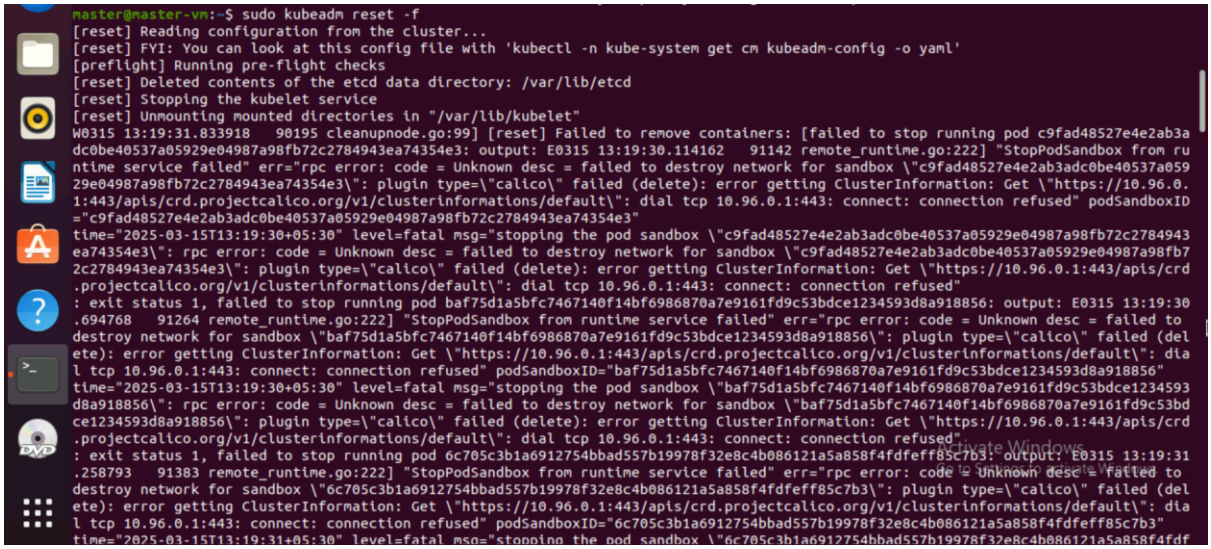
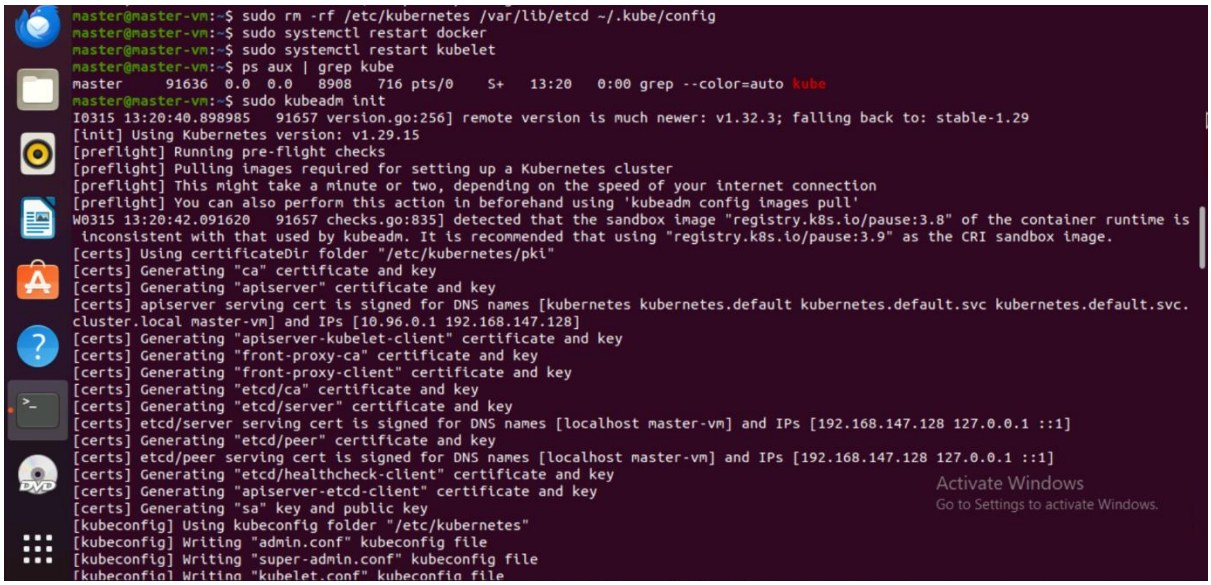
Kubernetes Project-02

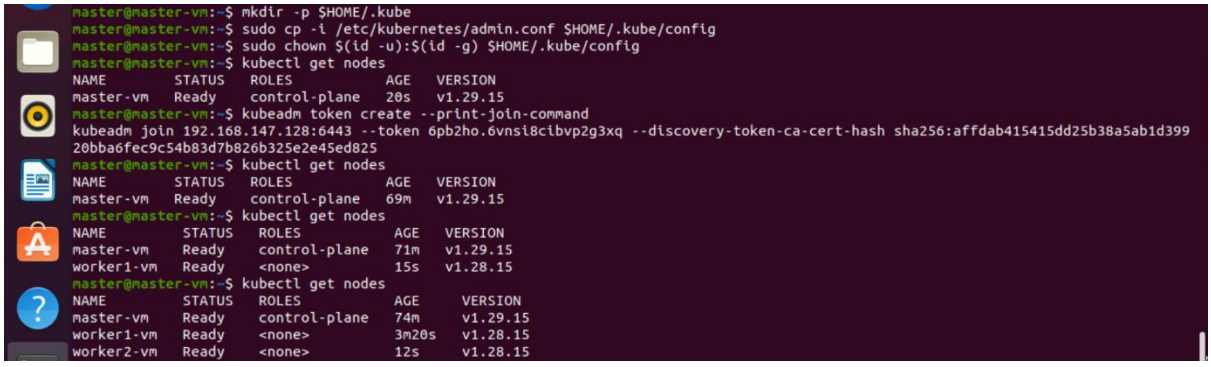
Kubernetes Multi-Tenant Project

Step 1: Check if Any Worker Node is Ready

kubectl get nodes



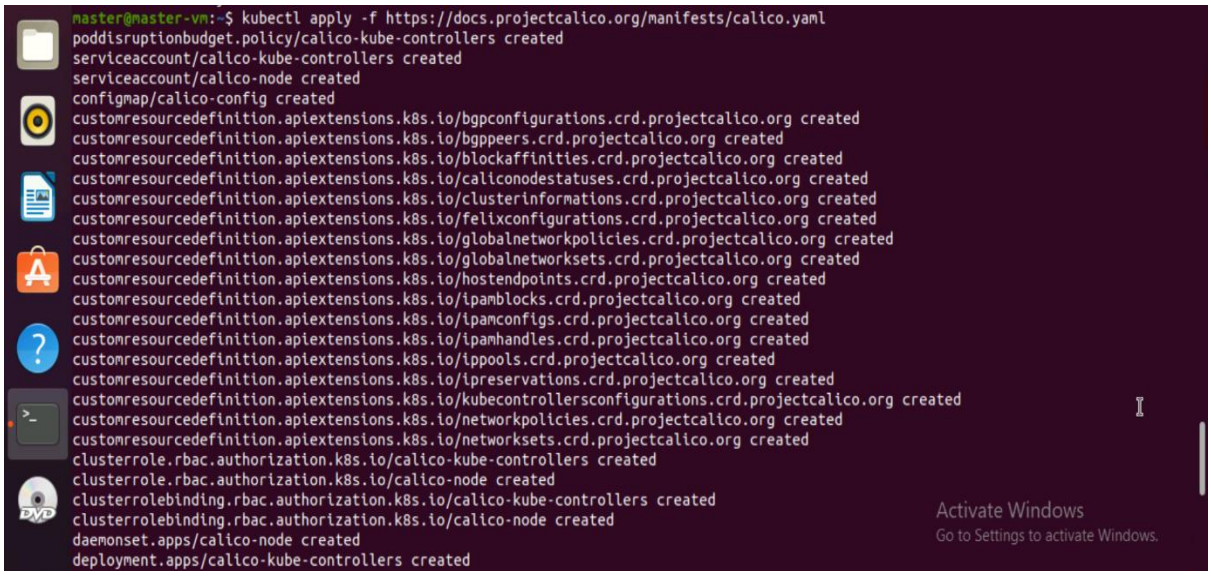




Step 2: Install Calico for Networking

Apply the Calico manifest to enable networking:

kubectl apply -f https://docs.projectcalico.org/manifests/calico.yaml



Step 3: Create Namespaces for Tenants

To isolate tenants, create separate namespaces:

kubectl create namespace tenant-a

kubectl create namespace tenant-b

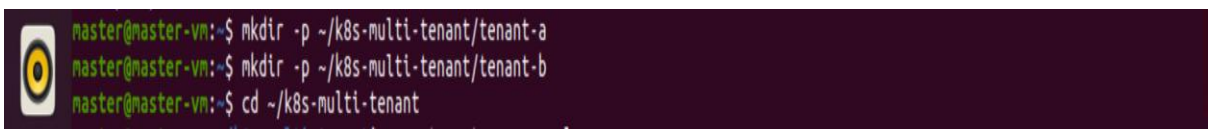
Step 4: Create Folder Structure for YAML Files

Create the folder structure to organize YAML files for each tenant:

mkdir -p ~/k8s-multi-tenant/tenant-a

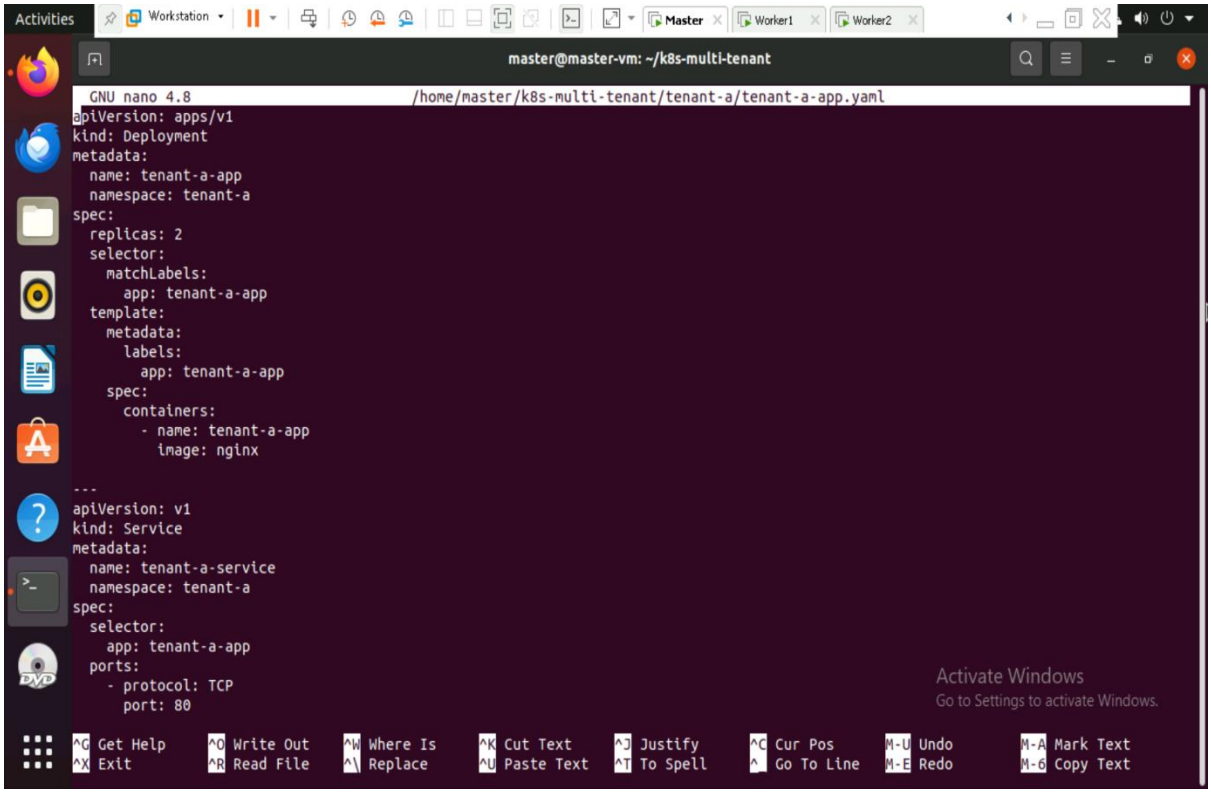
mkdir -p ~/k8s-multi-tenant/tenant-b

cd ~/k8s-multi-tenant



Step 5: Create Deployment and Service for Tenant A

nano ~/k8s-multi-tenant/tenant-a/tenant-a-app.yaml



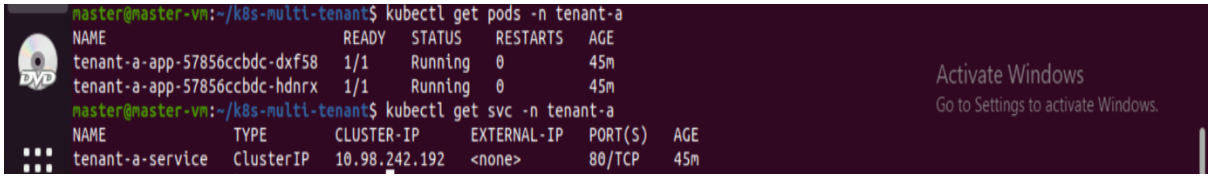
Apply the configuration:

kubectl apply -f ~/k8s-multi-tenant/tenant-a/tenant-a-app.yaml

Verify the deployment:

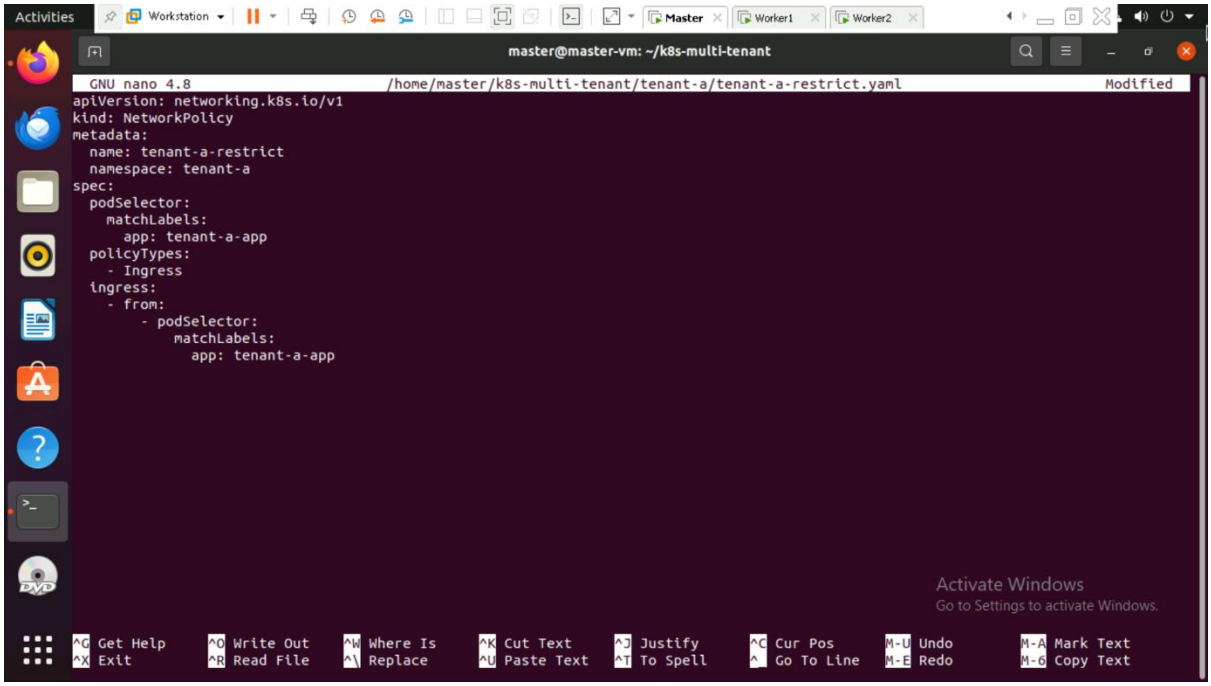
kubectl get pods -n tenant-a

kubectl get svc -n tenant-a



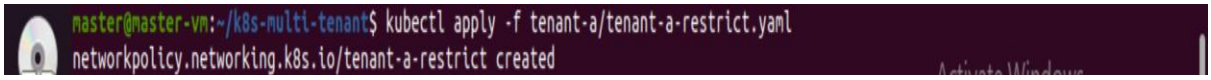
Step 6: Restrict Network Access for Tenant A

nano ~/k8s-multi-tenant/tenant-a/tenant-a-restrict.yaml



Apply the network policy:

kubectl apply -f ~/k8s-multi-tenant/tenant-a/tenant-a-restrict.yaml

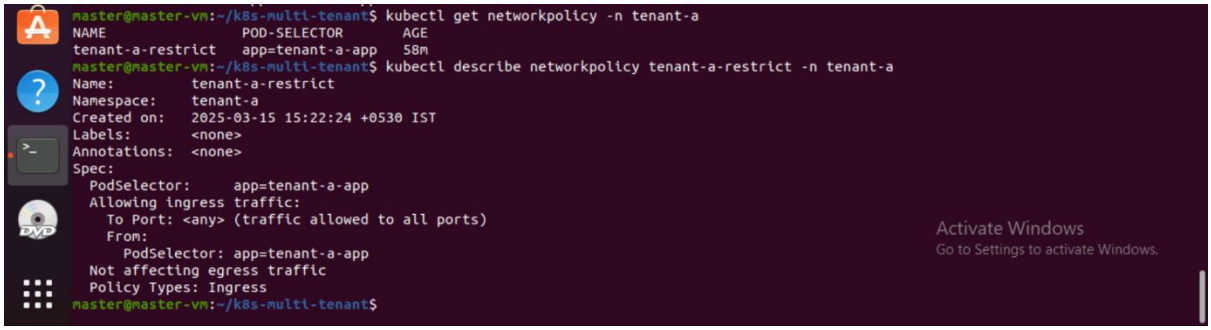


Verify Network Policy

To verify the network policy for Tenant A, run the following commands:

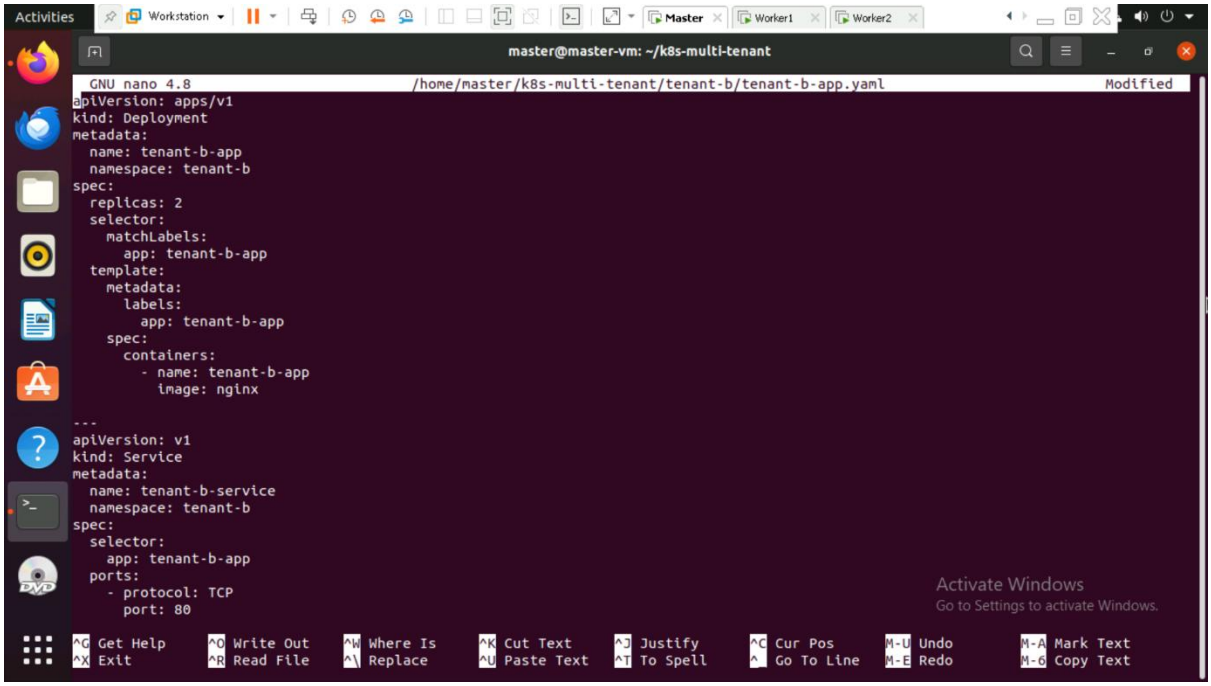
kubectl get networkpolicy -n tenant-a

kubectl describe networkpolicy tenant-a-restrict -n tenant-a



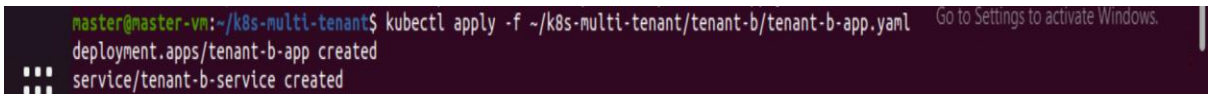
Step 7: Create Deployment and Service for Tenant B

nano ~/k8s-multi-tenant/tenant-b/tenant-b-app.yaml



Apply the configuration:

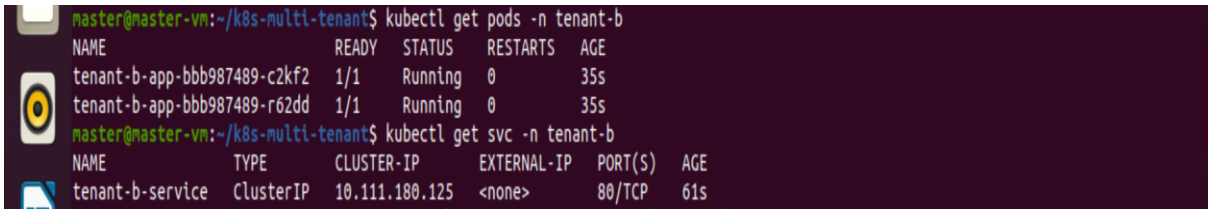
kubectl apply -f ~/k8s-multi-tenant/tenant-b/tenant-b-app.yaml



Verify the deployment:

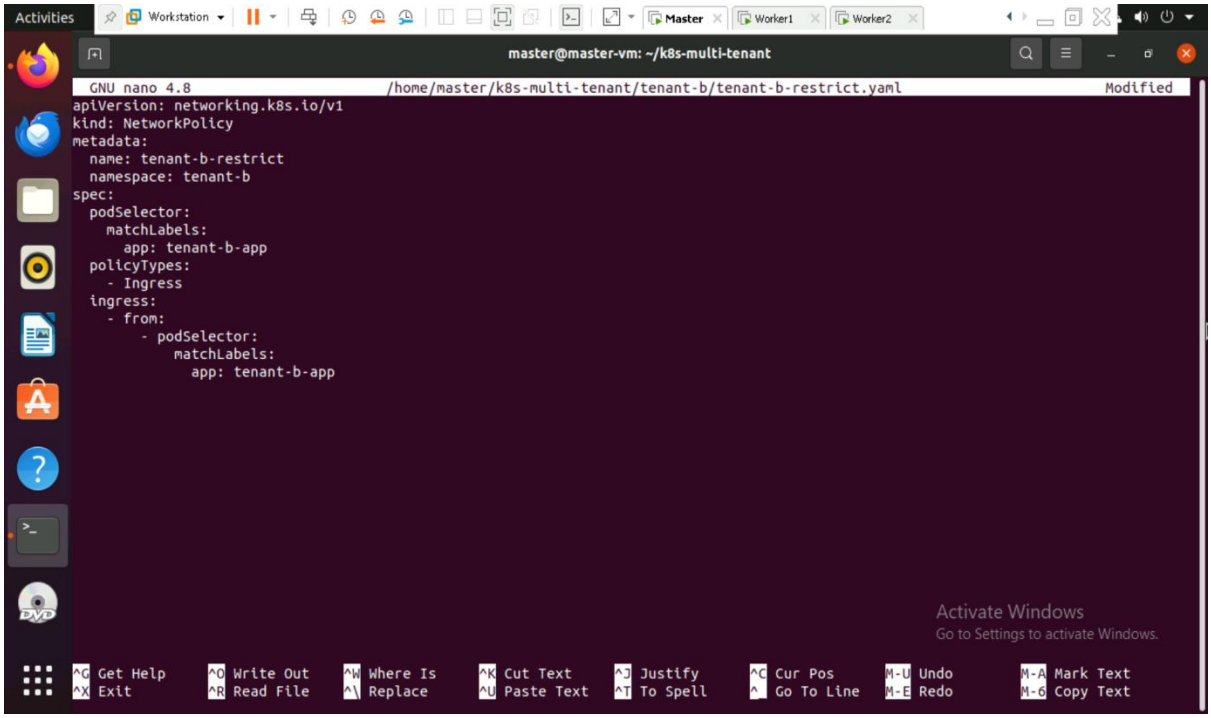
kubectl get pods -n tenant-b

kubectl get svc -n tenant-b



Step 8: Restrict Network Access for Tenant A

nano ~/k8s-multi-tenant/tenant-b/tenant-b-restrict.yaml



Apply the network policy:

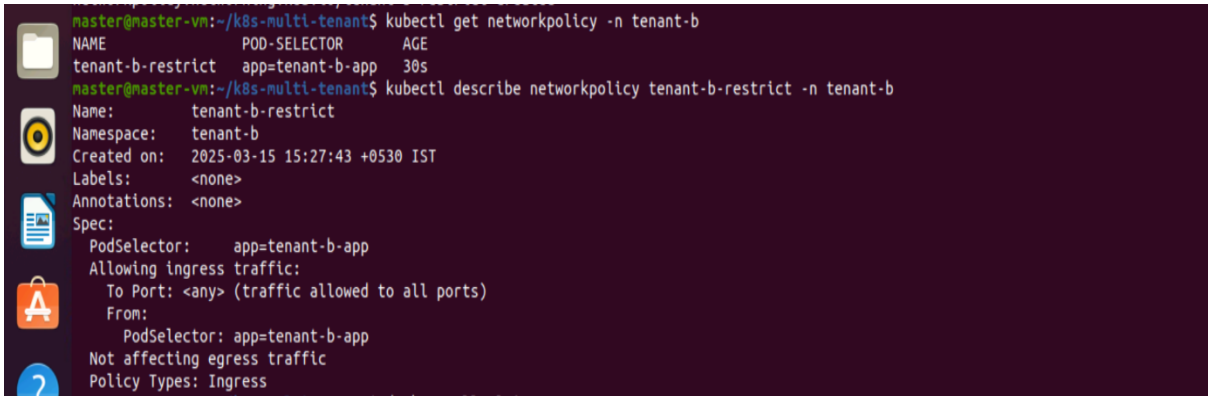
kubectl apply -f ~/k8s-multi-tenant/tenant-b/tenant-b-restrict.yaml

Step 9: Verify Network Policy

To verify the network policy for Tenant B, run the following commands:

kubectl get networkpolicy -n tenant-b

kubectl describe networkpolicy tenant-b-restrict -n tenant-b



Step 11: Test Tenant Isolation

Create a test pod in tenant-b and check access to tenant-a:

In worker docker run : docker pull alpine

kubectl run test-pod --image=alpine -n tenant-b --restart=Never -- sleep 3600

kubectl exec -it test-pod -n tenant-b -- wget --spider tenant-a-service.tenant-a