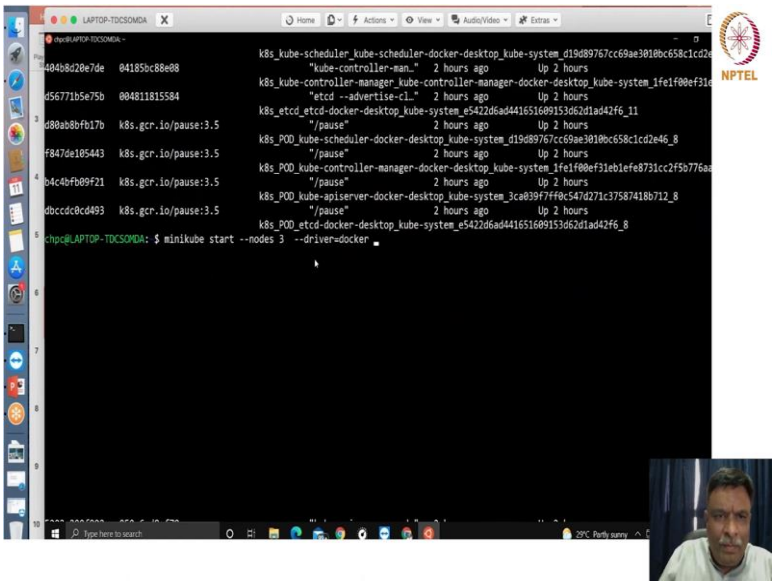
(Refer Slide Time: 07:36)



So, if you now do docker ps, if you see docker ps there are 3 things which are important for us to understand and see. We are now running 3 nodes, right, one is minikube, one is minikube-m02, and another one is minikube-m03, right. So, we have started minikube

with 3 nodes. And then; let where did it go, ok. So, minikube-m02, minikube-m03. So, I hope this is clear, right. So, we have created now.

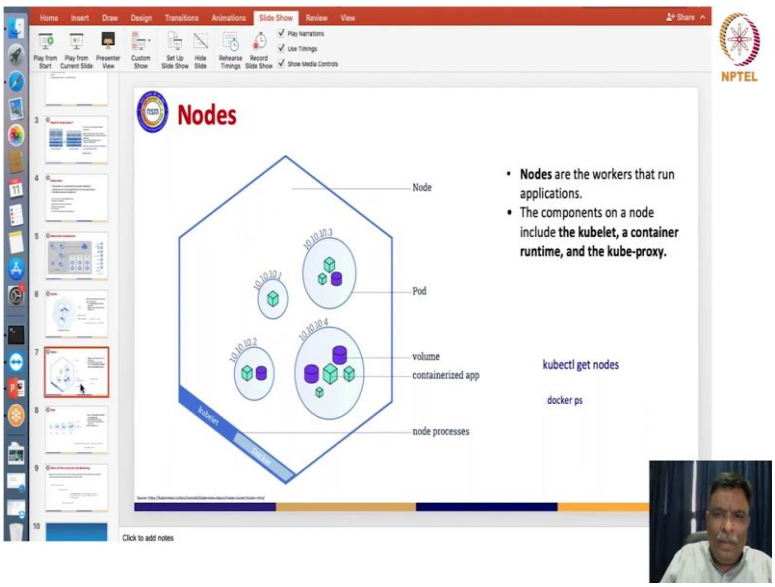
(Refer Slide Time: 08:32)



Now, next thing is; why is this display giving me the issues.

Next thing is we will try to check, ok, the nodes.

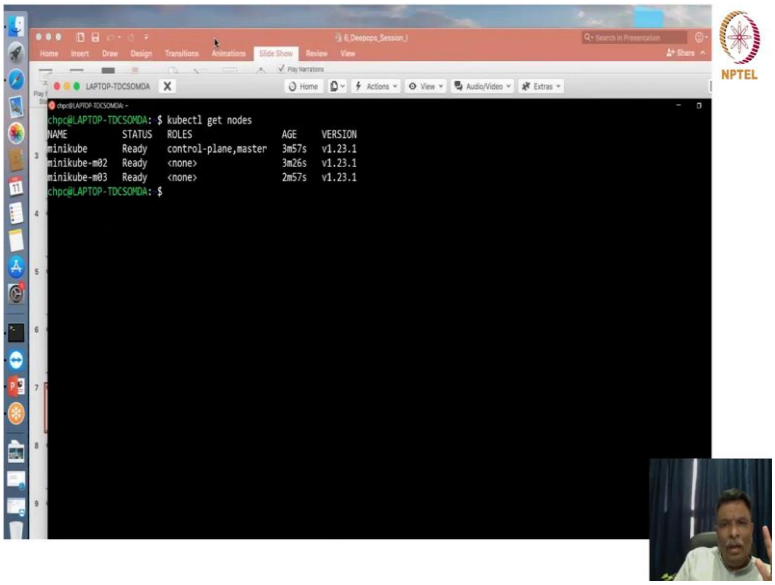
(Refer Slide Time: 08:48)



So, if you see the diagram here, we have got the node process which is a running as a docker, it has got various pods running with certain specific IP addresses.

So, nodes are the workers that are going to run the applications and we know the components on node will include kubelet, the runtime and the kube-proxy, right. So, this is a brief thing which we know.

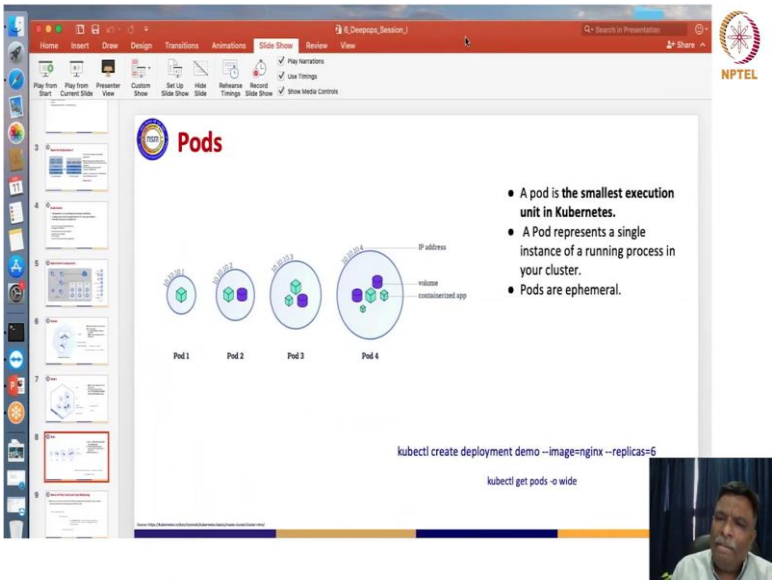
(Refer Slide Time: 09:26)



So, now, let us try to actually use this command which is kube control get nodes. So, if you see kubectl get nodes, what we get is we are getting what does our minikube cluster has, right. It has got a master which has a control plane which is having a name minikube.

Now, this particular minikube-m02 and minikube-m03 are two worker nodes, right; two worker nodes. So, they are ready for us to use, ok.

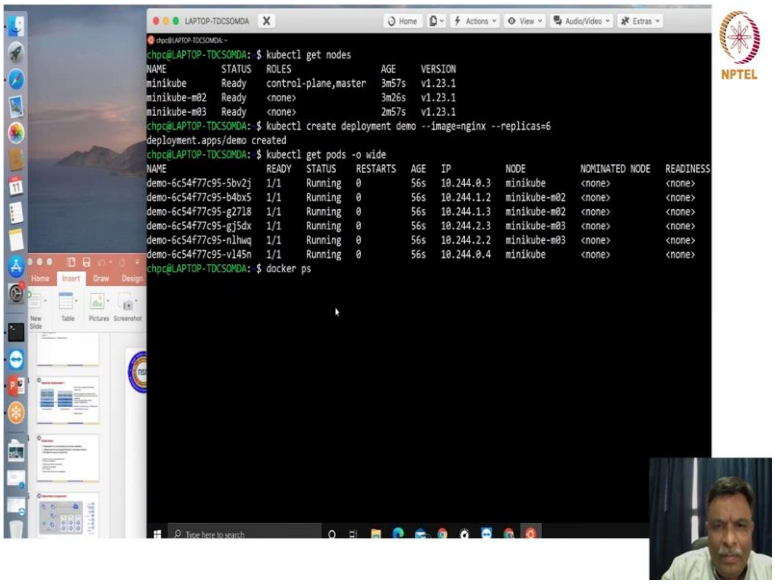
(Refer Slide Time: 10:24)



Now, once you get this let us try to understand and deploy certain workload on it, ok. So, what we have tried to do is, we have tried to create two virtual machines which are treated as these worker nodes because this is a single node cluster which we are trying to have. We have a master node; we have got two virtual machines which act as nodes.

So, these two nodes technically can be your node 1, node 2 which are physical servers also. But for us to understand the concept, we are trying to have 2 worker nodes and 1 master node. And now we will use this command kube control create deployment demo with images of NGINX web server and will replicate it into 6 deployment pods, ok.

(Refer Slide Time: 11:33)



So, let us try to see that. What we do is we will do not going to type, no, yes.

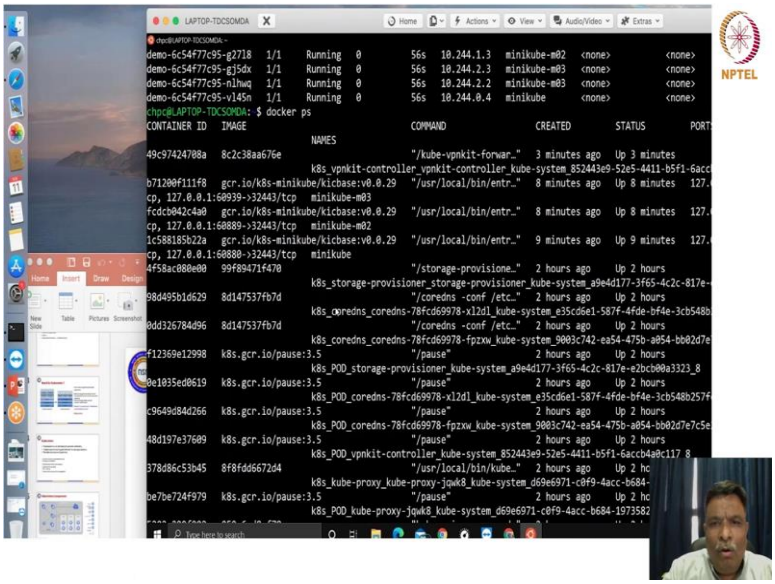
So, we are going to create, ok a deployment demo, ok and then we are going to actually use this in NGINX image, ok for just trying to what to say, trying to create, ok 6 such replicas, ok. So, let us try to do it. We have created, ok.

Now, once we have created let us try to again see whether those pods, ok are visible or not to us. And we use this command which is here which is kube control get pods -o wide. So, let us try to do this command and understand what is that we have created for demo, ok.

So, we have that command. So, do not want to type it again. So, the command is kube control get pods -o wide, ok. When you see this, what has happened actually? What effectively has happened is we have created 6 pods, which are of similar type, ok because you have created replicas of them.

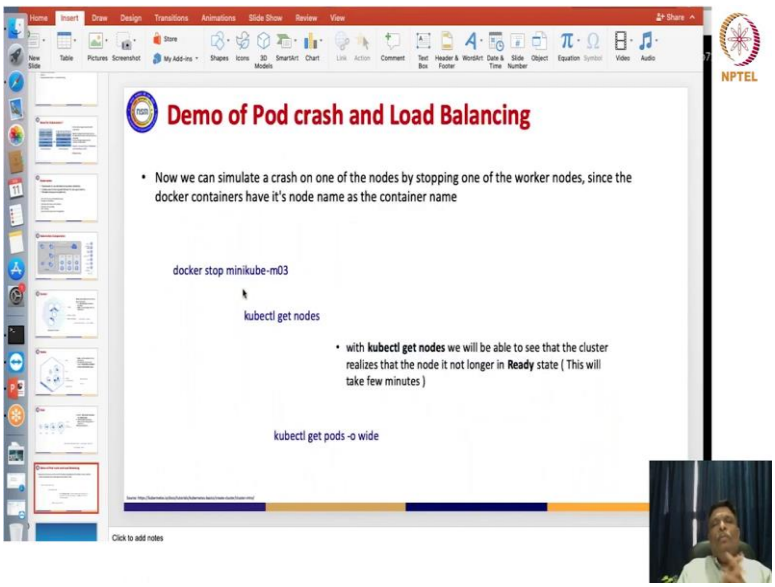
And what is going to happen is it has now got allocated, ok to this cluster of ours. Here minikube is one master node it has got two pods running, minikube-m02, yes, has got two pods, ok and then minikube-m03 which is another node has got two pods. So, we created 6 pods. Those automatically got scheduled, ok on these 3 nodes automatically.

(Refer Slide Time: 14:37)



Now, we will try to do a experiment. And in the meantime, we can also see if you do docker ps, there is no information about these pods, ok running on as containers from the docker perspective correct. So, I just wanted to be show you this.

(Refer Slide Time: 15:00)

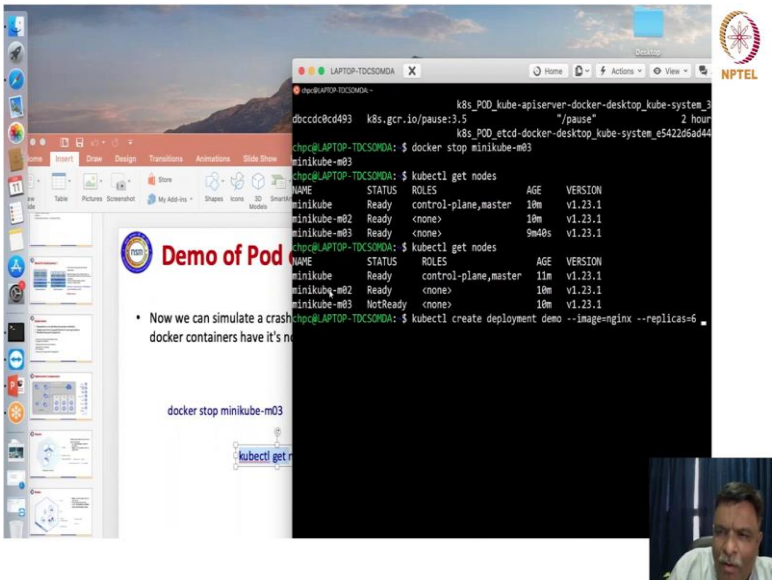


But now what we are going to do is we are going to actually do this demo of crashing a pod, ok crashing a pod and then doing some load balancing. What does that mean? That means that we can simulate a crash on one of the nodes by stopping one of the worker

nodes, ok as written here in the slide. And since the docker containers have its node name as the container name, it is very easy for us to understand.

So, let us try to do this and stop the dockers which are running on minikube-m03, ok. So, let me just do that. Let me just do docker stop, ok.

(Refer Slide Time: 15:43)

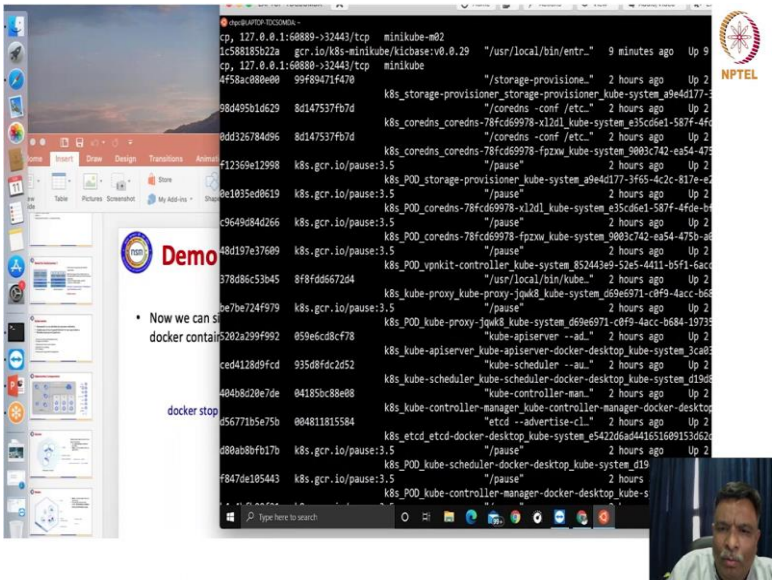


It is minikube-m03. So, once we do this it has stopped. So, now, let us try to again see this command, ok kubectl get nodes, right.

So, now, see the idea is that we have stopped the minikube-m03 docker, ok. When I say docker minikube-m03 has stopped, my node has to stop actually. So, it will take few minutes for getting that result because there has to have some synchronization, orchestration and the message passing to happen. So, let us try to see this.

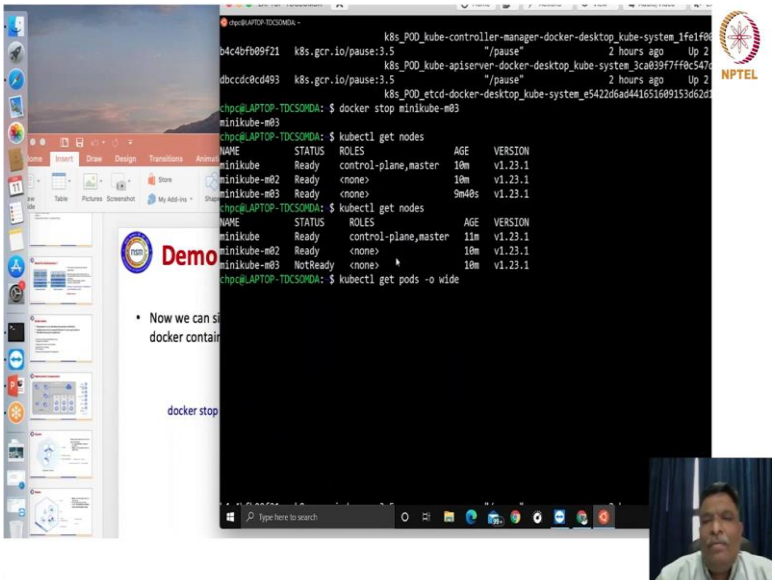
And now you have got the status as not ready. So, now, what we have tried to simulate is one of our nodes has failed. So, minikube-m03 node has failed. Now, we are only left with two nodes. So, what technically should happen? Since, Kubernetes has to do load balancing, it has to ensure, ok it has to ensure that the pods which were running on this node minikube-m03 should actually be relocated to or balanced to these two nodes which is minikube and minikube-m02, ok. So, let us try to do that also.

(Refer Slide Time: 18:15)



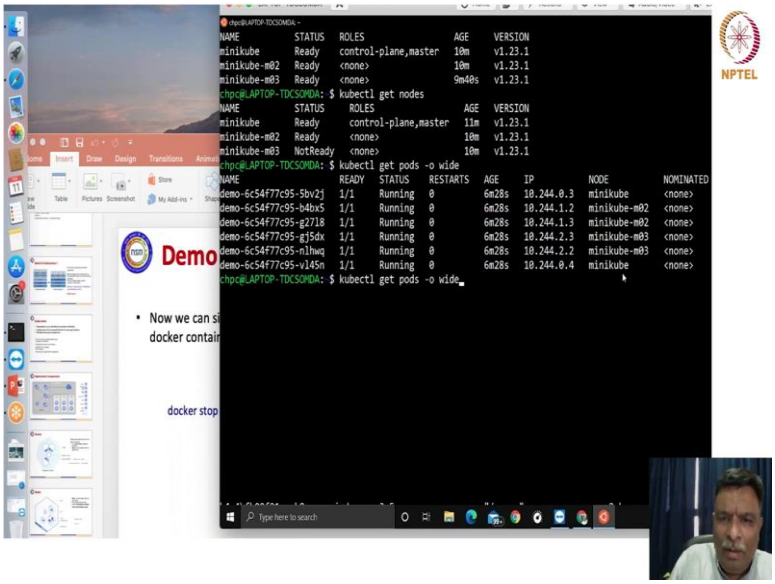
And what we are going to do is we are going to; sorry.

(Refer Slide Time: 18:20)



We are going to now see this information again. Please understand, we have not scheduled or we have not transferred the pods running on minikube-m03 to m02 and minikube. So, let us try to see where are those pods running.

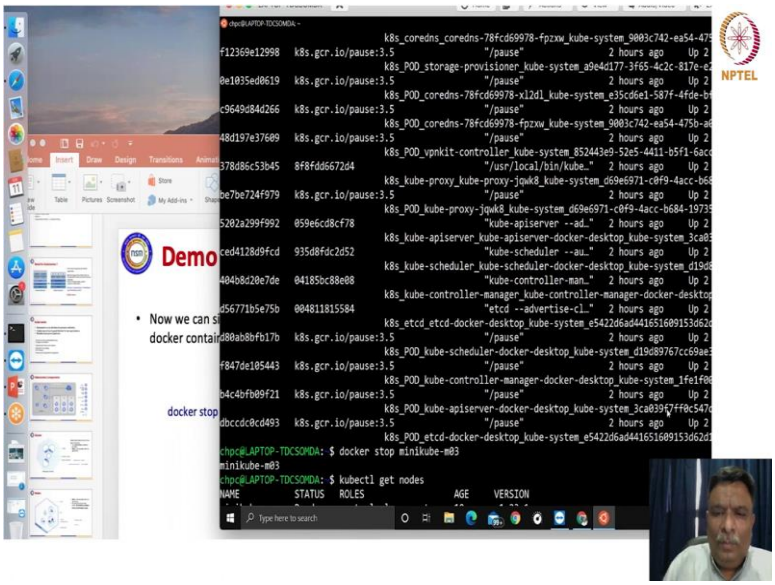
(Refer Slide Time: 18:39)



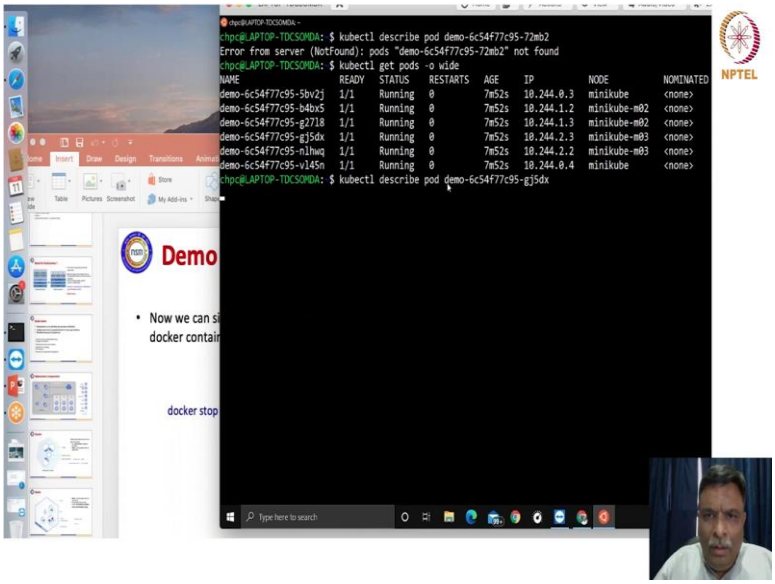
So, still if you see minikube-m03 is running this pod and minikube-m03 is running this pod and each of these nodes are running two pods each, right.

Since, I told you it will take a few minutes for us to get back that information, let us wait for a few minutes. In the meantime, let us try to see, ok the description of each pod, ok. So, why is this.

(Refer Slide Time: 19:22)



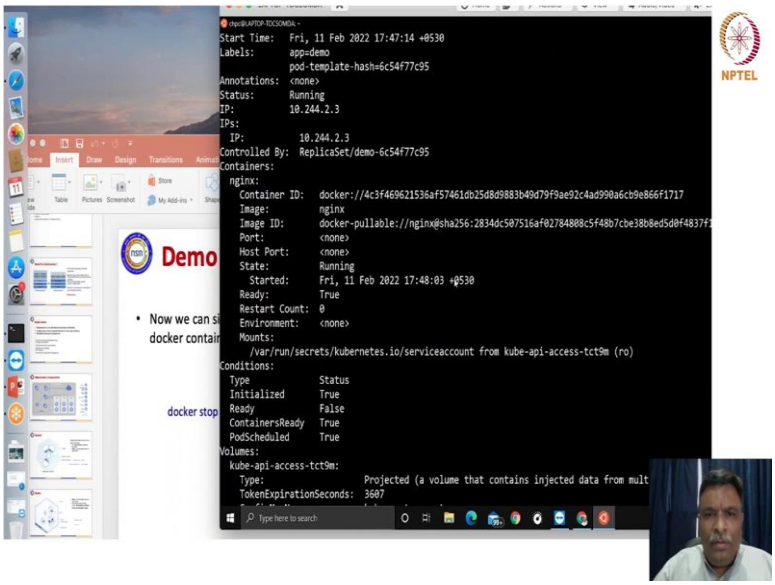
(Refer Slide Time: 19:33)



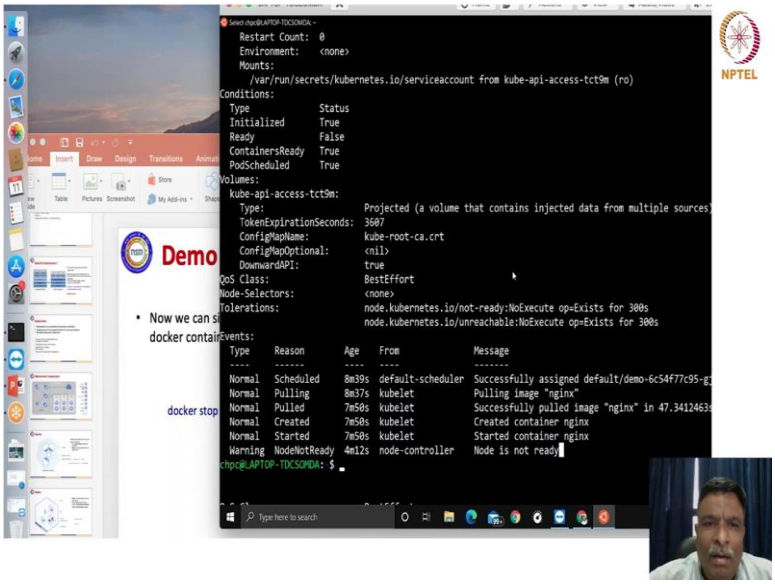
So, I will describe it using this command which is. See, described being a pod, why this pod is not available now, let us try to understand it. Because that pods name was a bit different. So, we will have to see some pod here like which was supposed to be running on minikube-m03 which we had stopped. So, let us try to see this pod, ok.

Let me just copy that information.Yes.

(Refer Slide Time: 20:51)

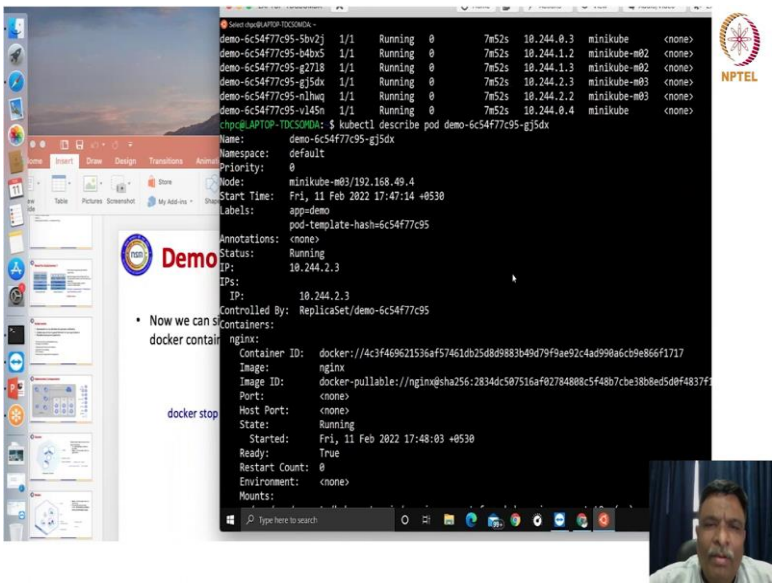


(Refer Slide Time: 20:54)

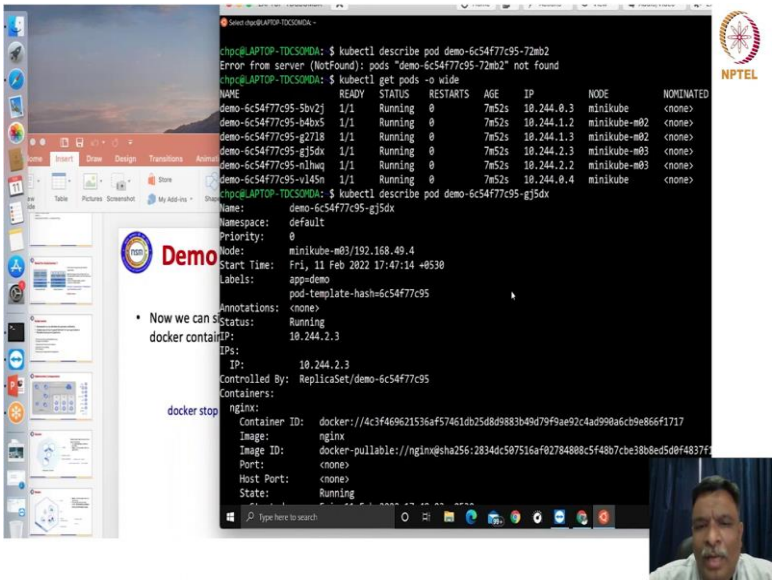


So, if you see this information about the pod, ok it has done scheduling, pulling, created, everything happened, right started the container, created the container, but at one point in time it shows node is not ready. This is the time when we stopped the node, ok.

(Refer Slide Time: 21:16)

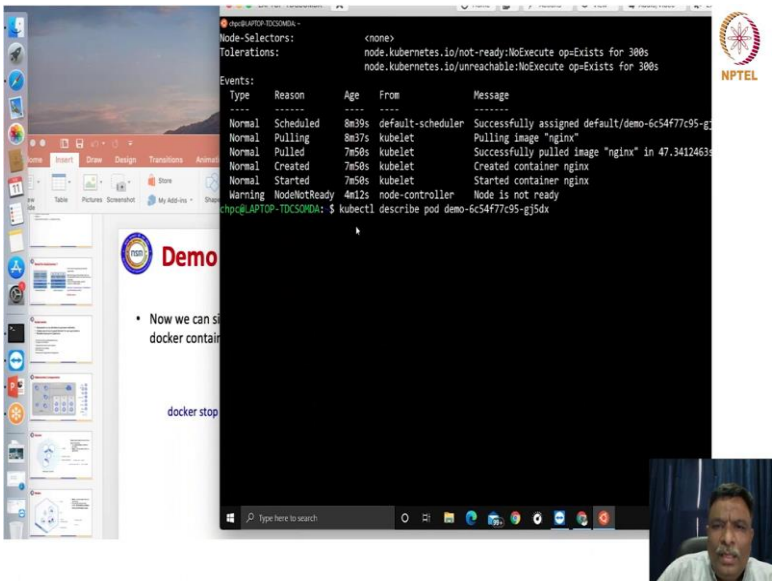


(Refer Slide Time: 21:17)



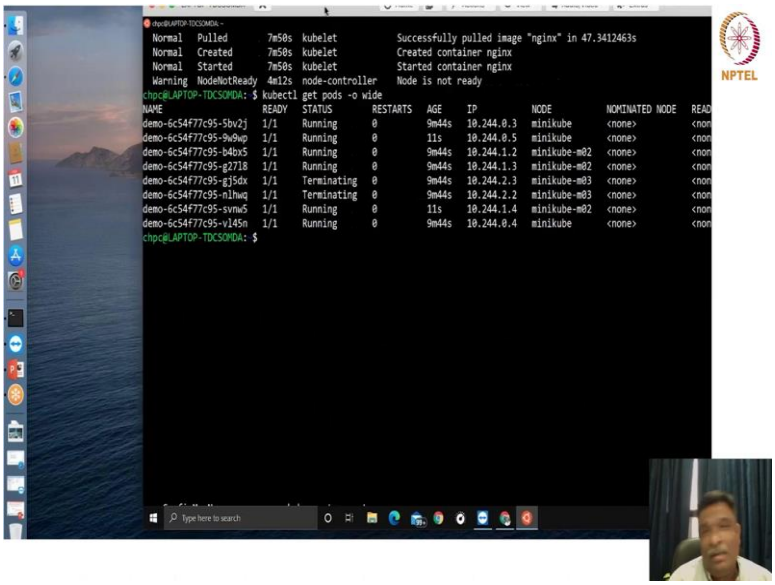
So, this has got a lot of information. We will discuss maybe in the days to come we will gradually understand, right what is this particular container ID, what is that image which you have put in, ok, what is the state of that particular pod actually, ok, like it was running, somewhere running, right, yeah state was running; at what time it started, right. So, how many times it has been restarted actually all of this. So, this is how you describe each pod.

(Refer Slide Time: 21:47)



So, let me just again go to and see whether we have actually been able to; yes check here now.

(Refer Slide Time: 21:51)



We initially had two pods, ok running on each of these nodes, we had 3 nodes. Then, we simulated a crash by stopping one of the nodes, and then we told that, ok there are only two nodes available. So, now, what has happened? If you see minikube-m03 node, the pod which was running here, ok has terminated, this pod also has terminated.

And if you see this has been shifted, ok to one particular pod to this particular minikube node and another one to minikube-m02 node. So, if you see this way, there are again 6 pods running, but those 6 pods are actually split or distributed among only two nodes, which is the master node and one worker node, because another worker mode actually is crashed or its not available.

Now, in this situation, since the master node is also running a worker node sorry a pod, it is not necessary that the pod could be scheduled on the master node. Generally, it does not happen, people do not try to do that or when the Kubernetes cluster is set up it is not done, it is not advisable. But here we are just showing there is a option of no schedule. So, if you do not schedule it, it goes to some other node instead of the master, right.

So, this is what actually we had thought of talking today. And in the next session, we will link this particular concept of scheduling, orchestration, ok to something which is

very specific to AI workloads which is DeepOps. We will see how we can see the dashboards, ok and try to see the Kubernetes dashboard, try to see how the visualization happens of the CPU, the memory, various pods, ok all of this in the next session.

So, I suppose today is this thing we are done.