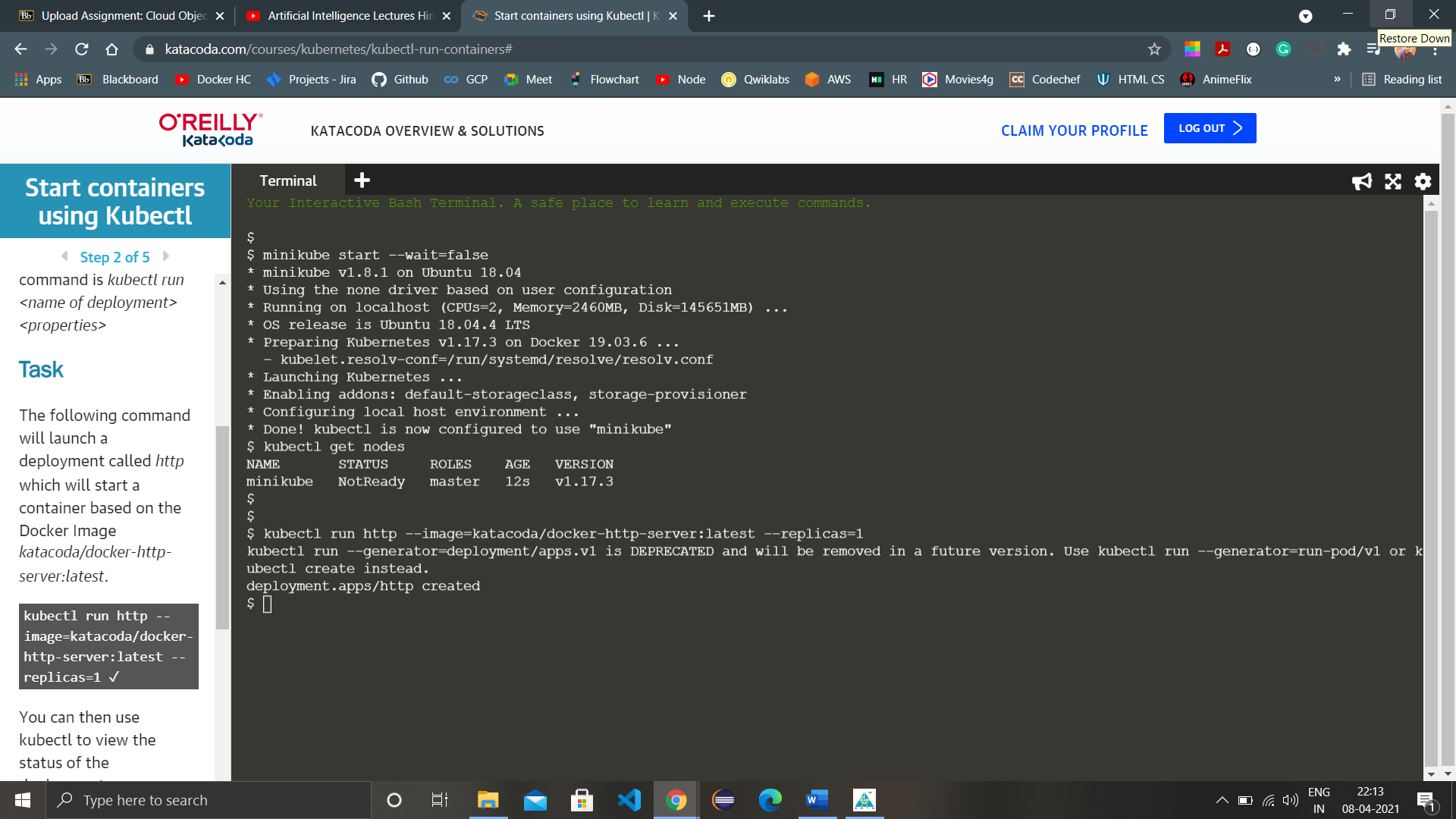
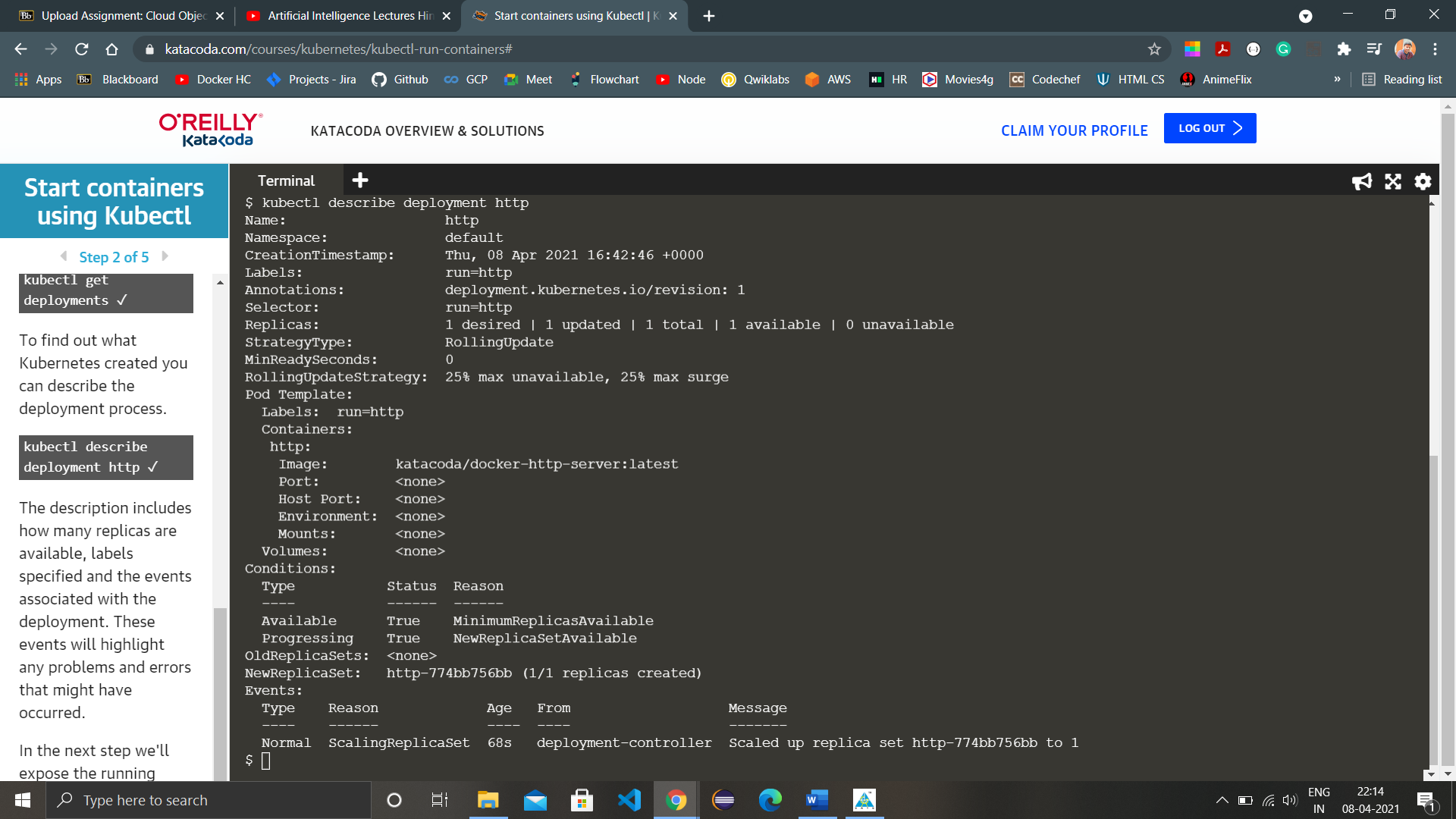
**Experiment 8**: Deploying containers using Kubectl and launch a single node cluster

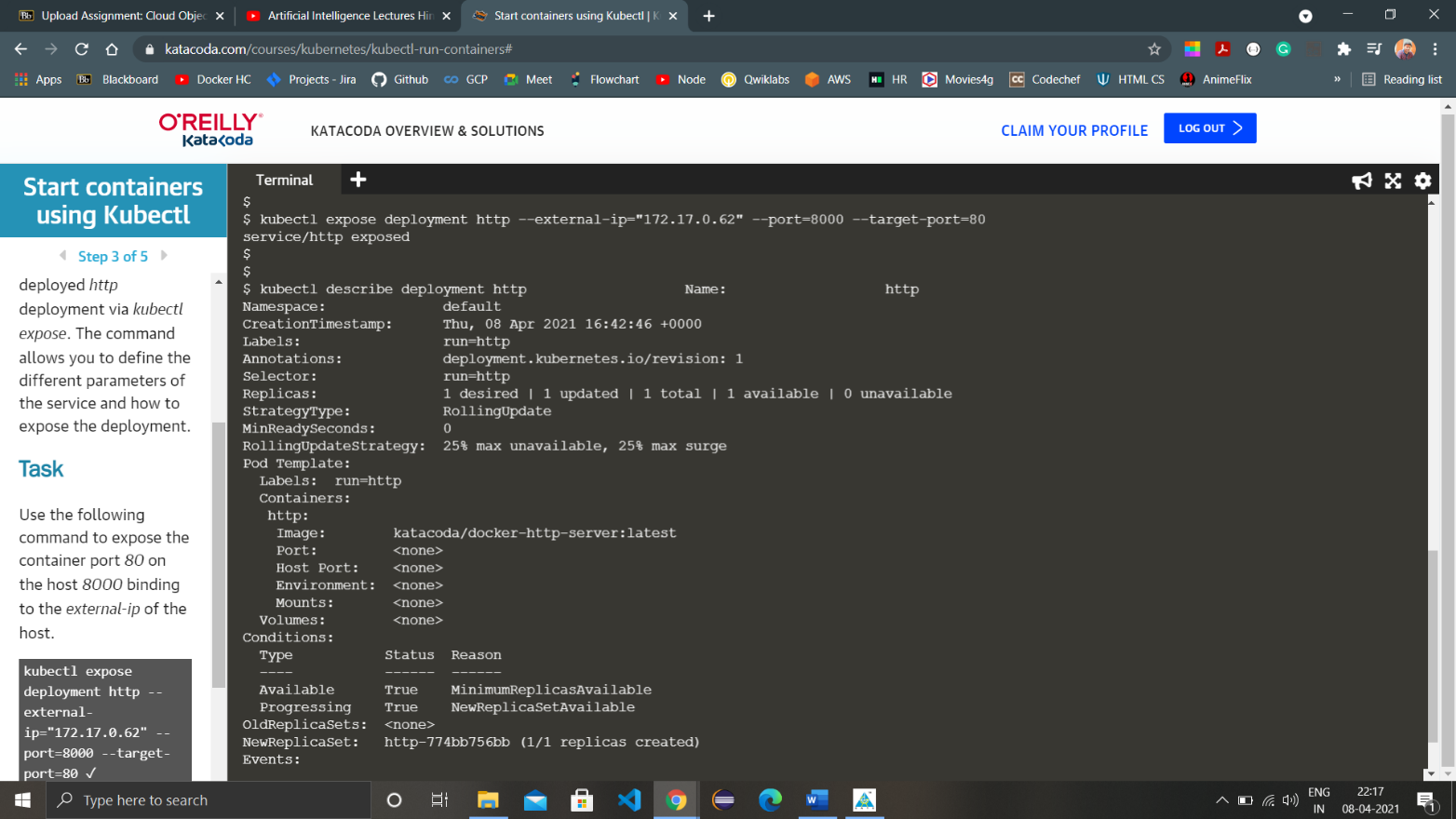
1. Deploying containers using Kubectl
2. First of all, we need to start the cluster and download kubectl command line using the command ‘miikube start –wait=false’.



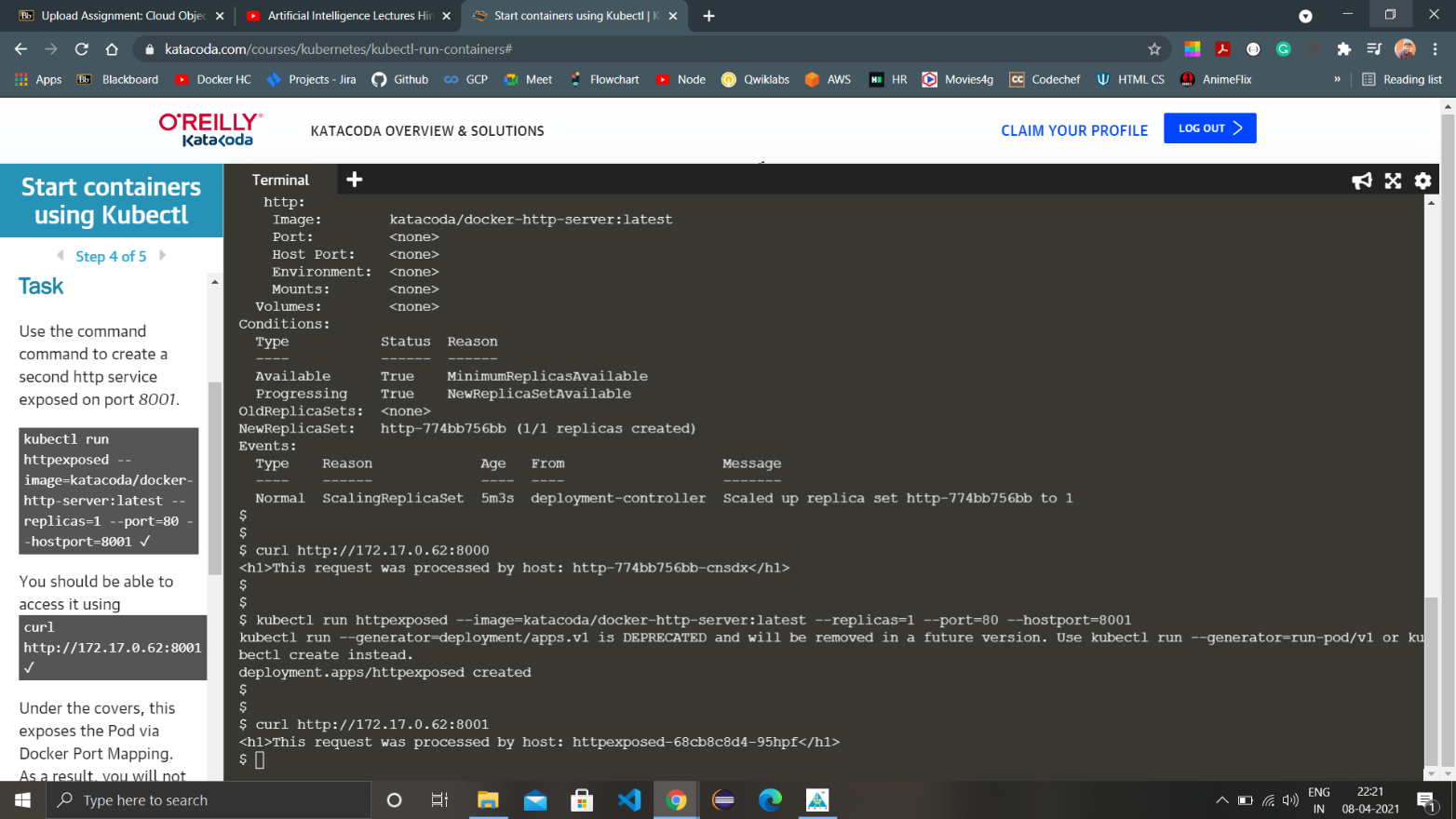
1. Once the cluster is started wait for the node to get ready and check the nodes using the command ‘kubectl get nodes’.
2. Now we have to run a deployment called http, at first we are considering only one replica. This will be done by the command ‘kubectl run http --image=katacoda/docker-http-server:latest --replicas=1’
3. For knowing about the status of deployment, run the command ‘kubectl get deployments’.
4. Also we can check information about a particular deployment and what all did Kubernetes created in that deployment with the command ‘kubectl describe deployment http’. Description also include number of replicas in the deployment.



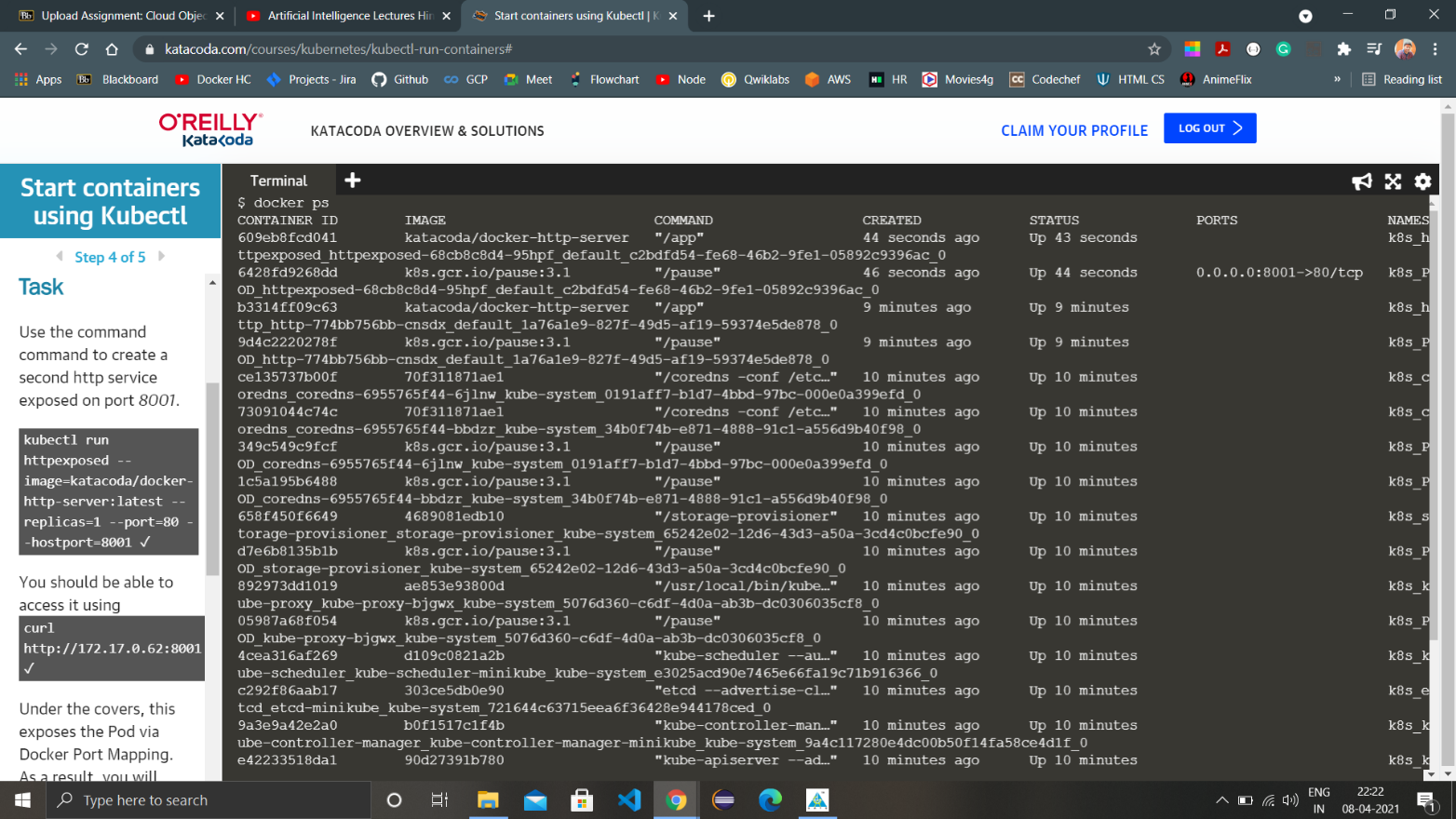
1. Now we have to expose the deployment to port 80 on the host 8000 binding to the external IP of the host using the command ‘kubectl expose deployment http --external-ip="172.17.0.48" --port=8000 --target-port=80’. Now we can access the HTTP service using the curl command ‘curl <http://172.17.0.48:8000>’.



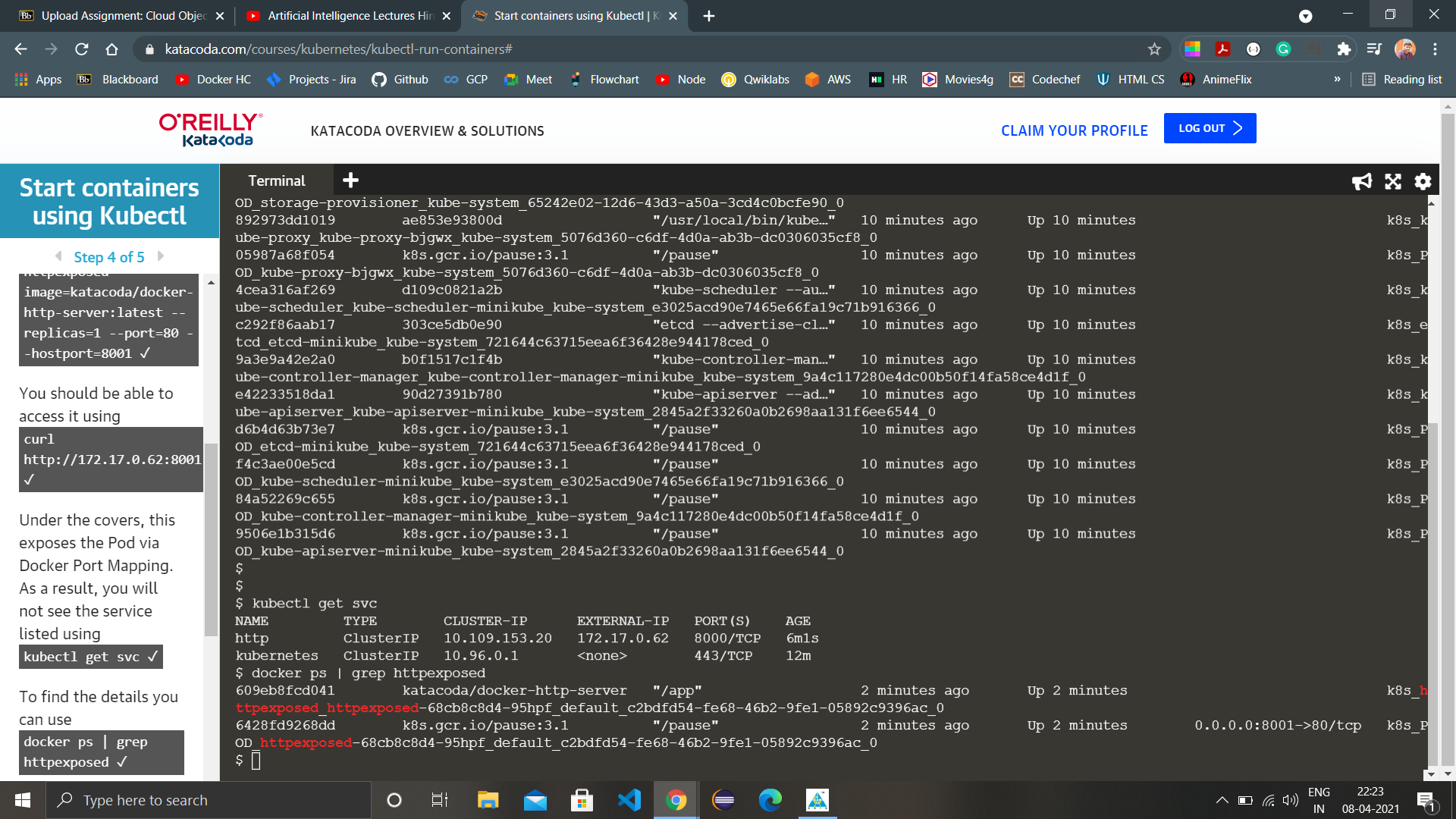
1. Create a second http service exposed on port 8001 using the command ‘kubectl run httpexposed --image=katacoda/docker-http-server:latest --replicas=1 --port=80 --hostport=8001’. Now the second service we created just now can be accessed using the curl command ‘curl <http://172.17.0.48:8001>’.



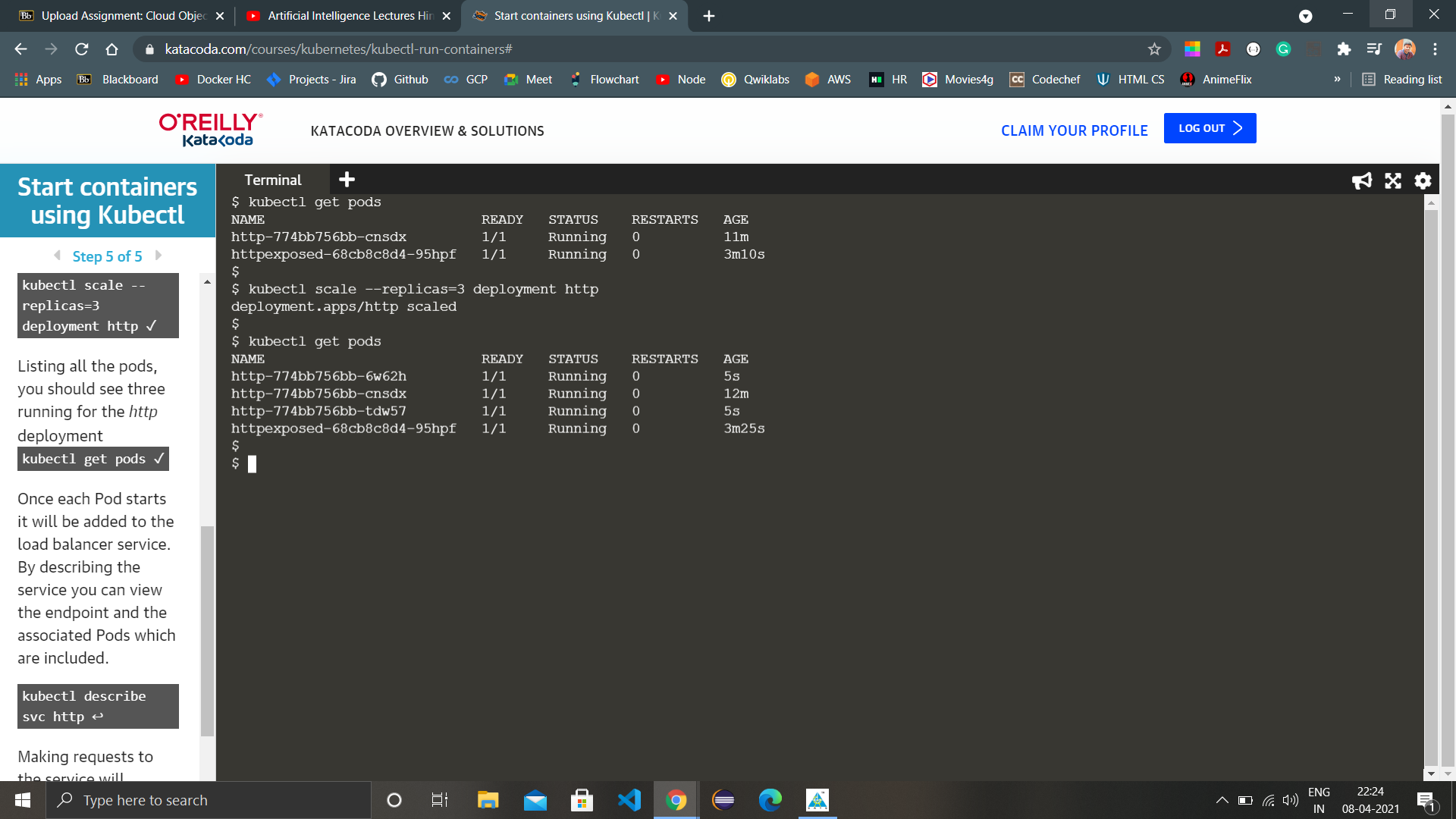
1. Run ‘docker ps’ command to get a list of all the containers running currently.



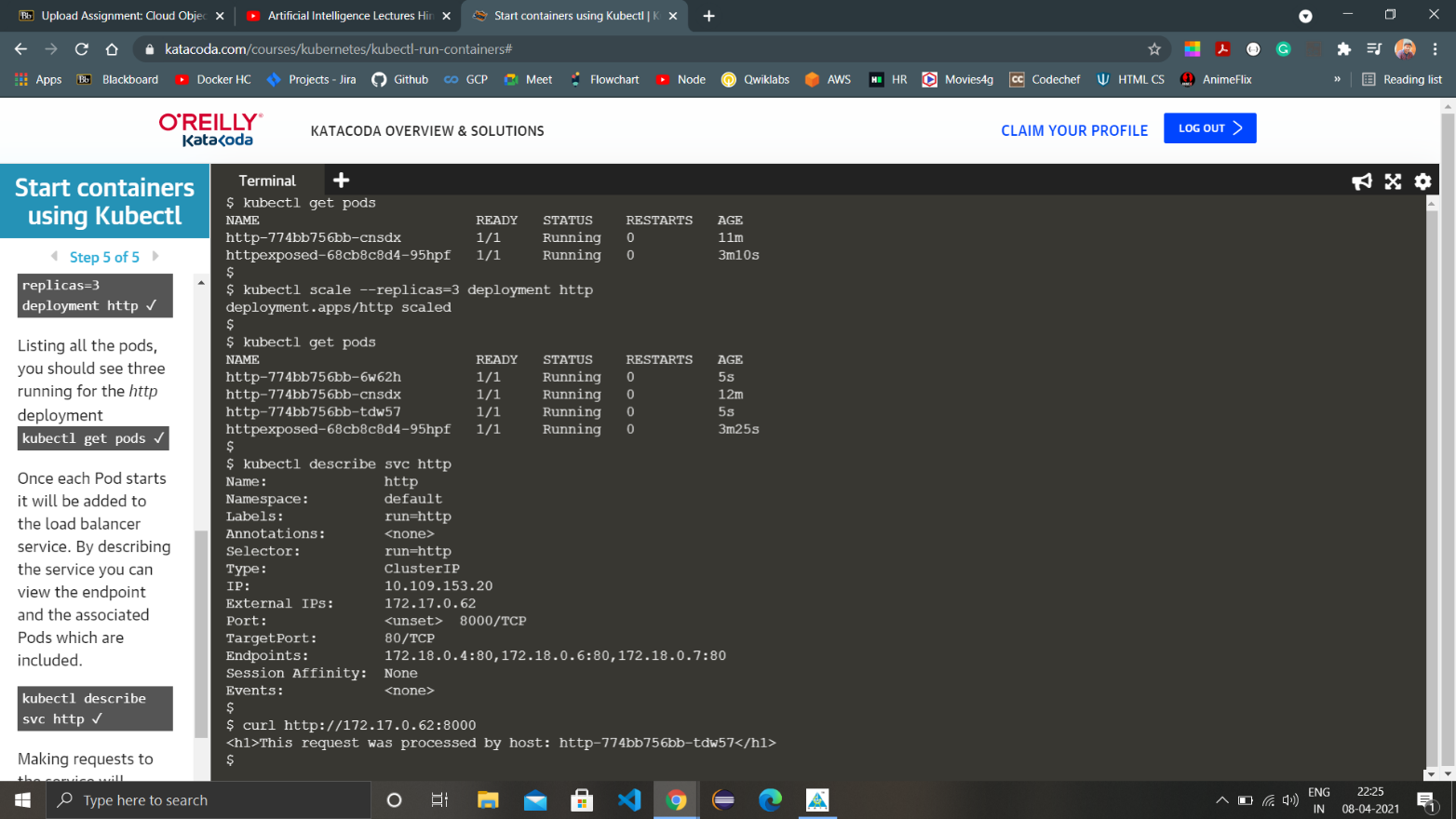
1. Run command ‘kubectl get svc’ and ‘docker ps | grep httpexposed’ to get the service listings along with details of the service.



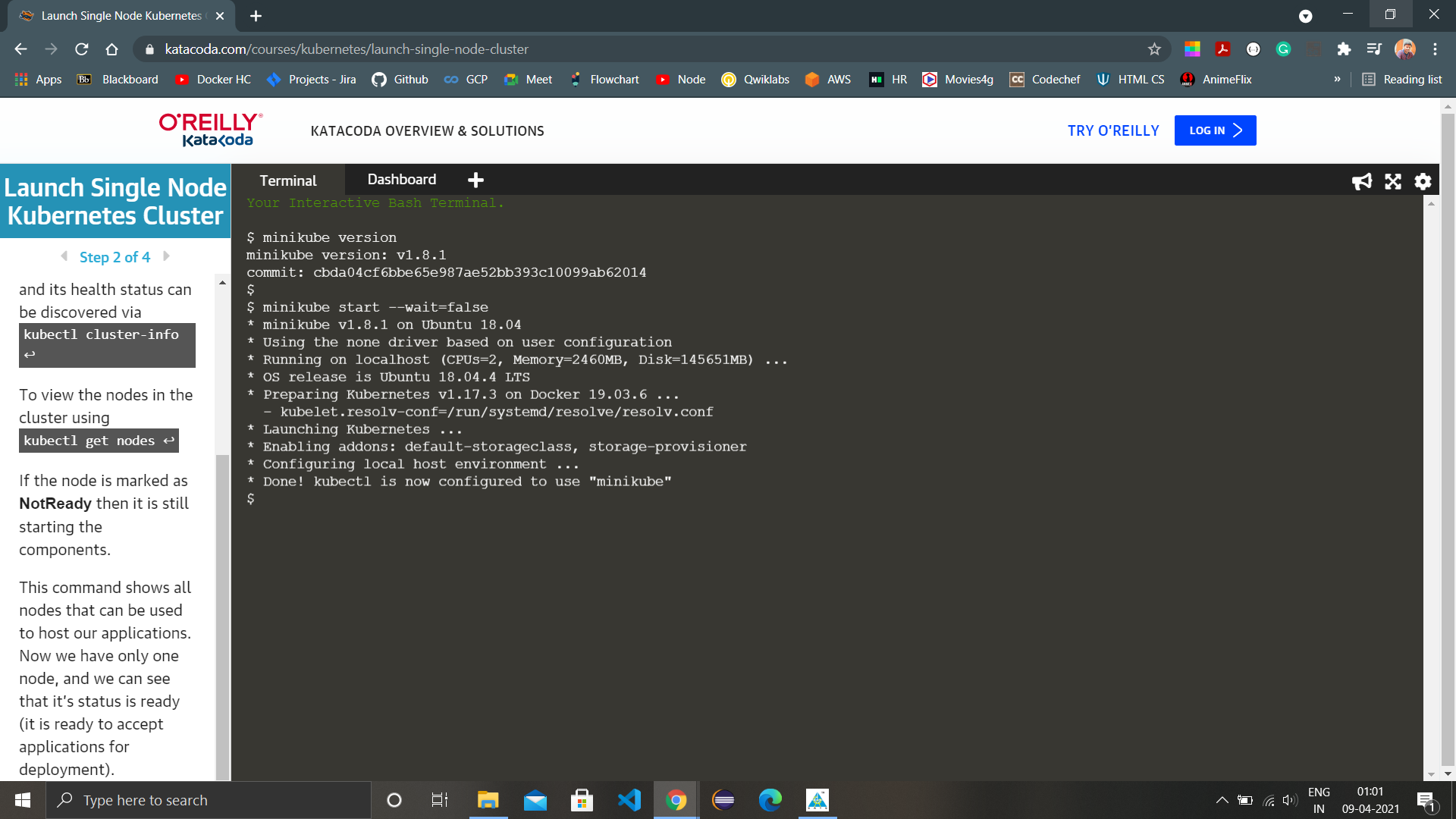
1. To scale a particular service, http deployment in this case, use the command ‘kubectl scale --replicas=3 deployment <deployment-name>’ which is http.



1. List all the pods using the command ‘kubectl get pods’.
2. After these alterations in the service, again describe the service to see the changes.
3. Finally test the accessing again using the curl command.



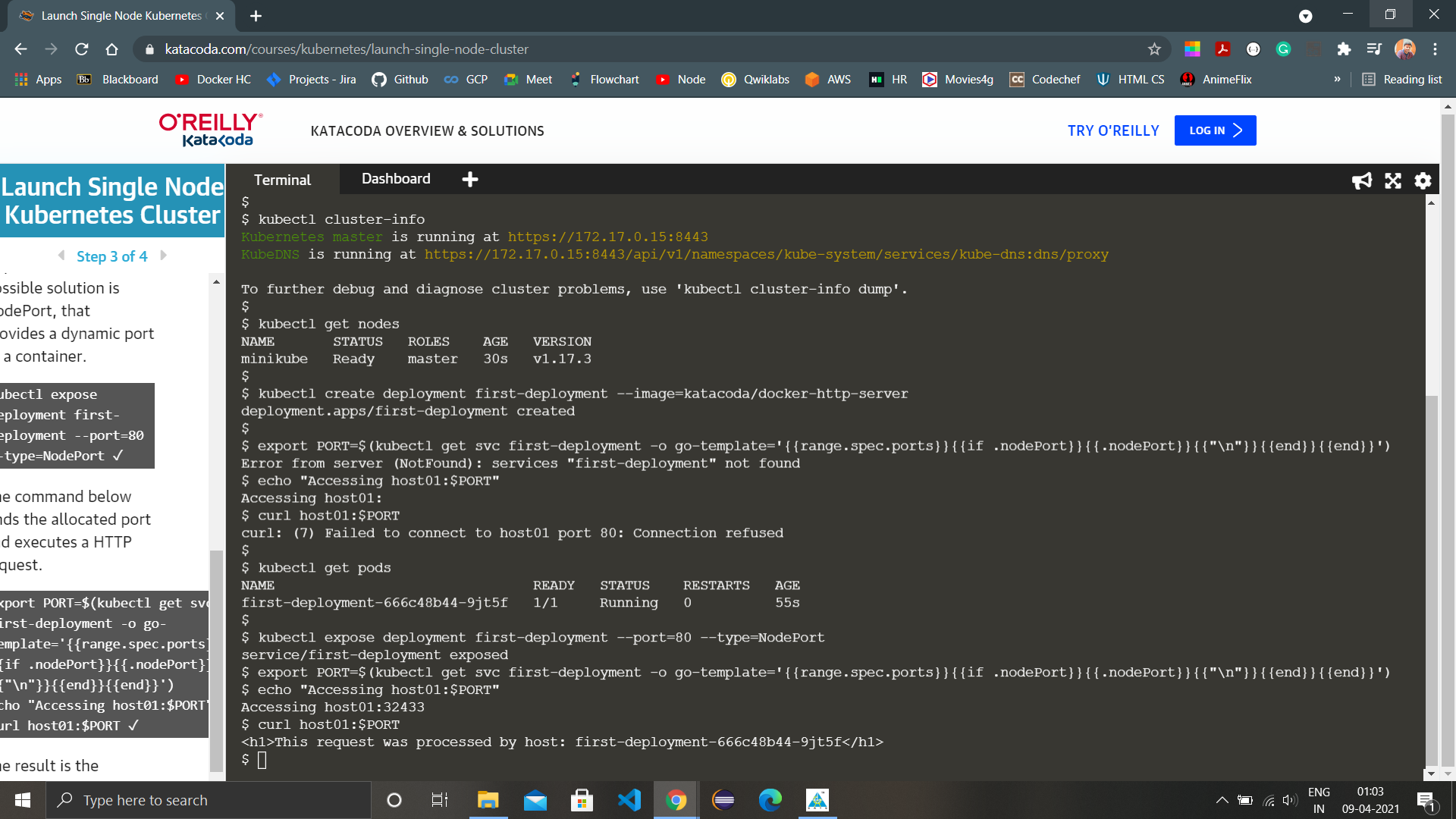
1. Launch a single node cluster
2. Check the version of minikube using the command ‘minikube version’.



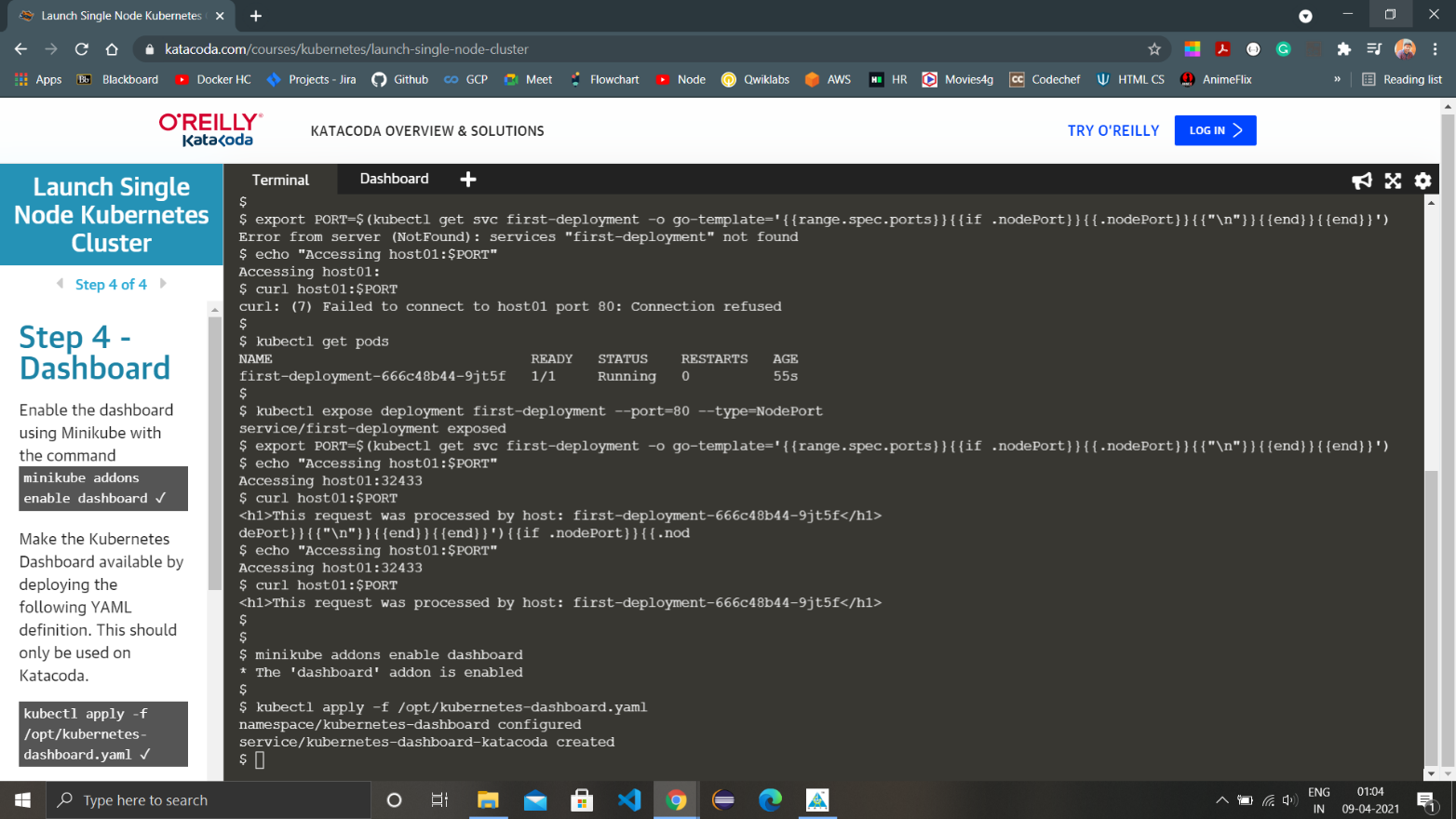
1. Start the cluster by executing the command ‘minikube start –wait=false’
2. To get the details and heath check results of a cluster, execute the command ‘kubectl cluster-info’.



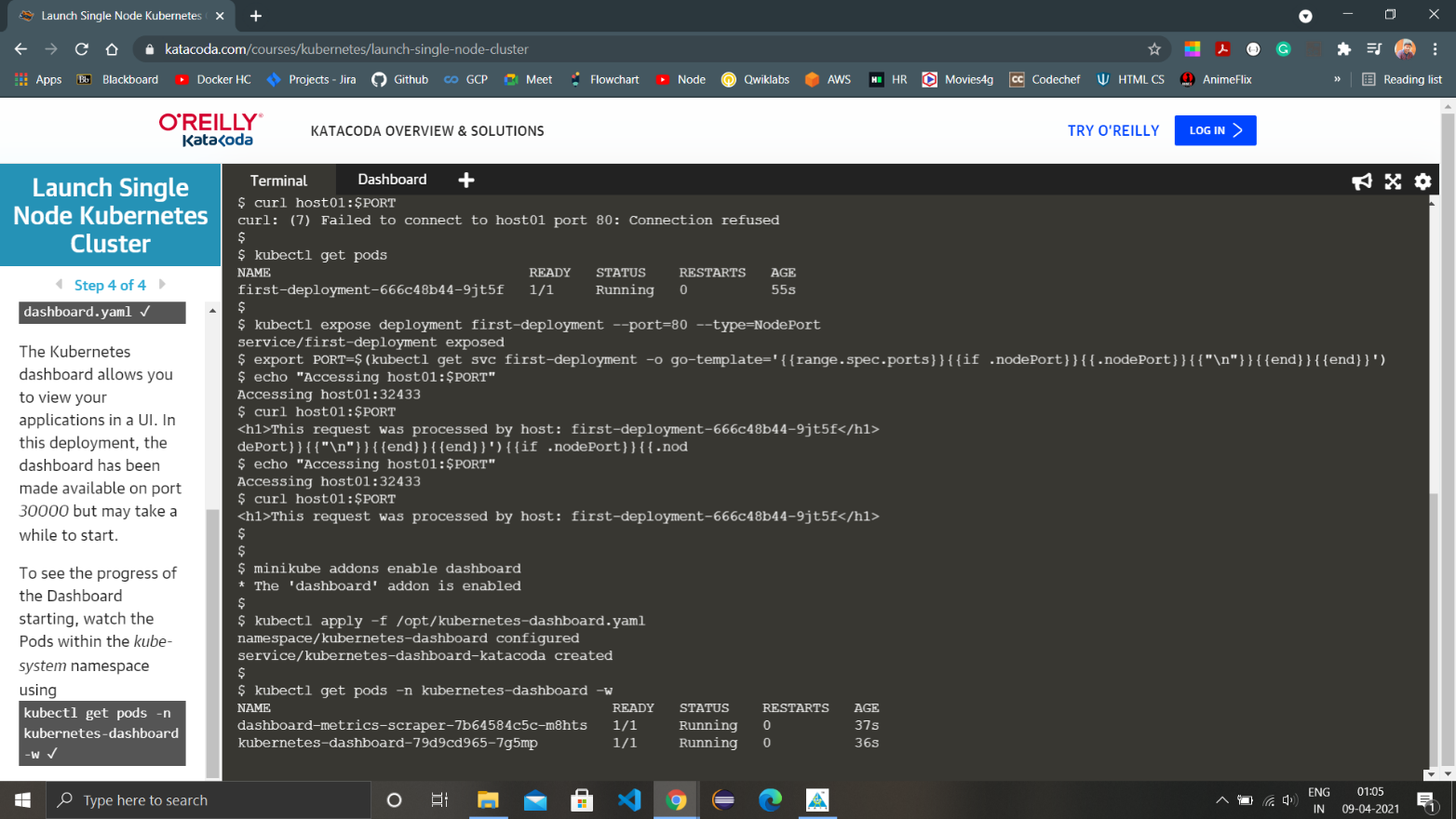
1. To list all the nodes in the cluster, execute the command ‘kubectl get nodes’.
2. Create a new deployment named first-deployment using the command ‘kubectl create deployment first-deployment --image=katacoda/docker-http-server’ and check the status using the command ‘kubectl get pods’.
3. Now expose the container using the command ‘kubectl expose deployment first-deployment --port=80 --type=NodePort’.
4. Find the allocated port using the command `export PORT=$(kubectl get svc first-deployment -o go-template='{{range.spec.ports}}{{if .nodePort}}{{.nodePort}}{{"\n"}}{{end}}{{end}}') echo "Accessing host01:$PORT"` and execute the HTTP request using ‘curl host01:$PORT’.



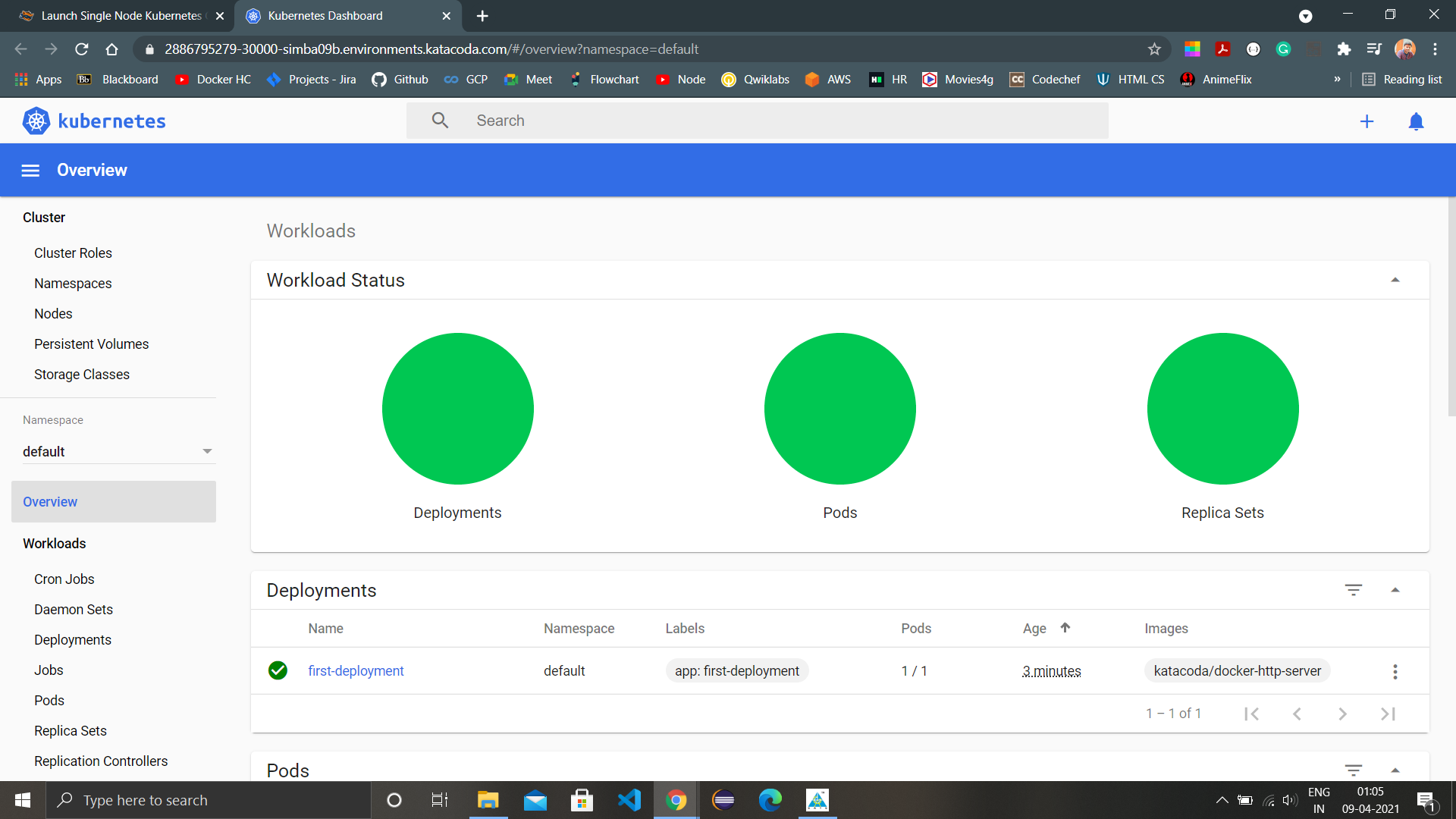
1. Now we have to enable the dashboard so that we can view all things related to our deployments there. To do that, execute the command ‘minikube addons enable dashboard’.



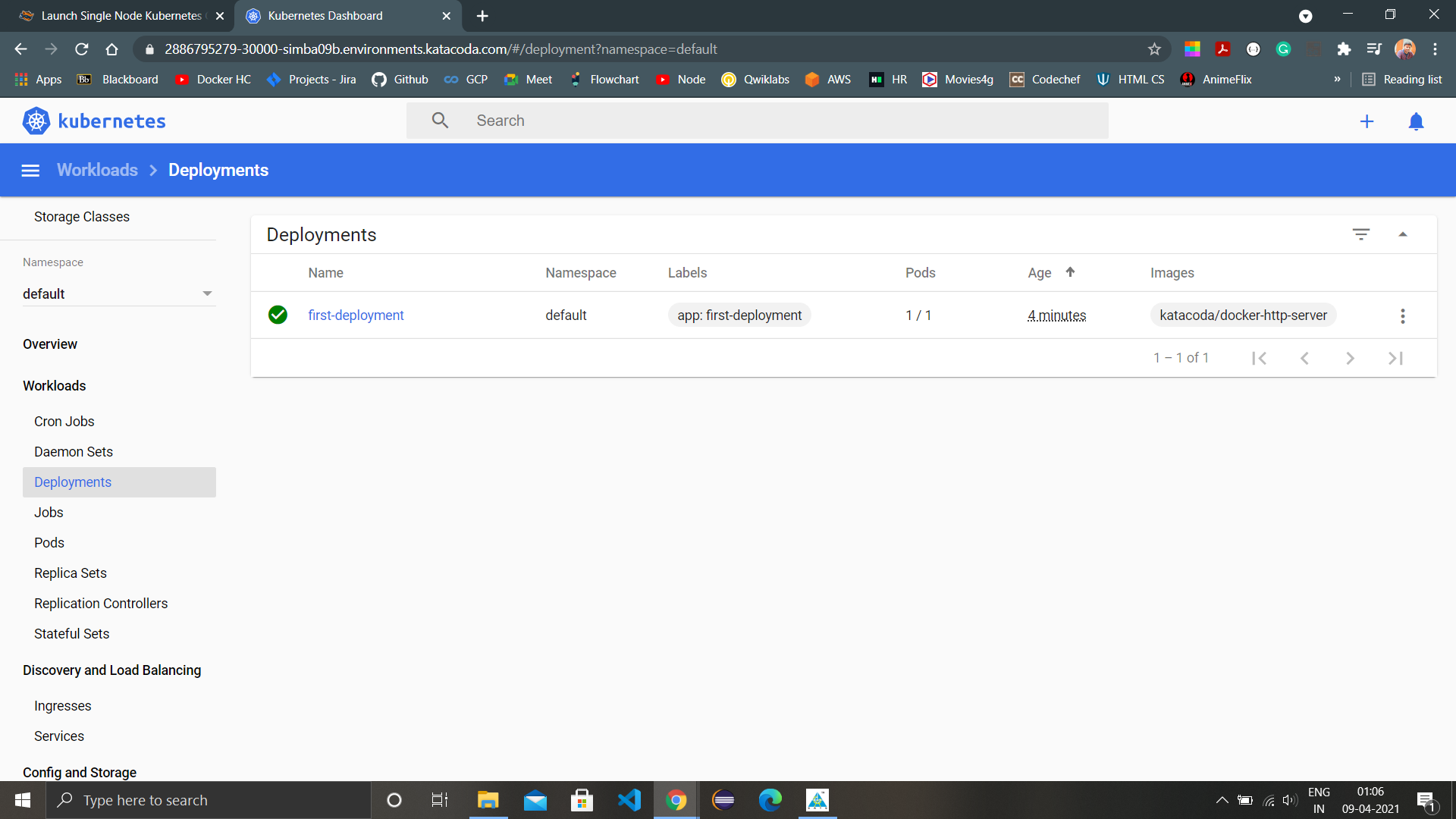
1. Apply the YAML file to complete dashboard availability using the command ‘kubectl apply -f /opt/kubernetes-dashboard.yaml’ Now, we just need to run the command ‘kubectl get pods -n kubernetes-dashboard -w’ and dashboard will start running.



1. Open  <https://2886795288-30000-simba09b.environments.katacoda.com/> on a new browser window and you will be headed to Kubernetes dashboard where we can see different clusters, workloads and much more.



1. In deployments we will have all the deployments listed, currently we are having only one.



1. Head back to the terminal and create a new deployment and again check deployments in the dashboard.

