Lab 3 Report



Name: Dawid Pionk

Document: Report for the third lab

Supervisor: Lei Shi

Submission Date: 18/05/25

Table of Contents

[1. Create a Kubernetes cluster Tutorial: Hello Minikube 3](#_Toc197028915)

[Creating a service 5](#_Toc197028916)

[Enable Addons 6](#_Toc197028917)

[Cleaning up 7](#_Toc197028918)

[2. Deploy an app Tutorial: Using kubectl to Create a Deployment 8](#_Toc197028919)

[kubectl basics 8](#_Toc197028920)

[Deploying an app 8](#_Toc197028921)

[Viewing the app 8](#_Toc197028922)

[3. Explore your app Tutorial: Viewing Pods and Nodes 9](#_Toc197028923)

[Check the application configuration 9](#_Toc197028924)

[Show the app in the terminal 10](#_Toc197028925)

[Executing commands on the container 10](#_Toc197028926)

[4. Expose your app publicly Tutorial: Using a Service to Expose Your App 11](#_Toc197028927)

[Step 1: Creating a new service 11](#_Toc197028928)

[Step 2: Using labels 12](#_Toc197028929)

[Step 3: Deleting a service 14](#_Toc197028930)

[5. Scale up your app Tutorial: Running Multiple Instances of Your App 15](#_Toc197028931)

[Scaling deployment 15](#_Toc197028932)

[Load balancing 16](#_Toc197028933)

[Scale down 17](#_Toc197028934)

[6. Update your app Tutorial: Performing a Rolling Update 18](#_Toc197028935)

[Update the version of the app 18](#_Toc197028936)

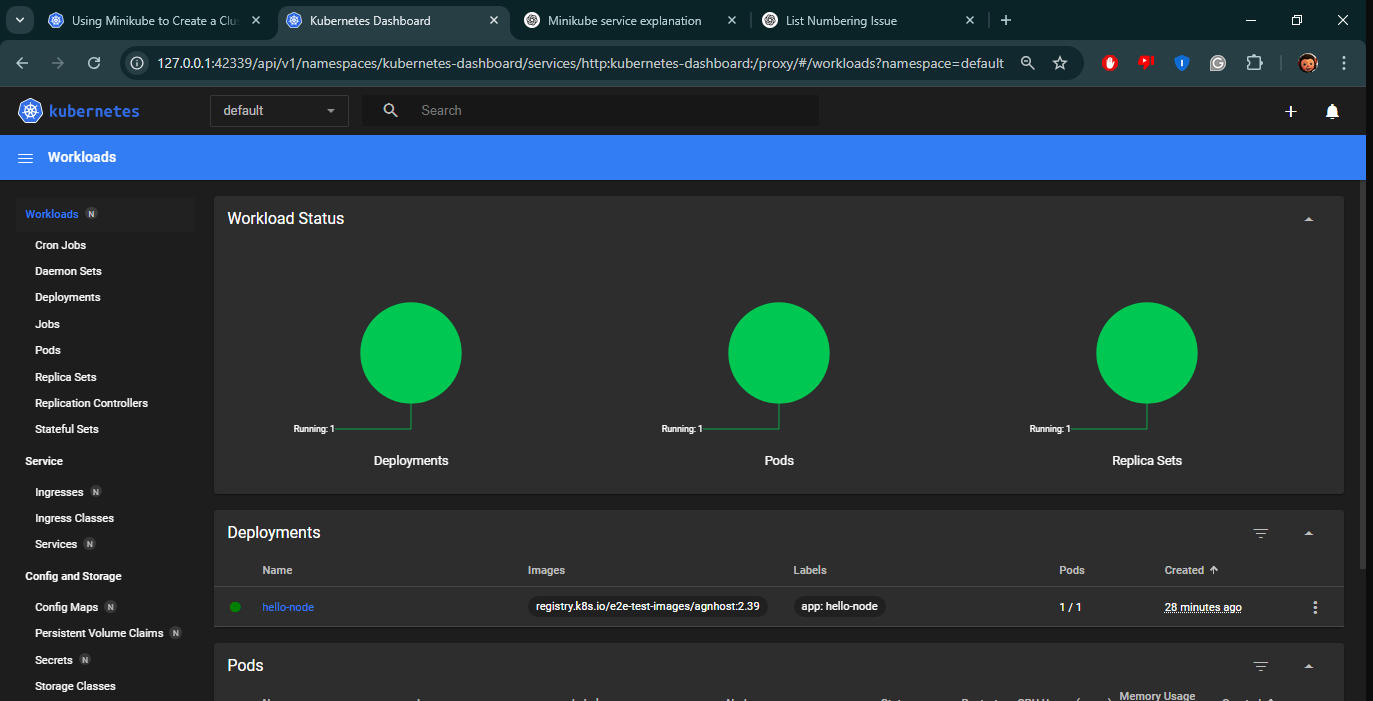
[Verify an update 19](#_Toc197028937)

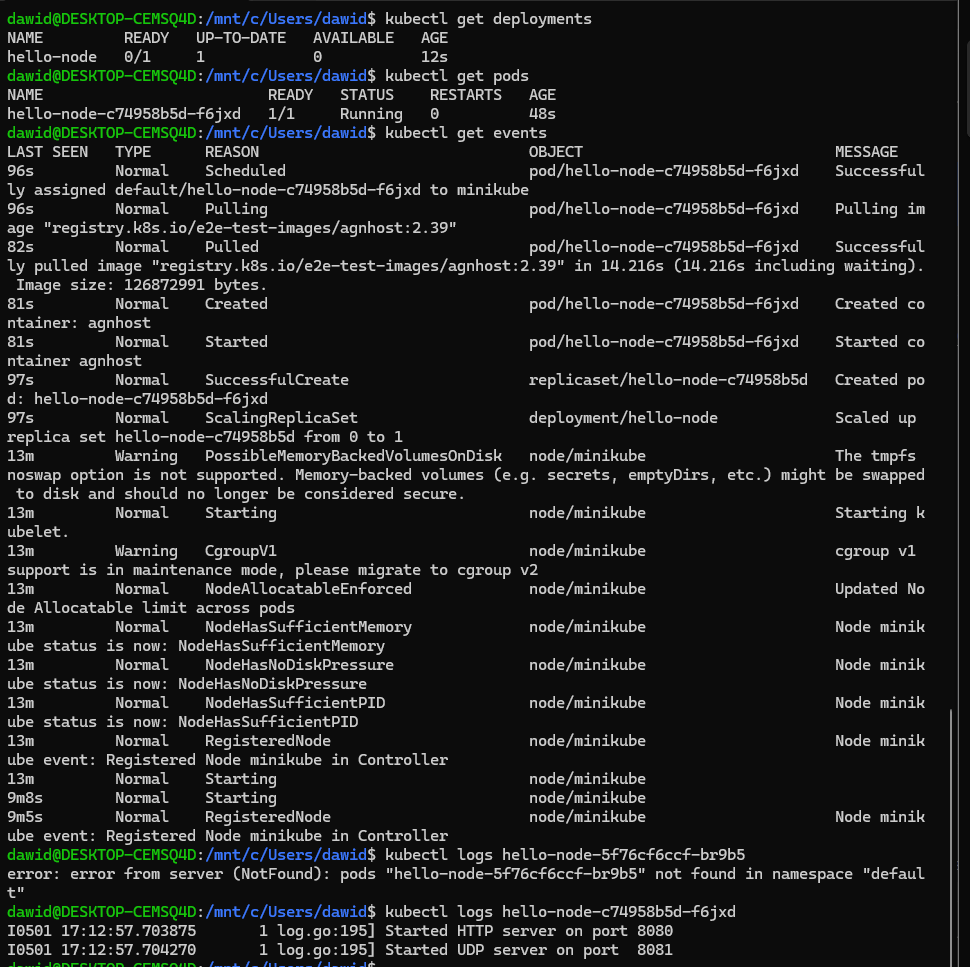
[Roll back an update 20](#_Toc197028938)

# 1. Create a Kubernetes cluster Tutorial: Hello Minikube

Getting started and creating a deployment

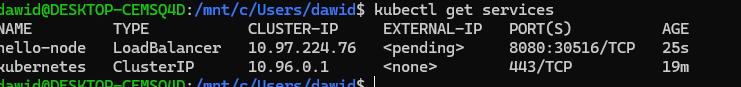
1. Downloaded and installed minikube on my Windows machine
2. Added minikube to my system path
3. Created a minikube cluster using:
   1. Minikube start
   2. Ran into an error:
      1. ❌ Exiting due to DRV\_NOT\_HEALTHY: Found driver(s) but none were healthy. See above for suggestions on how to fix installed drivers.
   3. Ran the command with admin privileges and ran into another errorExiting due to PR\_HYPERV\_MODULE\_NOT\_INSTALLED: Failed to start host: creating host: create: precreate: Hyper-V PowerShell Module is not available.
   4. Ran it in Linux instead, and it worked, most likely due to being connected to Docker.
4. Ran minikube dashboard to run the Kubernetes dashboard.
5. Started the deployment of a Kubernetes pod
   1. kubectl create deployment hello-node --image=registry.k8s.io/e2e-test-images/agnhost:2.39 -- /agnhost netexec --http-port=8080
6. View deployment using:
   1. kubectl get deployments
7. Viewed the pod using:
   1. kubectl get pods
8. Viewed cluster events using:
   1. kubectl get events
9. Viewed the kubectl configuration:
   1. kubectl config view
10. Viewed application logs for a container in my pod
    1. kubectl logs hello-node-c74958b5d-f6jxd
    2. hello-node-c74958b5d-f6jxd is the name of my pod

This is the minikube dashboard from step 4

Steps 6 to 10 – Creating and viewing the deployment

## Creating a service

1. Exposed the public to the public internet:
   1. kubectl expose deployment hello-node-- type=LoadBalancer-- port=8080
2. Viewed the services:
   1. kubectl get services

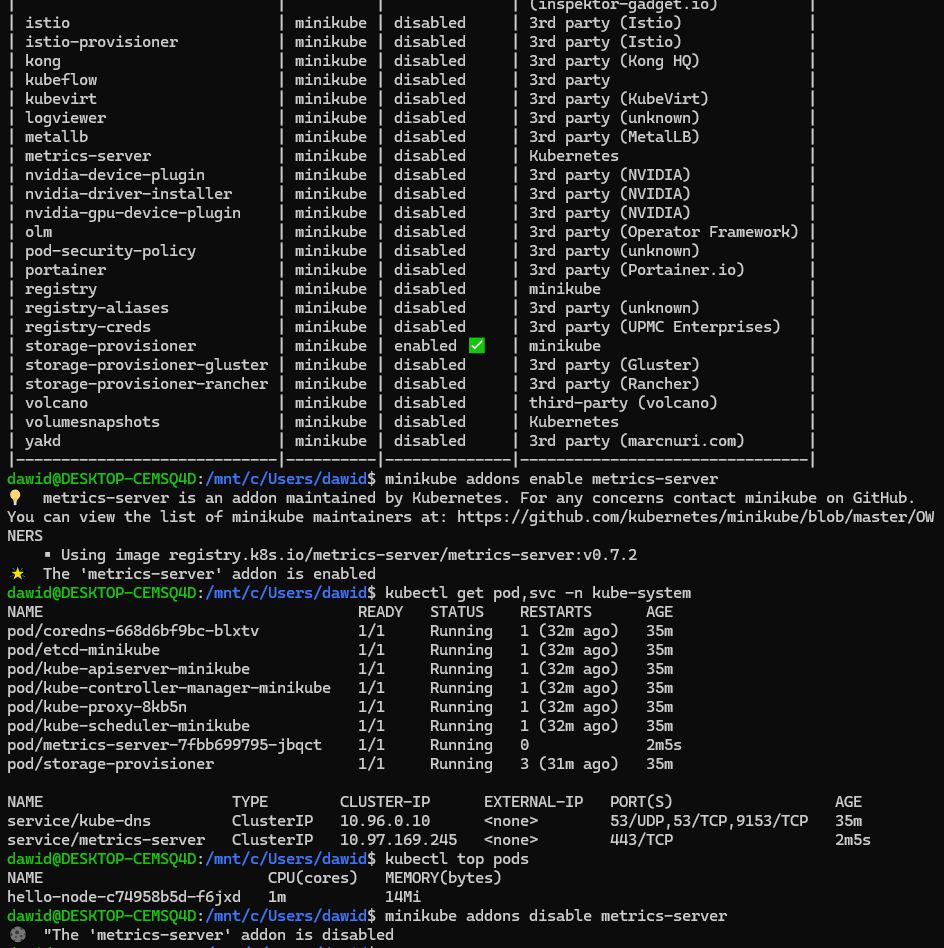
Here you can see the services that are running

1. Ran minikube service hello-node
   1. This is done to easily view the public address of the Hello node service well as making it available in your browser



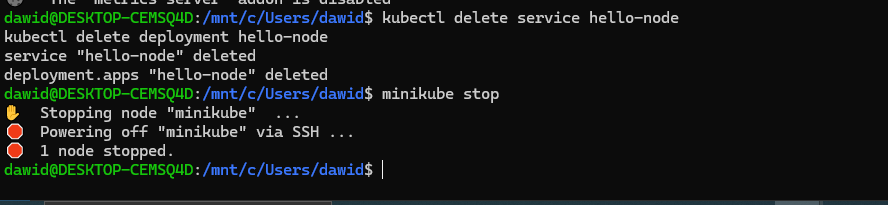
## Enable Addons

1. View the currently supported addons
2. Enabled an addon called metrics-saver
3. Viewed the pod and service for metrics-saver by installing the addon
4. Checked the output from the new addon
5. Disables the addon



## Cleaning up

1. Cleaned up the deployment and services I created during this tutorial.
2. Stopped the minikube cluster



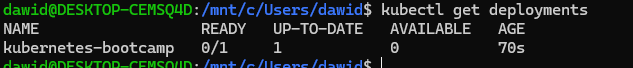
# 2. Deploy an app Tutorial: Using kubectl to Create a Deployment

## kubectl basics

1. Ran kubectl –help to see what commands I can use
2. Checked the version using kubectl version
3. Checked the nodes in a cluster using kubectl get nodes

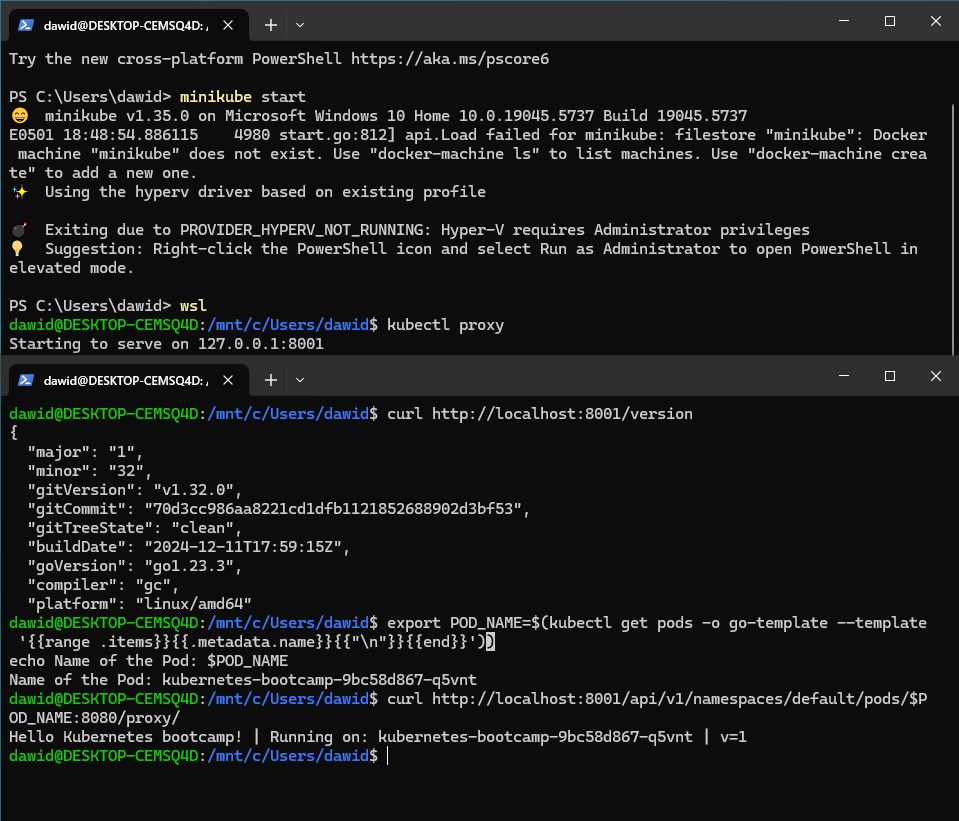
## Deploying an app

1. Deployed an app on Kubernetes using:
   1. kubectl create deployment kubernetes-bootcamp-- image=gcr.io/google-samples/kubernetes-bootcamp:v1
2. Listed my deployments
   1. kubectl get deployments

Here you can see the app I deployed

## Viewing the app

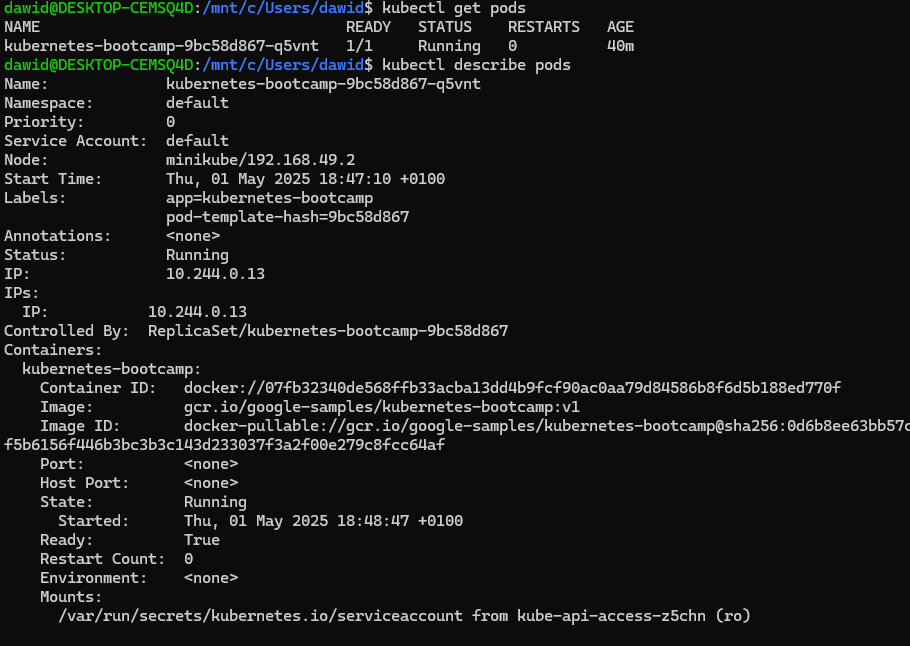
1. Created a proxy so that I could communicate with the Kubernetes cluster through my terminal
2. Ran curl <http://localhost:8001/version>
   1. To view all the APIs hosted through the proxy endpoint.
3. Saved the name of the pod in a variable called $POD\_NAME
4. Accessed the pod through he proxied API

All the commands ran in this exercise

# 3. Explore your app Tutorial: Viewing Pods and Nodes

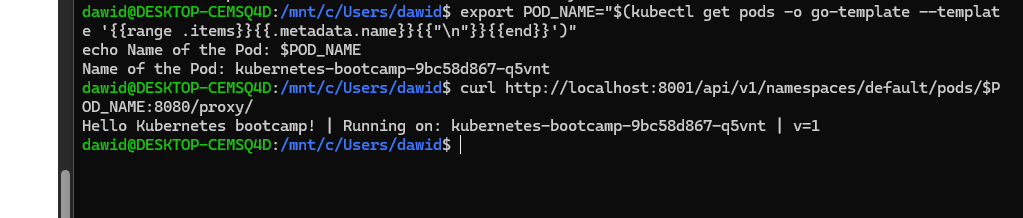
## Check the application configuration

1. Used kubectl get pods to check what pods are deployed
2. Ran kubectl describe pods to see details about the kubernetes-bootcamp-9bc58d867-q5vnt pod



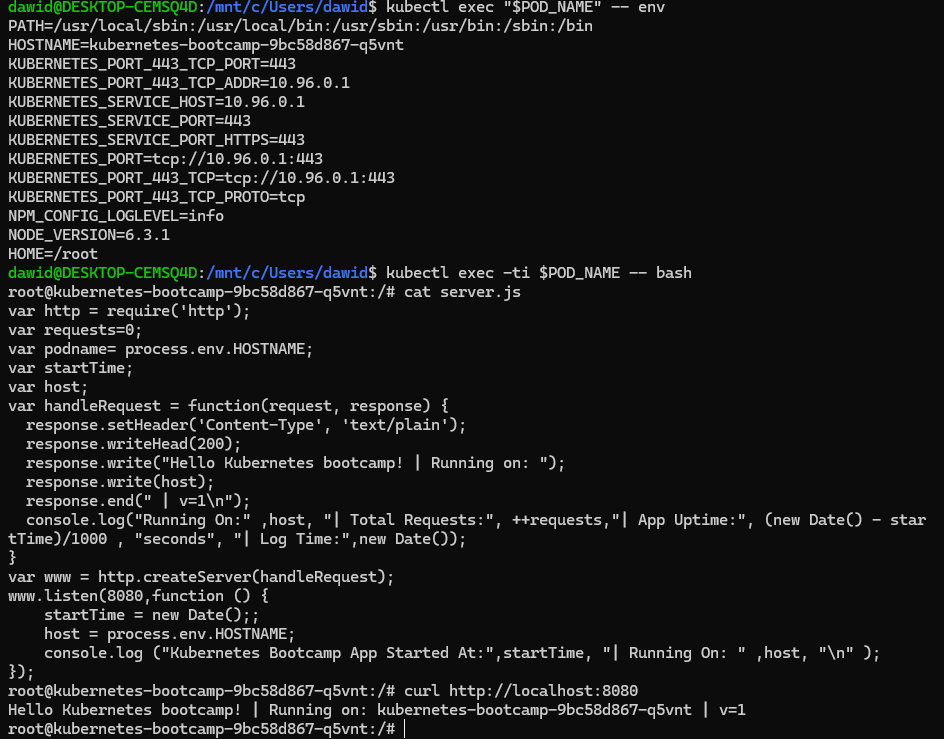
## Show the app in the terminal

1. Set up kubectl proxy
2. Assigned a pod name variable
3. Ran curl <http://localhost:8001/api/v1/namespaces/default/pods/$POD_NAME:8080/proxy/>
4. Note: These are the same steps as from a couple of sections before



## Executing commands on the container

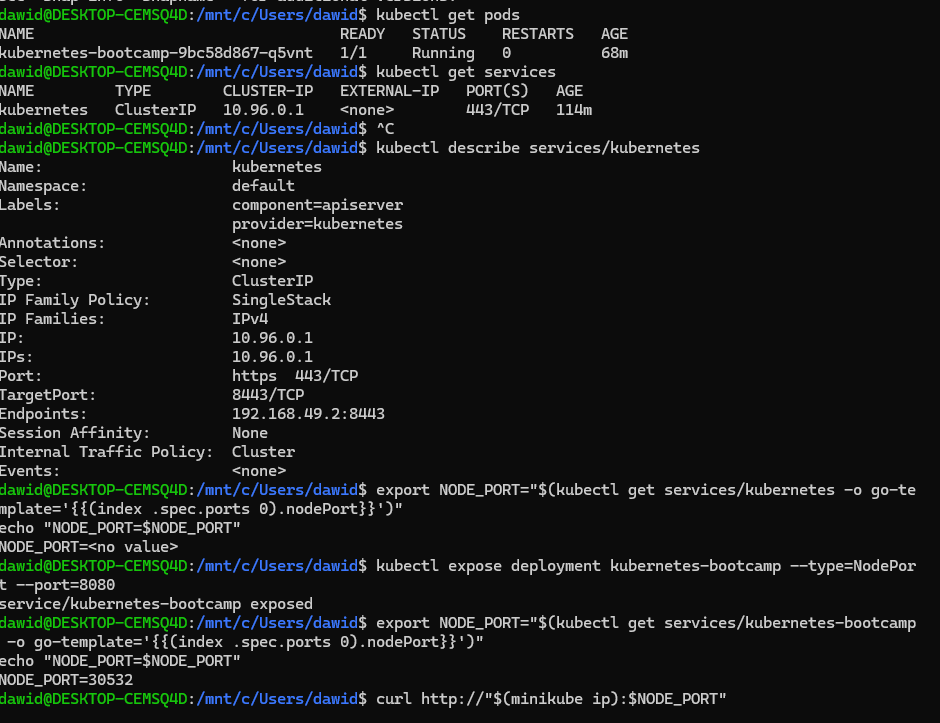
1. Used an exec command on the pod\_name variable, which showed the environmental variables for my pod.
2. Started a bash session in the pods container
3. Viewed the contents of the server.js file
4. Checked if the application was running using curl <http://localhost:8080>

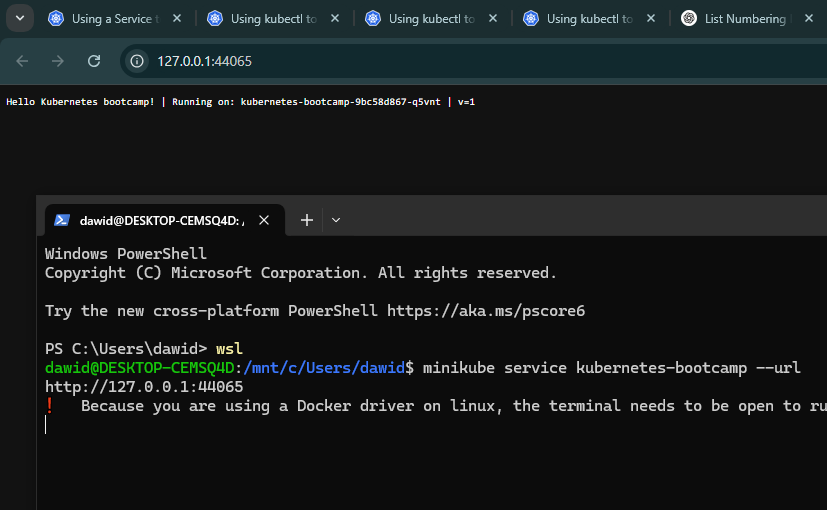


# 4. Expose your app publicly Tutorial: Using a Service to Expose Your App

### Step 1: Creating a new service

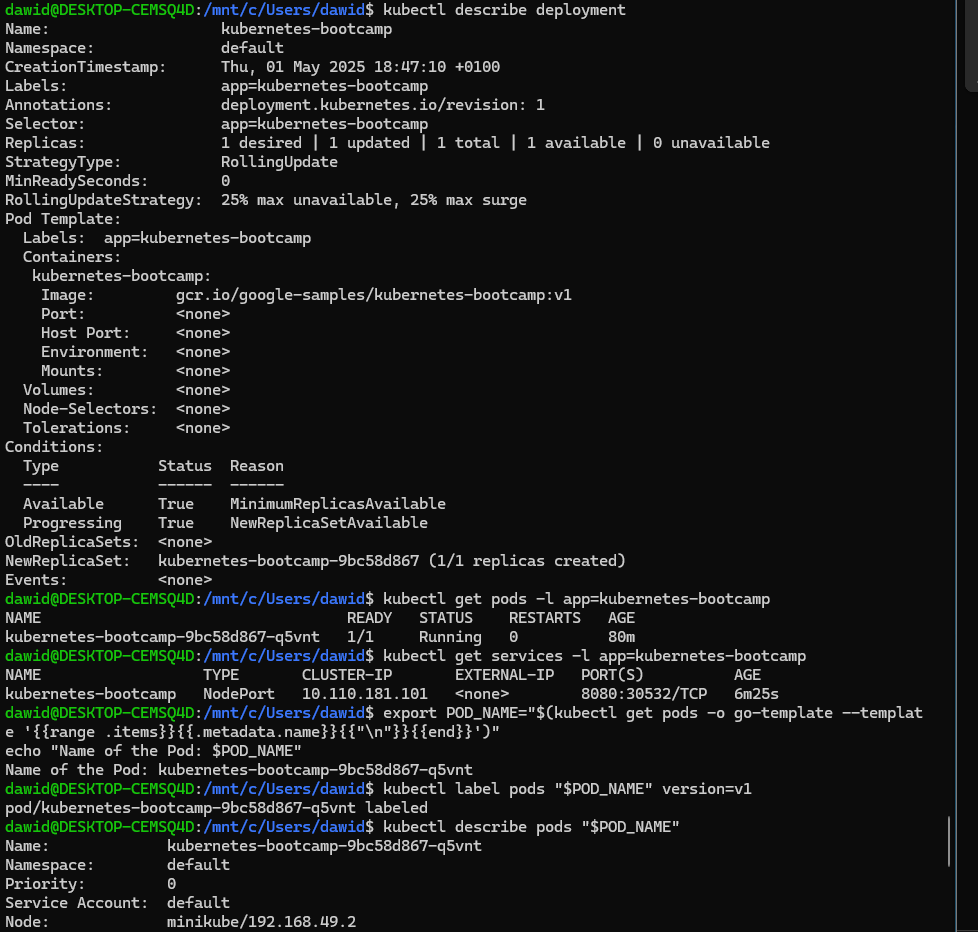
1. Checked if my application was running using kubectl get pods
2. Checked services from my cluster using kubectl get services
3. Ran a describe on the service didn’t have a Kubernetes-bootcamp service, but I used one called just Kubernetes that was already running
   1. Didn’t work since it has no node-port value
   2. Asked ChatGPT to create a service for Kubertes-bootcamp, it gave the following code:kubectl expose deployment kubernetes-bootcamp-- type=NodePort-- port=8080
4. Assigned a variable node port equal to the bootcamp node port value.
5. Ran curl http://"**$(**minikube ip**)**:$NODE\_PORT" to test the app.
6. Since I'm using minikube with Docker Desktop, I needed to perform extra steps.
7. Ran minikube service kubernetes-bootcamp –url on a separate terminal

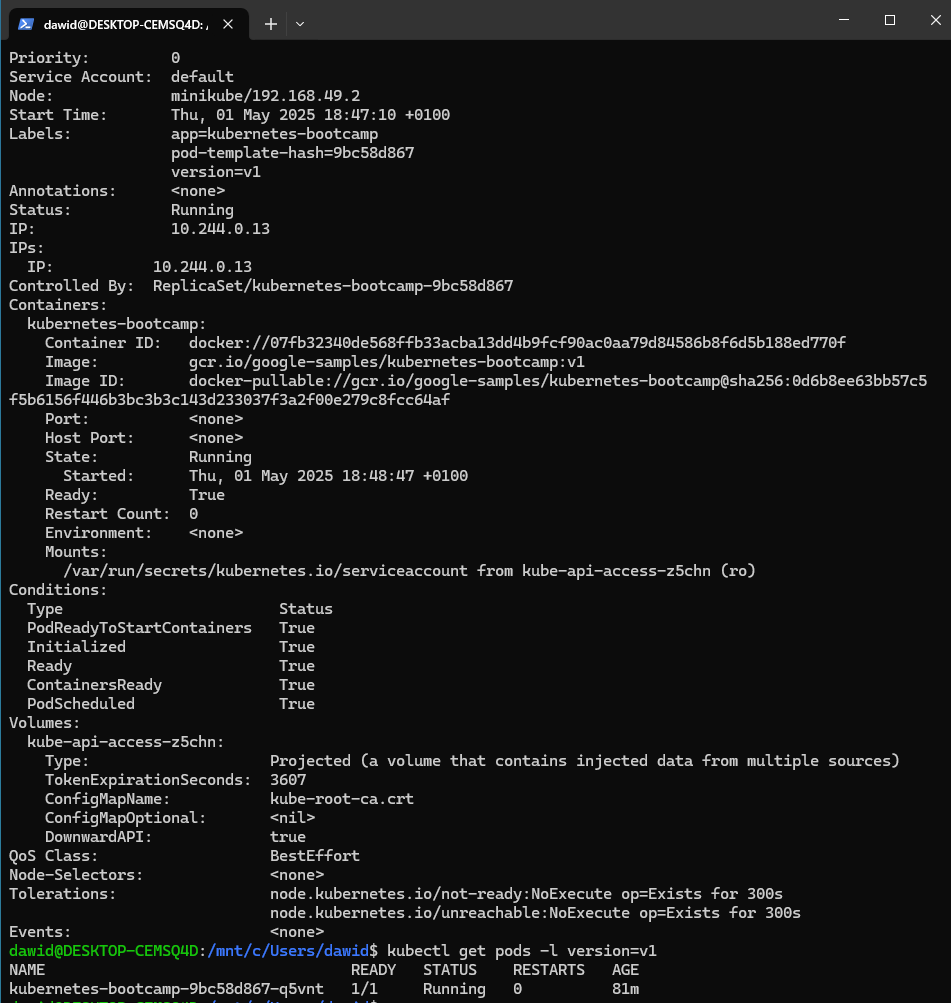




### Step 2: Using labels

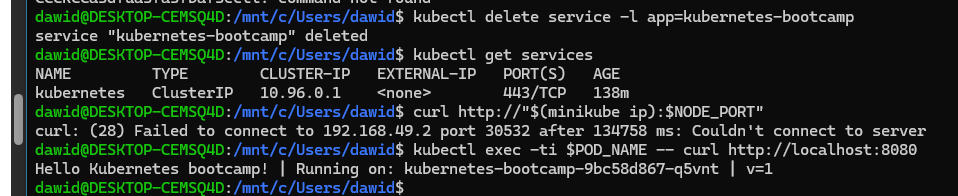
1. Got details about the deployment using:
   1. kubectl describe deployment
   2. Deployments automatically assign labels for pods
2. Ran kubectl get pods -l app=kubernetes-bootcamp
   1. This gets pods with a filter for pods with a Kubernetes-bootcamp label.
3. Ran kubectl get services -l app=kubernetes-bootcamp
   1. This does the same as above, except for services.
4. Got the pod's name and saved it in a pod\_name environmental variable.
5. Used kubectl label pods "$POD\_NAME" version=v1 to apply a new label
6. Viewed details about the pod using kubectl describe pods "$POD\_NAME"
7. kubectl get pods -l version=v1 works now since we have a pod with this label





### Step 3: Deleting a service

1. Removed a service using:
   1. kubectl delete service -l app=kubernetes-bootcamp
2. Verified service was gone using kubectl get services
3. Verified that the route is no longer exposed using:
   1. curl http://"**$(**minikube ip**)**:$NODE\_PORT"
   2. As well as the minikube service command from step 1
4. Verified that the app is running but not reachable from outside the cluster

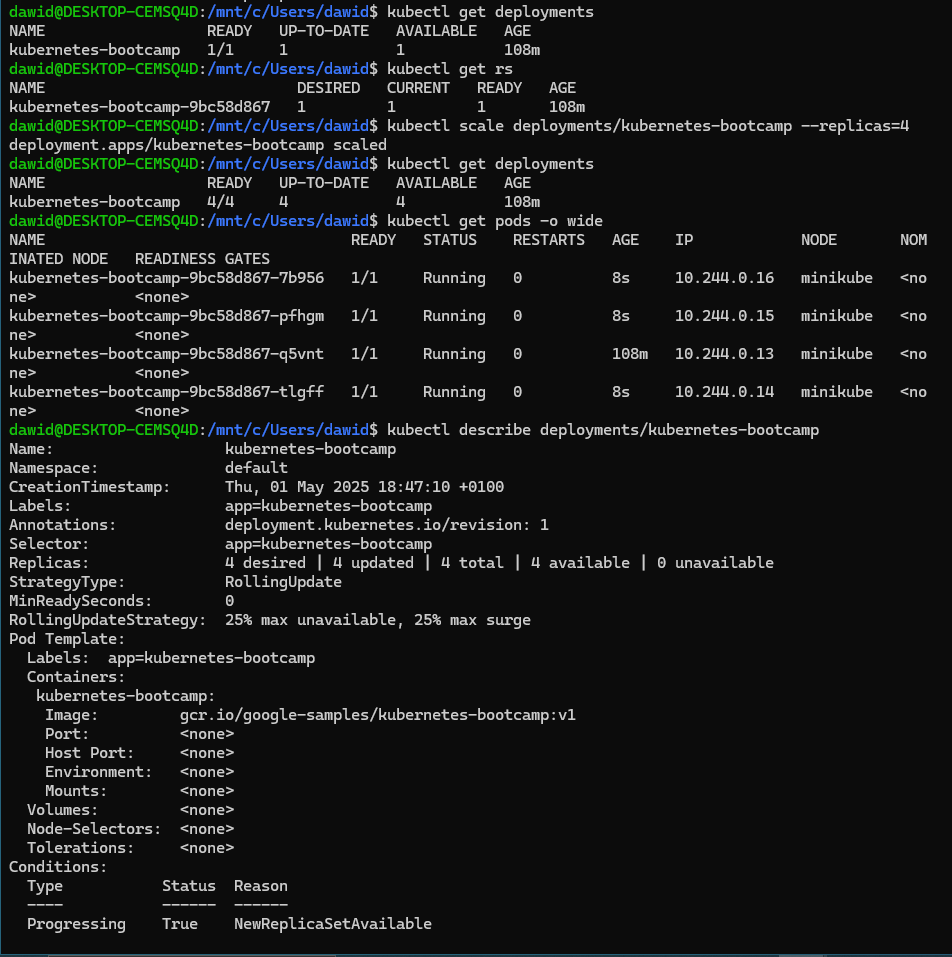


# 5. Scale up your app Tutorial: Running Multiple Instances of Your App

1. Set up the bootcamp again since it was deleted in the previous section
   * kubectl expose deployment/kubernetes-bootcamp-- type="LoadBalancer"-- port 8080

## Scaling deployment

1. Ran kubectl get deployments to check the deployments
2. Ran kubectl get rs
   1. This was done to see the ReplicaSet run by the deployment. ReplicaSet is a controller that makes sure a certain number of identical pods are running at once.
3. Scaled the deployment to 4 replicas:
   1. kubectl scale deployments/kubernetes-bootcamp-- replicas=4
4. Checked deployments again using kubectl get deployments, there are 4 replicas available instead of 1.
5. Checked if the number of pods changed using
   1. kubectl get pods -o wide
6. Checked if the change was applied using the describe command.
   1. kubectl describe deployments/kubernetes-bootcamp



## Load balancing

1. Ran kubectl describe services/kubernetes-bootcamp
   1. To check exposed IP and PORT, like before
2. Assigned a node port variable to the service node port
   1. . export NODE\_PORT="**$(**kubectl get services/kubernetes-bootcamp -o go-template='{{(index. .spec.ports 0) nodePort}}'**)**"
   2. echo NODE\_PORT=$NODE\_PORT -> to check
3. Ran the command multiple times using the following commands:
   1. minikube service kubernetes-bootcamp –url
   2. curl http://127.0.0.1:45409

A screenshot of a computer

AI-generated content may be incorrect.

## Scale down

1. Scaled down the number of deployment replicas from 4 to 2
   1. kubectl scale deployments/kubernetes-bootcamp-- replicas=2
2. Checked deployments
3. Checked the number of pods, and 2 were terminating

A screenshot of a computer program

AI-generated content may be incorrect.

# 6. Update your app Tutorial: Performing a Rolling Update

## Update the version of the app

1. Got the list of deployments
2. Got the list of pods
3. Viewed the image version of the app by looking at the Image field after running:
   1. kubectl describe pods
4. Ran kubectl set image deployments/kubernetes-bootcamp kubernetes-bootcamp=docker.io/jocatalin/kubernetes-bootcamp:v2
   1. This told the deployment to use a different image of the app.
   2. Essentially, image in this context means version
5. Checked the list of pods again to see if the above command executed properly.

A screenshot of a computer screen

AI-generated content may be incorrect.

A screen shot of a computer program

AI-generated content may be incorrect.

## Verify an update

1. Checked if the service was up and running using:
   1. kubectl expose deployment/kubernetes-bootcamp --type="NodePort" --port 8080
   2. The service was already up.
2. Set the node port variable like in the previous sections
3. Ran curl to check execution, and the v=2, which means the image update worked.
4. Confirmed the update rollout status using
   1. kubectl rollout status deployments/kubernetes-bootcamp
5. Checked the image field in the pods to see if version 2 was applied.

A screenshot of a computer screen

AI-generated content may be incorrect.



## Roll back an update

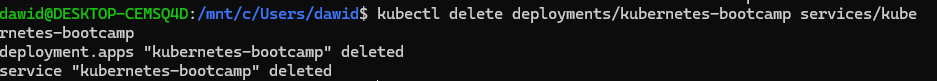
1. Performed another update using:
   1. kubectl set image deployments/kubernetes-bootcamp kubernetes-bootcamp=gcr.io/google-samples/kubernetes-bootcamp:v10
2. Checked deployments to see the status of the deployments
3. Checked the pods to see that a pod has an ErrImagePull status
4. Ran describe on pods to see more on this statusIn the events section, v10 had an error.
5. Rolled back the deployment with:
   1. kubectl rollout undo deployments/kubernetes-bootcamp
6. List all the pods again
7. Ran describe on pods to see if the deployment is back to v2

A computer screen with white text

AI-generated content may be incorrect.

A screen shot of a computer

AI-generated content may be incorrect.

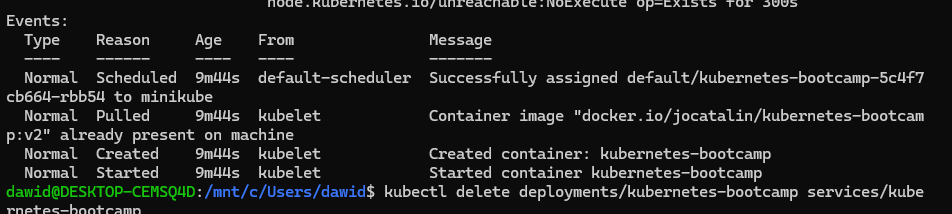




A computer screen shot of a computer program

AI-generated content may be incorrect.Pre-rollback



Post rollback