**Kubernetes Lab Exercises Submission**

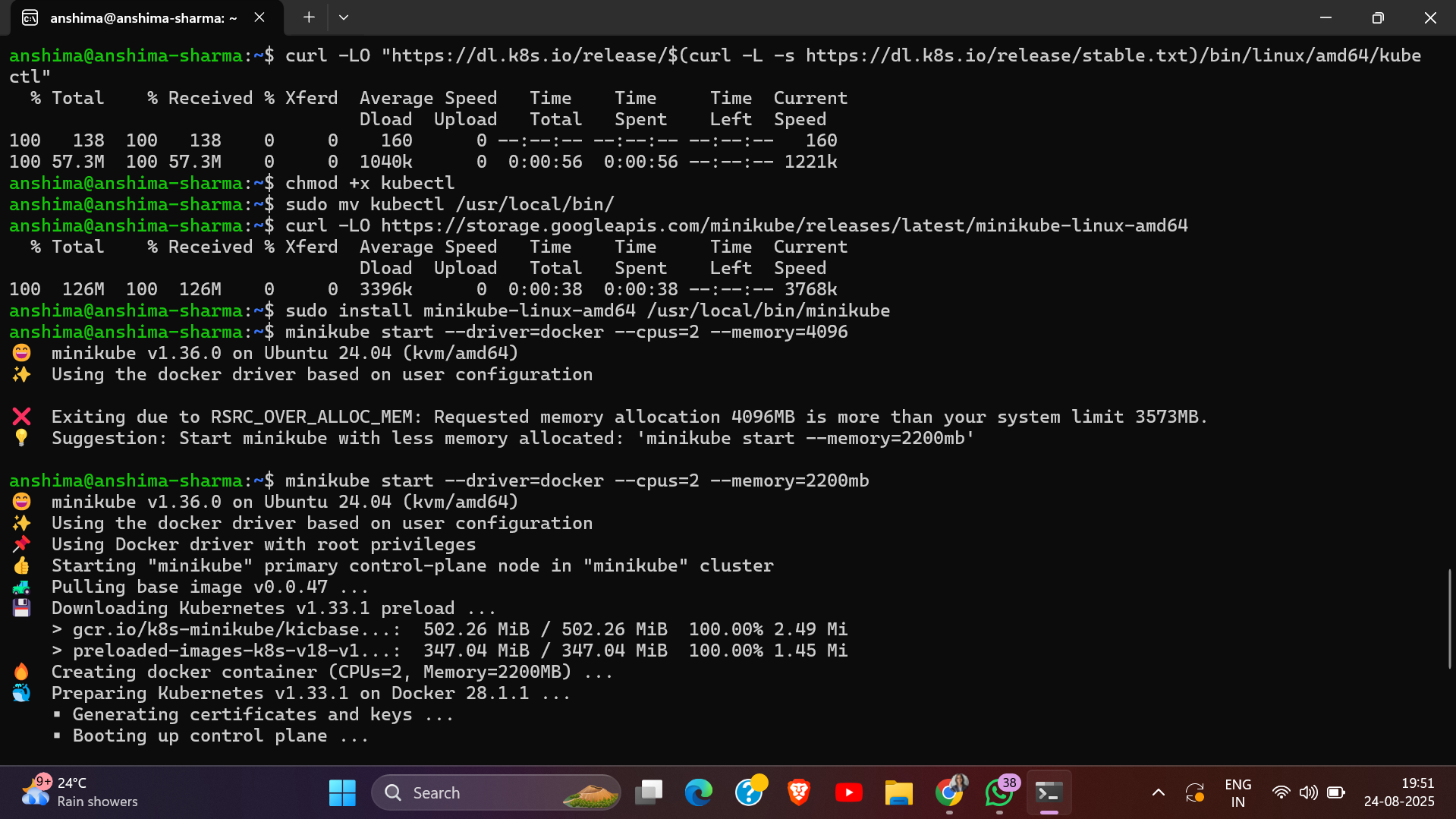
# Exercise 1: Setting Up Your Kubernetes Cluster

* Install Minikube and kubectl.
* Start a Minikube cluster with minikube start.
* Use kubectl cluster-info to verify your cluster is running.
* List all nodes using kubectl get nodes.

## Commands Used

* curl -LO [https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl](https://dl.k8s.io/release/$(curl%20-L%20-s%20https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl)
* curl -LO <https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64>
* minikube start --driver=docker --cpus=2 --memory=220mb
* minikube status
* kubectl cluster-info
* kubectl get nodes

## Screenshots / Output Evidence



## 

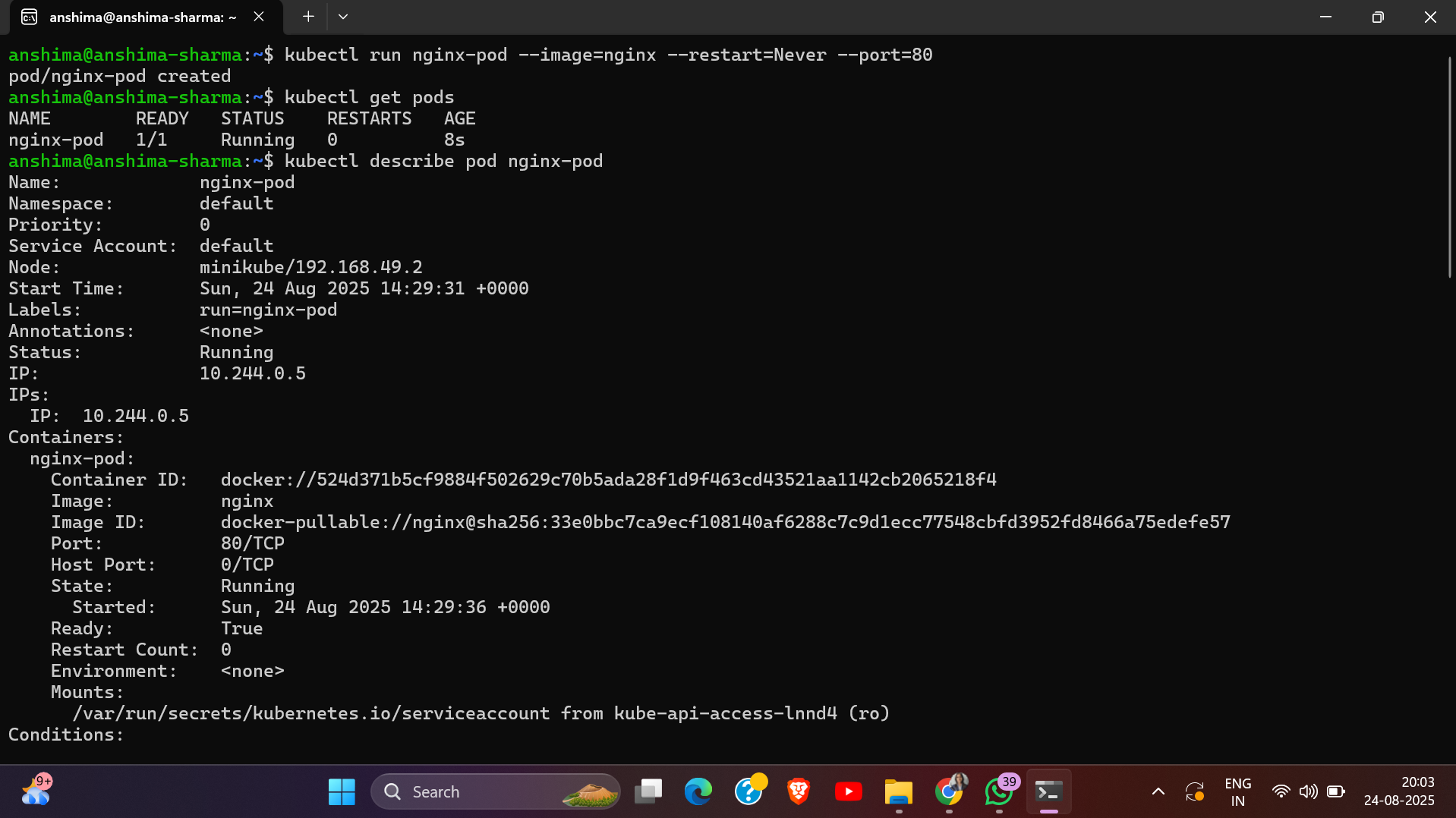
# Exercise 2: Creating and Managing Pods

* Create a simple pod using a predefined image like nginx
* Check the pod status.
* View pod logs
* Expose the pod via a service

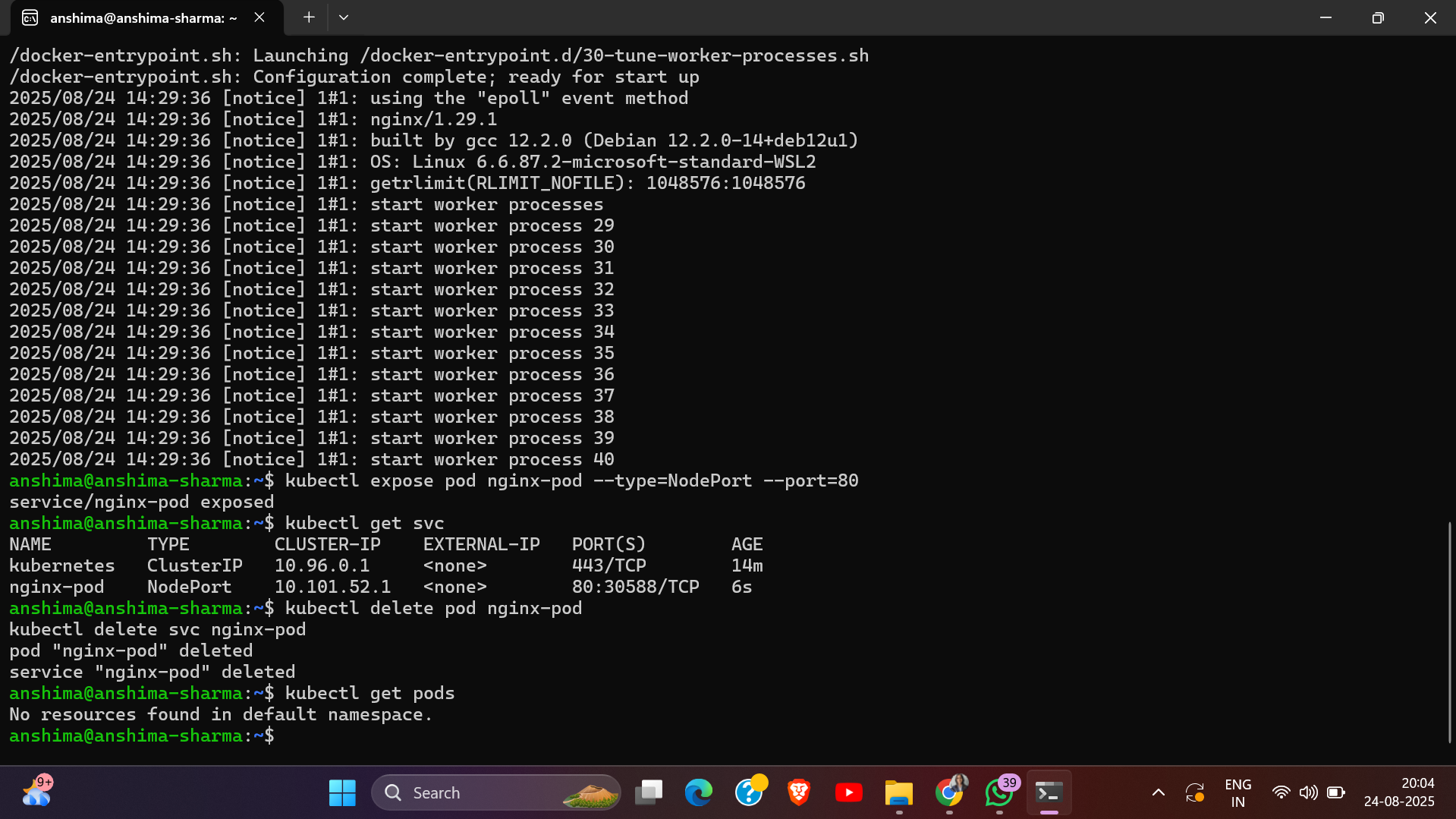
## Commands Used

* kubectl run nginx-pod --image=nginx --restart=Never --port=80
* kubectl get pods
* kubectl describe pod nginx-pod
* kubectl logs nginx-pod
* kubectl expose pod nginx-pod --type=NodePort --port=80
* kubectl get svc
* kubectl delete pod nginx-pod
* kubectl get pods

## Screenshots / Output Evidence







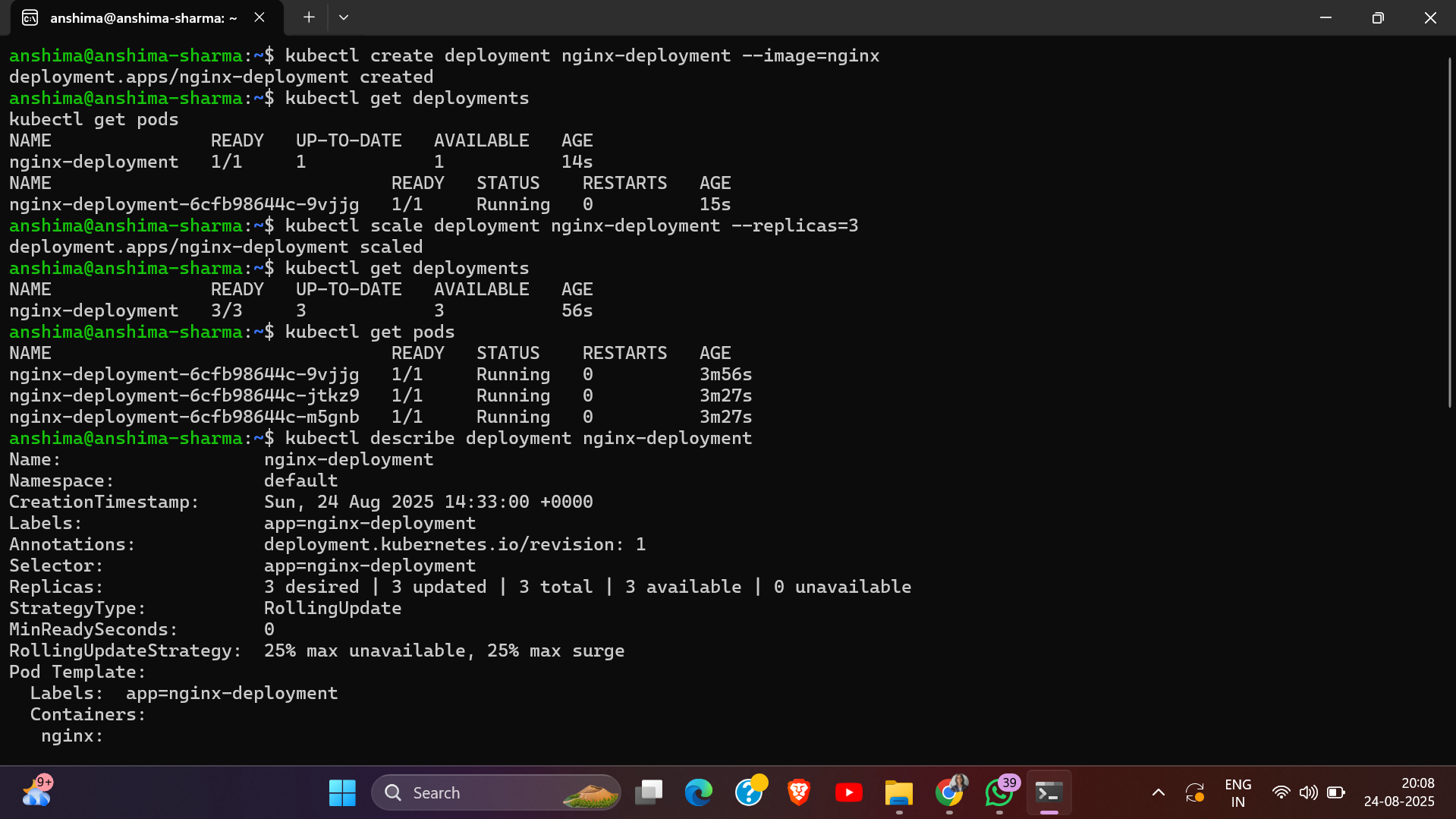
# Exercise 3: Working with Deployments

* Create a deployment with using nginx image
* Scale the deployment to 3 replicas
* Verify the deployment
* Update the deployment by changing the image(imperative way)

## Commands Used

* kubectl create deployment nginx-deployment --image=nginx
* kubectl get deployments
* kubectl scale deployment nginx-deployment --replicas=3
* kubectl get deployments
* kubectl get pods
* kubectl describe deployment nginx-deployment
* kubectl set image deployment/nginx-deployment nginx=nginx:1.19
* kubectl rollout history deployment/nginx-deployment
* kubectl rollout undo deployment/nginx-deployment

## Screenshots / Output Evidence





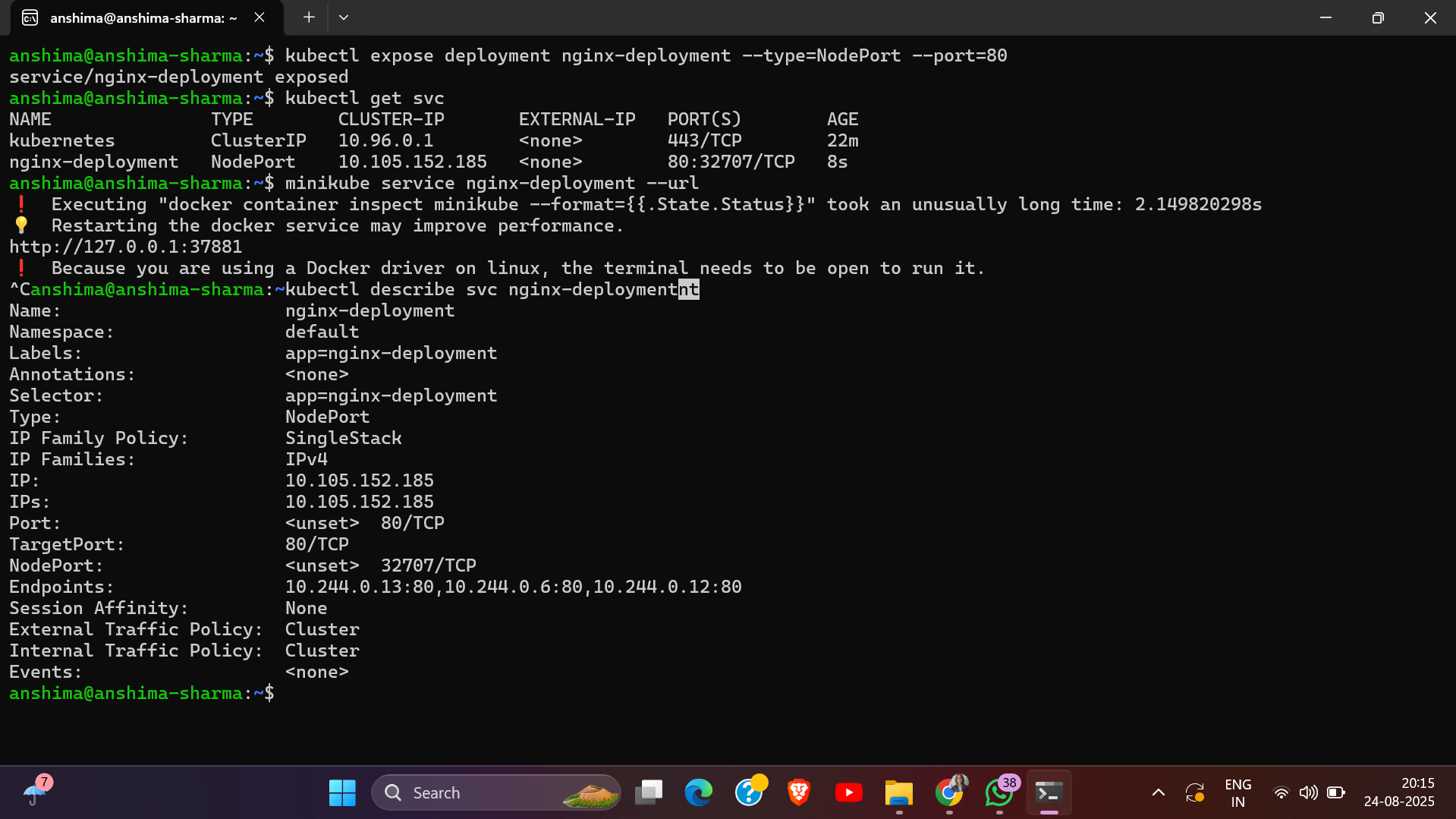
# Exercise 4: Services and Networking

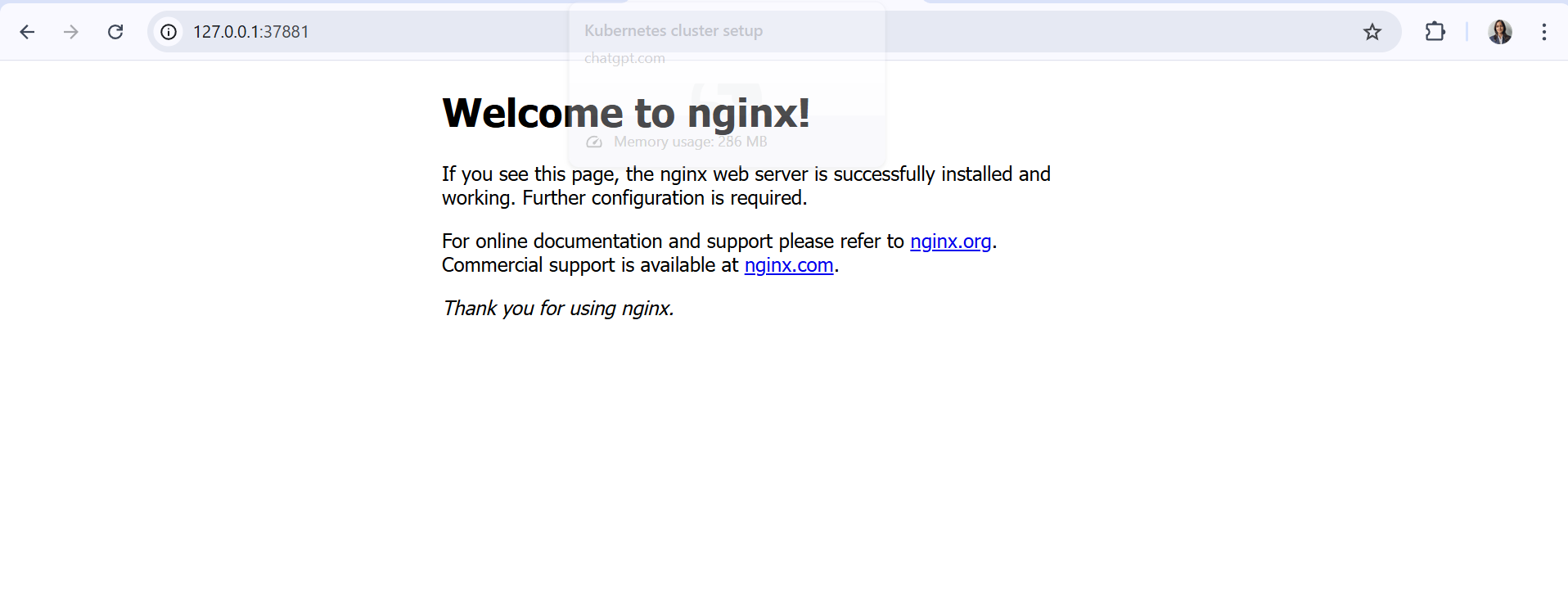
* Expose your nginx deployment using a Service:
* Create a service of type NodePort to make it accessible externally.
* View the service details.
* Test access to the nginx service through the browser.

## Commands Used

* kubectl expose deployment nginx-deployment --type=NodePort --port=80
* kubectl get svc
* minikube service nginx-deployment –url
* kubectl describe svc nginx-deployment

## Screenshots / Output Evidence





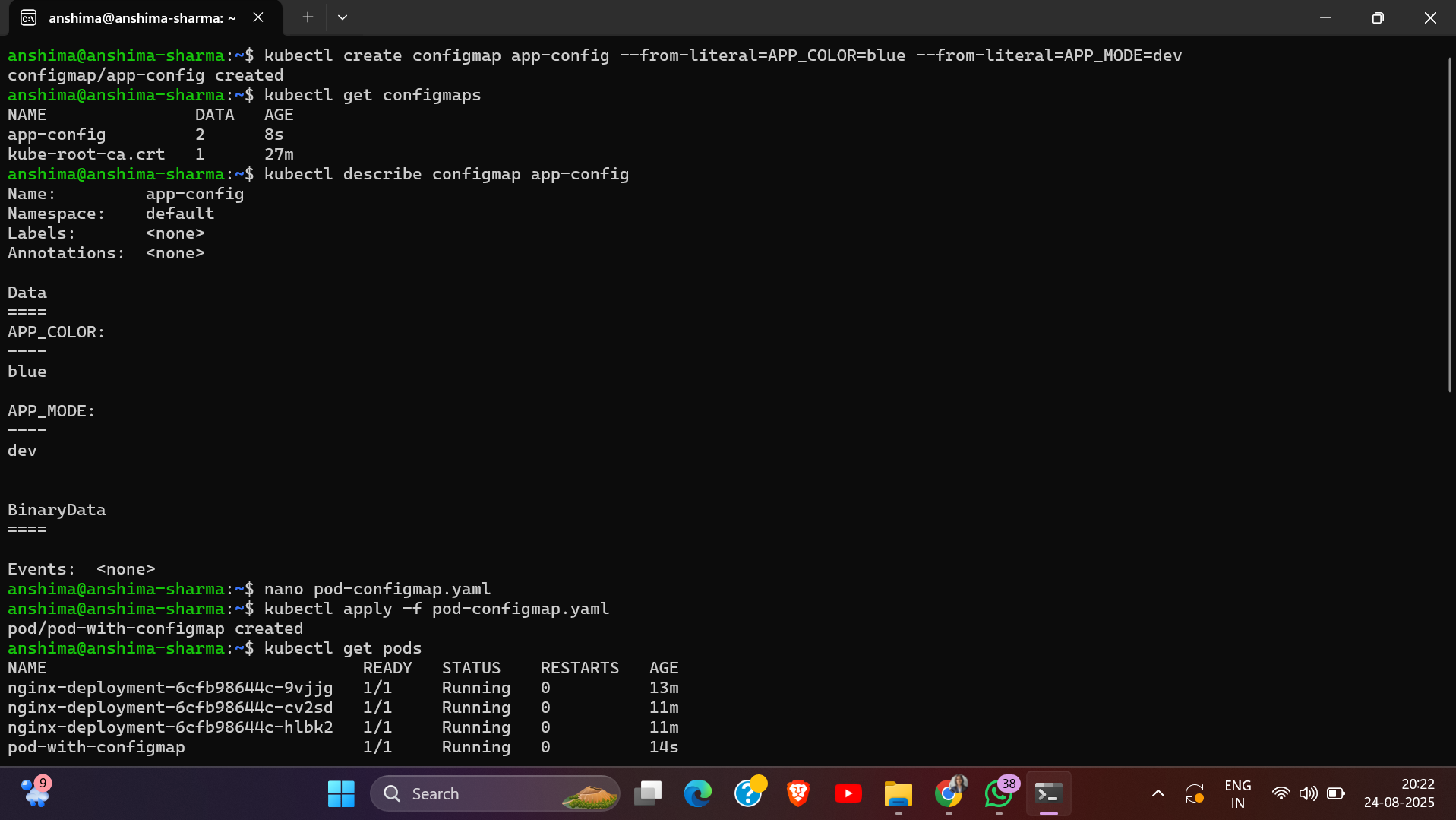
# Exercise 5: ConfigMaps and Secrets

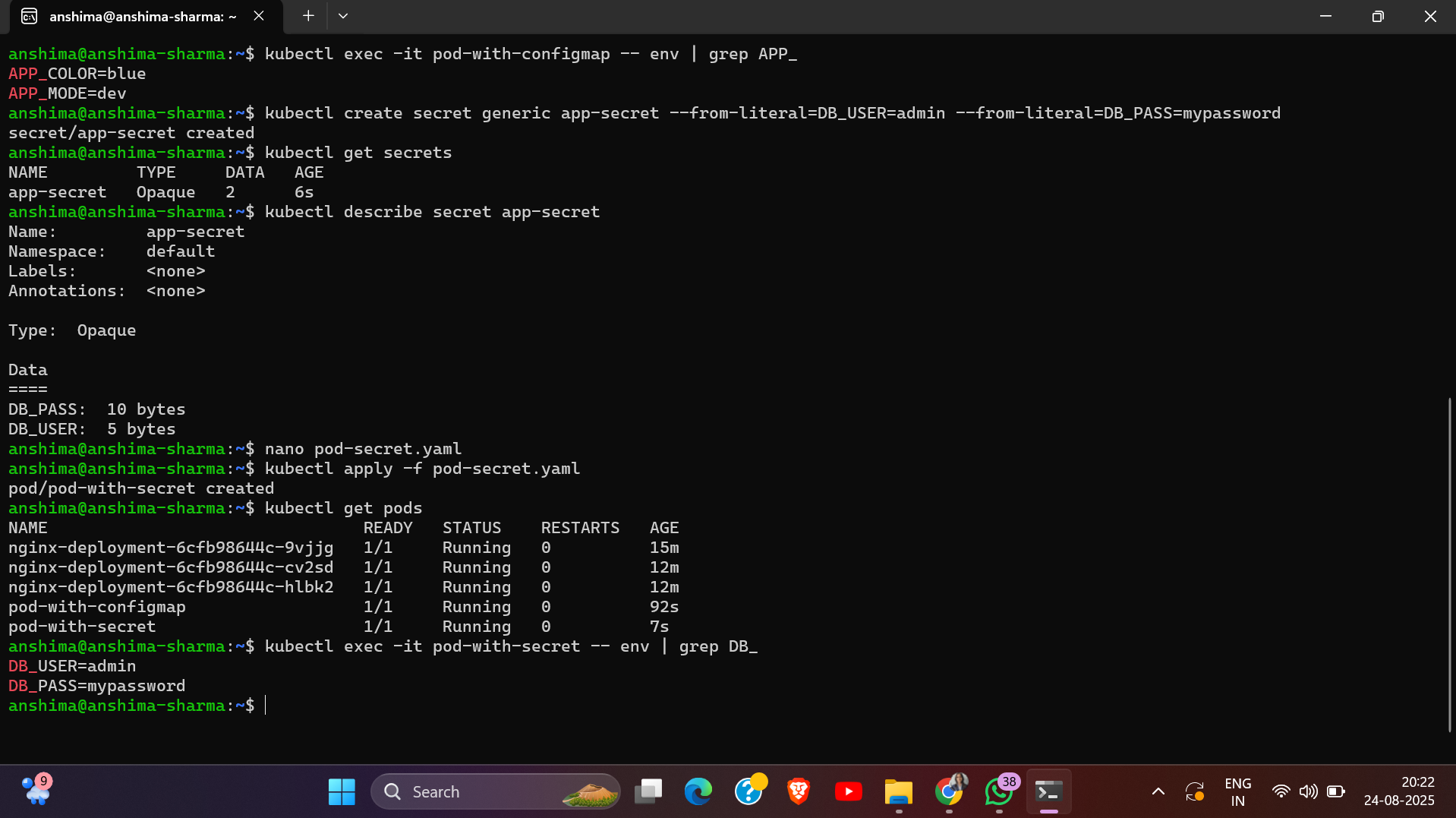
* Create a ConfigMap using a key-value pair:
* Mount the ConfigMap as environment variables in a pod.
* Create a Secret:
* Access the Secret in the pod via environment variables.

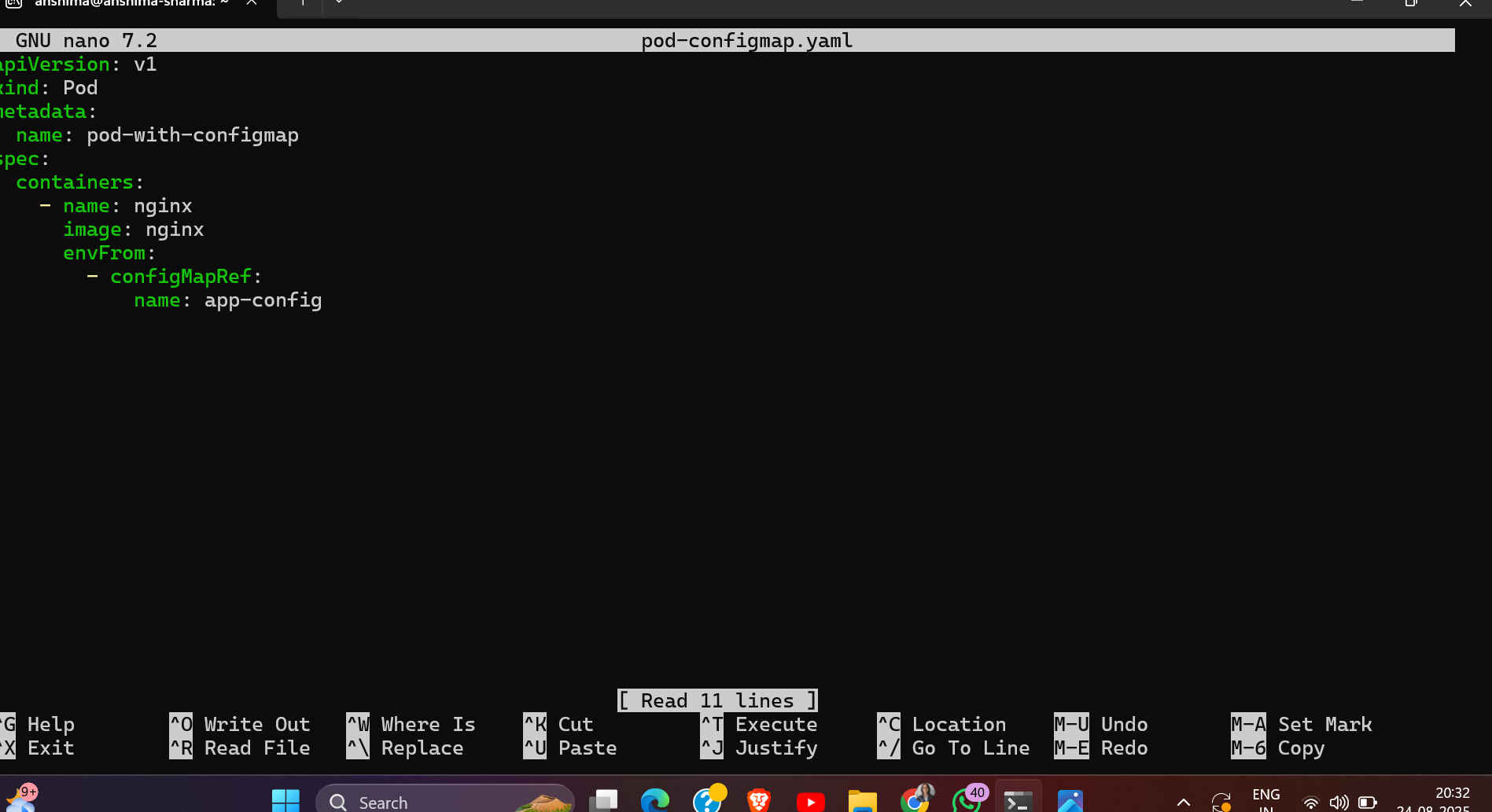
## Commands Used

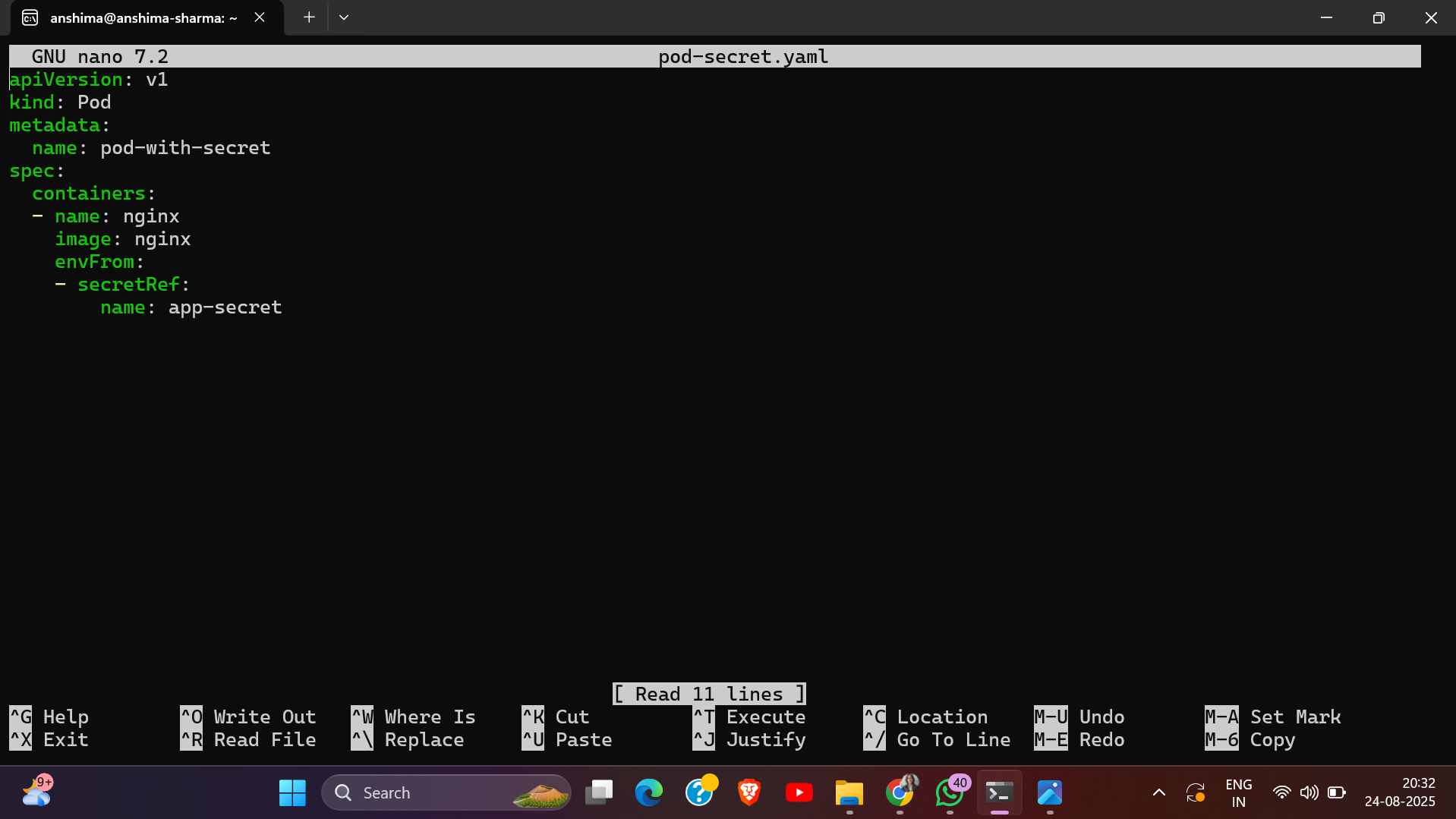
* kubectl create configmap app-config --from-literal=APP\_COLOR=blue --from-literal=APP\_MODE=dev
* kubectl get configmaps
* kubectl describe configmap app-config
* nano pod-configmap.yaml
* kubectl apply -f pod-configmap.yaml
* kubectl get pods
* kubectl create secret generic app-secret --from-literal=DB\_USER=admin --from-literal=DB\_PASS=mypassword
* kubectl get secrets
* kubectl describe secret app-secret
* nano pod-secret.yaml
* kubectl apply -f pod-secret.yaml
* kubectl get pods
* kubectl exec -it pod-with-secret -- env | grep DB\_

## Screenshots / Output Evidence









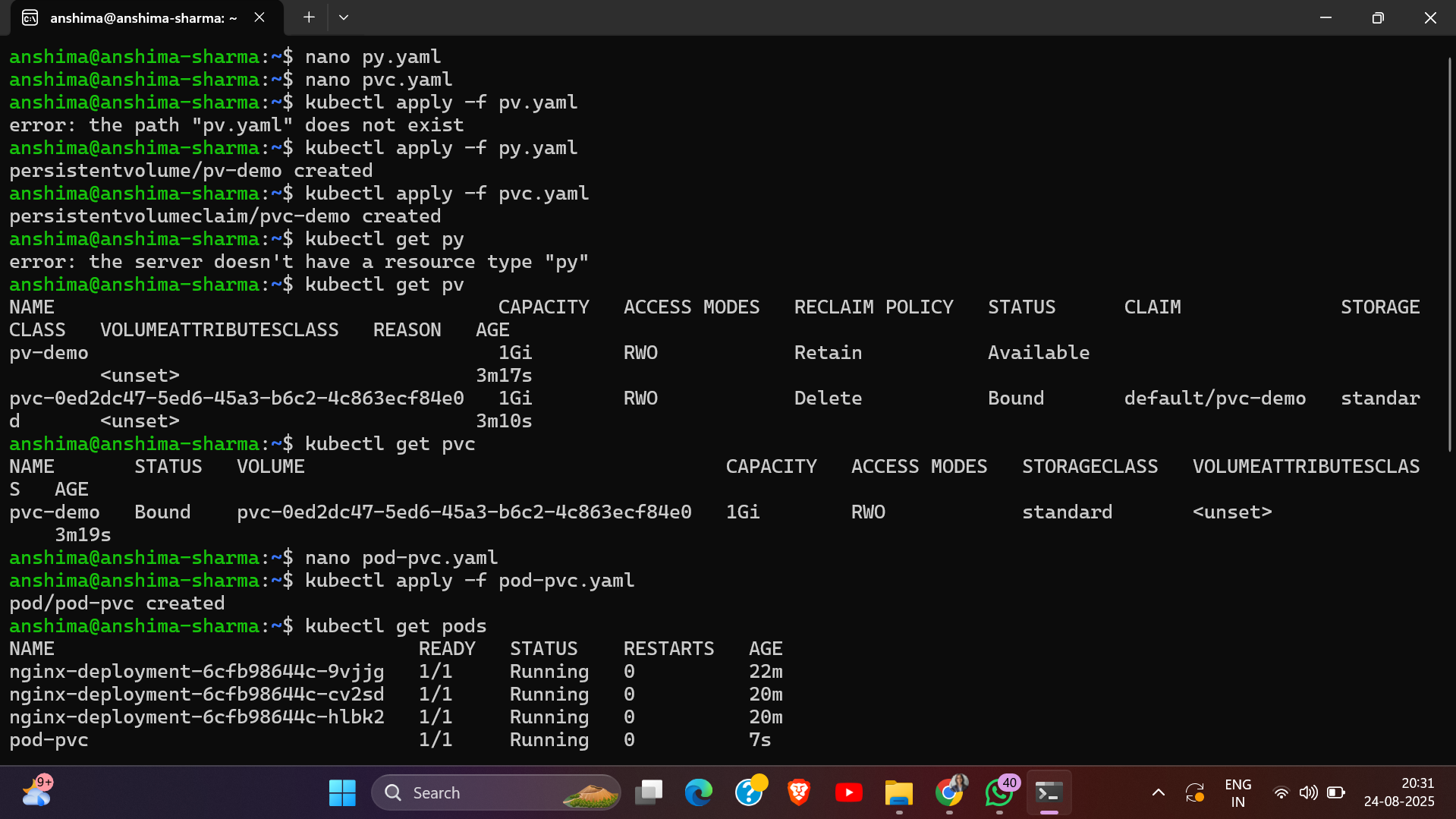
# Exercise 6: Persistent Volumes (PVs) and Persistent Volume Claims (PVCs)

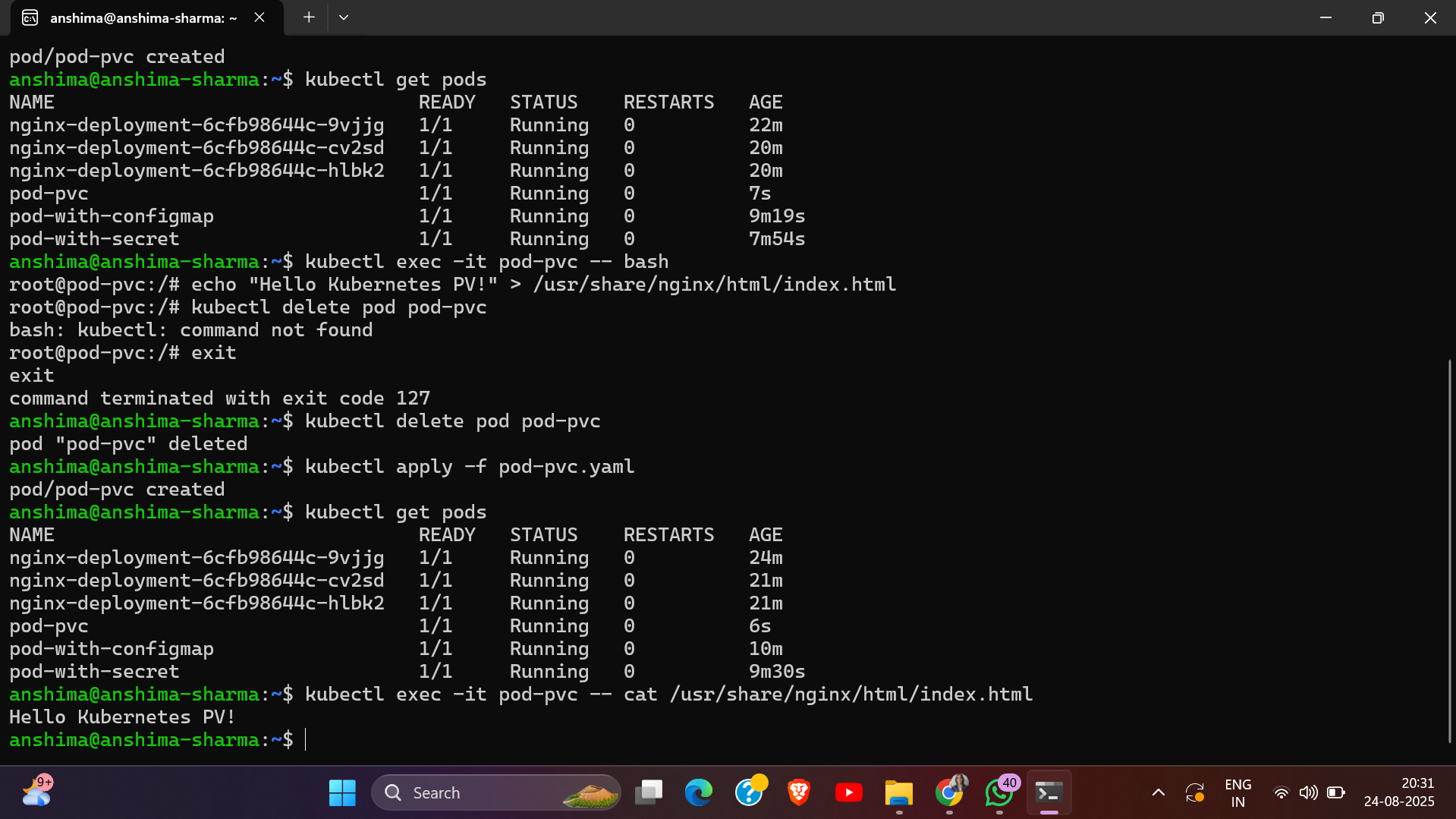
* Create a Persistent Volume (PV) and a Persistent Volume Claim (PVC) in YAML.
* Apply the YAML files to create the PV and PVC.
* Create a pod that uses the PVC to mount the volume.
* Write data to the volume and verify its persistence by restarting the pod.

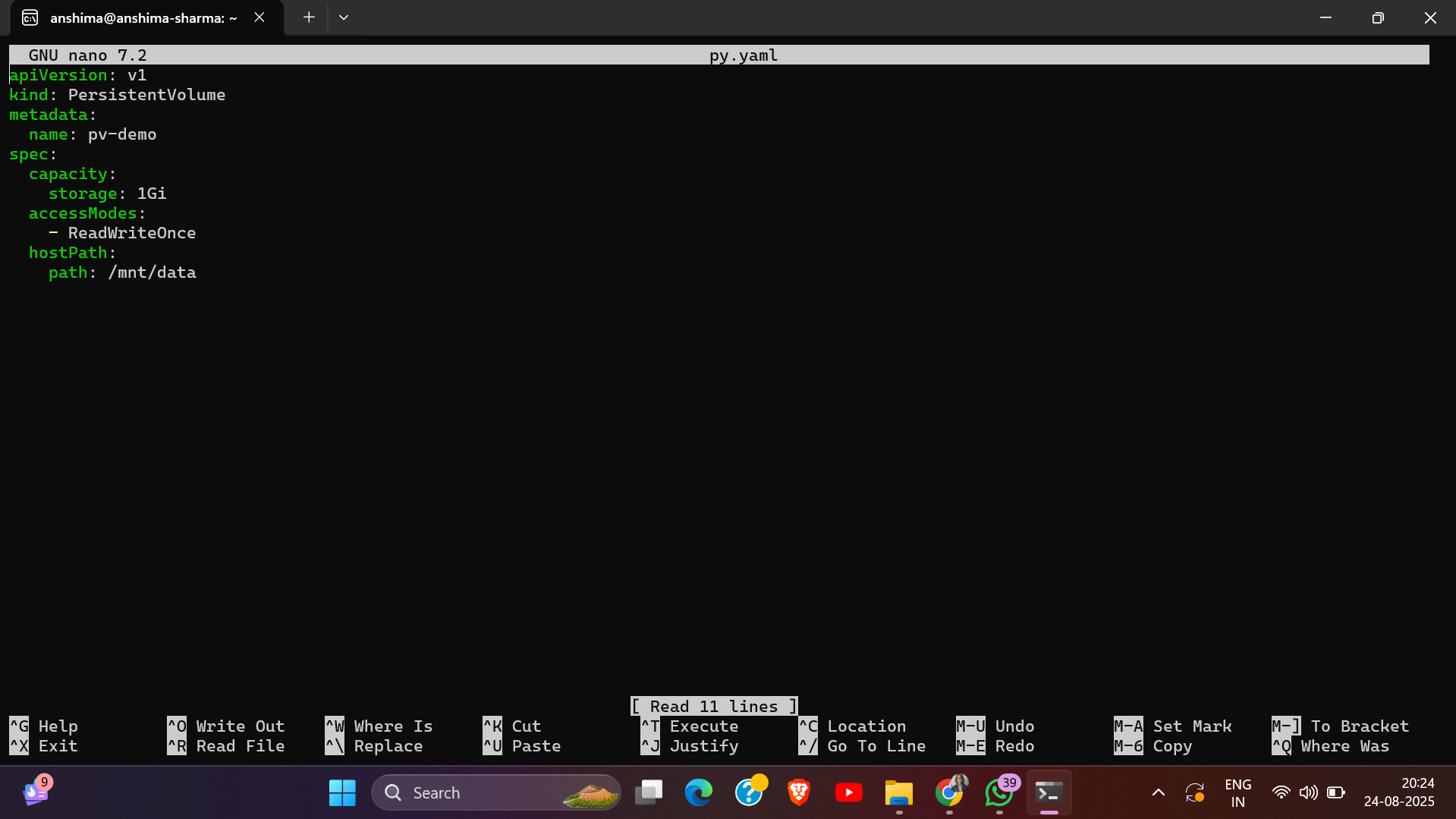
## Commands Used

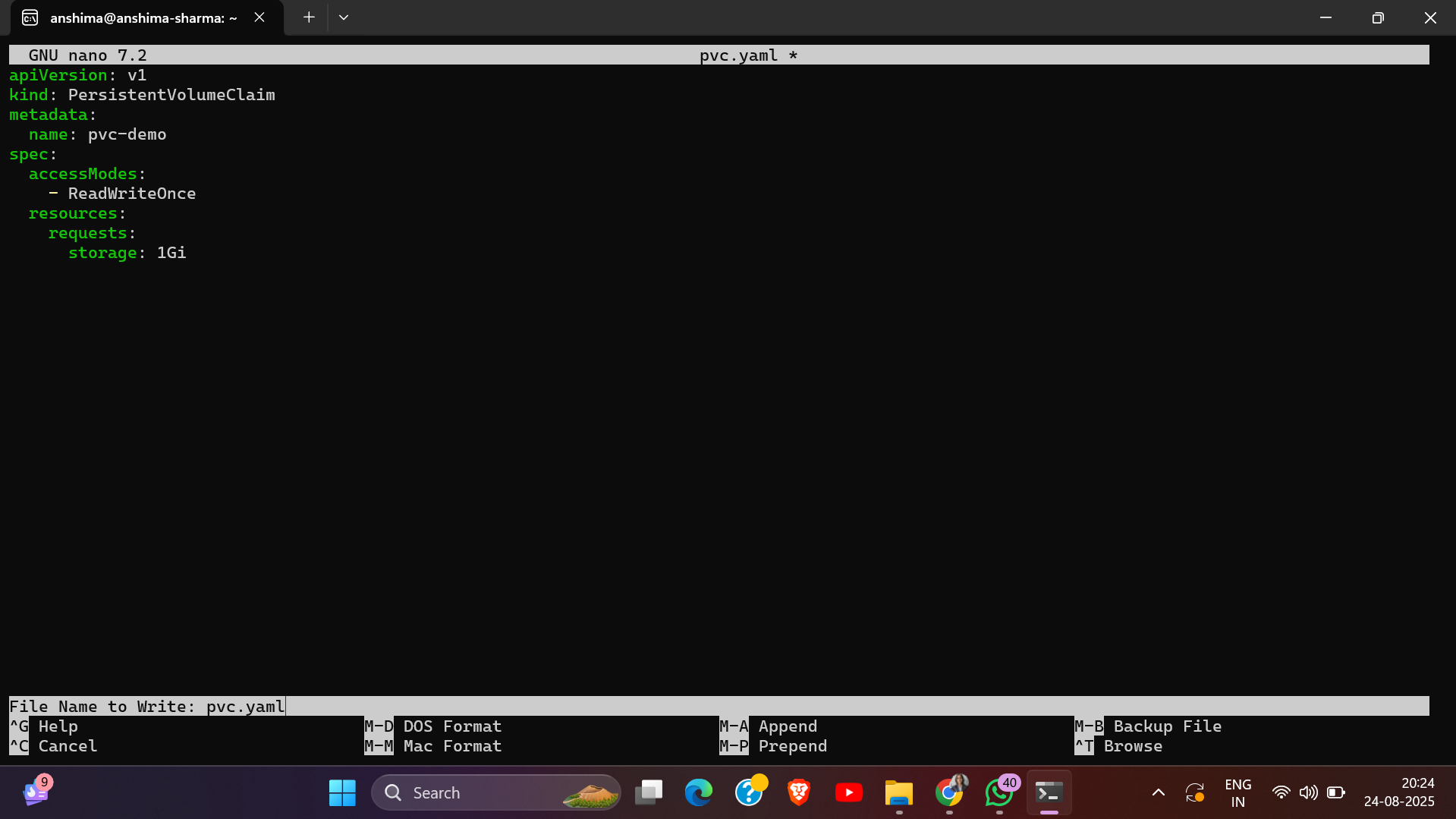
* nano pv.yaml
* nano pvc.yaml
* kubectl apply -f pv.yaml
* kubectl apply -f pvc.yaml
* kubectl get pv
* kubectl get pvc
* nano pod-pvc.yaml
* kubectl apply -f pod-pvc.yaml
* kubectl get pods
* kubectl exec -it pod-pvc – bash
* # echo "Hello Kubernetes PV!" > /usr/share/nginx/html/index.html and exit
* kubectl delete pod pod-pvc
* kubectl apply -f pod-pvc.yaml
* kubectl exec -it pod-pvc -- cat /usr/share/nginx/html/index.html

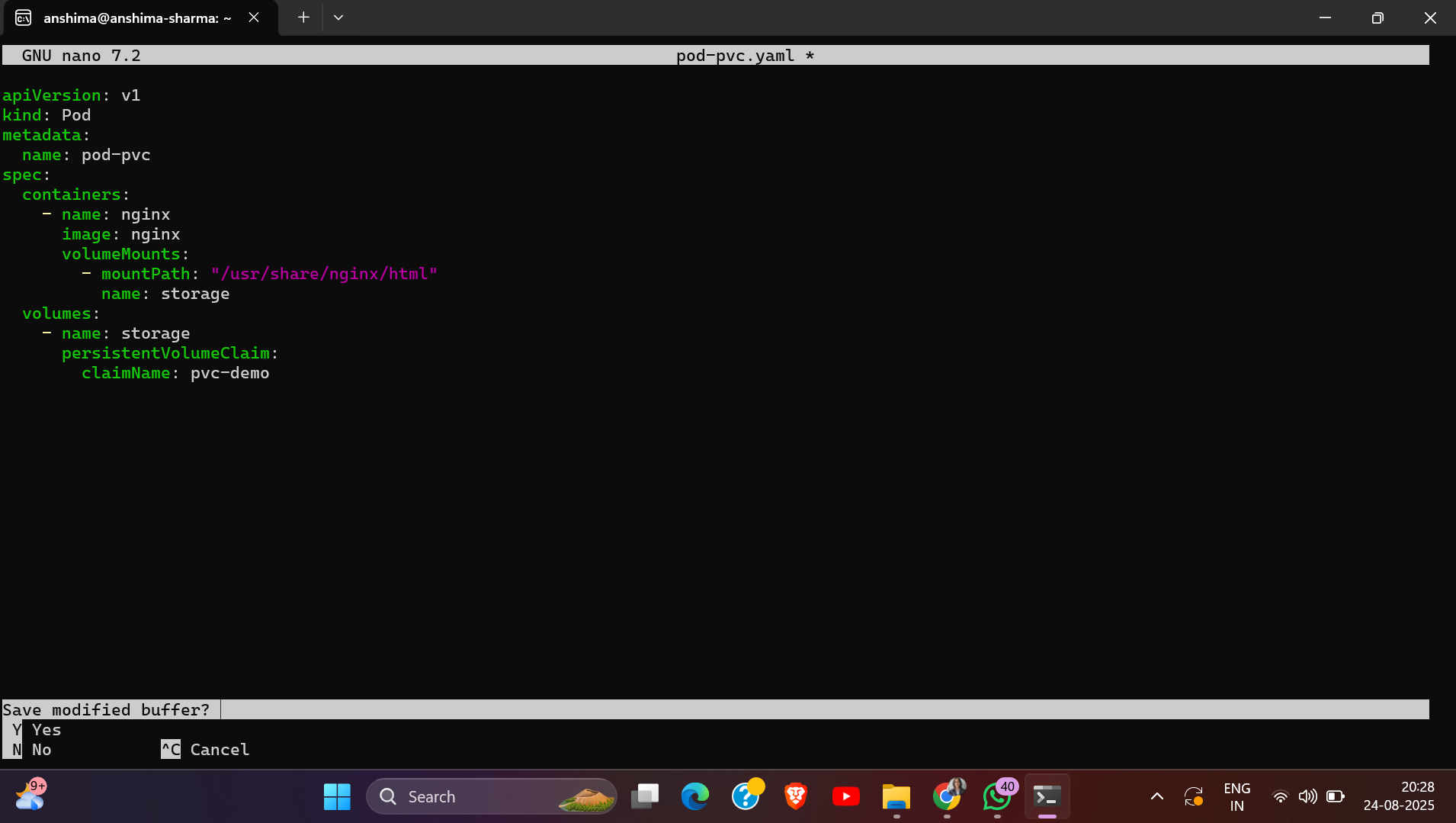
## Screenshots / Output Evidence











# Exercise 7: StatefulSets

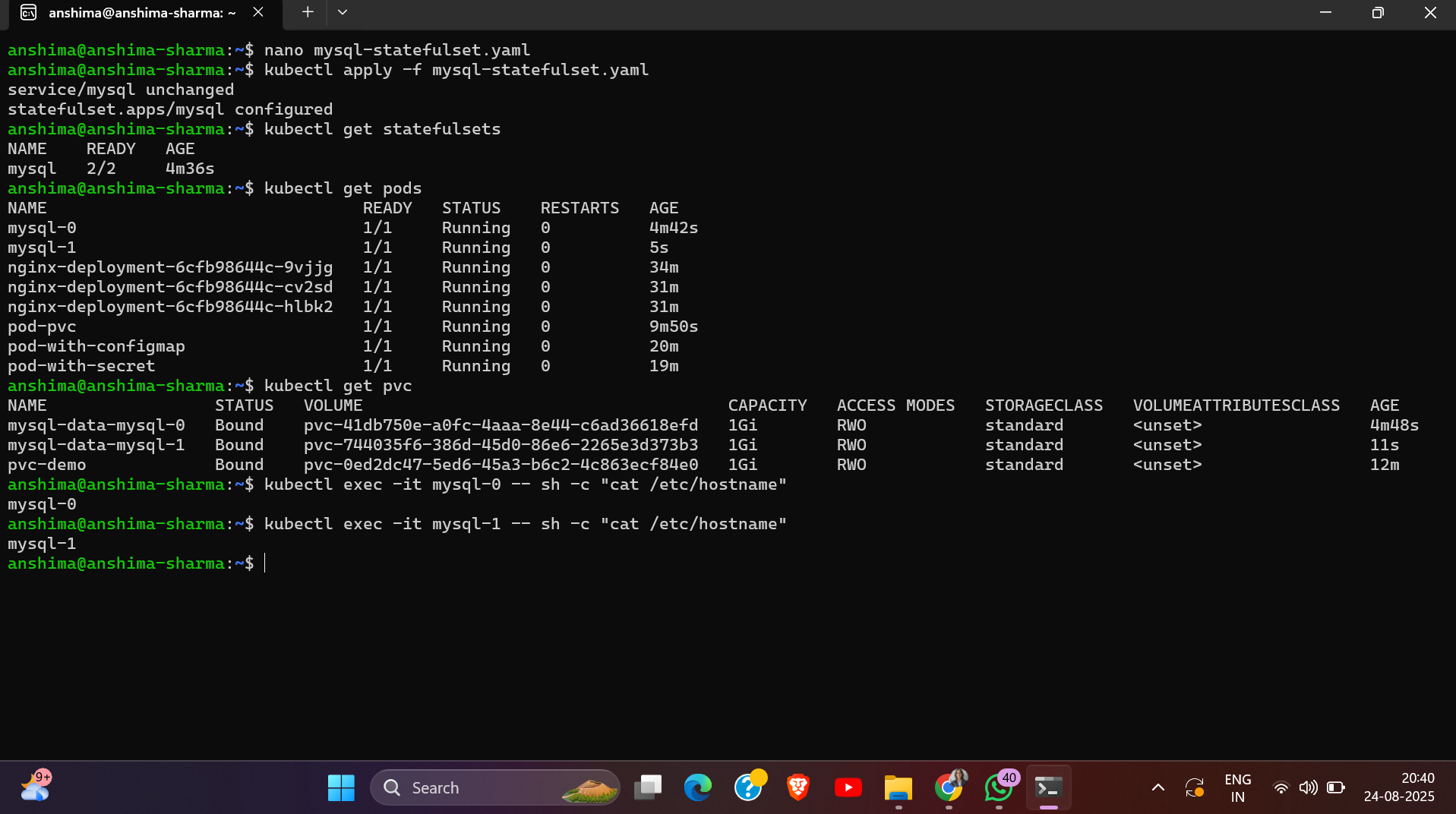
* Deploy a StatefulSet with (create YAML for an app like MySQL).
* View the StatefulSet.
* Create a headless service for the StatefulSet and access the pod by its stable network

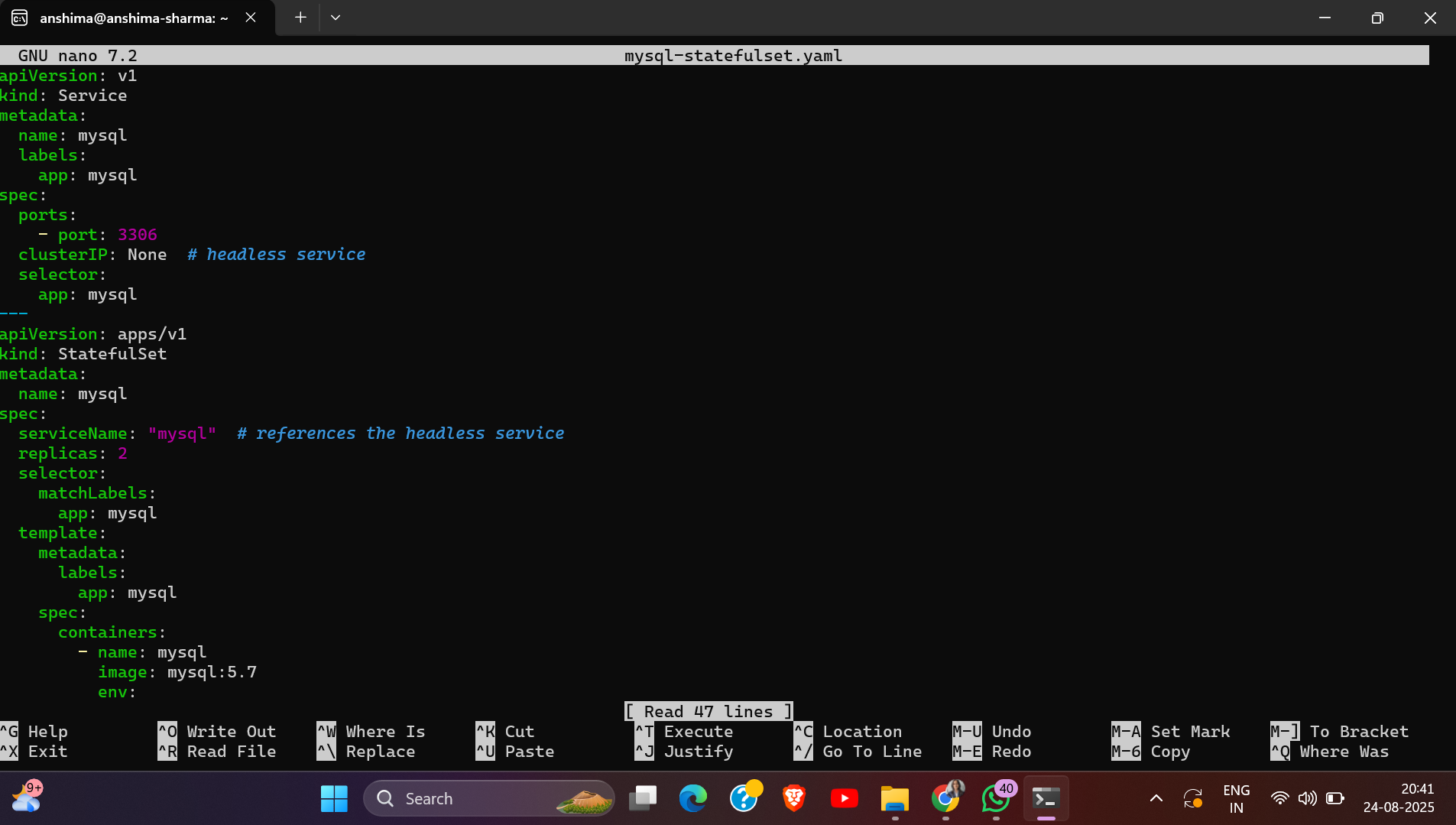
identity.

## Commands Used

* nano mysql-statefulset.yaml
* kubectl apply -f mysql-statefulset.yaml
* kubectl get statefulsets
* kubectl get pods
* kubectl get pvc
* kubectl exec -it mysql-0 -- sh -c "cat /etc/hostname"
* kubectl exec -it mysql-1 -- sh -c "cat /etc/hostname"

## Screenshots / Output Evidence





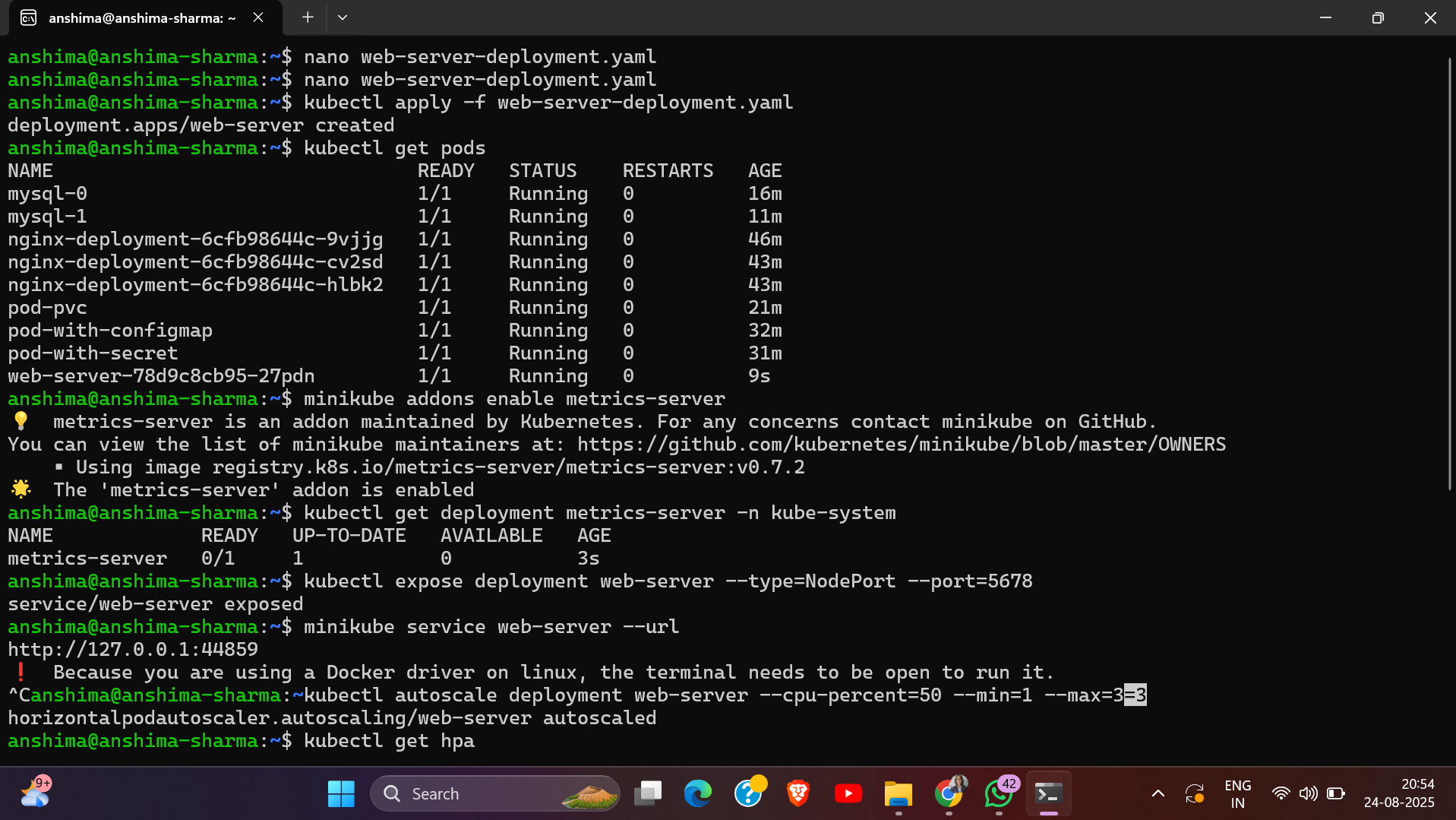
# Exercise 8: Horizontal Pod Autoscaling (HPA)

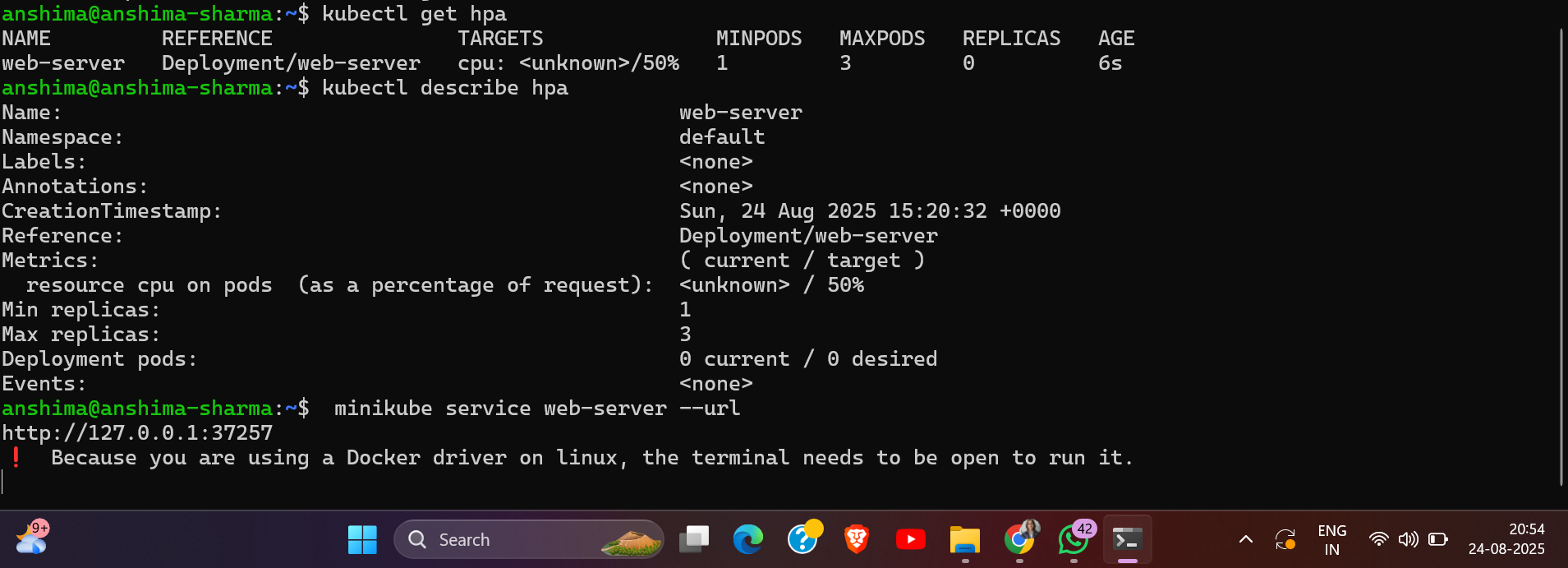
* Create a deployment (e.g., a simple HTTP server).
* Enable metrics server for autoscaling (e.g., Minikube).
* Create an HPA to scale the deployment based on CPU utilization:
* Test autoscaling by generating load on the deployment.

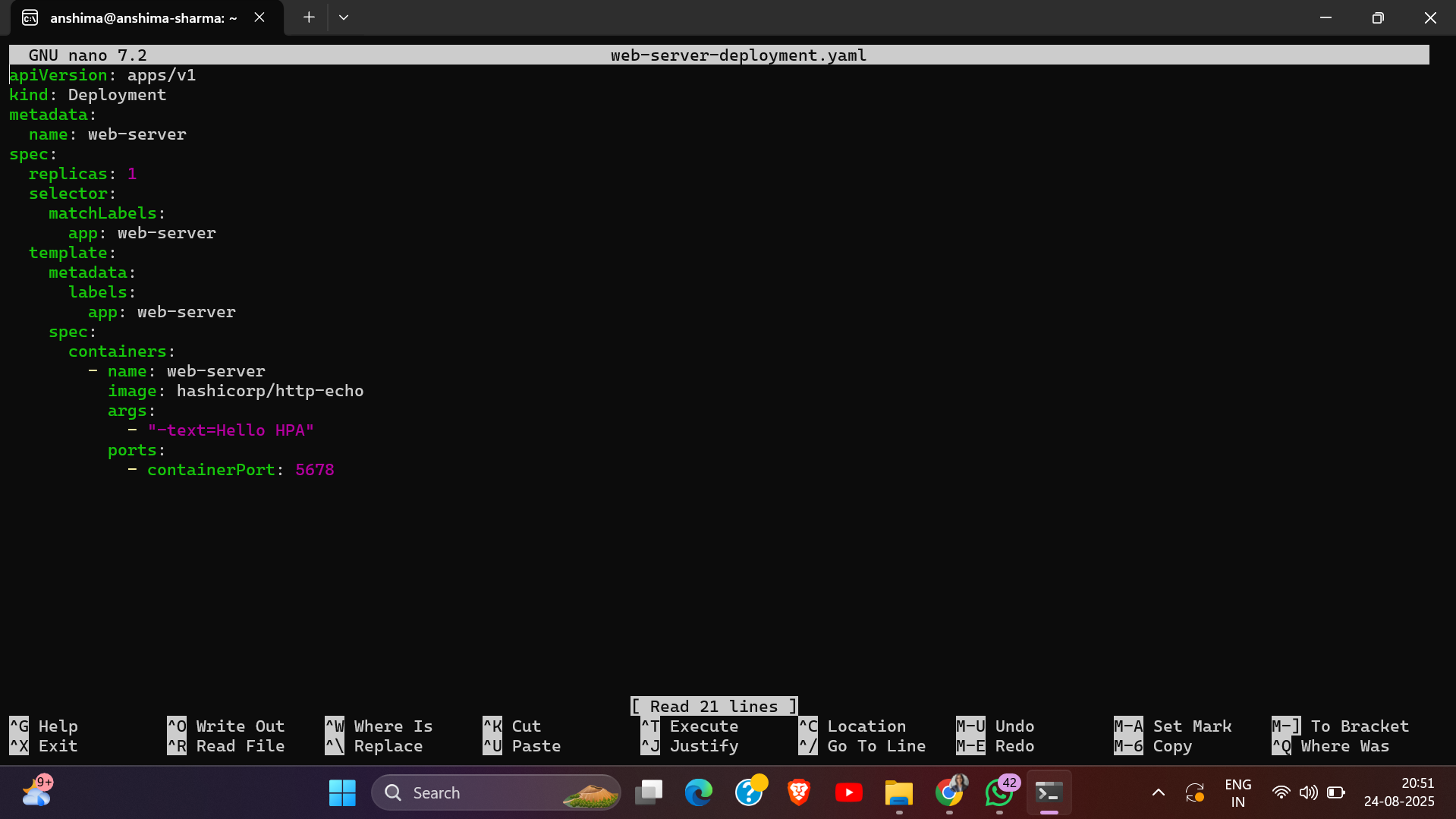
# Commands Used

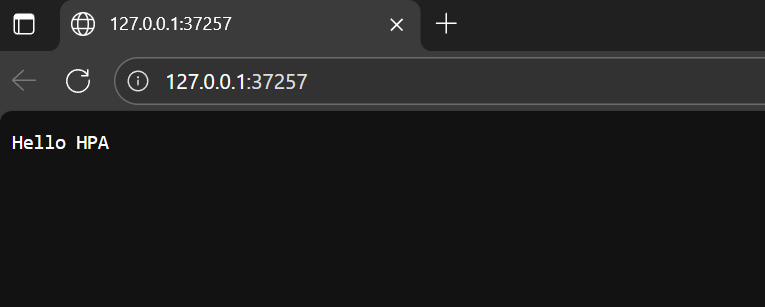
* nano web-server-deployment.yaml
* kubectl apply -f web-server-deployment.yaml
* kubectl get pods
* kubectl create deployment web-server --image=hashicorp/http-echo --replicas=1 -- \
* -text="Hello HPA"
* kubectl get deployments
* kubectl get pods
* minikube addons enable metrics-server
* kubectl get deployment metrics-server -n kube-system
* kubectl expose deployment web-server --type=NodePort --port=5678
* minikube service web-server –url
* kubectl autoscale deployment web-server --cpu-percent=50 --min=1 --max=3
* kubectl get hpa
* kubectl describe hpa

## Screenshots / Output Evidence









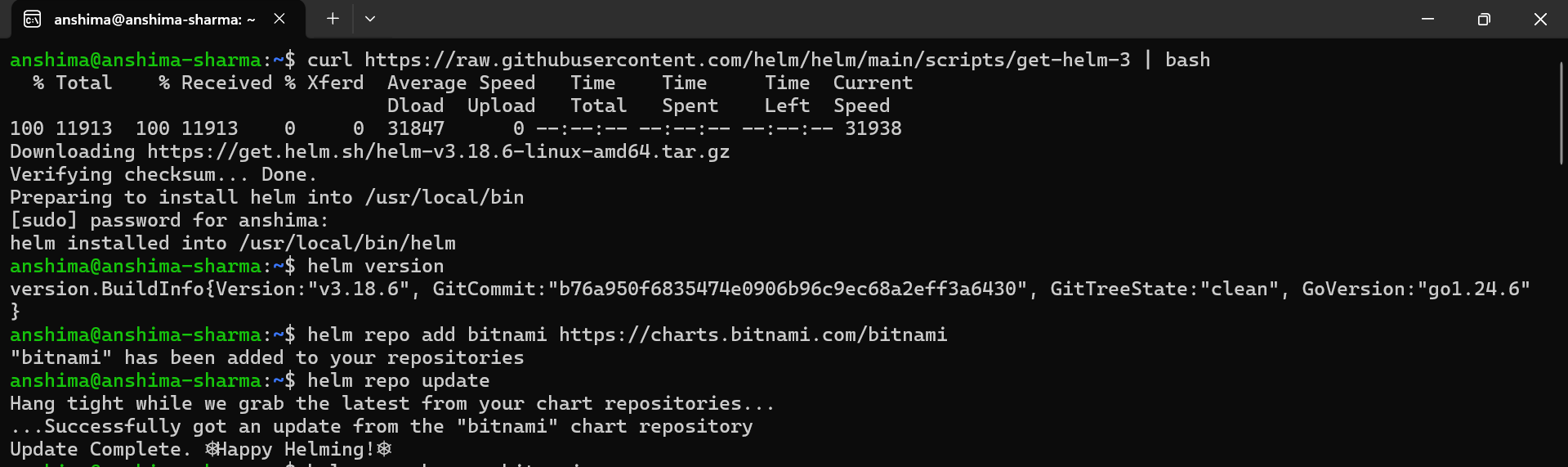
# Exercise 9: Helm Basics

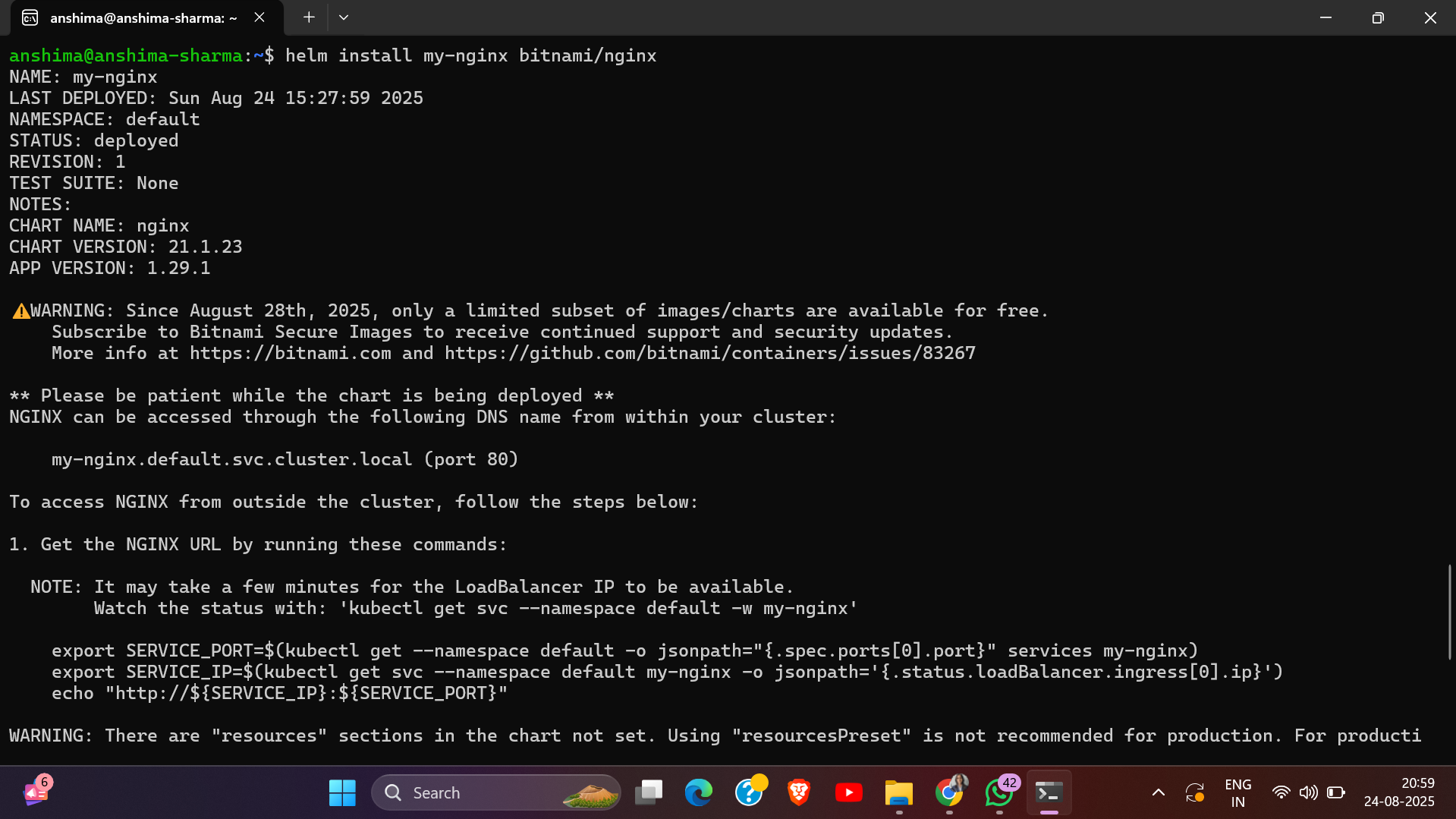
* Install Helm on your local machine.
* Add a Helm chart repository:
* Install a package from the Helm chart repository, e.g., Nginx:
* Verify the installation using

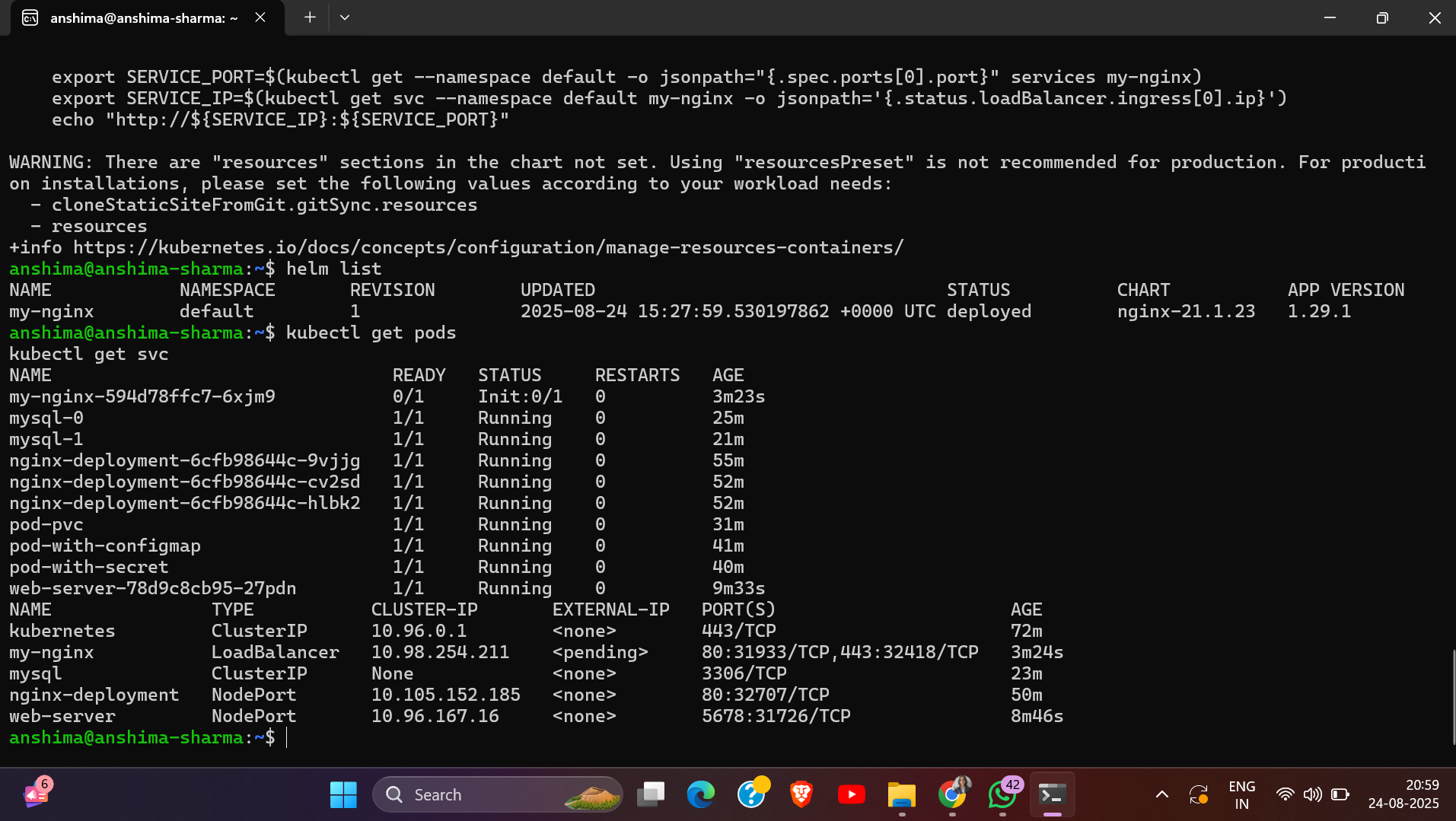
## Commands Used

* curl https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3 | bash
* helm version
* helm repo add bitnami <https://charts.bitnami.com/bitnami>
* helm repo update
* helm install my-nginx bitnami/nginx
* helm list
* kubectl get pods

## Screenshots / Output Evidence







# Exercise 10: Debugging and Troubleshooting

* Identify pod issues using describe command
* Check the status of nodes and pods
* View events related to the pod
* View logs for troubleshooting(pods and deployment)

# Commands Used

kubectl get pods

kubectl describe pod pod-pvc

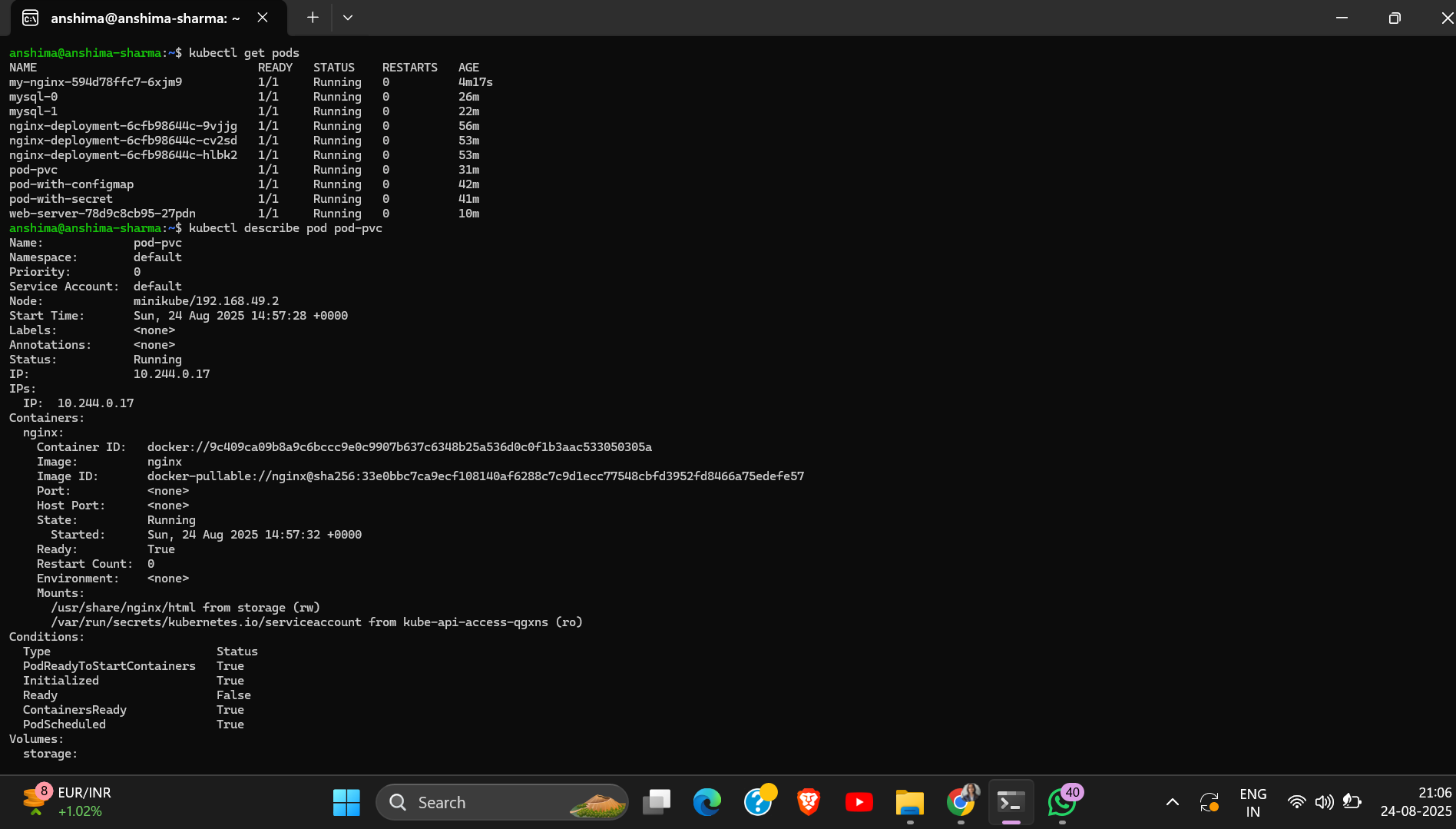
kubectl get nodes

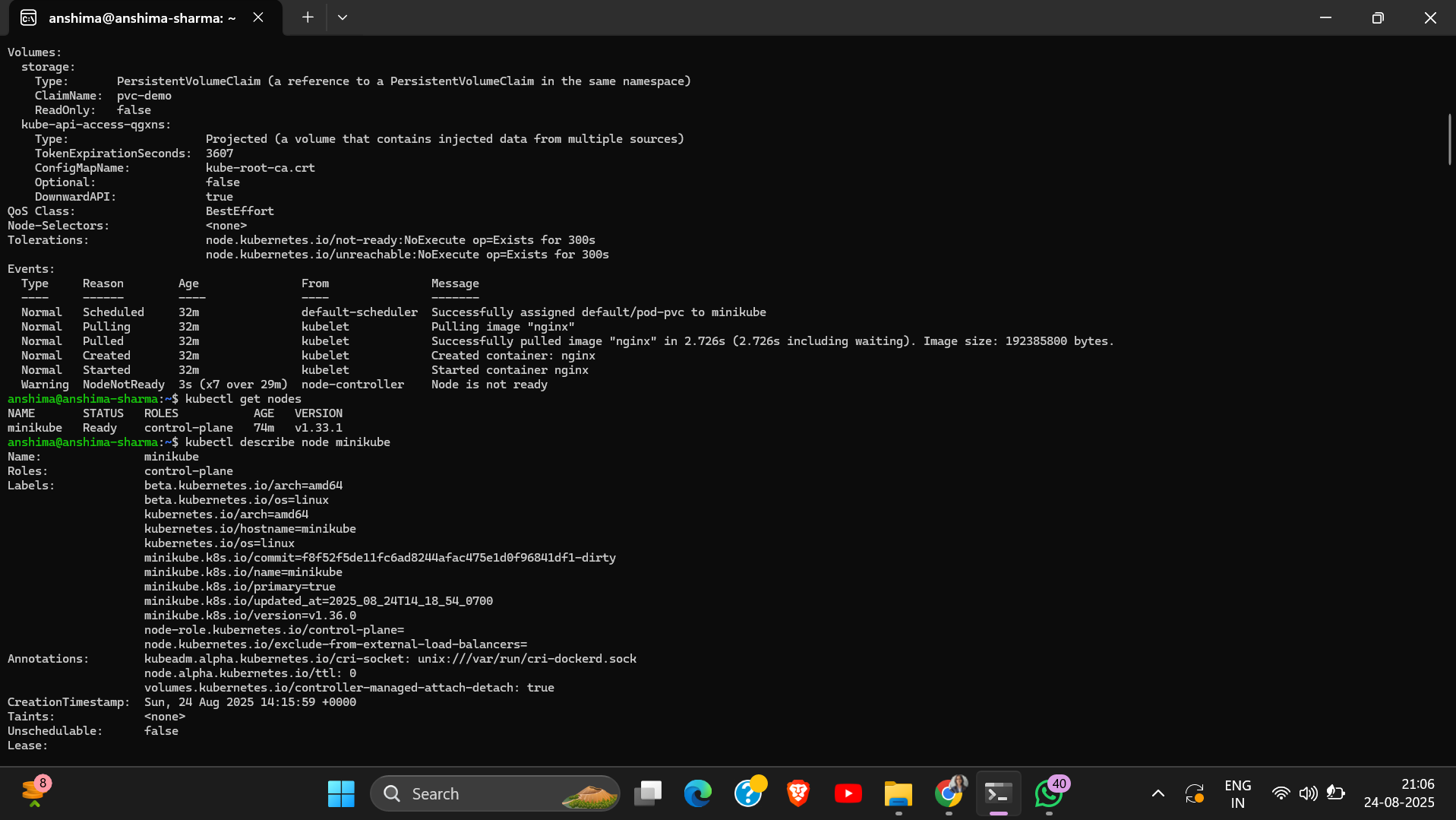
kubectl describe node minikube

kubectl get events --sort-by=.metadata.creationTimestamp

kubectl logs pod-pvc

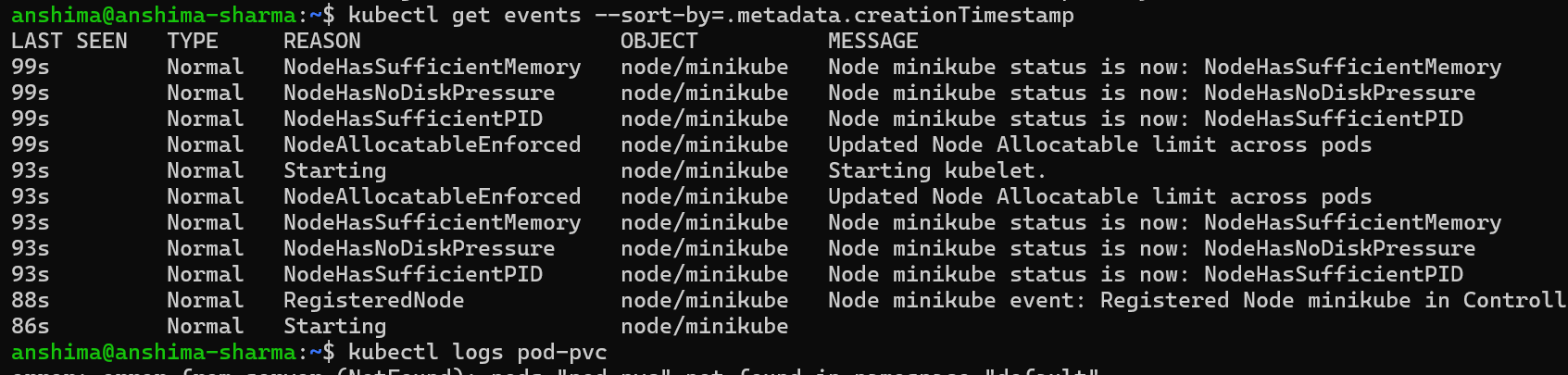
## Screenshots / Output Evidence





## 

## 



# Capstone Project: Kubernetes Application Deployment

## Commands Used

## Screenshots / Output Evidence