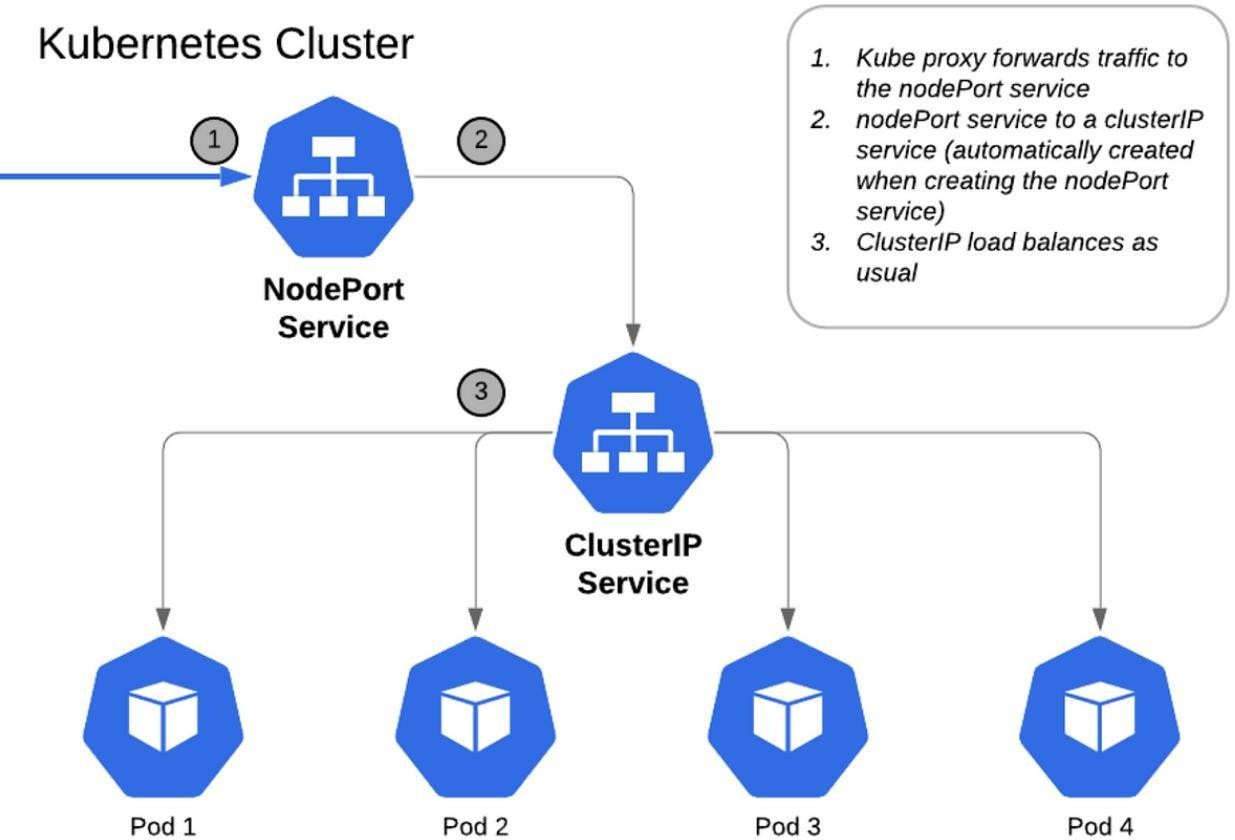
# KUBERNETES PROJECT WITH NODEPORT SERVICE

****

**Files and Configurations Used in This Project**

#### NGINX Configuration

* + NGINX Deployment with 3 Replicas
  + Expose in NodePort Service at Port 30007

#### Database Configuration

* + MySQL Headless
  + MySQL Persistent Volume (PV)
  + MySQL StatefulSet

#### Secrets and Configurations

* + MySQL Secrets
  + MySQL ConfigMaps
  + Service Account for Secrets and ConfigMaps
  + Read-Only Roles for Secrets and ConfigMaps
  + Read-Only Role-Binding for Secrets and ConfigMaps

#### Monitoring and Logging

* + DaemonSet (Prometheus)
  + Prometheus RBAC
  + Prometheus NodePort Exposed on Port 30090
  + Promtail DaemonSet
  + Loki DaemonSet
  + Loki NodePort Exposed on Port 30091
  + Grafana Daemon with Exposed Port 30080

**Project Summary**

#### NGINX Deployment and Services

* + Sabse pehle NGINX web server ko 3 replicas ke saath deploy karna.
  + Phir, NodePort service ka use karke expose karna.

#### Database Configuration

* + MySQL StatefulSet deploy karna.
  + Persistent Volume (PV) configure karna taake data storage manage ho sake.

#### Secrets & ConfigMaps

* + Security ke liye Secrets (MYSQL Database ka password aur username store karne ke liye
  + ConfigMaps (port, hostname, database name ke liye) ka use.

#### Security Enhancements

* + Service Accounts create karna.
  + RBAC roles aur RoleBindings configure karna jo Secrets aur ConfigMaps ke liye read-only access provide karein.

#### Monitoring & Logging

* + Prometheus DaemonSet deploy karna, RBAC configure karna aur port 30090 par expose karte hue read-write permissions dena.
  + Promtail ka use karna logs collect karne ke liye.
  + Loki ko logging backend banakar port 30091 par expose karna.
  + Grafana deploy karna jo metrics visualize karega aur port 30080 par expose hoga.

**Understanding Each File**

#### Is project me hum har YAML aur Configuration file ko samjhenge ki kaise ye Deployment, Networking ko Manage karne, Security implement karne, aur Monitoring Setup karne me madad kar rahi hai.

**Achievements:**

#### Successfully Deployed NGINX with NodePort Service

NGINX web server ko 3 replicas ke saath deploy kiya aur NodePort service ka use karke expose kiya, jo direct access allow karta hai specific nodes ke through.

#### Persistent & Secure MySQL Database

StatefulSet aur Persistent Volumes (PV) ka use karke MySQL database ko secure aur reliable banaya, taake data loss na ho.

#### Secrets & ConfigMaps Implementation

Sensitive information jaise database credentials, hostname, aur configuration settings securely manage kiye.

#### RBAC-based Access Control

Service Accounts, RBAC roles, aur RoleBindings implement kiye jo Secrets aur ConfigMaps ke liye read-only access provide karte hain, taake security maintain ho.

#### Monitoring & Logging Setup

Prometheus, Loki, aur Grafana ka use karke complete observability stack setup kiya jo system ki health aur performance track karne me madad karta hai.

#### Note: Sabse pehle Minikube ki installation and Setup karni hai, jaise hum pehle Installation part mein kar chuke hain. Agar tumne Minikube setup nahi kiya hai, toh neeche diye gaye URL ko kholo aur Minikube Cluster ka setup complete karo.

**URL :**

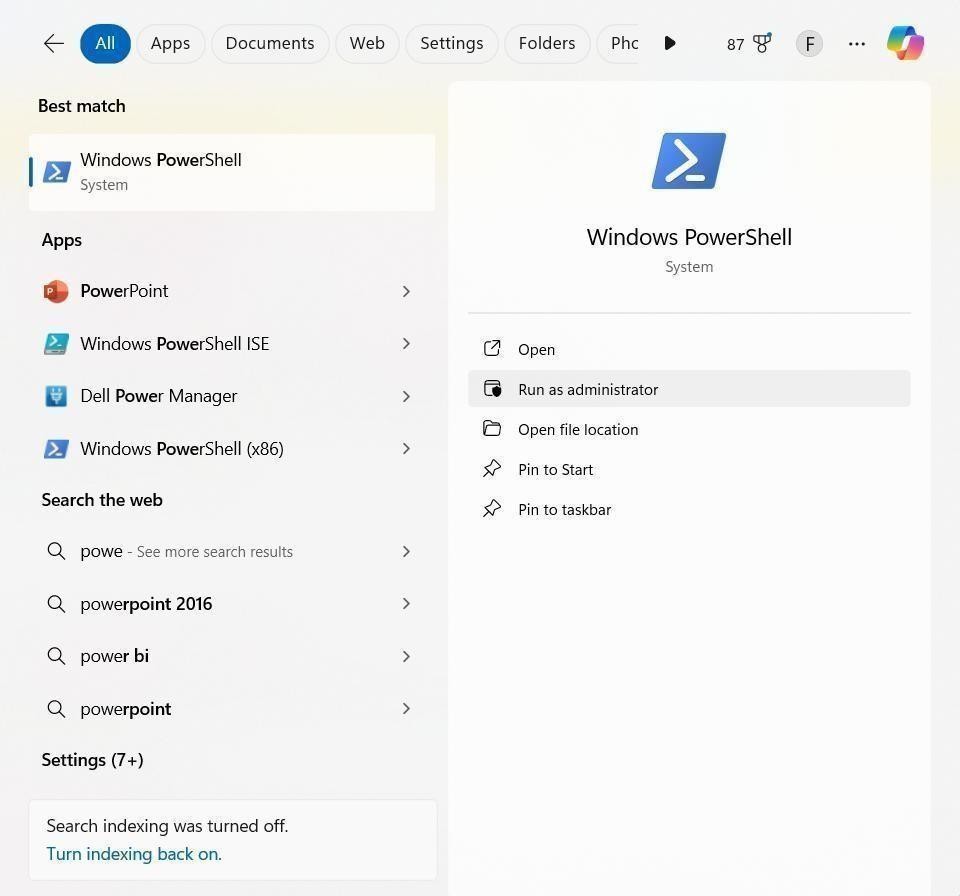
[https://github.com/Faikhan147/Kubernetes/blob/main/03-Installations/01-Minikube-Installation.docx](fg)

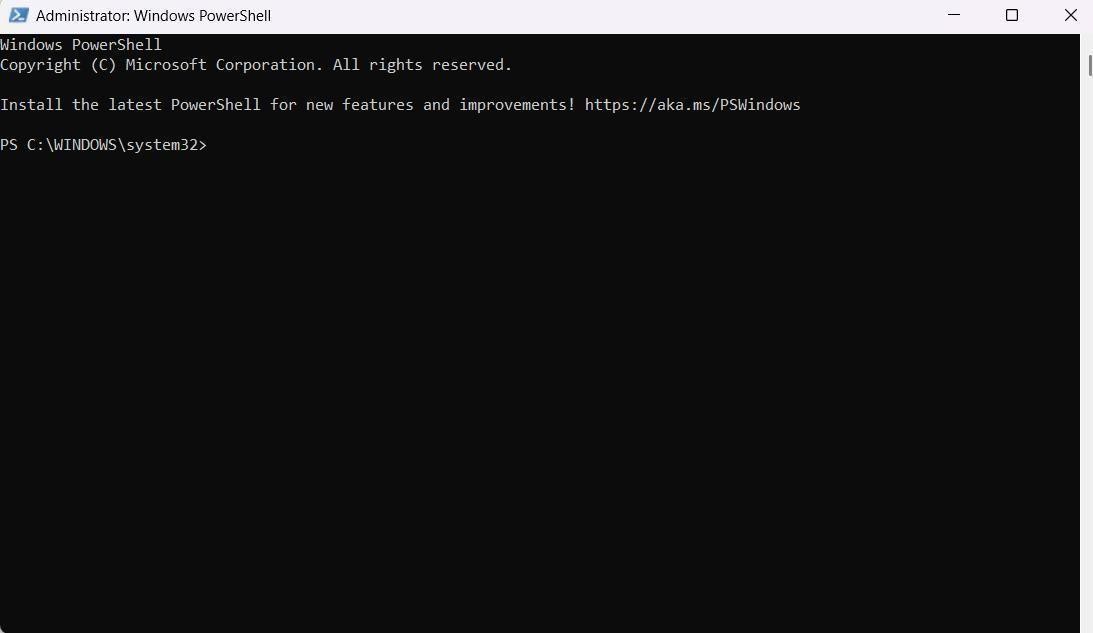
## 

## [P](j)art 1: Minikube Cluster Creation and Setup

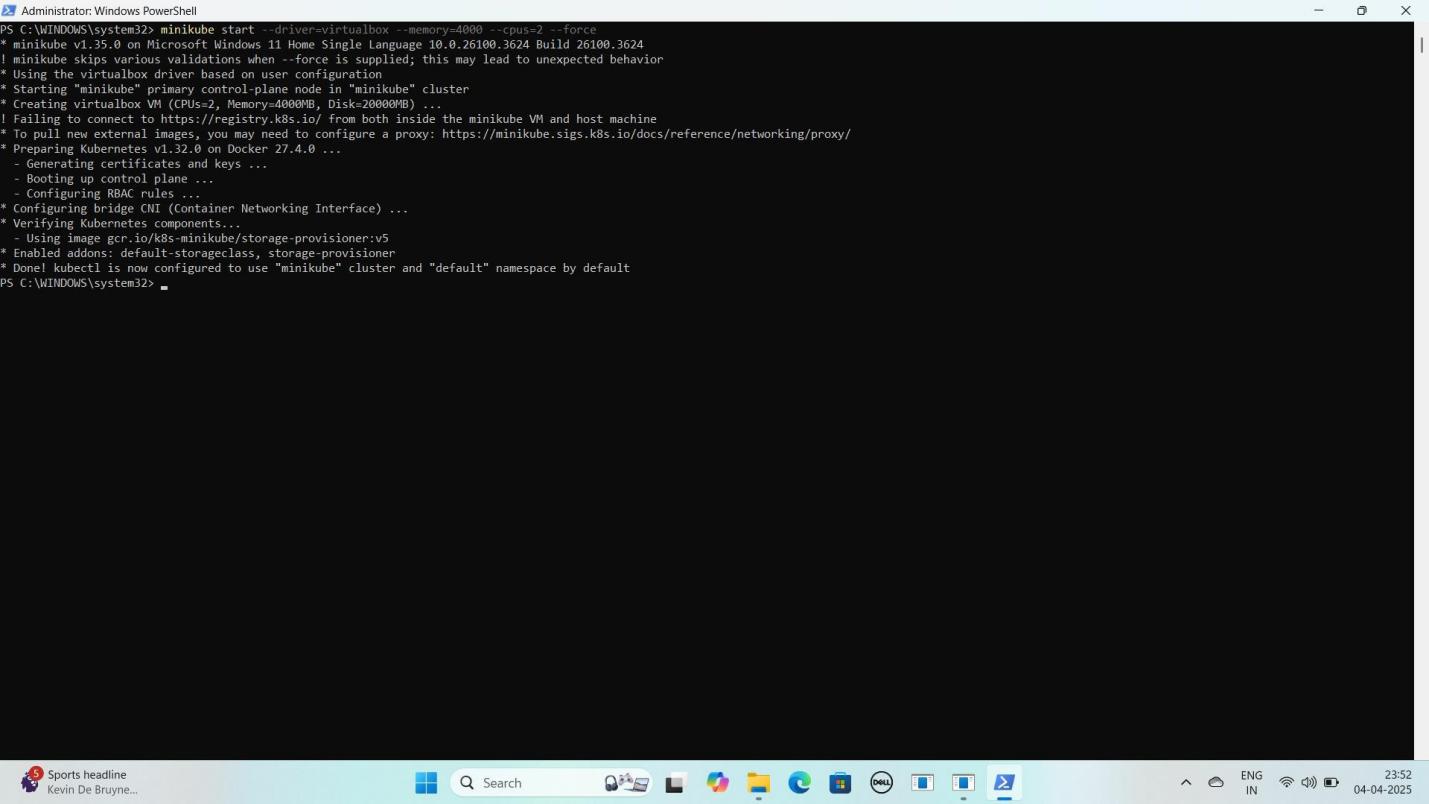
### Step 1: Windows search bar se PowerShell ko 'Run as administrator' ke through open karo aur Minikube start karo.

YE KUCH ISTARHA LAGEGA



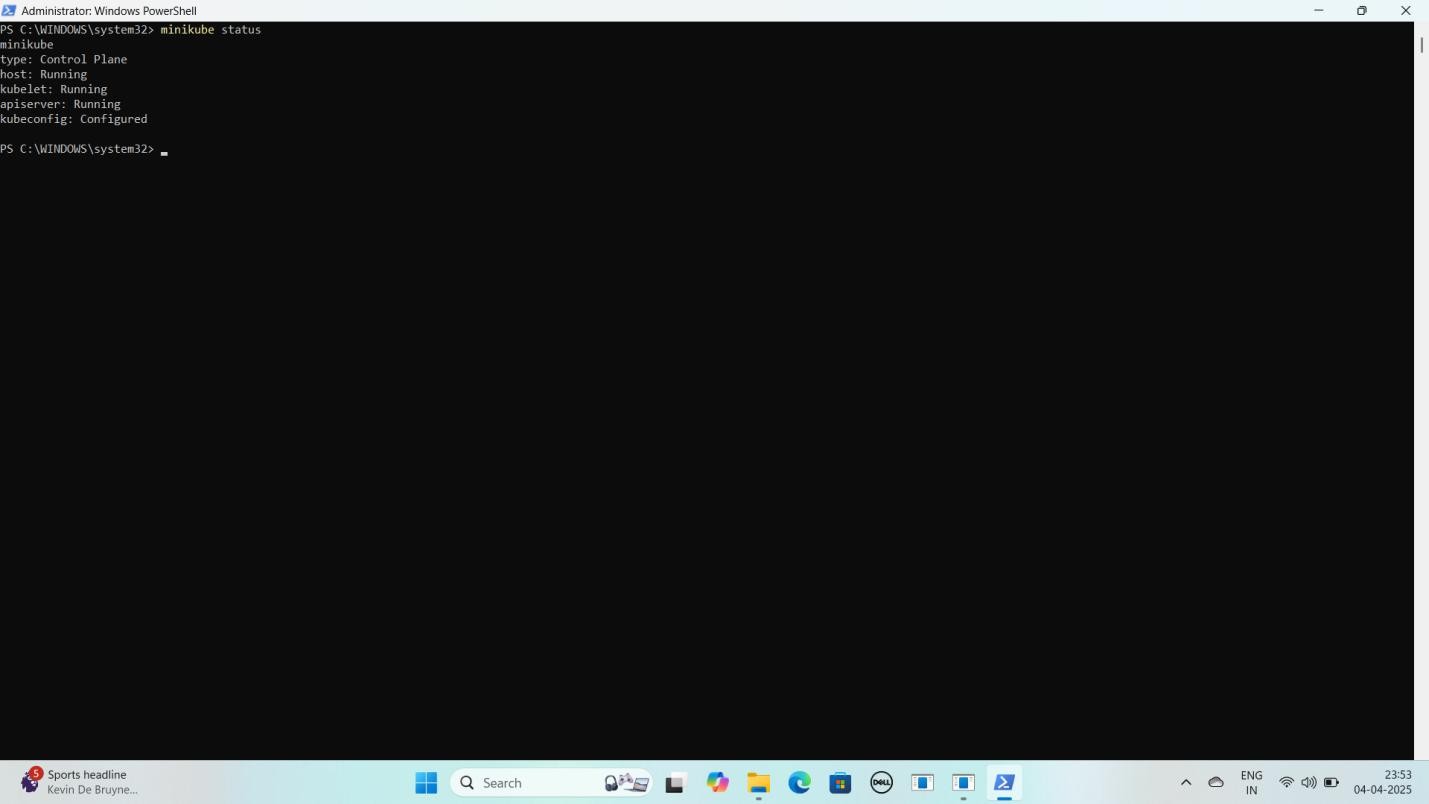


#### Minikube ko start karne ke liye ye command run kariye

minikube start --driver=virtualbox --memory=4000 --cpus=2 --force YE KUCH ISTARHA LAGEGA

#### Ab Minikube ka status check karne ke liye ye command run kariye

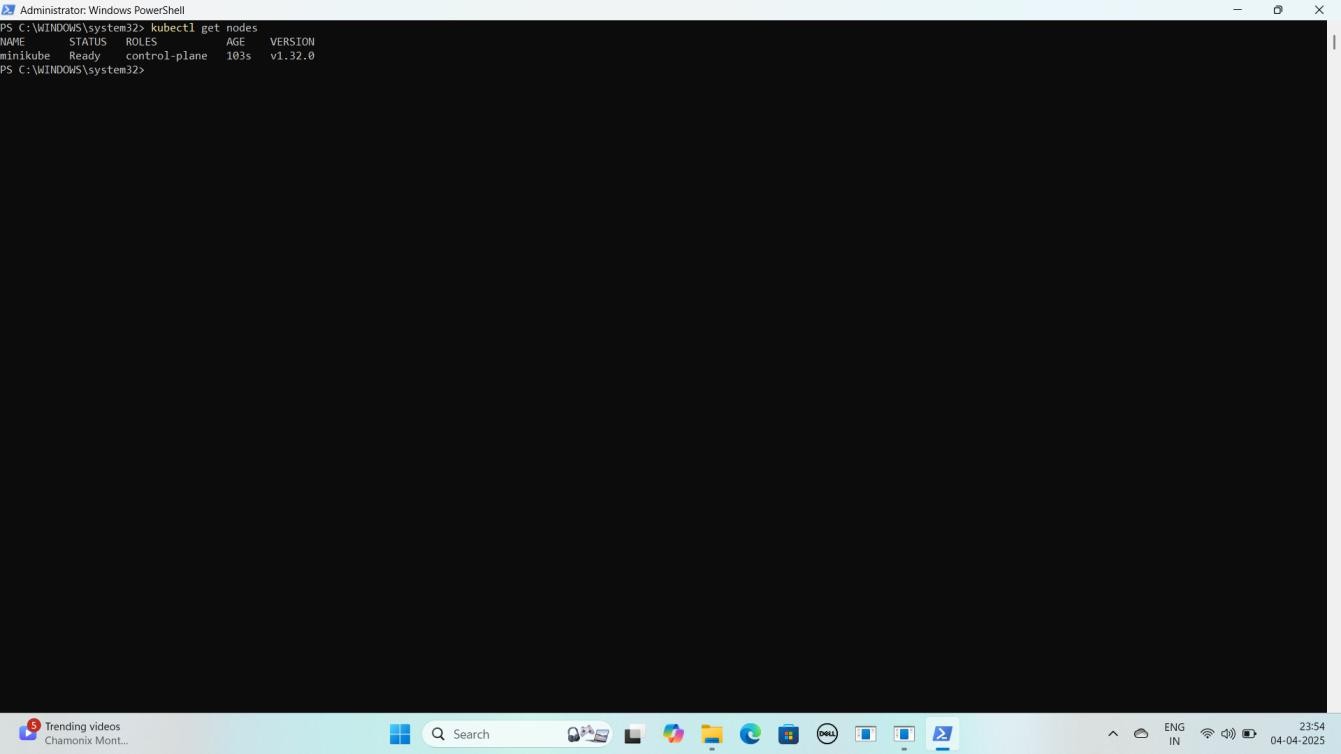
minikube status

YE KUCH ISTARHA LAGEGA

#### Ab Cluster Nodes check karne ke liye ye command run kare

kubectl get nodes

YE KUCH ISTARHA LAGEGA



**NOTE: Agar STATUS Ready show kar raha hao to sab sahi hai ab aap deployment kar sakte hai**

### Step 2: Project Ki GitHub Repository Clone Karein

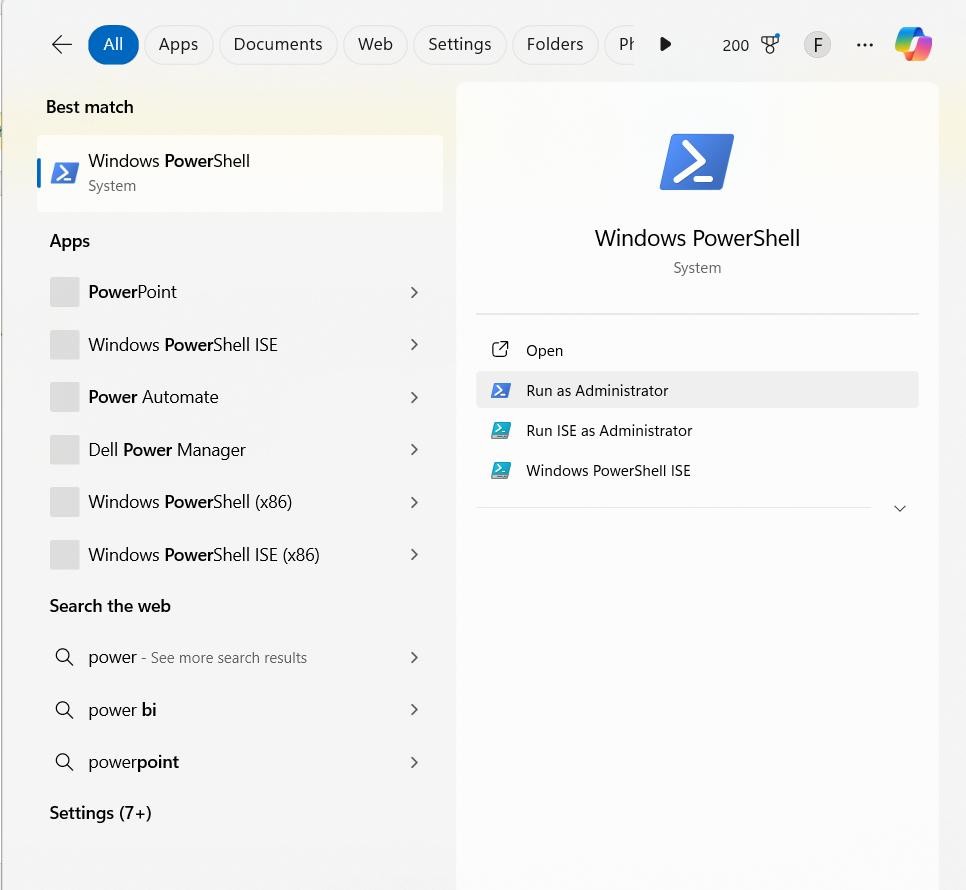
#### Laptop Mein Documents Folder Mein Jao

YE KUCH ISTARHA LEGAGA



#### PowerShell Ko "Run as Administrator" Open Karo

YE KUCH ISTARHA LAGEGA

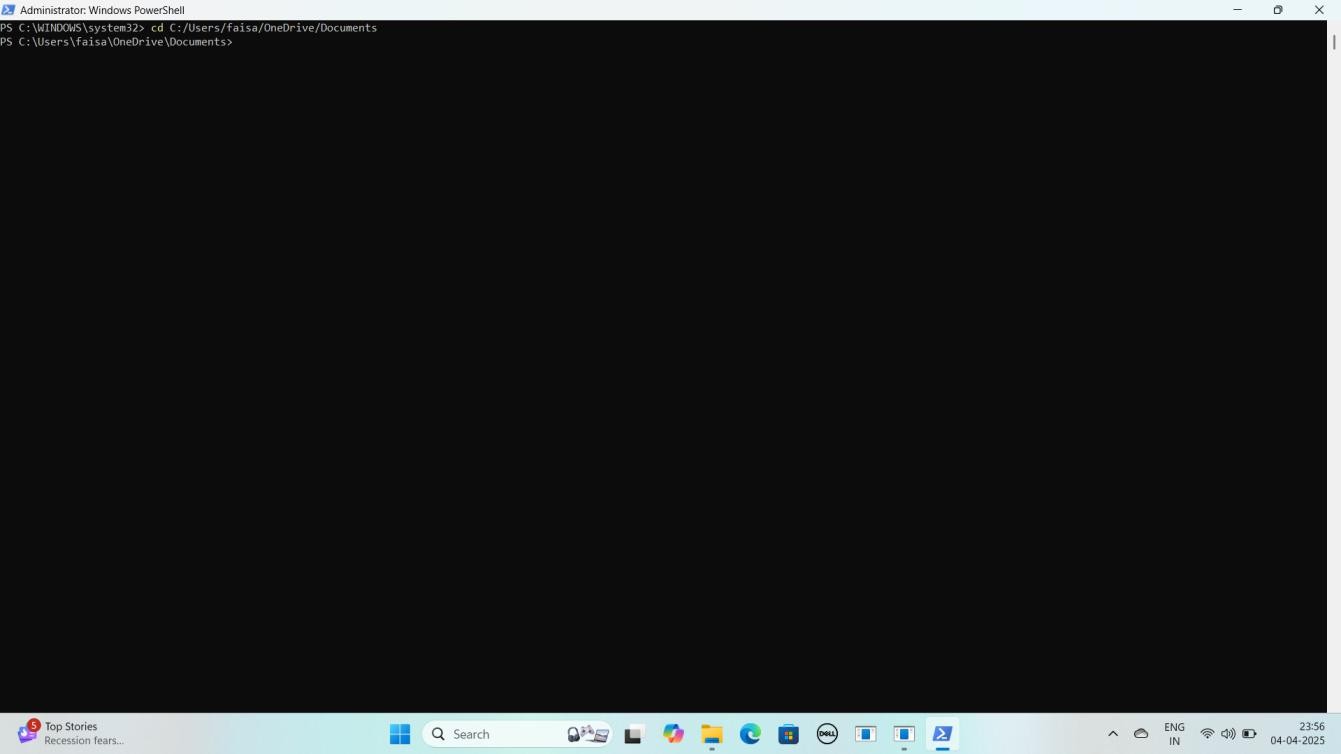


#### PowerShell me cd command ka use karke Documents folder ka path paste karein.

**Jaisa ki mere case me, path kuch aisa hai**

cd C:/Users/faisa/OneDrive/Documents YE KUCH ISTARHA LAGEGA





**NOTE: Isse aap Documents folder me Navigate ho jaoge."**

1. **GitHub Repository Clone Karo**

**As it is niche diya gaya URL copy karo aur PowerShell me paste karo**

git clone https://github.com/Faisalkhan45/MINIKUBE-CLUSTER-NODEPORT.git

YE KUCH ISTARHA LAGEGA



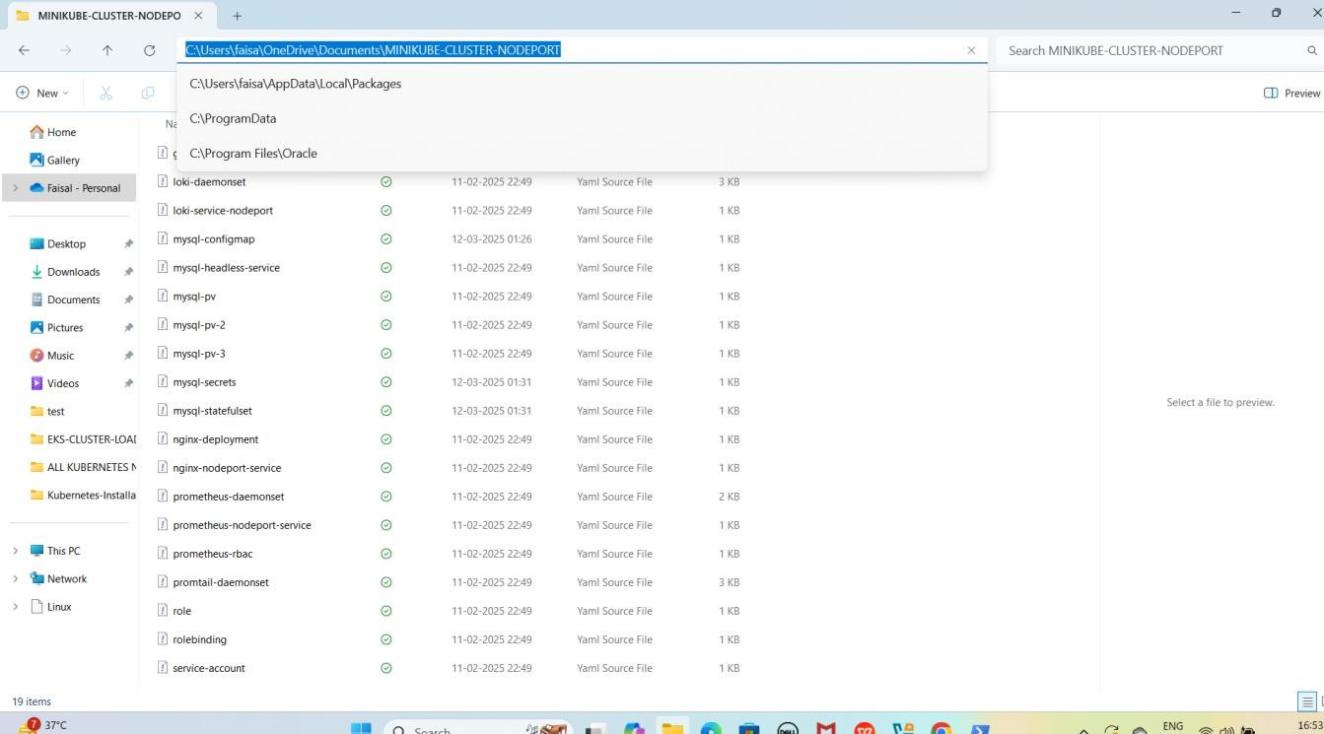
**NOTE: Repository clone hone ke baad, aapko "MINIKUBE-CLUSTER- NODEPORT" naam ka folder Documents me dikhega**

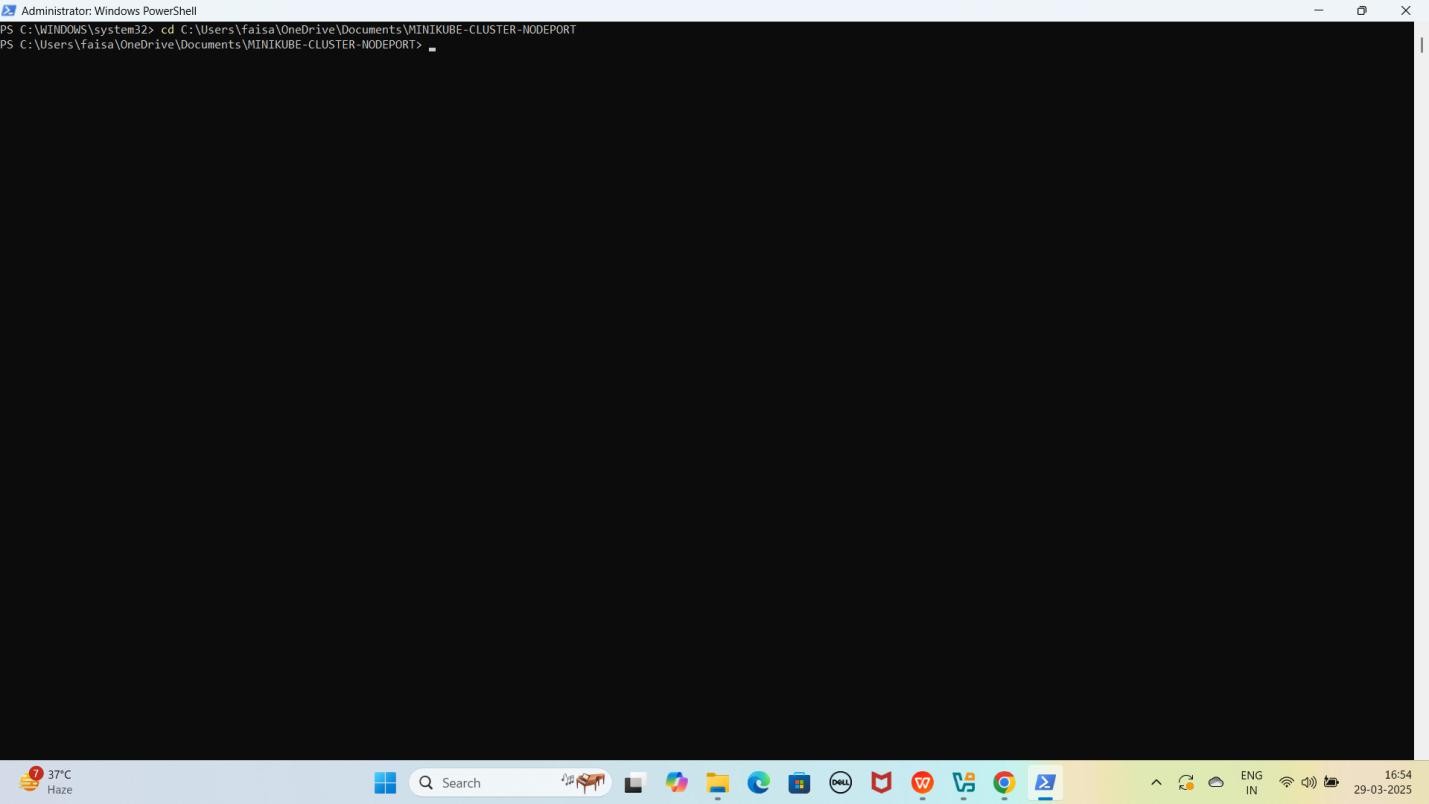
1. **EKS-CLUSTER-NODEPORT Folder Me Jao PowerShell me cd command ka use karo aur apna path paste karo**

**Jaise ki mere case me path kuch aisa hoga**

cd C:\Users\faisa\OneDrive\Documents\MINIKUBE-CLUSTER-NODEPORT

YE KUCH ISTARHA LAGEGA



**NOTE: Is command se aap PowerShell me MINIKUBE-CLUSTER-NODEPORT folder me Navigate ho jaoge.**

## Part 2: Nginx pods and Service Deployment

### Step 1: nginx-deployment.yaml File Ka Kaam

Yeh **file NGINX ke 3 replicas create** karne ke liye use hoti hai. Isko

**Kubernetes Deployment** ke andar likha jata hai.

JAISE KI:-

apiVersion: apps/v1 kind: Deployment metadata:

name: nginx-deployment spec:

replicas: 3 # 3 replicas for high availability selector:

matchLabels:

app: nginx template:

metadata:

labels:

app: nginx spec:

containers:

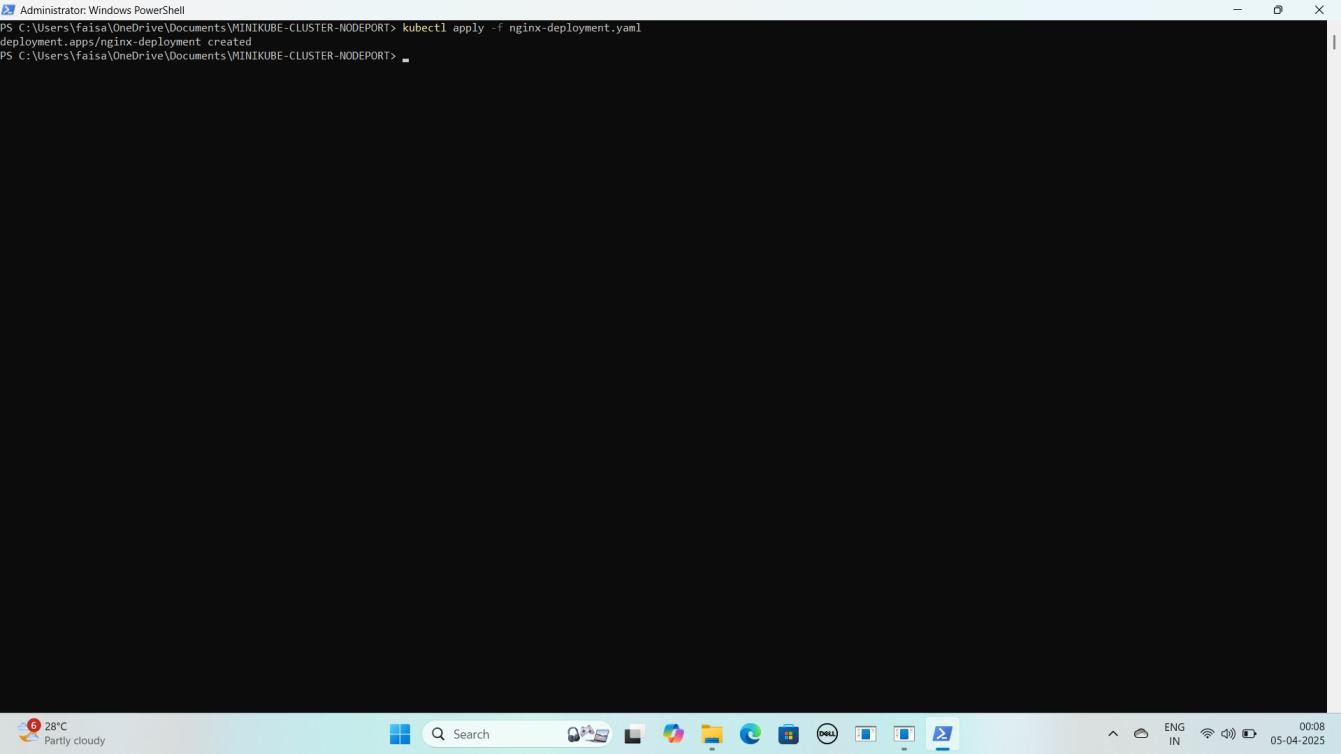
- name: nginx

image: nginx:latest # Latest Nginx image ports:

- containerPort: 80 # Expose port 80 in the container

### Deployment Apply Karo

kubectl apply -f nginx-deployment.yaml

YE KUCH ISTARHA LAGEGA

### Step 2: nginx-nodeport-service.yaml File Ka Kaam

Yeh **file NGINX pods** ko **expose** karne ke liye use hoti hai. Isme **NodePort service** define ki jati hai jo **NGINX pods** ko port **30007** par **access** karne ki ijazat deti hai.

JAISE KI:-

apiVersion: v1 kind: Service metadata:

name: nginx-service spec:

selector:

app: nginx ports:

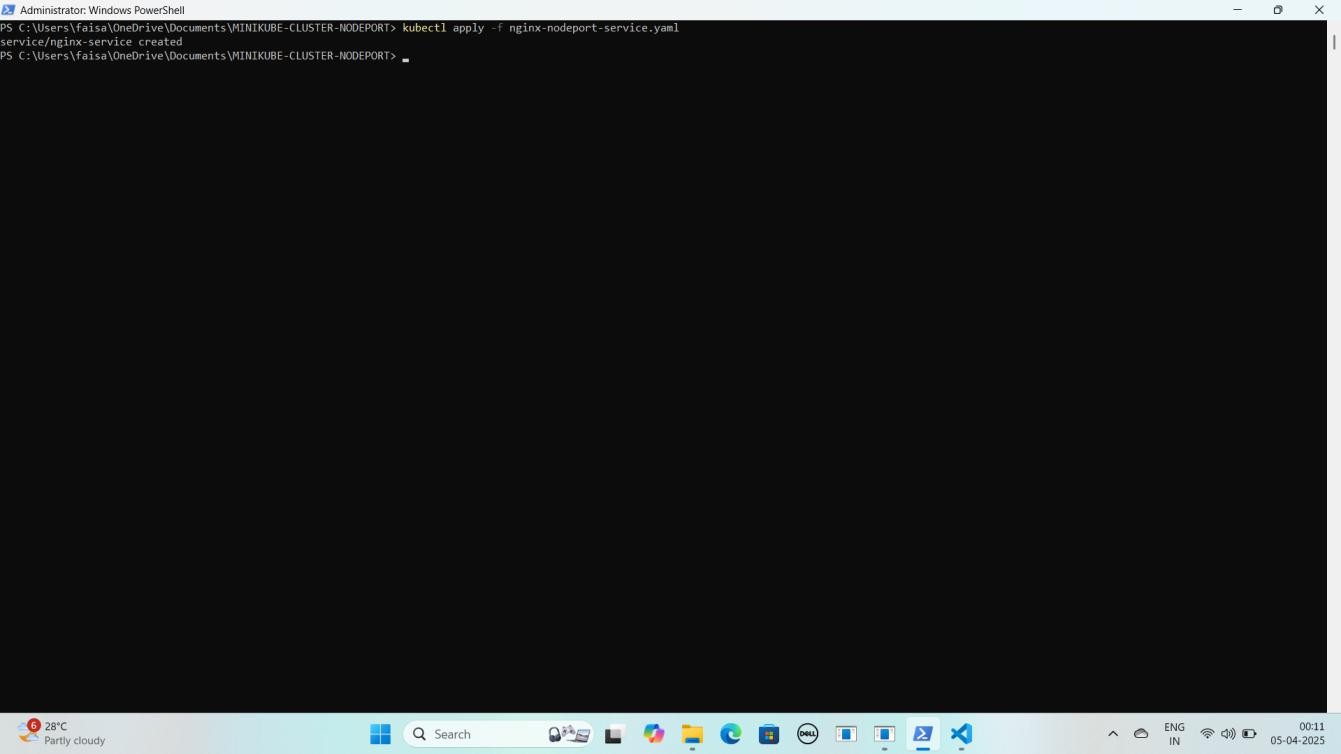
- port: 80 # Port inside the cluster

targetPort: 80 # Port to forward traffic to in the container

nodePort: 30007 # External port on which the service will be accessible type: NodePort # Type set to NodePort to expose externally

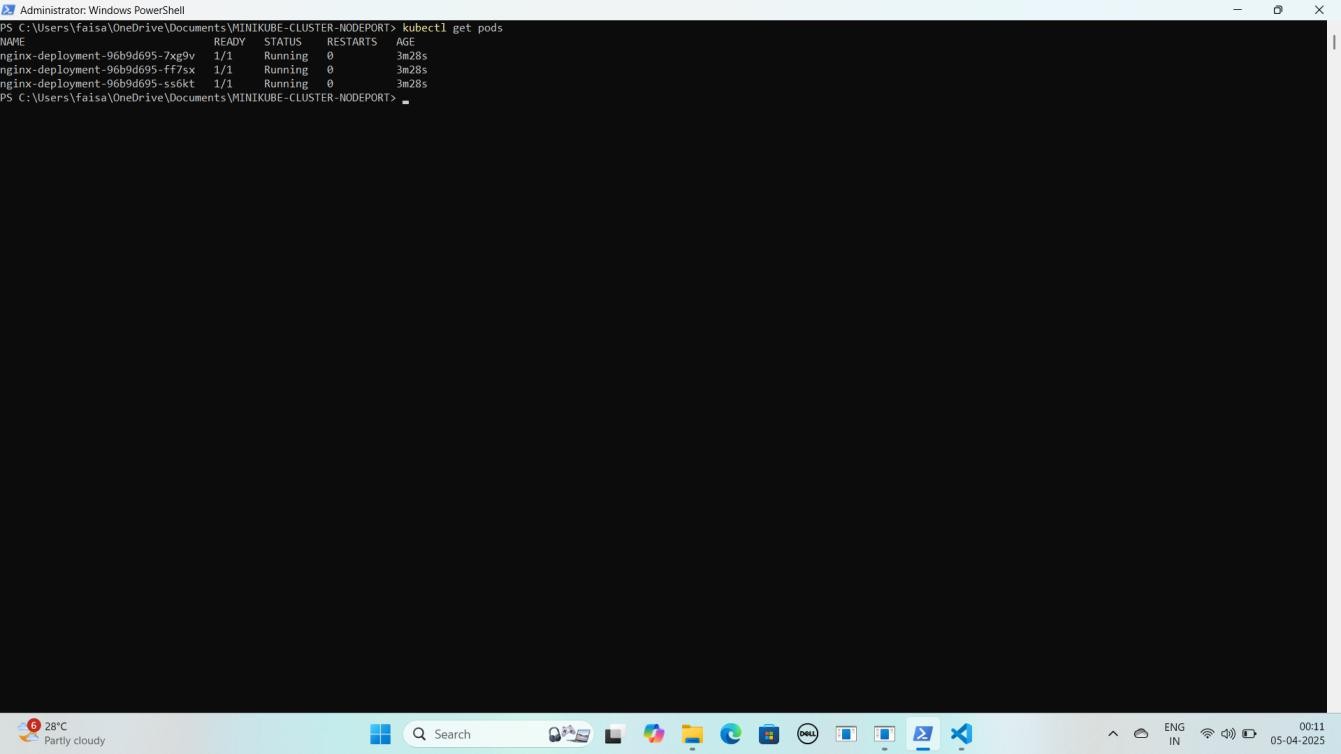
### Nginx Service Apply Karo

kubectl apply -f nginx-nodeport-service.yaml

YE KUCH ISTARHA LAGEGA

#### Pods check karne ke liye ye command run kariye

kubectl get pods

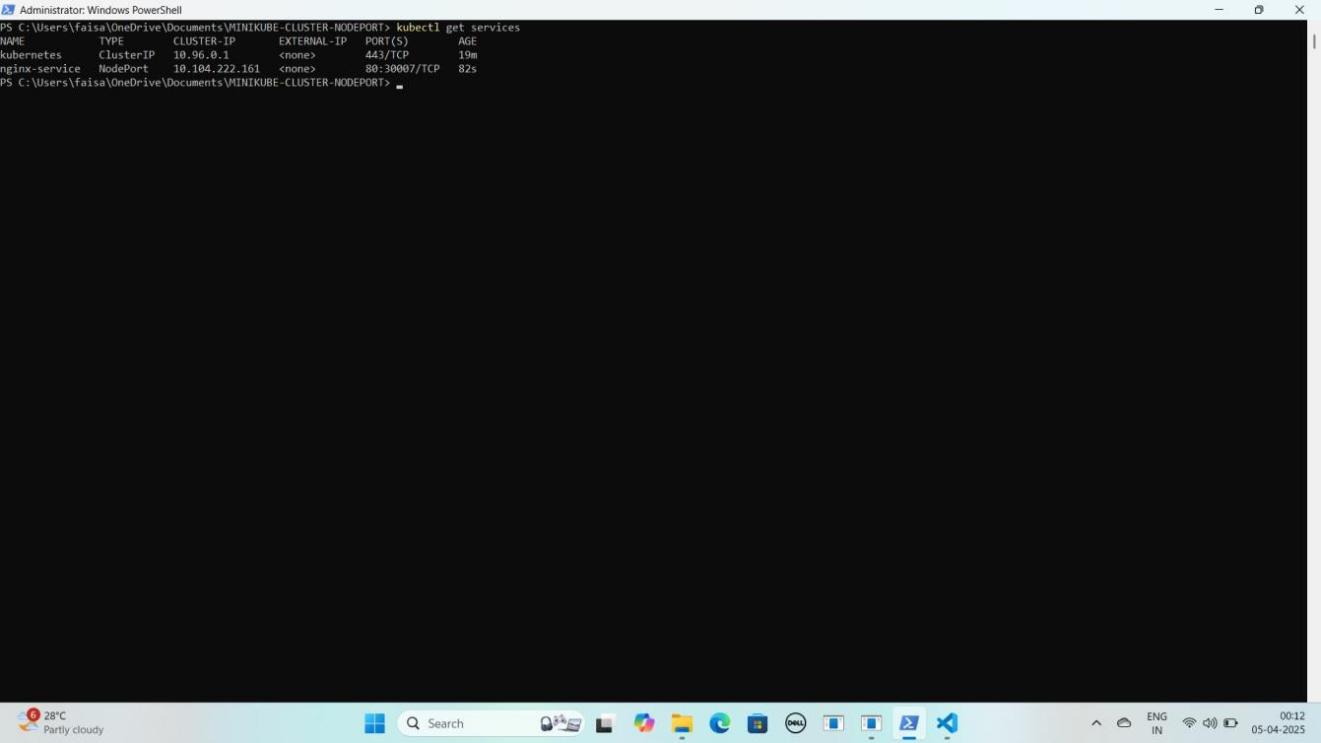
YE KUCH ISTARHA LAGEGA

#### NOTE: Agar STATUS Running show karraha hai to sab kuch sahi hai

1. **Serives check karne ke liye ye commad run kariye**

kubectl get services

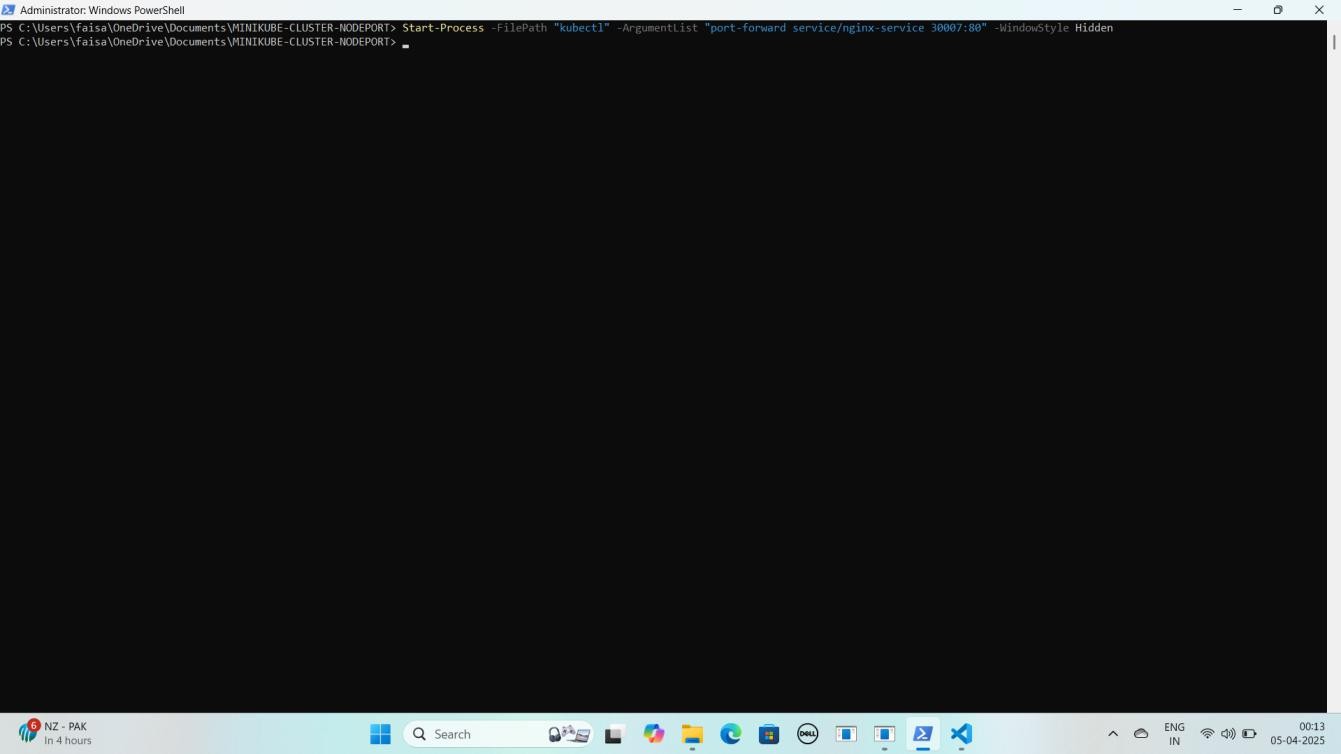
YE KUCH ISTARHA LAGEGA



#### Port Forwarding Karne ke liye ye command run karein

Start-Process -FilePath "kubectl" -ArgumentList "port-forward service/nginx- service 30007:80" -WindowStyle Hidden

YE KUCH ISTARHA LAGEGA

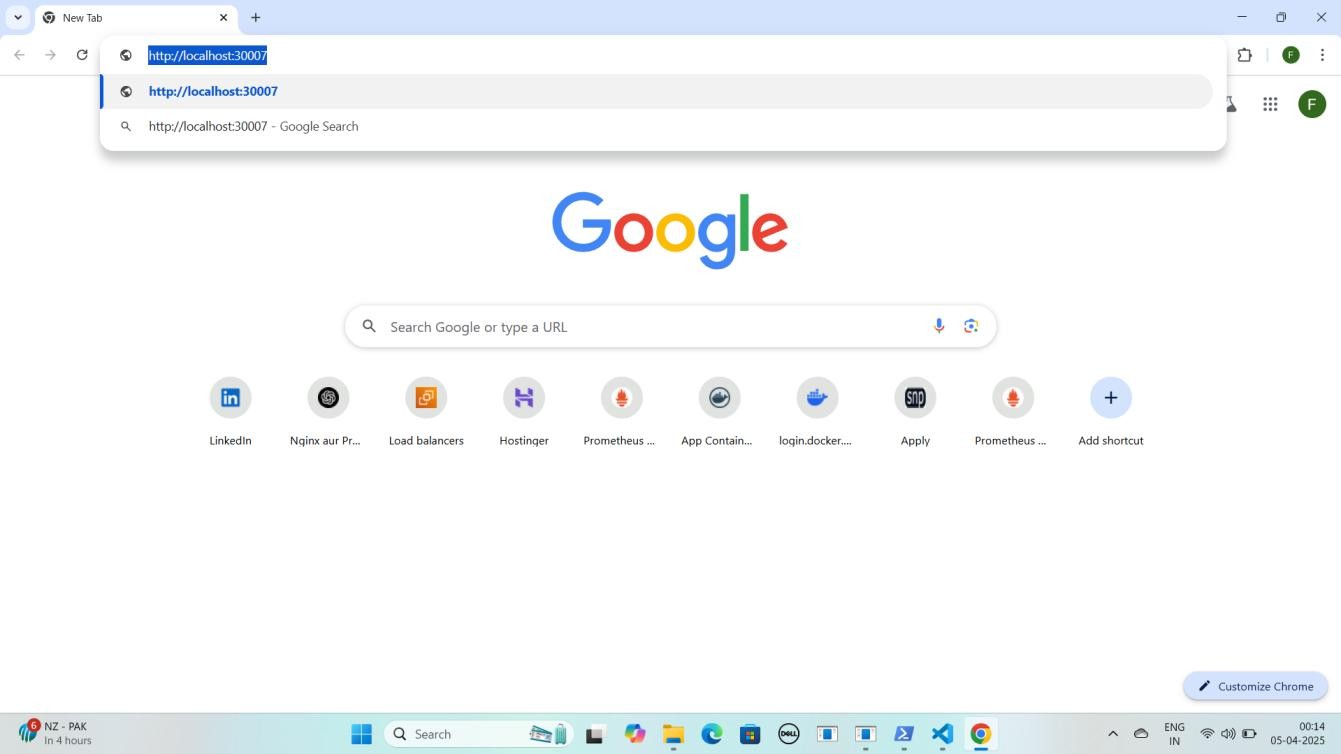


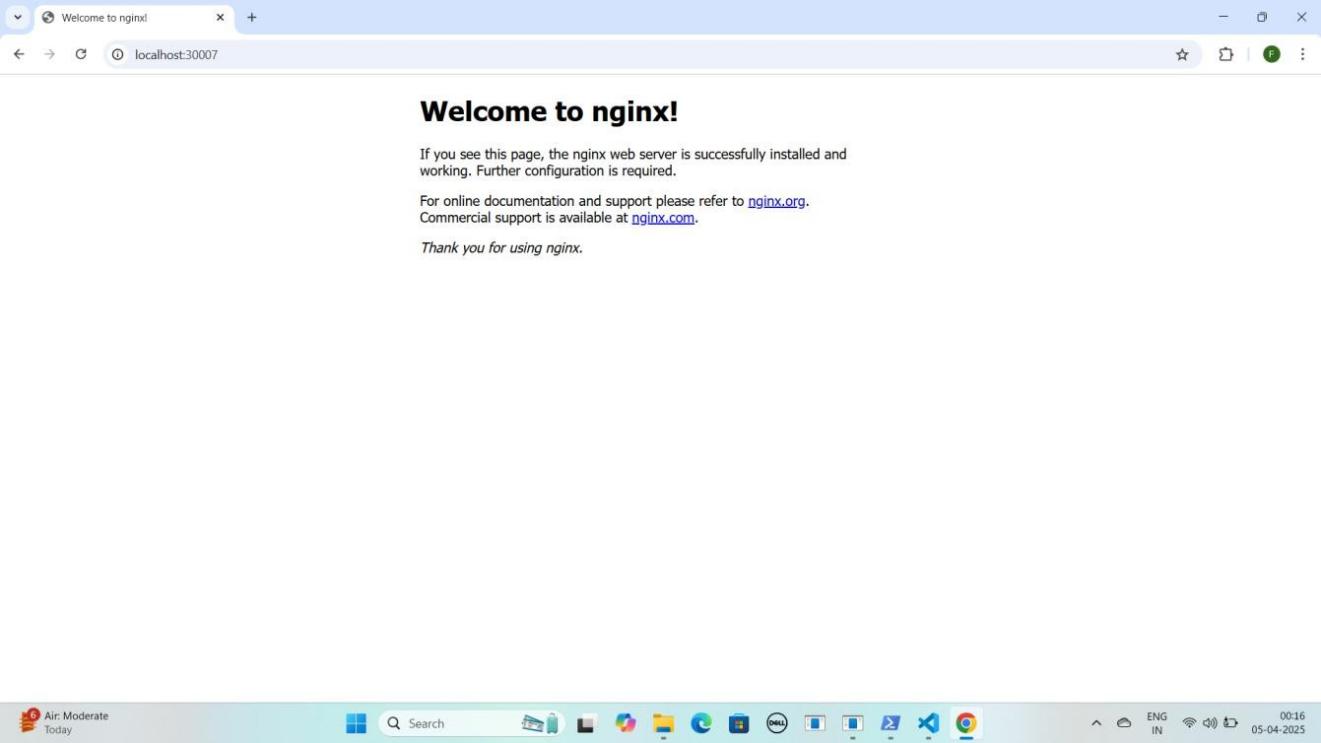
#### NOTE: Ab aapka Nginx expose ho chuka hai. Ab apne local system par Nginx ko NodePort ke saath browser me run kariye.

**NGINX Localhost NodePort Address :**

[http://localhost:30007](kj)

YE KUCH ISTARHA LAGEGA





## Part 3: MySQL Secrets and ConfigMap Configuration

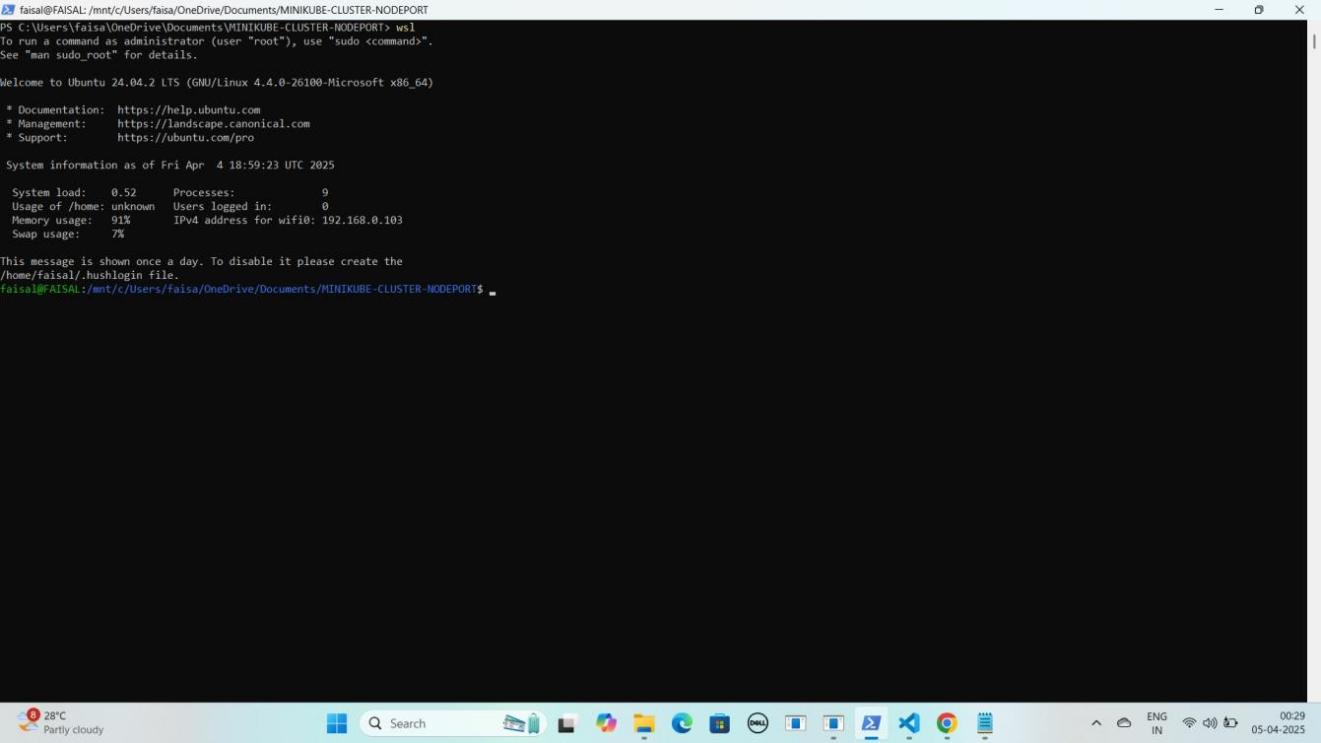
### Step 1: mysql-secrets.yaml File Ka Kaam

Ye **file username aur password ko securely store** karne ke liye use hoti hai. Kyunki **Secrets secure hoti hain**, is wajah se hum **Base64 encoding** ka use karke **username aur password** ko **encrypt** kar dete hain.

#### MYSQL ka username aur password encode karna:-

1. **Windows ke WSL (Windows Subsystem for Linux) ka use kar ke hum Base64 encoding kar sakte hain. wsl type karein aur yeh commands run karein**

wsl

YE KUCH ISTARHA LAGEGA

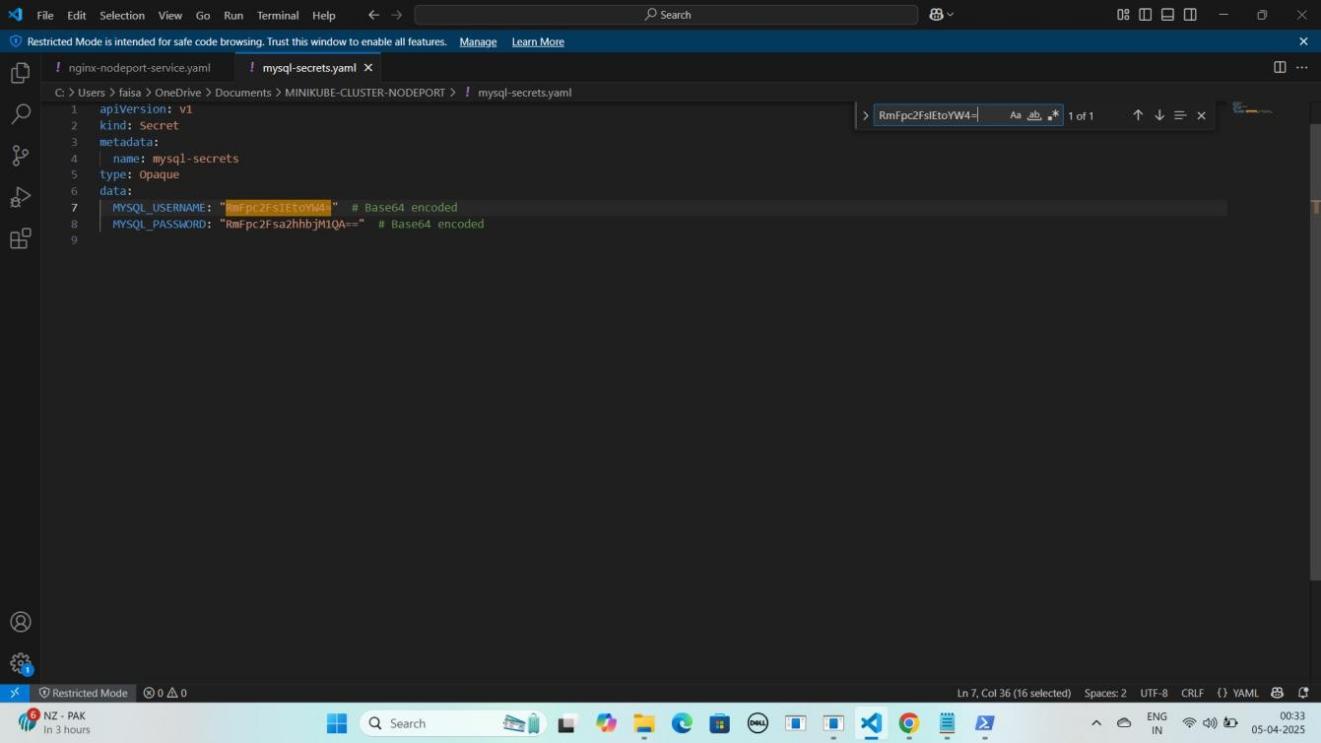
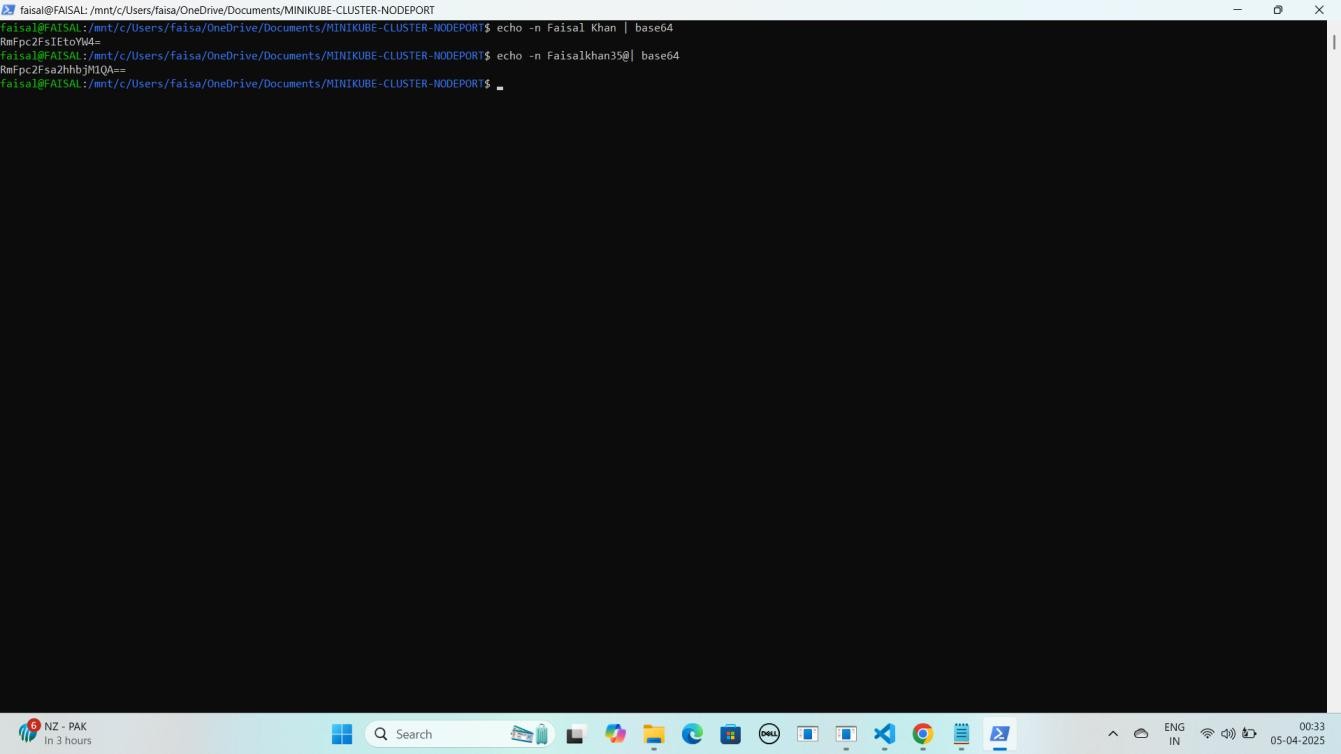
#### Username encode karein

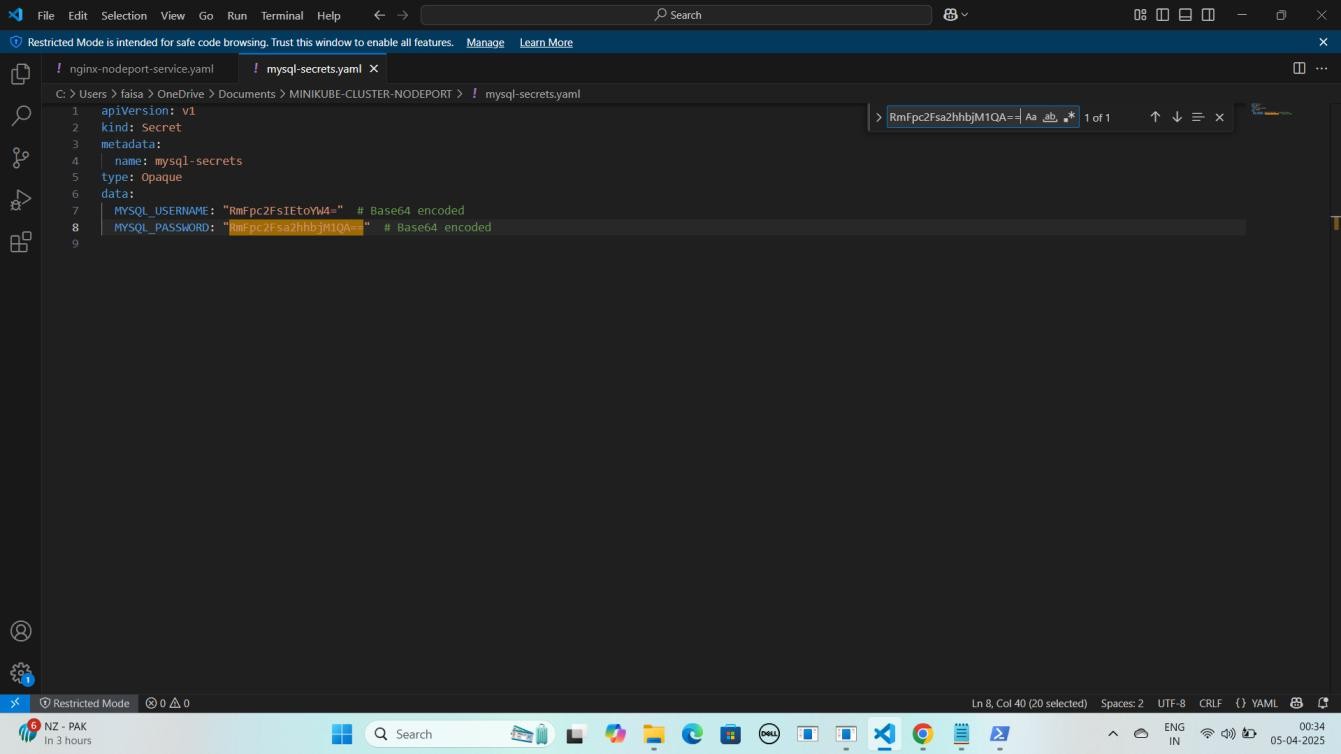
* + echo -n Faisal Khan | base64

#### Password encode karein

* + echo -n Faisalkhan35@| base64

YE KUCH ISTARHA LAGEGA





#### NOTE: Ye command aapko Base64 encoded values degi, jo hum mysql- secrets.yaml file me store karenge. Lekin aapko apne Username aur Password ke hisaab se mysql-secrets.yaml file update karni hogi.

JAISE KI:-

apiVersion: v1 kind: Secret metadata:

name: mysql-secrets type: Opaque

data:

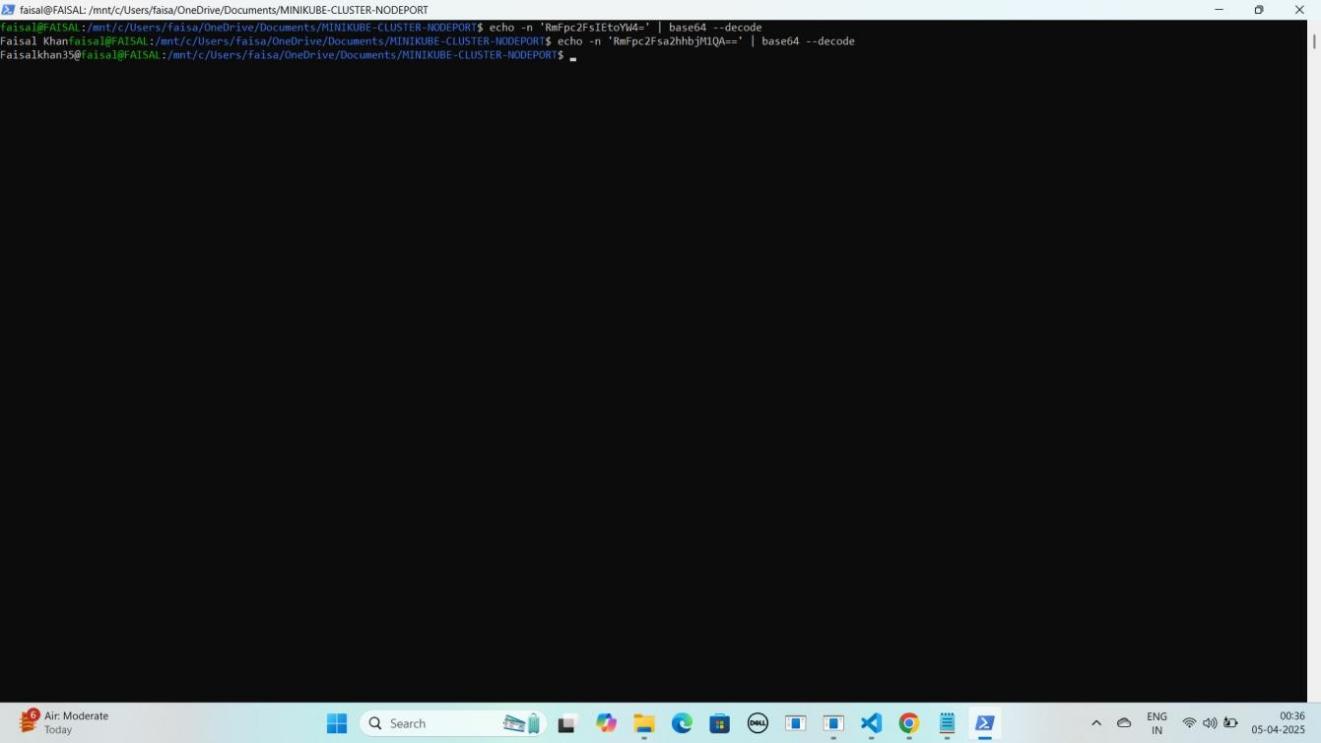
MYSQL\_USERNAME: "RmFpc2FsIEtoYW4=" # Base64 encoded MYSQL\_PASSWORD: "RmFpc2Fsa2hhbjM1QA==" # Base64 encoded

#### Agar aap encoded username aur password ko wapas original form me laana chahte hain, toh WSL me yeh commands run karein

1. **Username decode karein**
   * echo -n 'RmFpc2FsIEtoYW4=' | base64 --decode

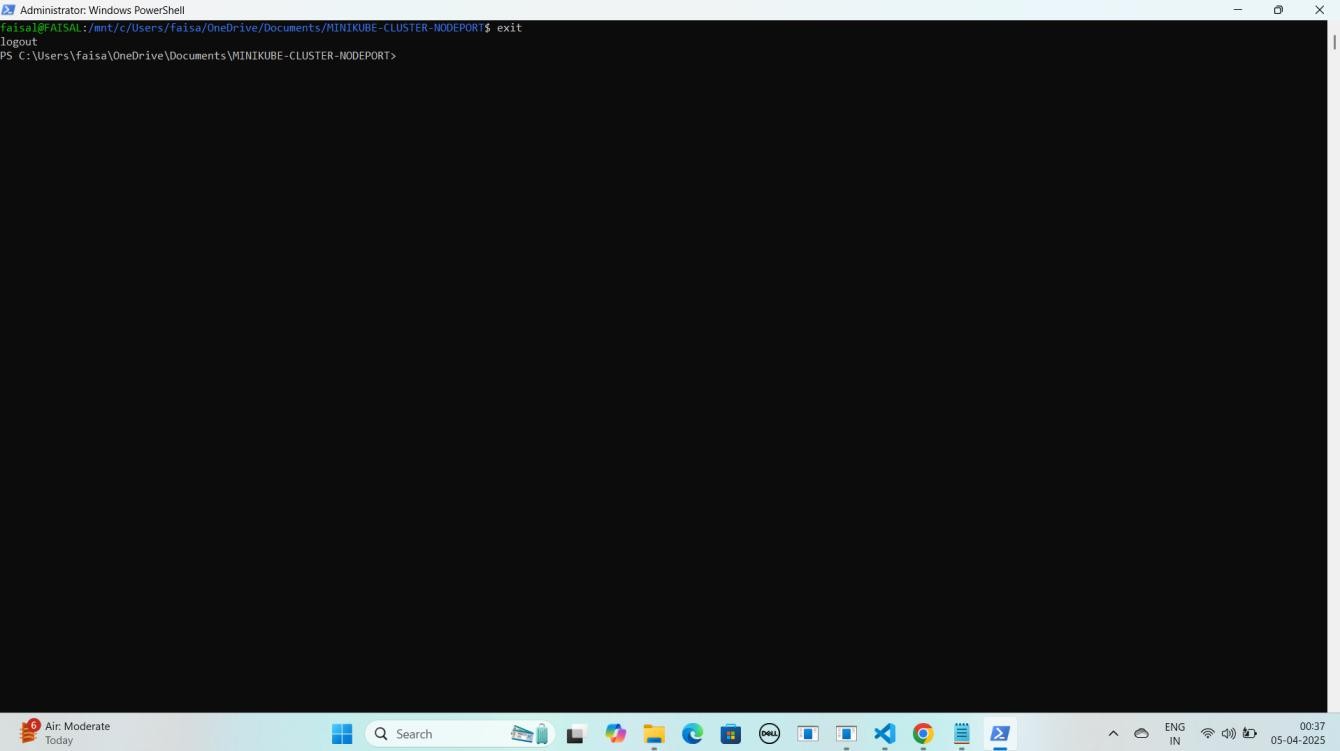
#### Password decode karein

* + echo -n 'RmFpc2Fsa2hhbjM1QA==' | base64 --decode

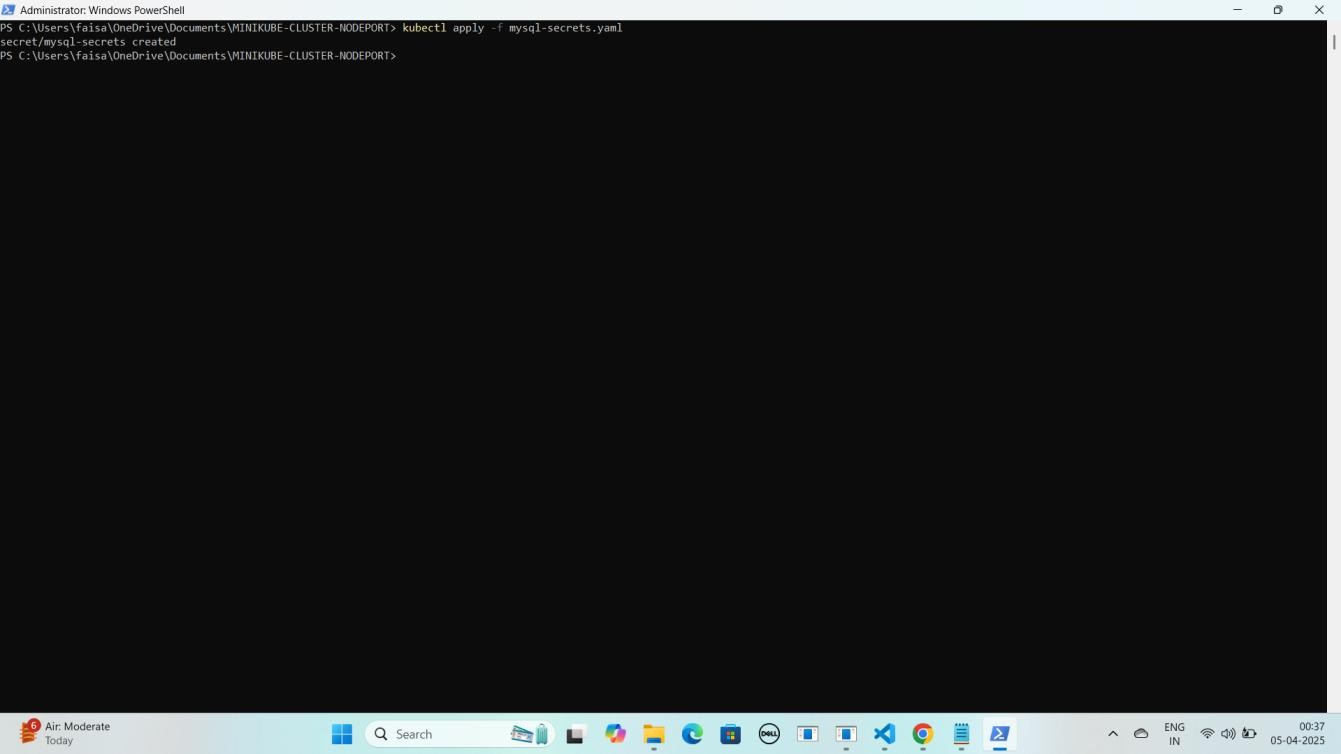
YE KUCH ISTARHA LAGEGA

#### Ab WSL se exit hogao exit hone ke liye exit type karo

exit

YE KUCH ISTARHA LAGEGA

### Secret Apply Karo

kubectl apply -f mysql-secrets.yaml YE KUCH ISTARHA LAGEGA

### Step 2: mysql-configmap.yaml File Ka Kaam

Yeh **file non-sensitive data** store karne ke liye use hoti hai, jaise

* **Database ka naam**
* **Host ka naam (hostname)**
* **Port number**

Isko **Kubernetes ConfigMap** ke andar likha jata hai.

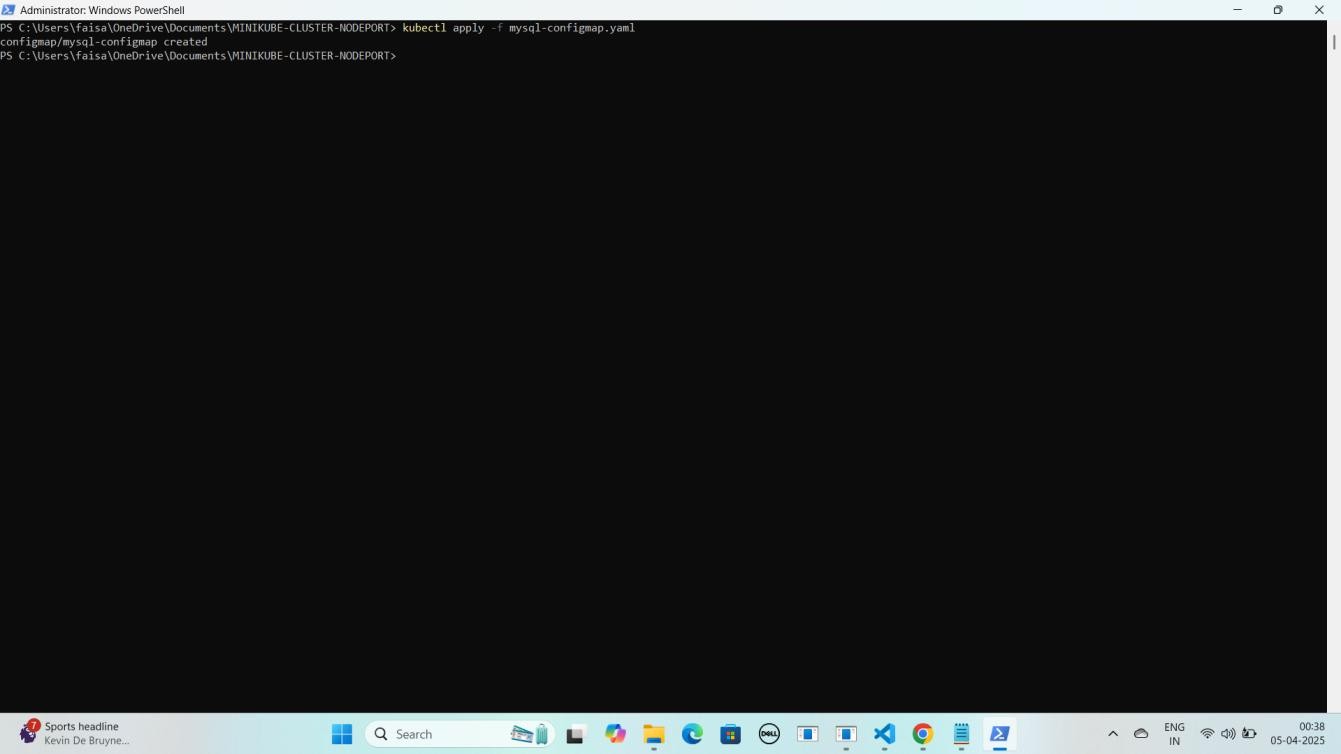
JAISE KI:-

apiVersion: v1 kind: ConfigMap metadata:

name: mysql-configmap data:

MYSQL\_DATABASE: "faisal-db" MYSQL\_HOST: "mysql-headless-service" MYSQL\_PORT: "3306"

### ConfigMap Apply Karo

kubectl apply -f mysql-configmap.yaml YE KUCH ISTARHA LAGEGA

### Step 3: service-account.yaml File Ka Kaam

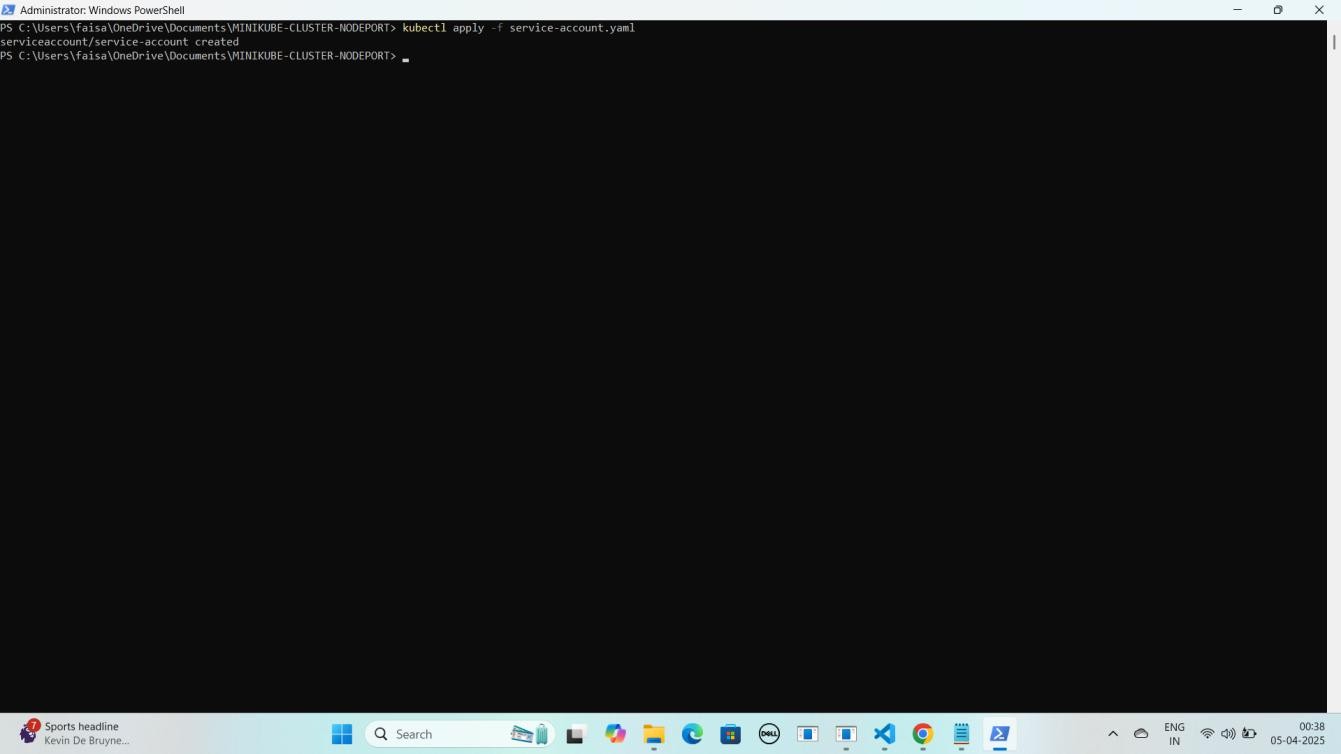
Yeh **file Secrets aur ConfigMaps** ko **access** karne ke liye **permissions** dene ke liye use hoti hai. Isko **Kubernetes Service Account** banane ke liye likha jata hai.

JAISE KI:-

apiVersion: v1 kind: ServiceAccount metadata:

name: service-account

### Service Account Apply Karo

kubectl apply -f service-account.yaml YE KUCH ISTARHA LAGEGA

### Step 4: role.yaml File Ka Kaam

Yeh **file Secrets aur ConfigMaps** ke liye **read-only permissions** define karne ke liye use hoti hai. Isko **Kubernetes Role** banane ke liye likha jata hai.

JAISE KI:-

apiVersion: rbac.authorization.k8s.io/v1 kind: Role

metadata:

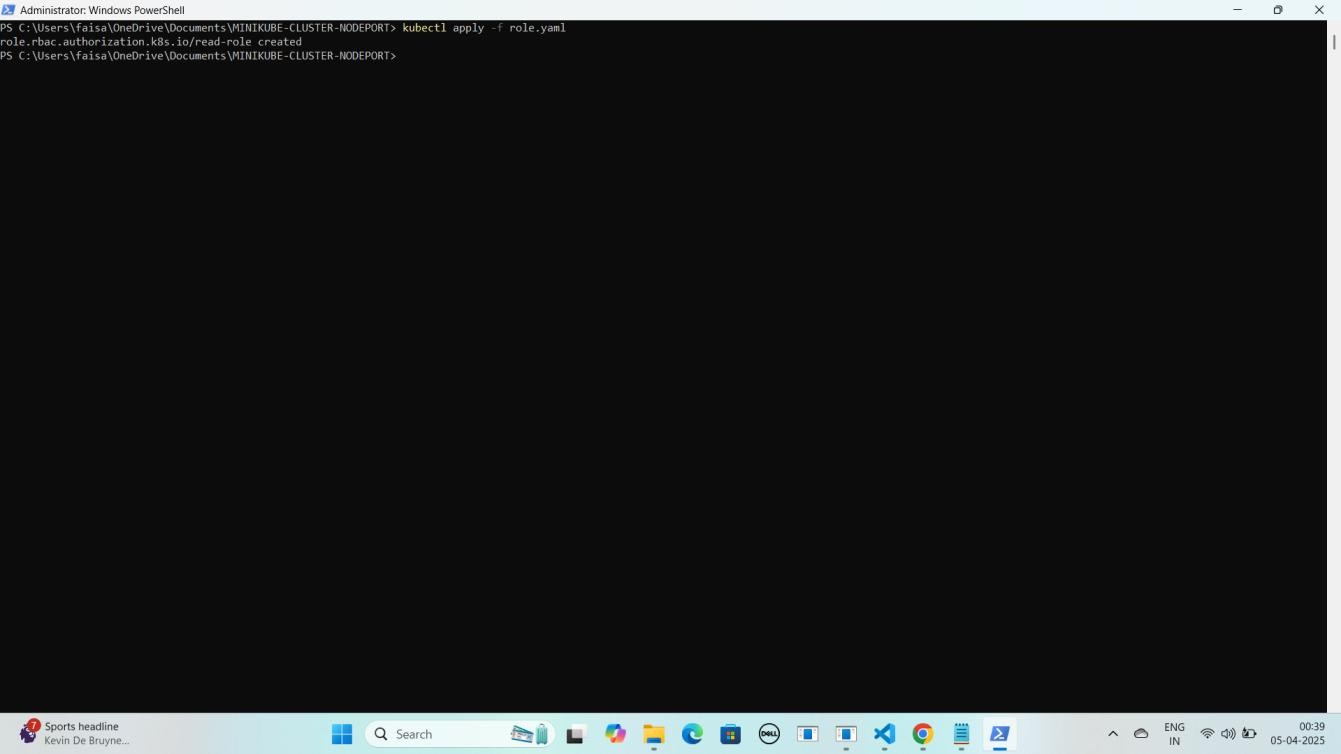
name: read-role rules:

* apiGroups: [""]

resources: ["secrets", "configmaps"] verbs: ["get", "list"]

* apiGroups: [""] resources: ["pods"] verbs: ["get", "list"]
* apiGroups: [""] resources: ["events"] verbs: ["get", "list"]
* apiGroups: [""] resources: ["endpoints"] verbs: ["get", "list"]

### Role Apply Karo

kubectl apply -f role.yaml YE KUCH ISTARHA LAGEGA

### Step 5: rolebinding.yaml File Ka Kaam

Yeh **file Service Account** ko **Secrets aur ConfigMaps** ke liye **read-only permissions** assign karne ke liye use hoti hai. Isko **Kubernetes RoleBinding** banane ke liye likha jata hai.

JAISE KI:-

apiVersion: rbac.authorization.k8s.io/v1 kind: RoleBinding

metadata:

name: read-write-only-binding subjects:

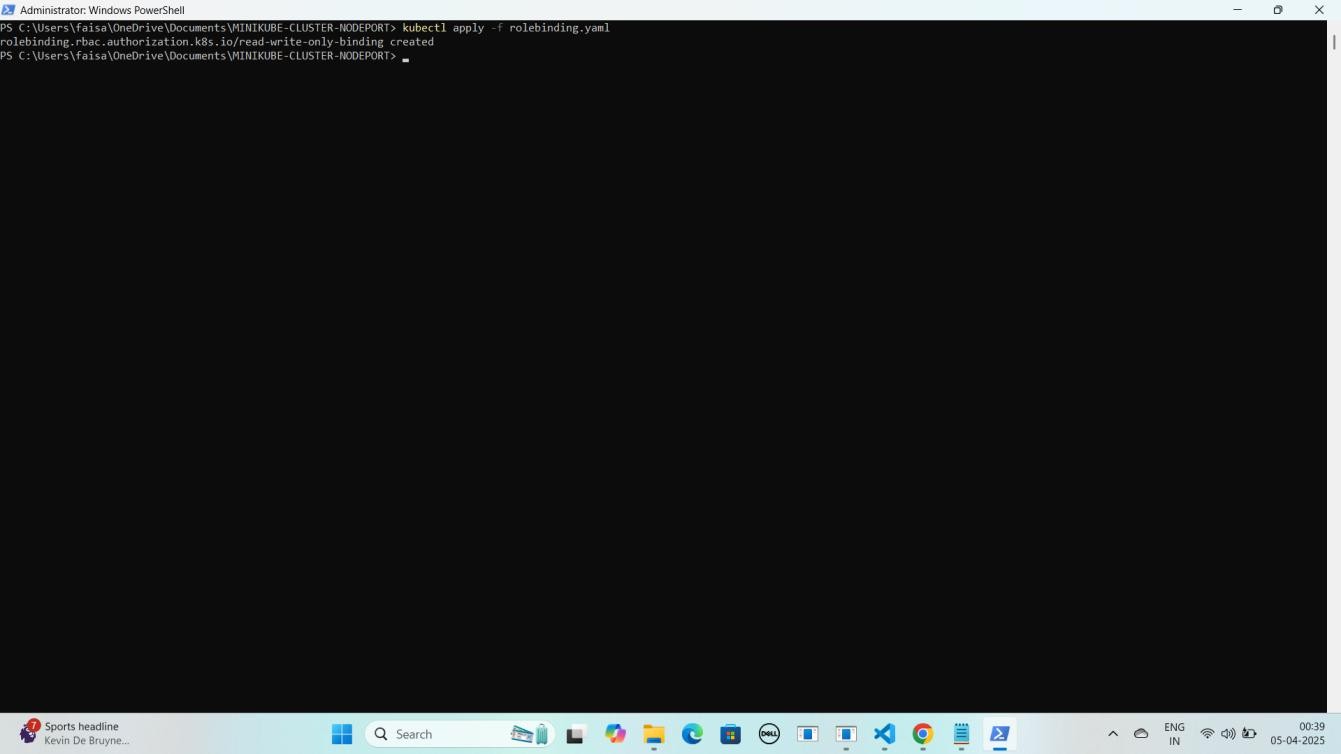
- kind: ServiceAccount name: service-account namespace: default

roleRef:

kind: Role name: read-role

apiGroup: rbac.authorization.k8s.io

### RoleBinding Apply Karo

kubectl apply -f rolebinding.yaml YE KUCH ISTARHA LAGEGA

## Part 4: MYSQL Database Pods and Services Deployment

### Step 1: mysql-headless-service.yaml File Ka Kaam

Yeh **file MySQL pods** ke beech **data synchronization** aur sharing ke liye use hoti hai. Isko **Kubernetes Headless Service** ke taur par define kiya jata hai.

JAISE KI:-

apiVersion: v1 kind: Service metadata:

name: mysql-headless-service spec:

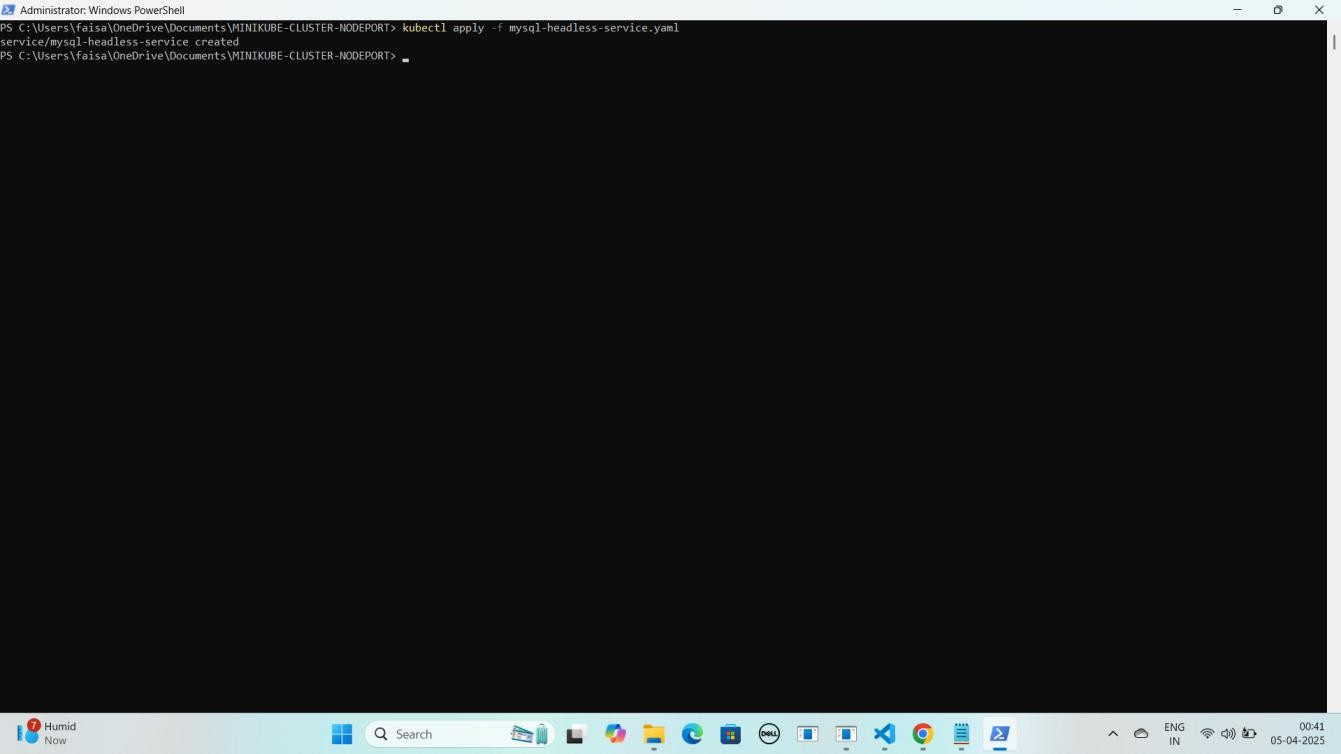
clusterIP: None # Makes it a headless service selector:

app: mysql ports:

- name: mysql port: 3306

targetPort: 3306

### Headless Service Apply Karo

kubectl apply -f mysql-headless-service.yaml YE KUCH ISTARHA LAGEGA

### Step 2: mysql-pv.yaml File Ka Kaam

Yeh **file cluster** mein ek **static volume** banane ke liye use hoti hai. Isko **Kubernetes Persistent Volume (PV)** create karne ke liye likha jata hai. Hum **3 PV** create karenge kyunki next step mein **StatefulSet 3 database pods** banayega, jinko **storage** ki zaroorat hogi.

JAISE KI:-

apiVersion: v1

kind: PersistentVolume metadata:

name: mysql-pv spec:

capacity:

storage: 5Gi accessModes:

- ReadWriteOnce persistentVolumeReclaimPolicy: Retain hostPath:

path: "/mnt/mysql-data"

apiVersion: v1

kind: PersistentVolume metadata:

name: mysql-pv-2

spec:

capacity:

storage: 5Gi accessModes:

- ReadWriteOnce persistentVolumeReclaimPolicy: Retain hostPath:

path: /mnt/data/mysql2

apiVersion: v1

kind: PersistentVolume metadata:

name: mysql-pv-3 spec:

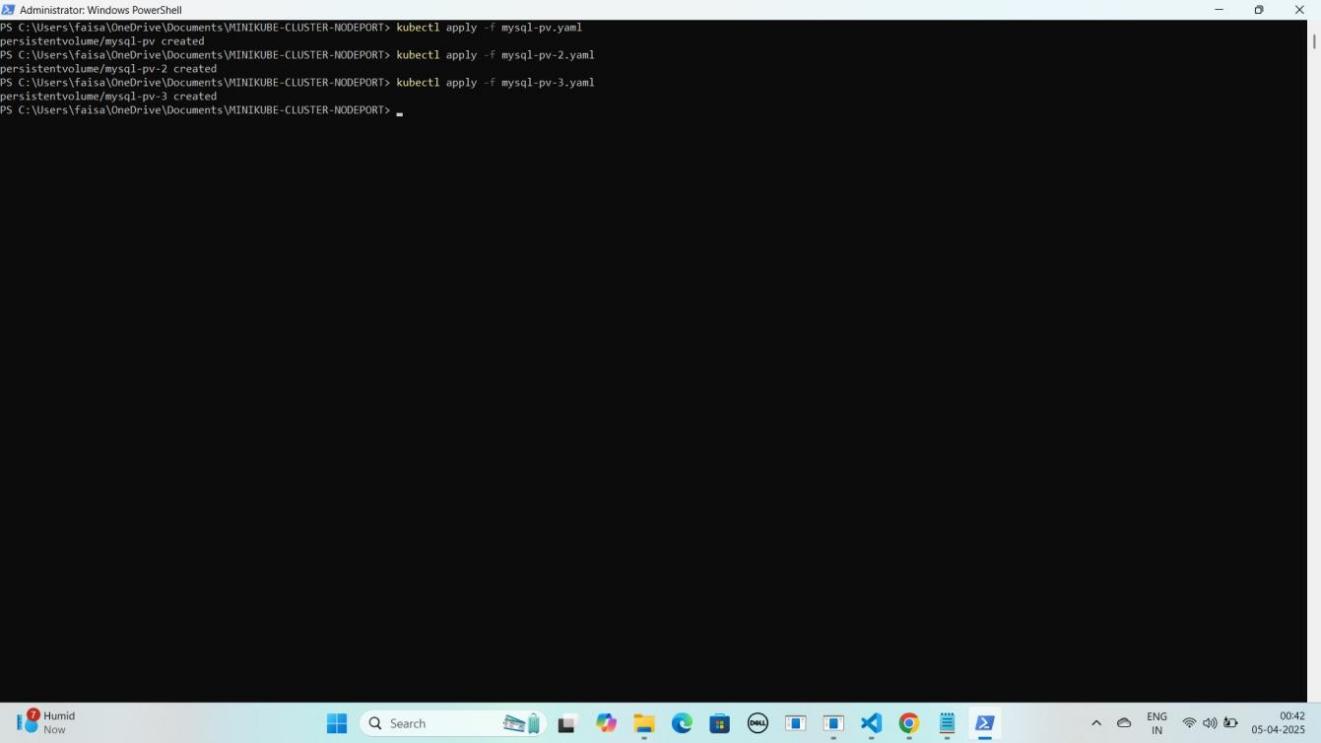
capacity:

storage: 5Gi accessModes:

- ReadWriteOnce persistentVolumeReclaimPolicy: Retain hostPath:

path: /mnt/data/mysql-3

**Persistent Volumes Apply Karo** kubectl apply -f mysql-pv.yaml kubectl apply -f mysql-pv-2.yaml kubectl apply -f mysql-pv-3.yaml

YE KUCH ISTARHA LAGEGA

### Step 3: mysql-statefulset.yaml File Ka Kaam

Yeh **file database** ke liye use hoti hai jisme hum **MySQL ke 3 replicas** define karte hain. Isme **Persistent Volume Claim (PVC)** use hoti hai jo **StatefulSet** ke **deployments** ko **Persistent Volume (PV)** se connect karti hai.

JAISE KI:-

apiVersion: apps/v1 kind: StatefulSet metadata:

name: mysql spec:

serviceName: "mysql-headless-service" # Links to the headless service replicas: 3 # Number of MySQL replicas

selector:

matchLabels:

app: mysql template:

metadata:

labels:

app: mysql spec:

containers:

* name: mysql

image: mysql:8.0 # MySQL 8.0 image ports:

* + containerPort: 3306 env:
  + name: MYSQL\_ROOT\_PASSWORD

valueFrom: secretKeyRef:

name: mysql-secrets key: MYSQL\_PASSWORD

volumeMounts:

* + name: mysql-data mountPath: /var/lib/mysql

volumeClaimTemplates:

- metadata:

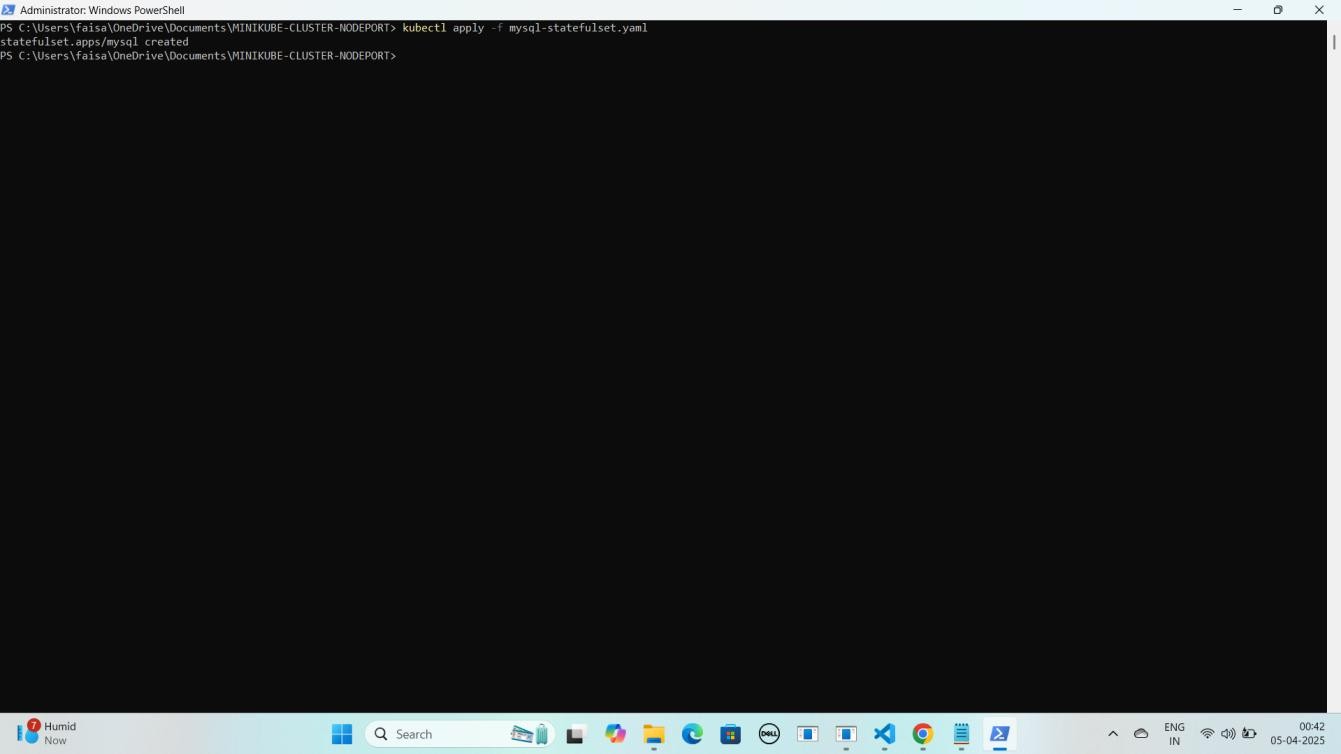
name: mysql-data spec:

accessModes:

- ReadWriteOnce resources:

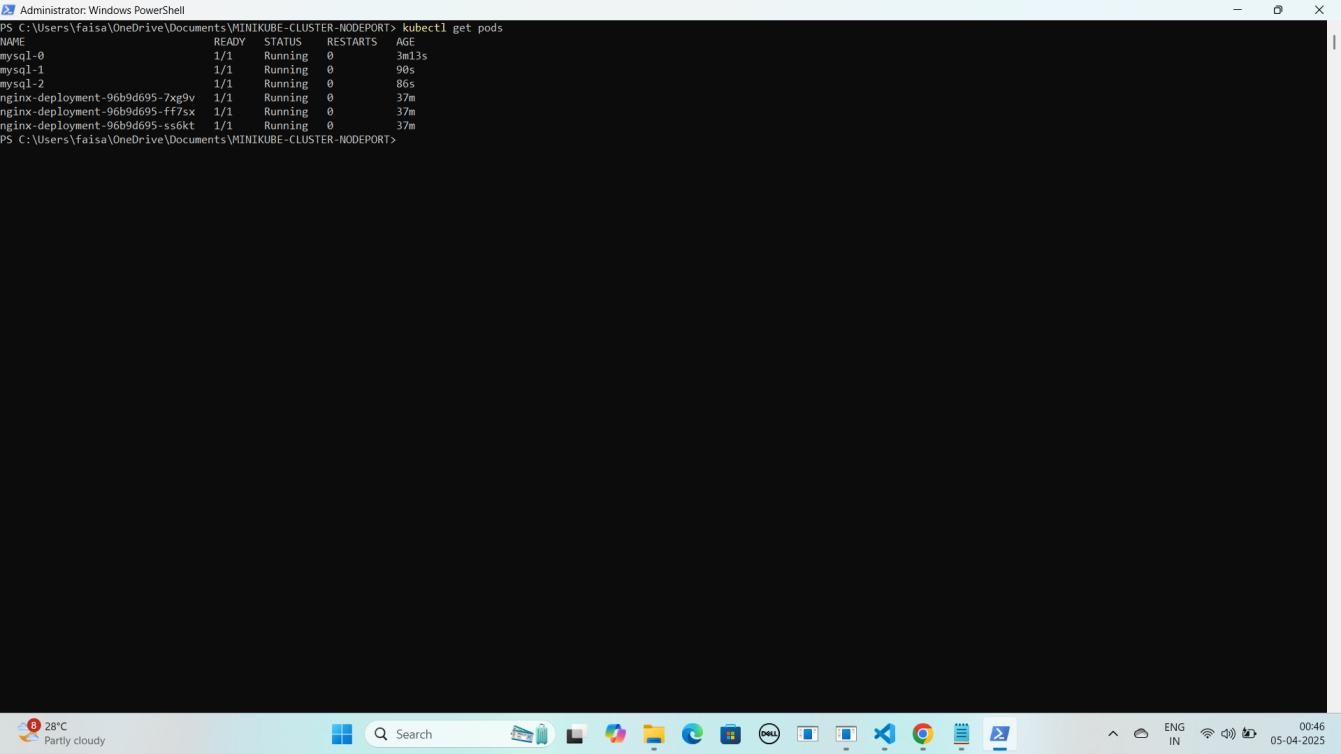
requests: storage: 5Gi

### StatefulSet Apply Karo

kubectl apply -f mysql-statefulset.yaml YE KUCH ISTARHA LAGEGA

#### Pods check karne ke liye ye command run kariye

kubectl get pods

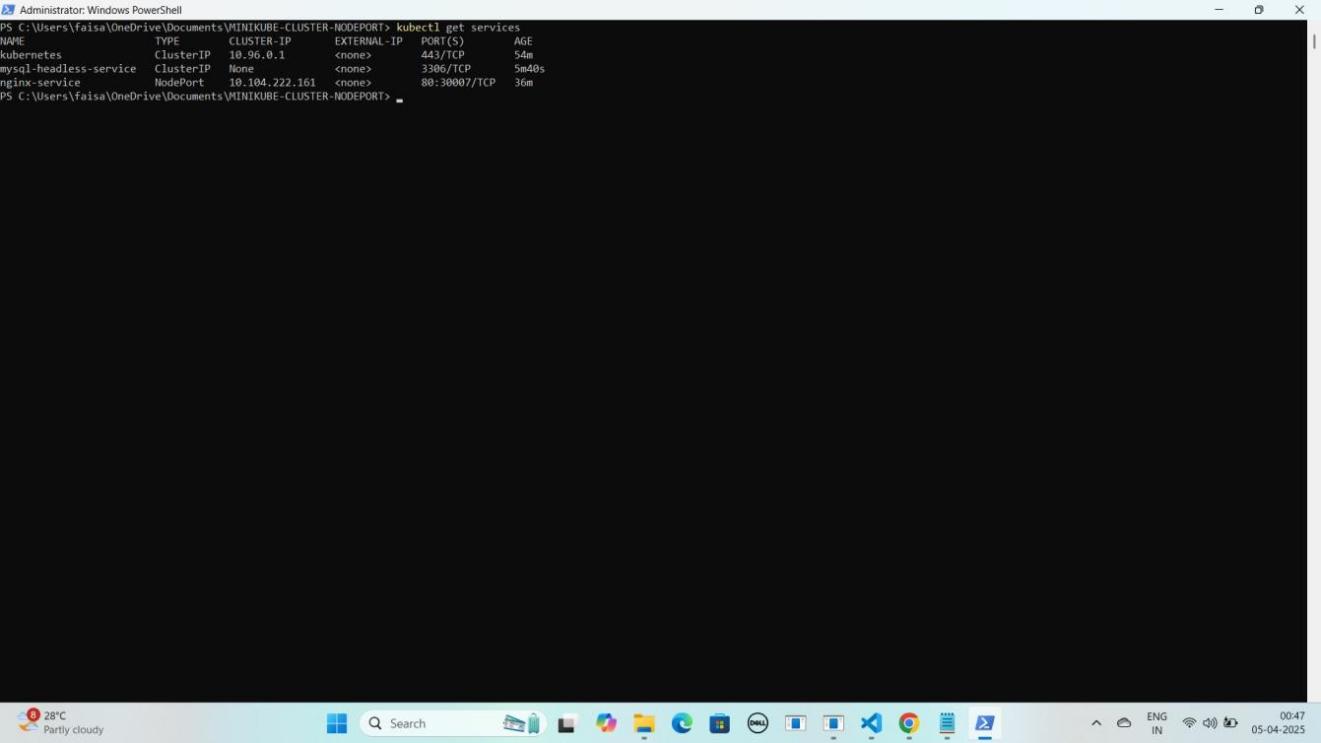
YE KUCH ISTARHA LAGEGA

#### NOTE: Agar sabhi Pods ka STATUS Running show karraha hai to sab kuch sahi hai

1. **Serives check karne ke liye ye commad run kariye**

kubectl get services

YE KUCH ISTARHA LAGEGA



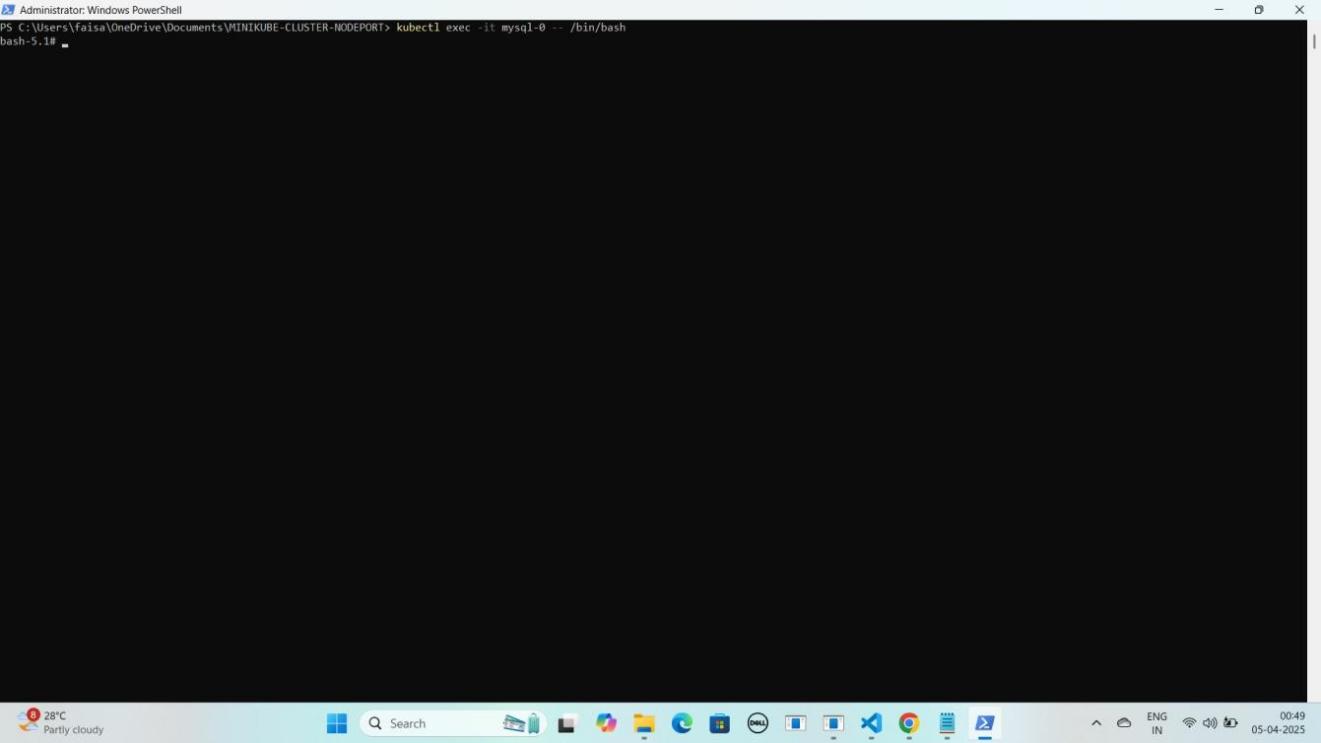
## Part 5: Accessing MySQL Database in Minikube Cluster

### Step 1: MySQL Database Access Karo

#### Pod ke andar jane ke liye ye command run karo

kubectl exec -it mysql-0 -- /bin/bash

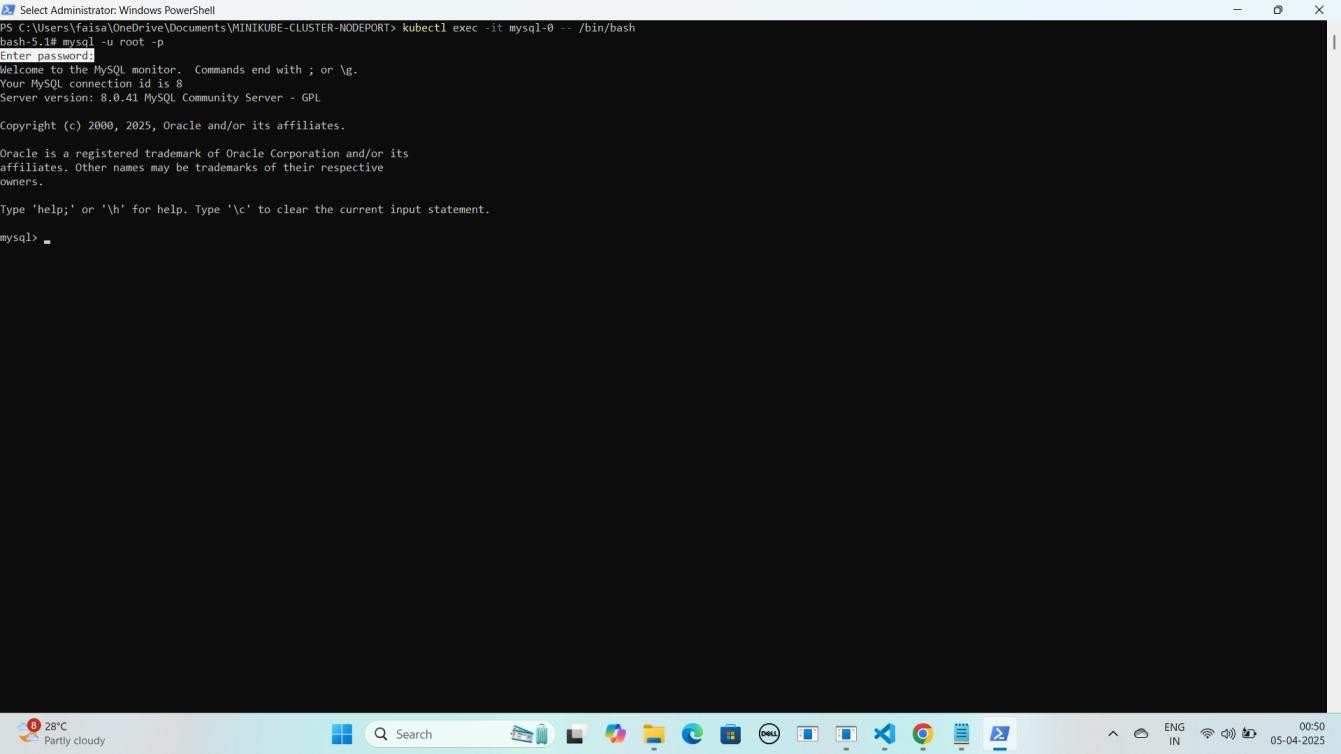
YE KUCH ISTARHA LAGEGA



#### MySQL database se connect karne ke liye ye command kuch istarha hogi

mysql -u root -p

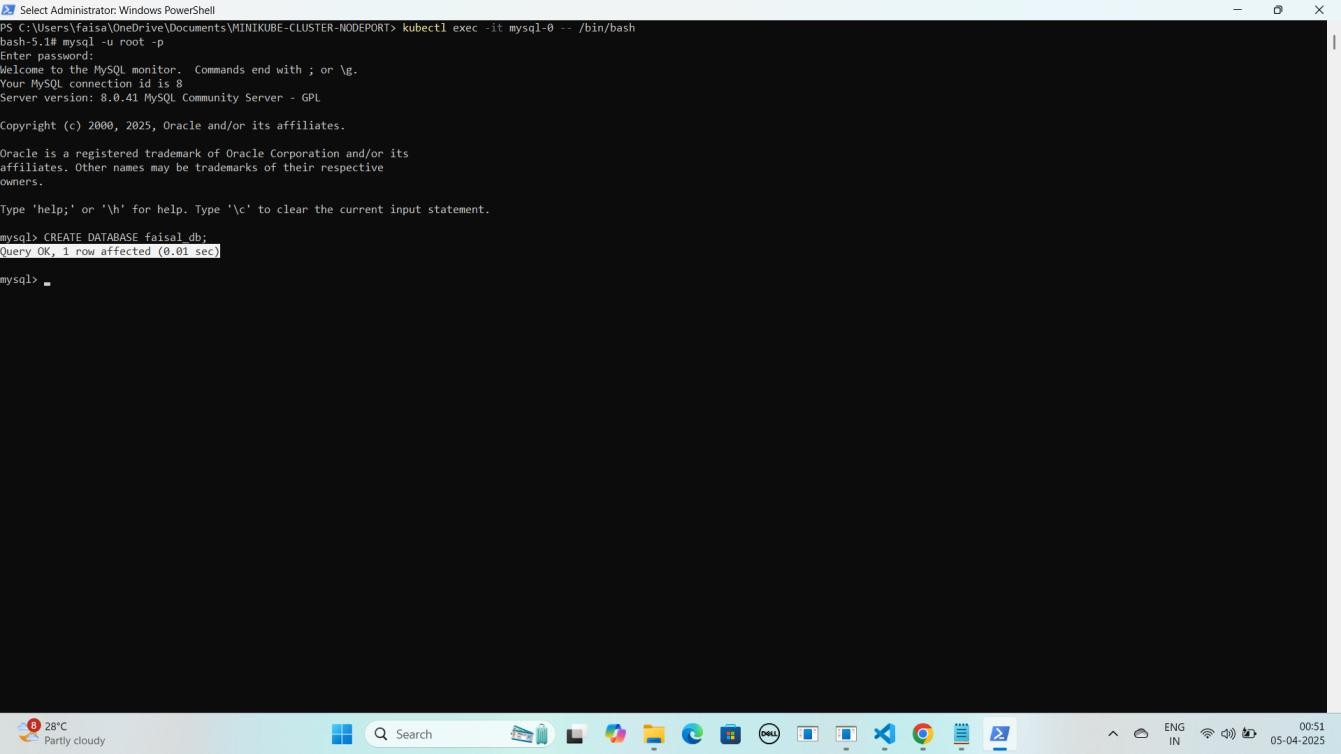
YE KUCH ISTARHA LAGEGA



#### Note: Jab aap se password maanga jaaye, to wahi password use karein jo pehle Base64 encode karke mysql-secrets.yaml file me store kiya gaya tha. Maine ‘Faisalkhan35@’ diya tha, lekin password enter karte waqt yeh screen par show nahi hoga.

1. **Database create karne keliye ye command run karo**

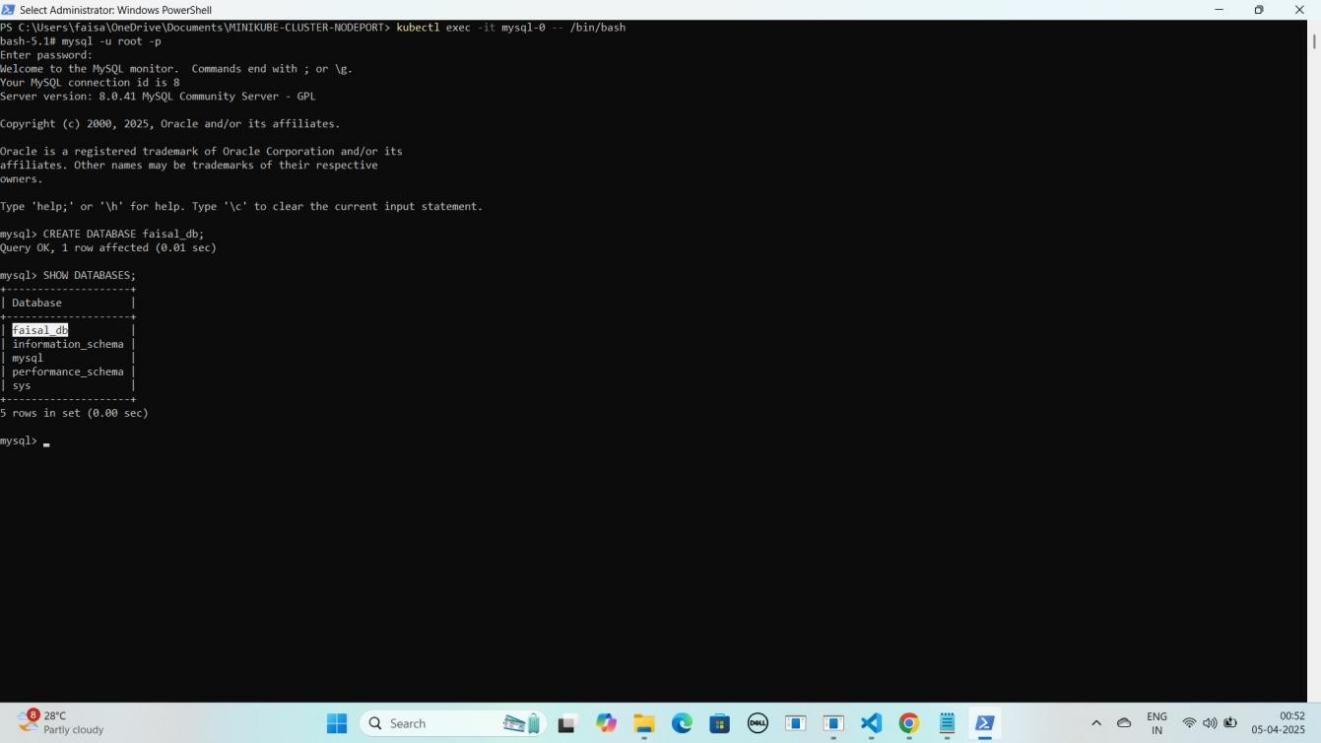
CREATE DATABASE faisal\_db; YE KUCH ISTARHA LAGEGA



#### Databases check karne ke liye ye command run kariye

SHOW DATABASES;

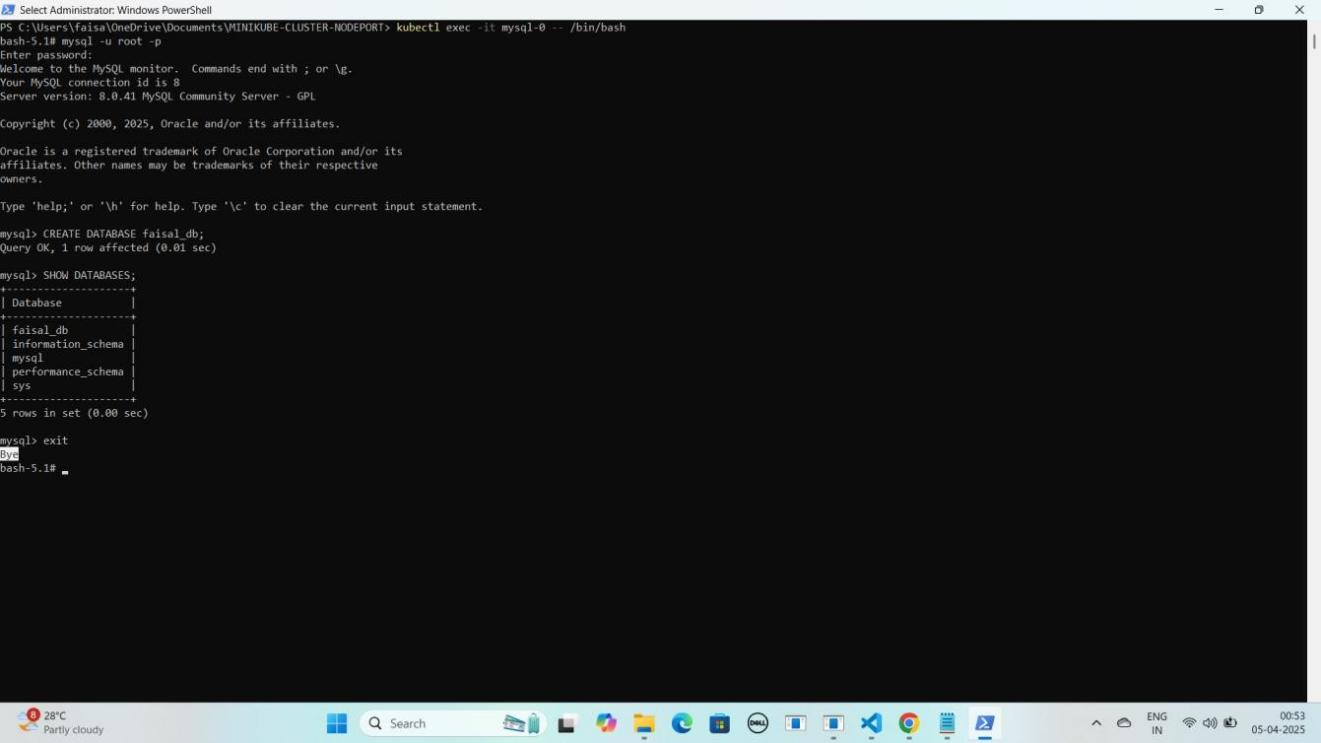
YE KUCH ISTARHA LAGEGA



#### Ab MYSQL database se exit hoagaye exit hone ke liye exit type kariye

exit

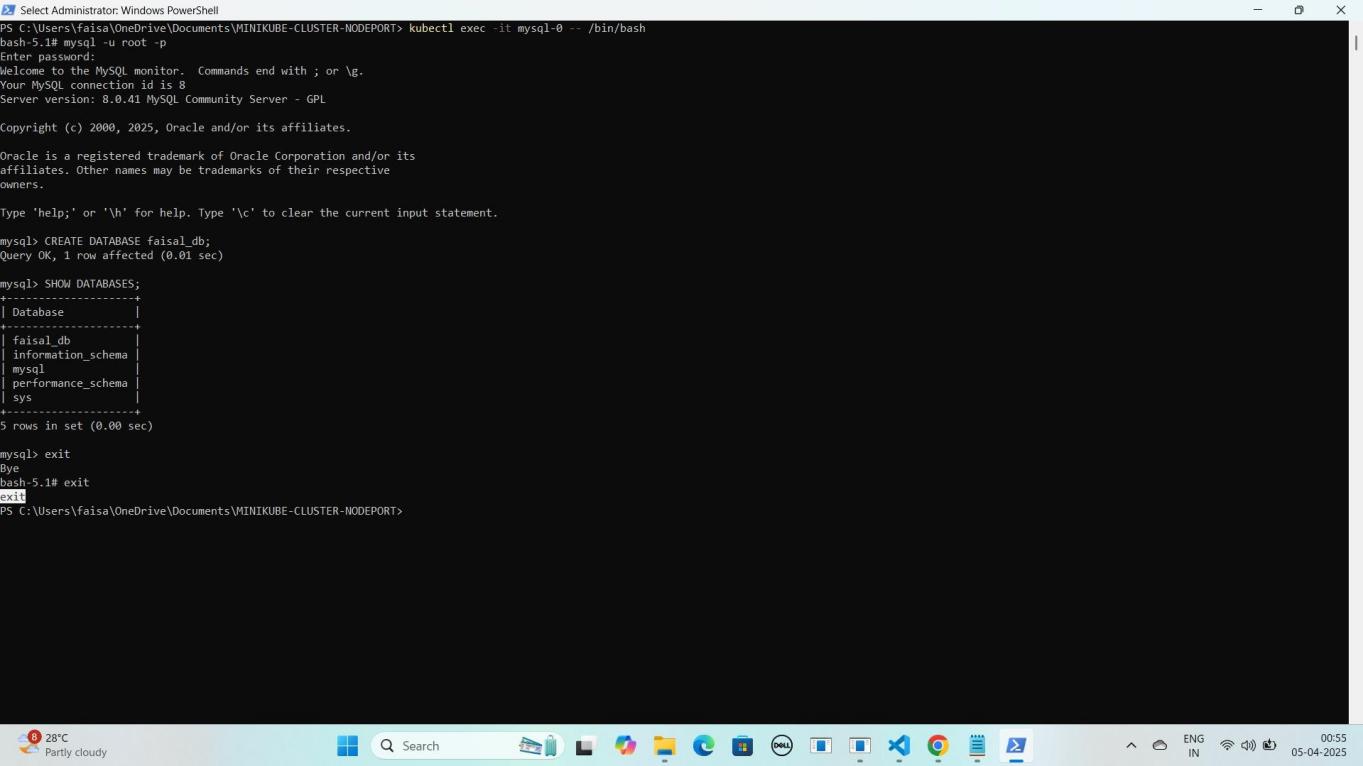
YE KUCH ISTARHA LAGEGA



#### Ab MYSQL database Pod se bhi exit hoagaye exit hone ke liye exit type kariye

exit

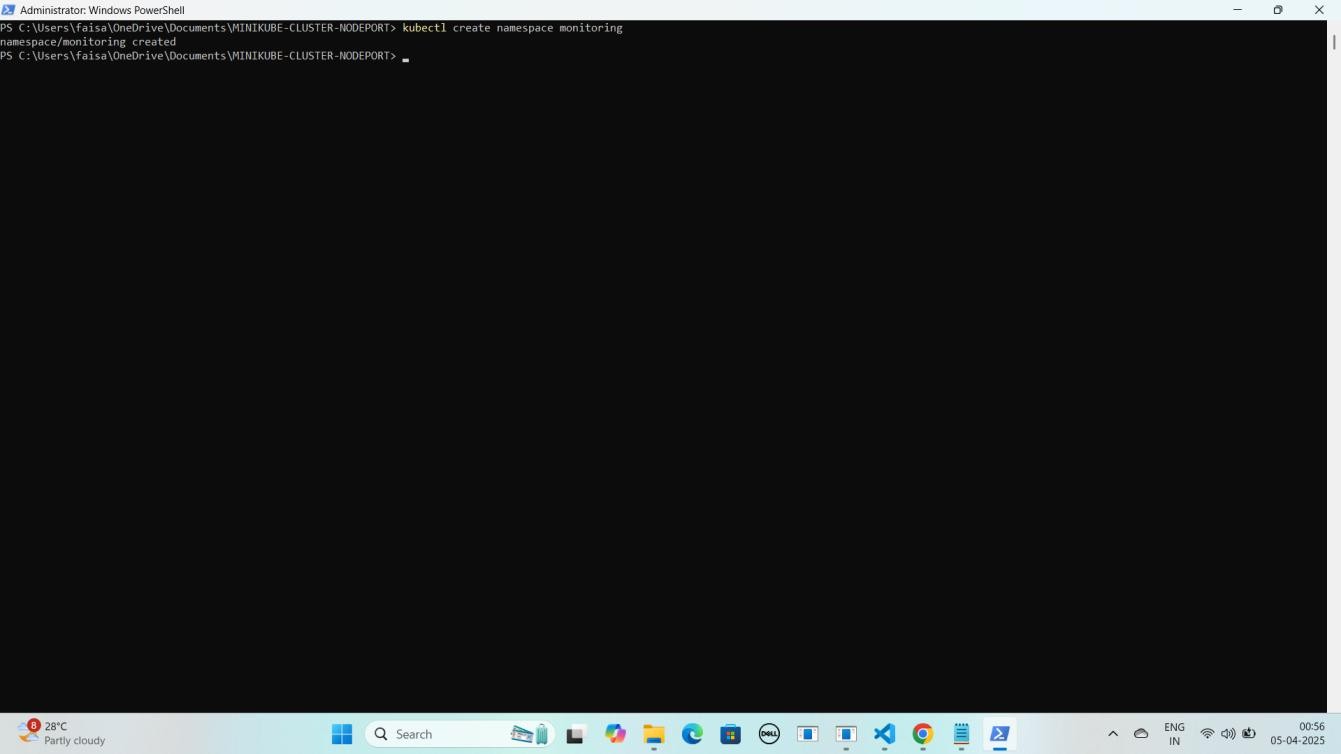
YE KUCH ISTARHA LAGEGA



**NOTE : Aapko faisal\_db naam ka database dikhega, aur isi tarah aap mysql-1 aur mysql-2 ke liye bhi access kar sakte hain**.

## Part 6: Monitoring Using Prometheus and Loki With Grafana

#### Kubernetes me ek naya namespace 'monitoring' create karna hoga. Namespace create karne ke liye ye command run kariye:

kubectl create namespace monitoring YE KUCH ISTARHA LAGEGA

### Step 1: prometheus-daemonset.yaml File Ka Kaam

Yeh **file Prometheus monitoring tool deploy** karne ke liye use hoti hai. Isko **cluster ka CPU usage, memory, aur network monitor** karne ke liye **Kubernetes DaemonSet** ke andar likha jata hai.

JAISE KI:-

apiVersion: apps/v1 kind: DaemonSet metadata:

name: prometheus namespace: monitoring

spec:

selector:

matchLabels:

app: prometheus template:

metadata:

labels:

app: prometheus spec:

containers:

* name: prometheus

image: prom/prometheus:v2.31.1 args:

* + "--config.file=/etc/prometheus/prometheus.yml"
  + "--storage.tsdb.path=/prometheus"
  + "--web.enable-lifecycle" volumeMounts:
  + name: prometheus-config mountPath: /etc/prometheus
  + name: prometheus-storage mountPath: /prometheus

volumes:

* name: prometheus-config configMap:

name: prometheus-config

* name: prometheus-storage emptyDir: {}

---

apiVersion: v1 kind: ConfigMap metadata:

name: prometheus-config namespace: monitoring

data:

prometheus.yml: | global:

scrape\_interval: 15s # Default scrape interval for all jobs

scrape\_configs:

# Kubernetes Nodes Service Discovery for Node Exporters

* job\_name: 'kubernetes-nodes' kubernetes\_sd\_configs:
  + role: node # Specify the role as 'node' relabel\_configs:
  + source\_labels: [ meta\_kubernetes\_node\_name] # Scrape metrics from nodes

target\_label: kubernetes\_node

- source\_labels: [ meta\_kubernetes\_node\_label\_kubernetes\_io\_hostname] # Include the node hostname as a label

target\_label: kubernetes\_hostname

# Kubernetes Pods Service Discovery (for scraping Kubernetes pods)

* job\_name: 'kubernetes-pods' kubernetes\_sd\_configs:
  + role: pod # Scrape all pods (can be further refined with relabeling) relabel\_configs:
  + source\_labels: [ meta\_kubernetes\_pod\_label\_app] # Use pod labels for further

filtering

target\_label: app

### Prometheus DaemonSet Apply Karo

kubectl apply -f prometheus-daemonset.yaml YE KUCH ISTARHA LAGEGA

### Step 2: prometheus-rbac.yaml File Ka Kaam

Yeh **file Prometheus** ko **required permissions** dene ke liye use hoti hai, taake woh **cluster** ke **resources** ko **access** kar sake.

JAISE KI:-

apiVersion: rbac.authorization.k8s.io/v1 kind: ClusterRole

metadata:

name: prometheus-scrape-role rules:

* apiGroups: [""]

resources: ["pods", "nodes"]

verbs: ["get", "list", "watch"]

* apiGroups: ["metrics.k8s.io"] resources: ["nodes", "pods"]

verbs: ["get", "list", "watch"]

* apiGroups: ["apps"]

resources: ["deployments", "replicasets", "daemonsets"] verbs: ["get", "list", "watch"]

* apiGroups: ["extensions"] resources: ["ingresses"] verbs: ["get", "list", "watch"]
* apiGroups: [""]

resources: ["services", "endpoints"] verbs: ["get", "list", "watch"]

---

apiVersion: rbac.authorization.k8s.io/v1 kind: ClusterRoleBinding

metadata:

name: prometheus-scrape-role-binding subjects:

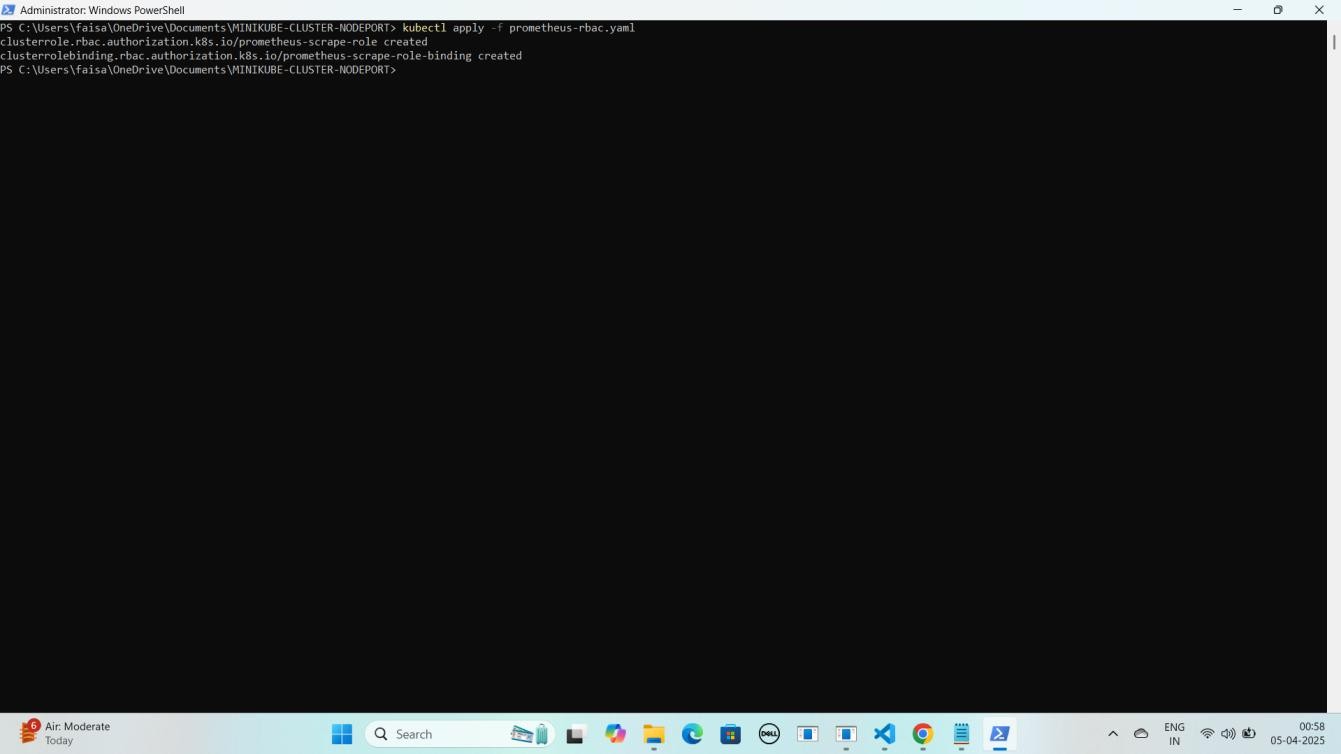
* kind: ServiceAccount name: default namespace: monitoring

roleRef:

kind: ClusterRole

name: prometheus-scrape-role apiGroup: rbac.authorization.k8s.io

### Prometheus RBAC Apply Karo

kubectl apply -f prometheus-rbac.yaml YE KUCH ISTARHA LAGEGA

### Step 3: prometheus-nodeport-service.yaml File Ka Kaam

Yeh **file Prometheus** ko port **30090** par **NodePort service** ke zariye **expose** karne ke liye use hoti hai, taake hum usko **browser** me **access** kar sakein.

JAISE KI:-

apiVersion: v1 kind: Service metadata:

name: prometheus-server namespace: monitoring

spec:

ports:

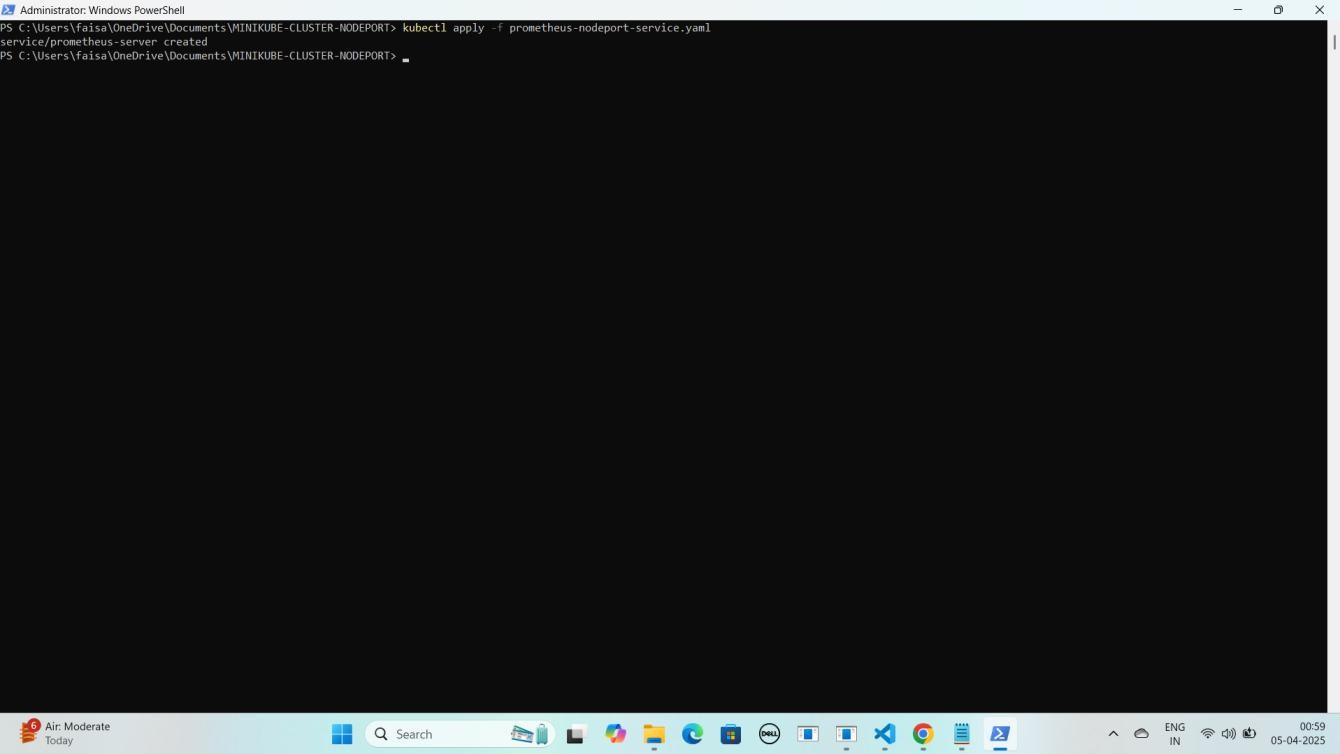
- port: 80

targetPort: 9090

nodePort: 30090 selector:

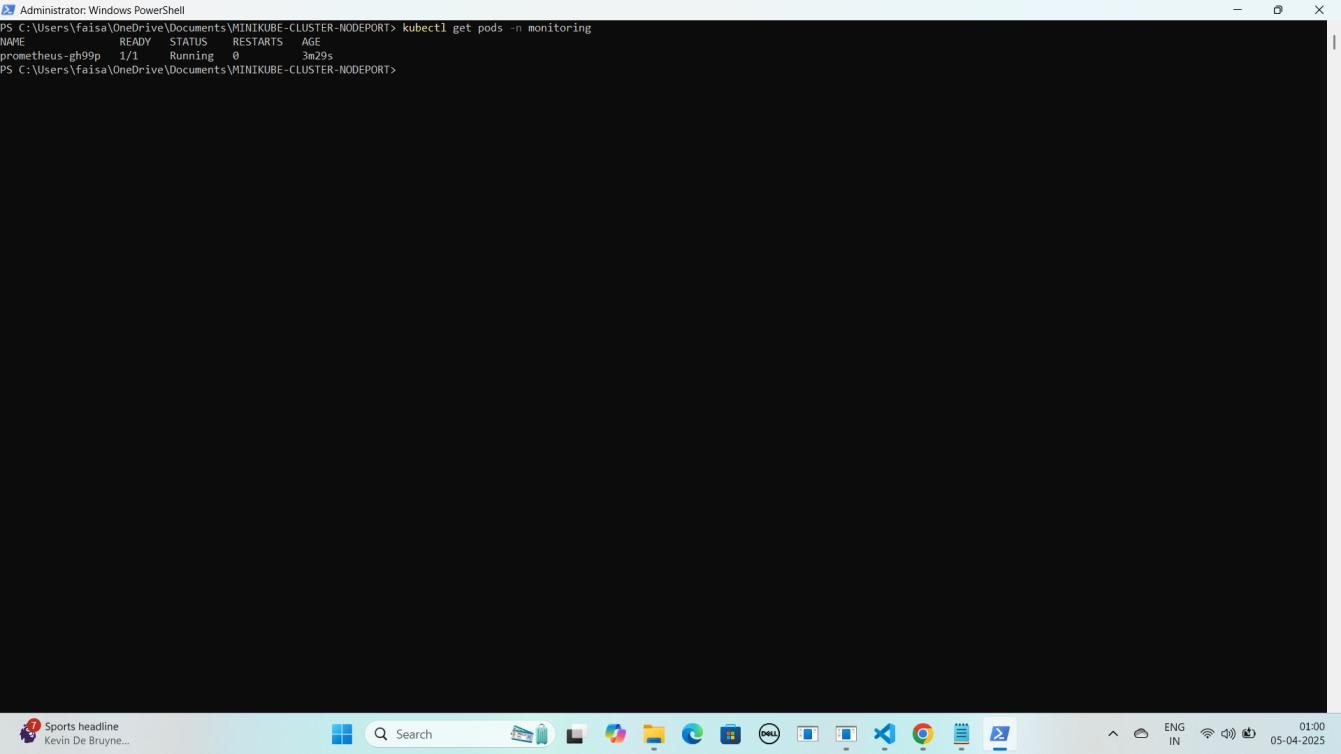
app: prometheus type: NodePort

### Prometheus Service Apply Karo

kubectl apply -f prometheus-nodeport-service.yaml YE KUCH ISTARHA LAGEGA

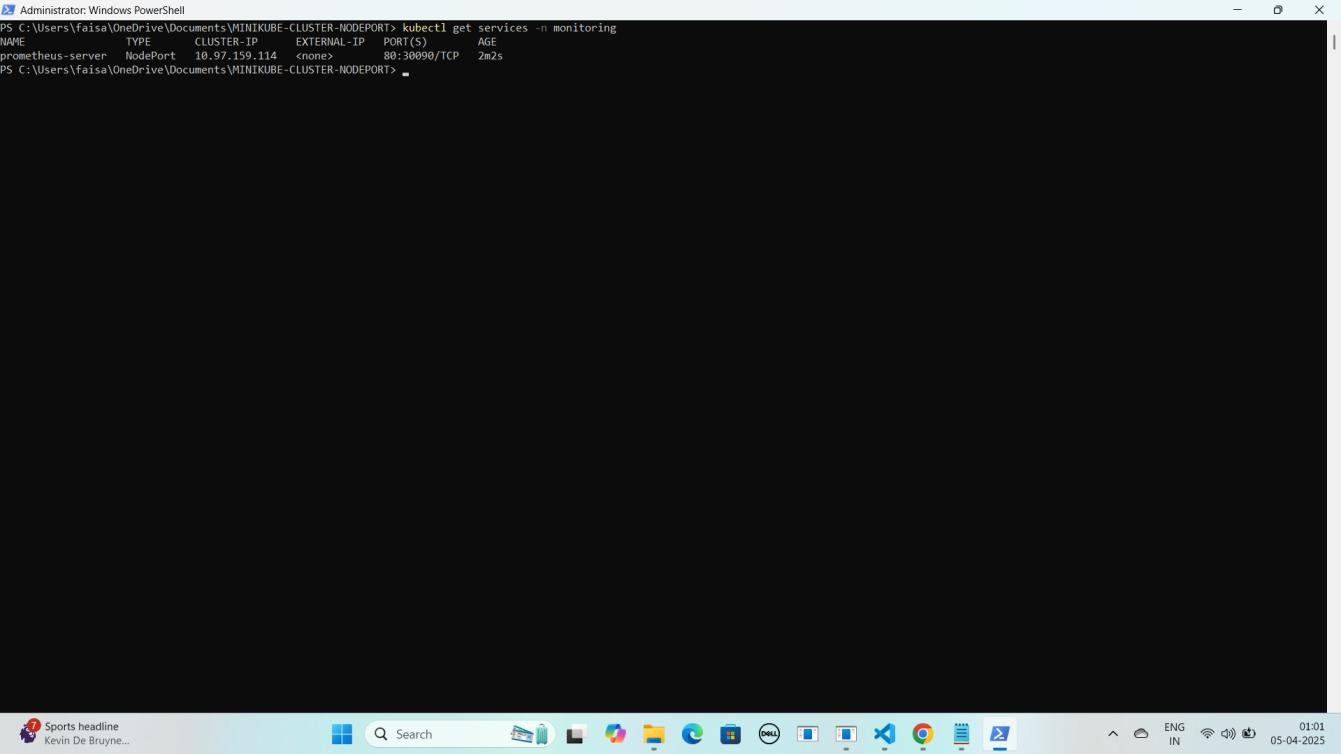
#### Pods check karne ke liye ye command run kariye

kubectl get pods -n monitoring

YE KUCH ISTARHA LAGEGA

#### NOTE: Agar STATUS Running show karraha hai to sab kuch sahi hai

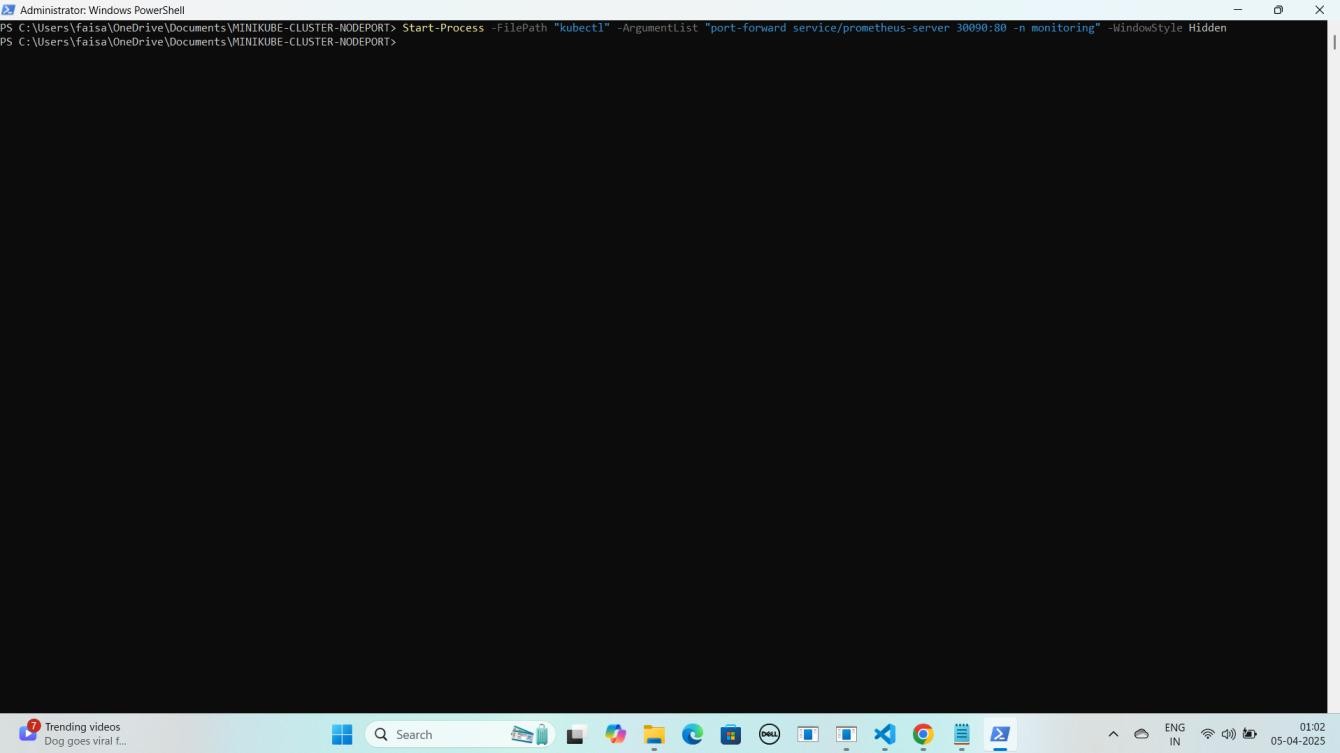
1. **Serives check karne ke liye ye commad run kariye**

kubectl get services -n monitoring YE KUCH ISTARHA LAGEGA

#### Port Forwarding Karne ke liye ye command run karein

Start-Process -FilePath "kubectl" -ArgumentList "port-forward service/prometheus-server 30090:80 -n monitoring" -WindowStyle Hidden

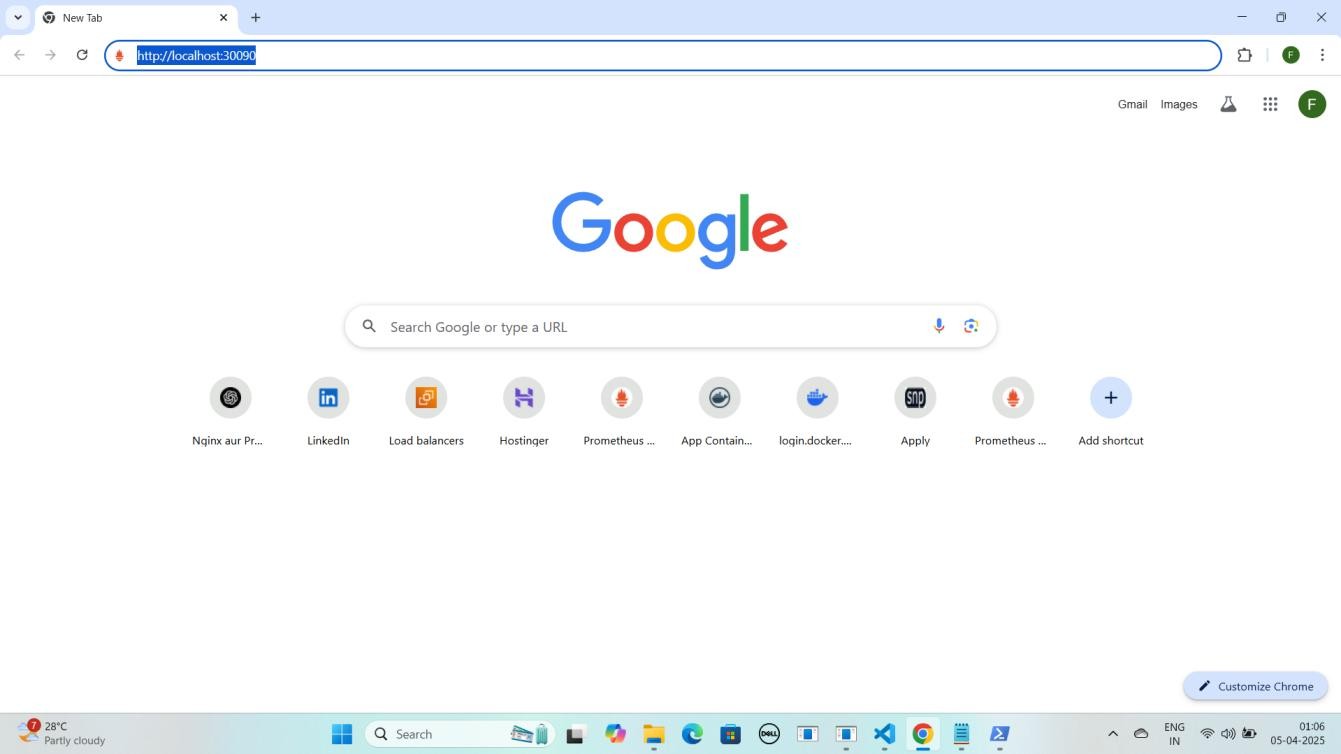
YE KUCH ISTARHA LAGEGA

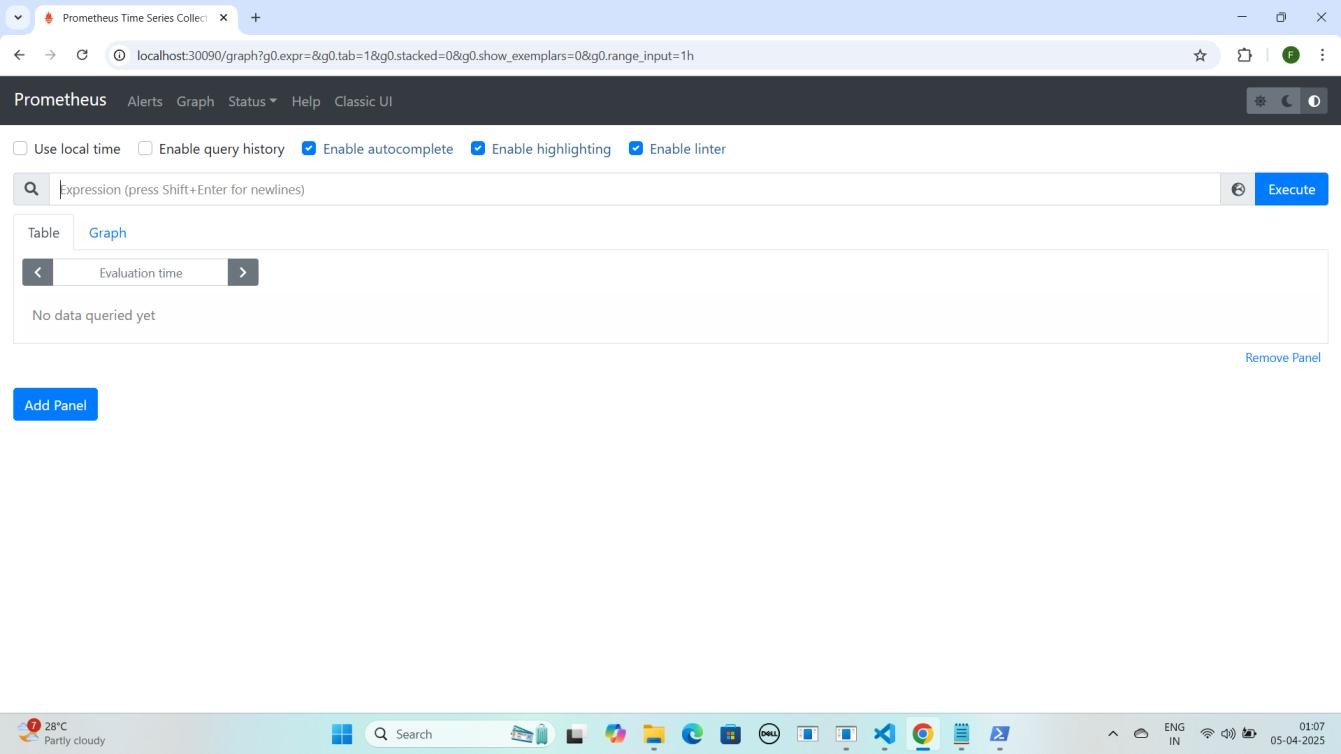


#### NOTE: Ab aapka Prometheus expose ho chuka hai. Ab apne local system par Prometheus ko NodePort ke saath browser me run kariye. Jaise ki mere case me kuch aisa hoga

**Prometheus:** [http://localhost:30090](u)

YE KUCH ISTARHA LAGEGA



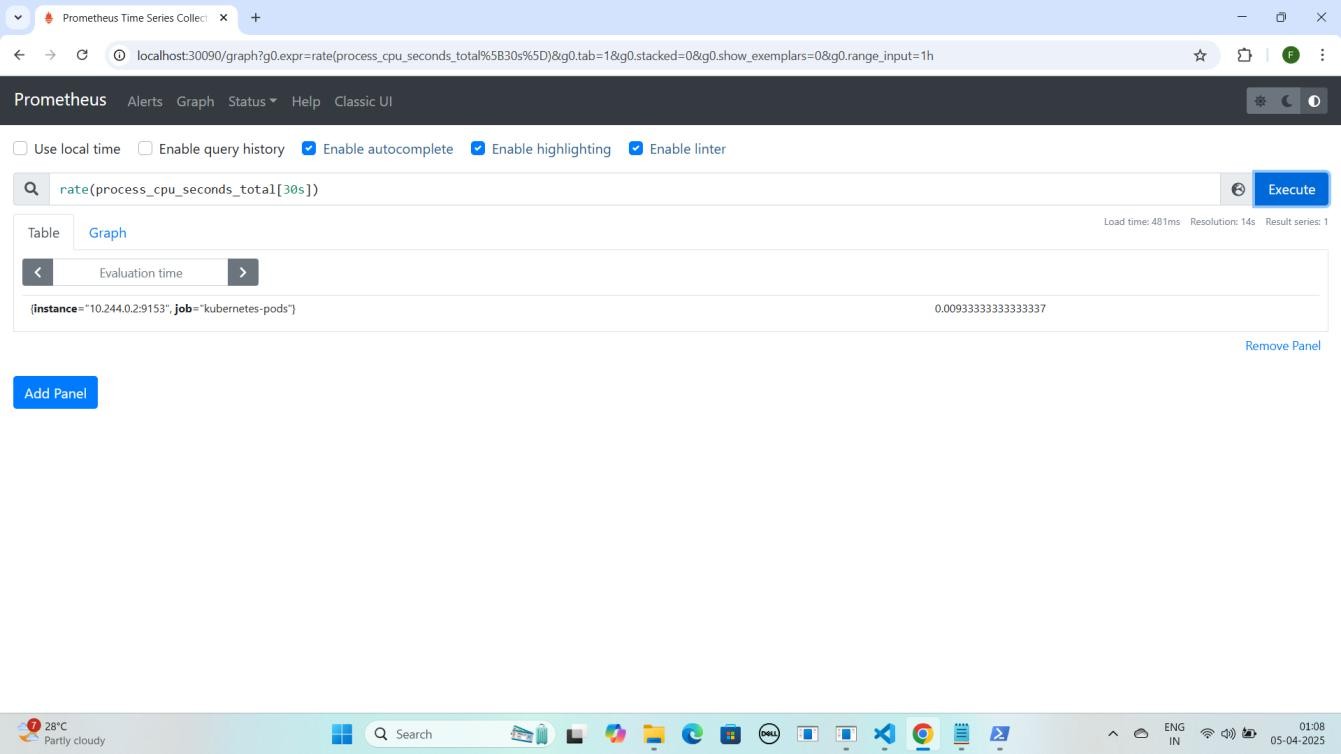


#### CPU usage check karne ke liye Prometheus me ye command run karein:

rate(process\_cpu\_seconds\_total[30s])

#### Note: Command Prometheus me paste karne ke baad Execute pe click karein.

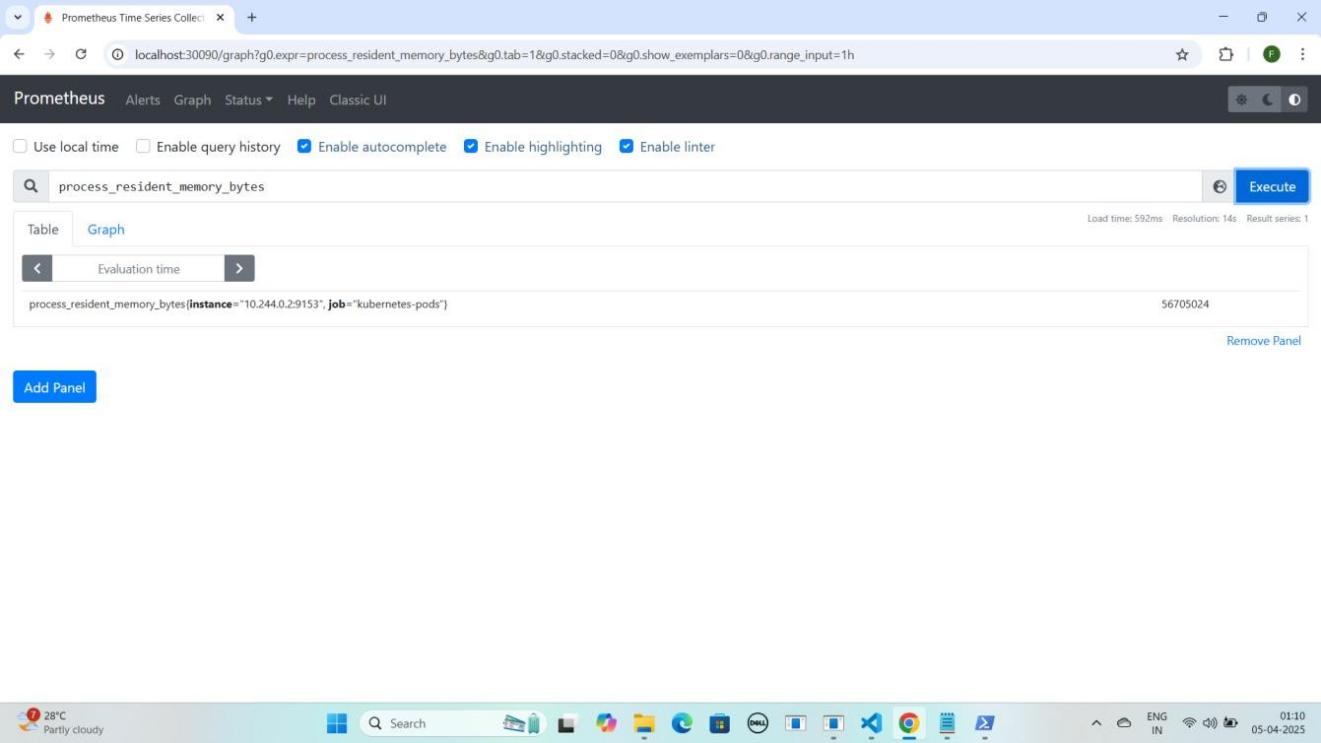
YE KUCH ISTARHA LAGEGA



#### Memory usage check karne ke liye Prometheus me ye command run karein

process\_resident\_memory\_bytes

#### Note: Command Prometheus me paste karne ke baad Execute pe click karein.

YE KUCH ISTARHA LAGEGA

### Step 4: promtail-daemonset.yaml File Ka Kaam

Yeh **file cluster** se **logs collect** karne aur unhe **Loki** par bhejne ke liye use hoti hai.

JAISE KI:-

apiVersion: apps/v1 kind: DaemonSet metadata:

name: promtail namespace: monitoring labels:

app: promtail spec:

selector:

matchLabels:

app: promtail template:

metadata:

labels:

app: promtail spec:

serviceAccountName: promtail containers:

* name: promtail

image: grafana/promtail:2.3.0 args:

* + -config.file=/etc/promtail/promtail.yaml volumeMounts:
  + name: config

mountPath: /etc/promtail

* + name: varlog mountPath: /var/log
  + name: containers

mountPath: /var/log/containers readOnly: true

* + name: docker

mountPath: /var/lib/docker/containers readOnly: true

volumes:

* name: config configMap:

name: promtail-config

* name: varlog hostPath:

path: /var/log

* name: containers hostPath:

path: /var/log/containers

* name: docker hostPath:

path: /var/lib/docker/containers

---

apiVersion: rbac.authorization.k8s.io/v1 kind: ClusterRole

metadata:

name: promtail rules:

* + apiGroups: [""] resources:
    - pods
    - nodes
* namespaces verbs:
* get
* watch
* list

---

apiVersion: rbac.authorization.k8s.io/v1 kind: ClusterRoleBinding

metadata:

name: promtail roleRef:

apiGroup: rbac.authorization.k8s.io kind: ClusterRole

name: promtail subjects:

- kind: ServiceAccount

name: promtail namespace: monitoring

---

apiVersion: v1 kind: ServiceAccount metadata:

name: promtail

namespace: monitoring

---

apiVersion: v1 kind: ConfigMap metadata:

name: promtail-config namespace: monitoring

data:

promtail.yaml: | server:

http\_listen\_port: 9080

grpc\_listen\_port: 0

positions:

filename: /tmp/positions.yaml

clients:

- url: <http://loki:3100/loki/api/v1/push>

scrape\_configs:

- job\_name: system static\_configs:

- targets:

- localhost labels:

job: varlogs

path : /var/log/\*\*/\*.log

- job\_name: containers static\_configs:

- targets:

- localhost labels:

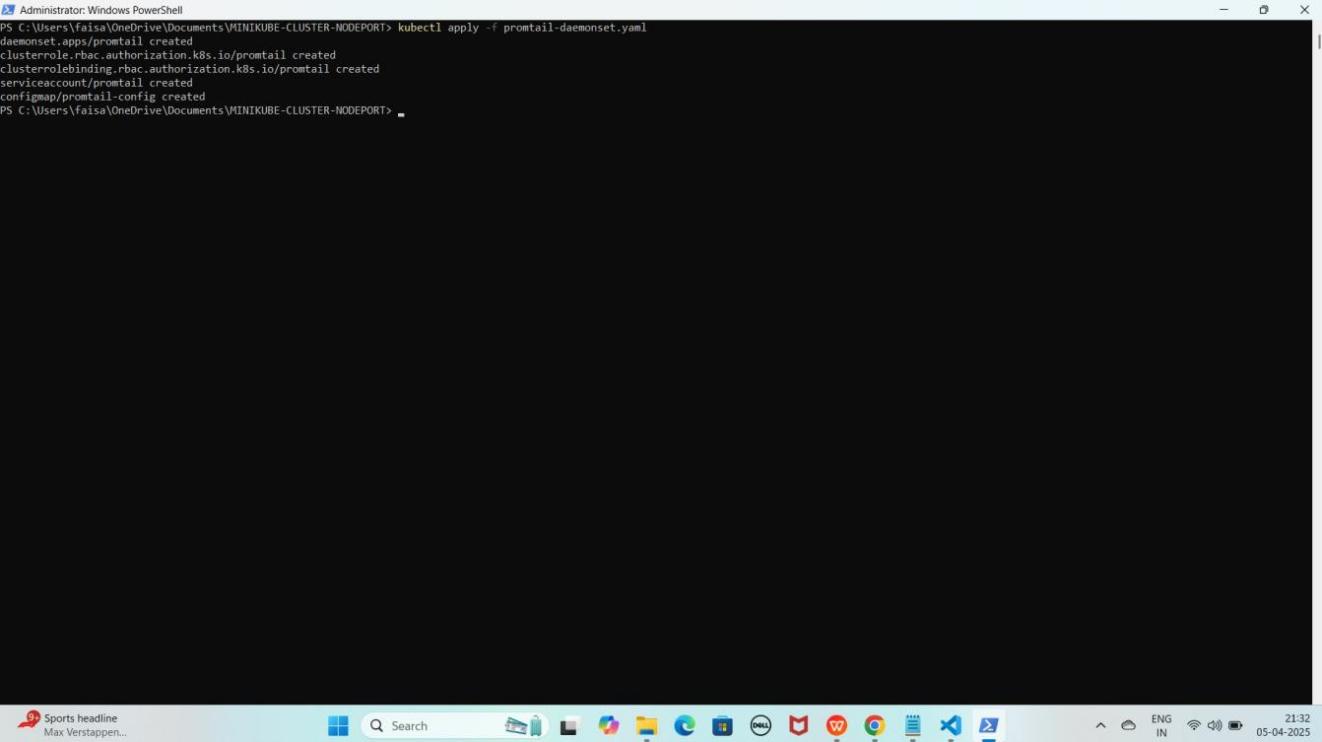
job: containers

path : /var/log/containers/\*.log

### Promtail DaemonSet Apply Karo

kubectl apply -f promtail-daemonset.yaml

YE KUCH ISTARHA LAGEGA



### Step 5: loki-daemonset.yaml File Ka Kaam

Yeh **file Loki** ko ek **logs database** ke taur par **setup** karne ke liye use hoti hai, jo **Promtail** se **logs collect** karke **store** karta hai.

JAISE KI:-

apiVersion: apps/v1 kind: DaemonSet metadata:

name: loki namespace: monitoring

spec:

selector:

matchLabels:

app: loki template:

metadata:

labels:

app: loki spec:

serviceAccountName: loki containers:

* name: loki

image: grafana/loki:2.3.0 args:

* + "-config.file=/etc/loki/loki-config.yaml" ports:
  + containerPort: 3100 name: http

resources:

requests:

cpu: "500m" memory: "512Mi"

limits:

cpu: "1" memory: "1Gi"

volumeMounts:

- name: loki-config

mountPath: /etc/loki/loki-config.yaml subPath: loki-config.yaml

volumes:

- name: loki-config configMap:

name: loki-config items:

- key: loki-config.yaml path: loki-config.yaml

---

apiVersion: v1 kind: ConfigMap metadata:

name: loki-config namespace: monitoring

data:

loki-config.yaml: |

auth\_enabled: false # Disable authentication explicitly

server:

http\_listen\_port: 3100

distributor: ring:

kvstore:

store: inmemory

ingester: lifecycler:

ring:

kvstore:

store: inmemory replication\_factor: 1

chunk\_idle\_period: 5m chunk\_retain\_period: 30s

max\_transfer\_retries: 0

schema\_config:

configs:

- from: 2020-10-24

store: boltdb-shipper object\_store: filesystem schema: v11

index:

prefix: index\_ period: 24h

storage\_config: boltdb\_shipper:

active\_index\_directory: /loki/index

cache\_location: /loki/cache shared\_store: filesystem

filesystem:

directory: /loki/chunks

compactor:

working\_directory: /loki/compactor shared\_store: filesystem compaction\_interval: 5m

limits\_config: enforce\_metric\_name: false reject\_old\_samples: true

reject\_old\_samples\_max\_age: 168h

chunk\_store\_config: max\_look\_back\_period: 0s

table\_manager: retention\_deletes\_enabled: true retention\_period: 168h

---

apiVersion: v1 kind: Service metadata:

name: loki namespace: monitoring

spec:

type: ClusterIP ports:

- port: 3100

targetPort: 3100 selector:

app: loki

---

apiVersion: v1 kind: ServiceAccount metadata:

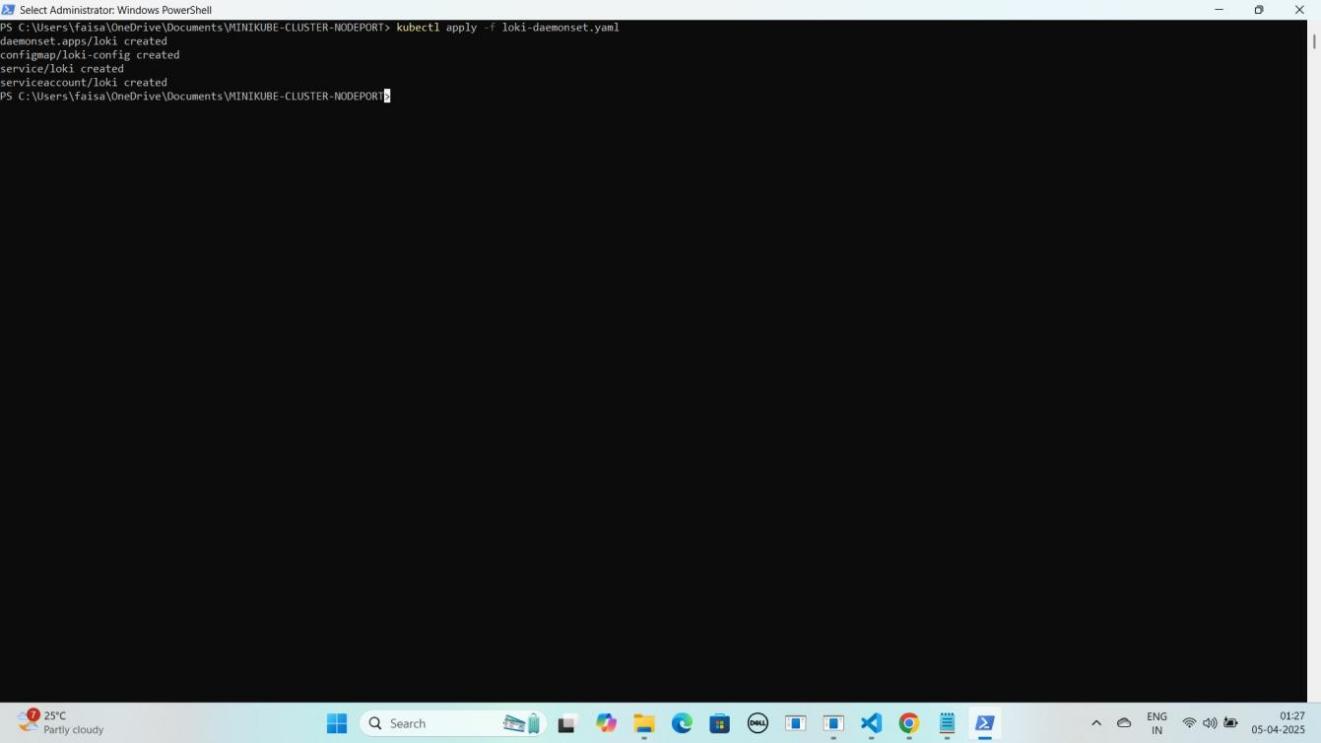
name: loki

namespace: monitoring

### Loki DaemonSet Apply Karo

kubectl apply -f loki-daemonset.yaml

YE KUCH ISTARHA LAGEGA



### Step 6: loki-nodeport-service.yaml File Ka Kaam

Yeh **file Loki** ko **expose** karne ke liye use hoti hai, taake hum **port 30091** ke zariye usko **access** kar sakein. Isko **Kubernetes NodePort Service** ke andar likha jata hai.

JAISE KI:-

apiVersion: v1 kind: Service metadata:

name: loki namespace: monitoring

spec:

ports:

- port: 3100

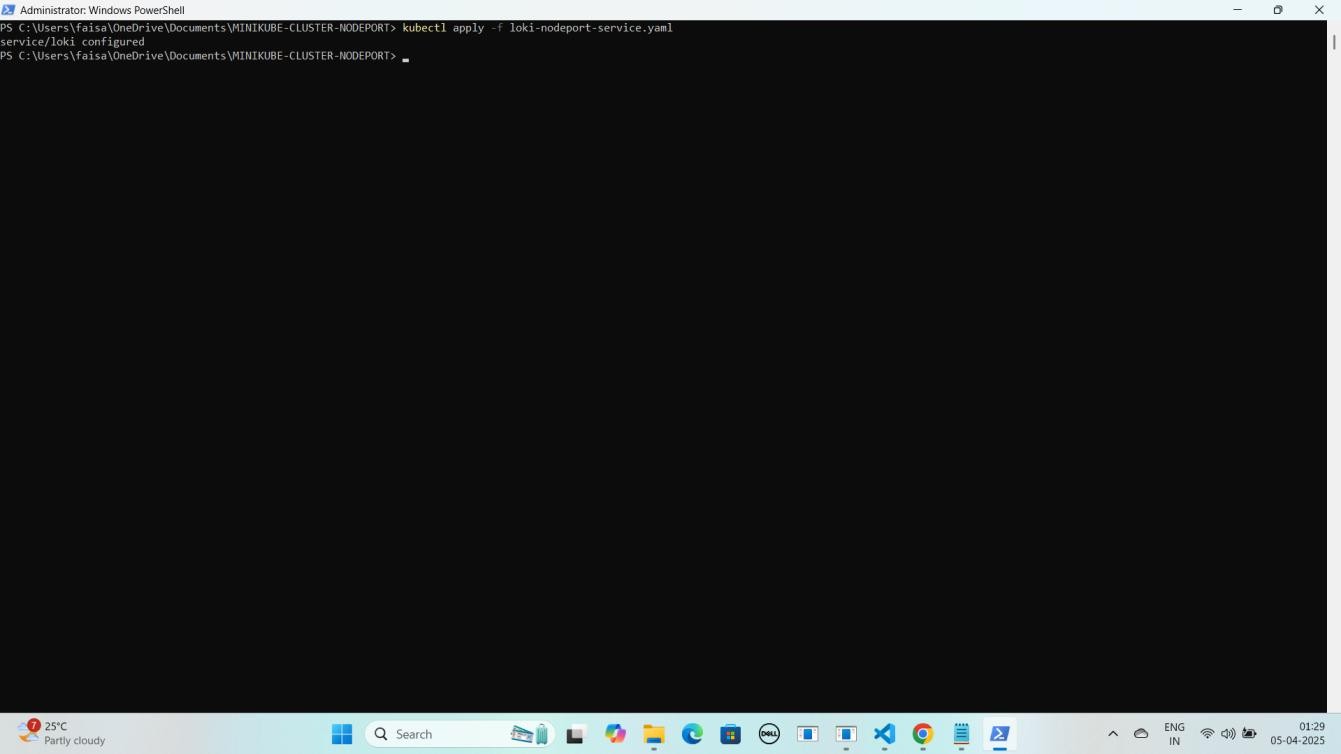
targetPort: 3100

nodePort: 30091 # This exposes Loki on port 30091 selector:

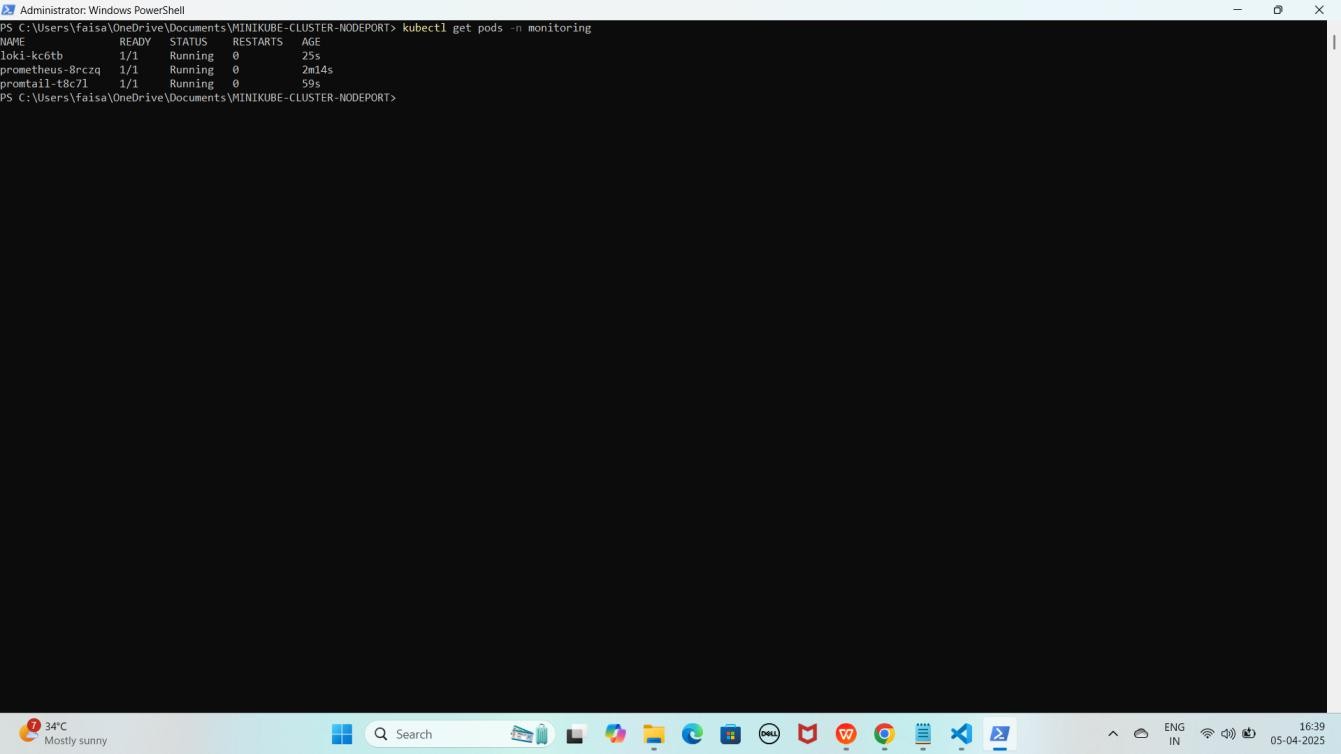
app: loki type: NodePort

### Loki Service Apply Karo

kubectl apply -f loki-nodeport-service.yaml

YE KUCH ISTARHA LAGEGA

#### Pods check karne ke liye ye command run kariye

kubectl get pods -n monitoring YE KUCH ISTARHA LAGEGA

#### NOTE: Agar STATUS Running show karraha hai to sab kuch sahi hai

1. **Serives check karne ke liye ye commad run kariye**

kubectl get services -n monitoring

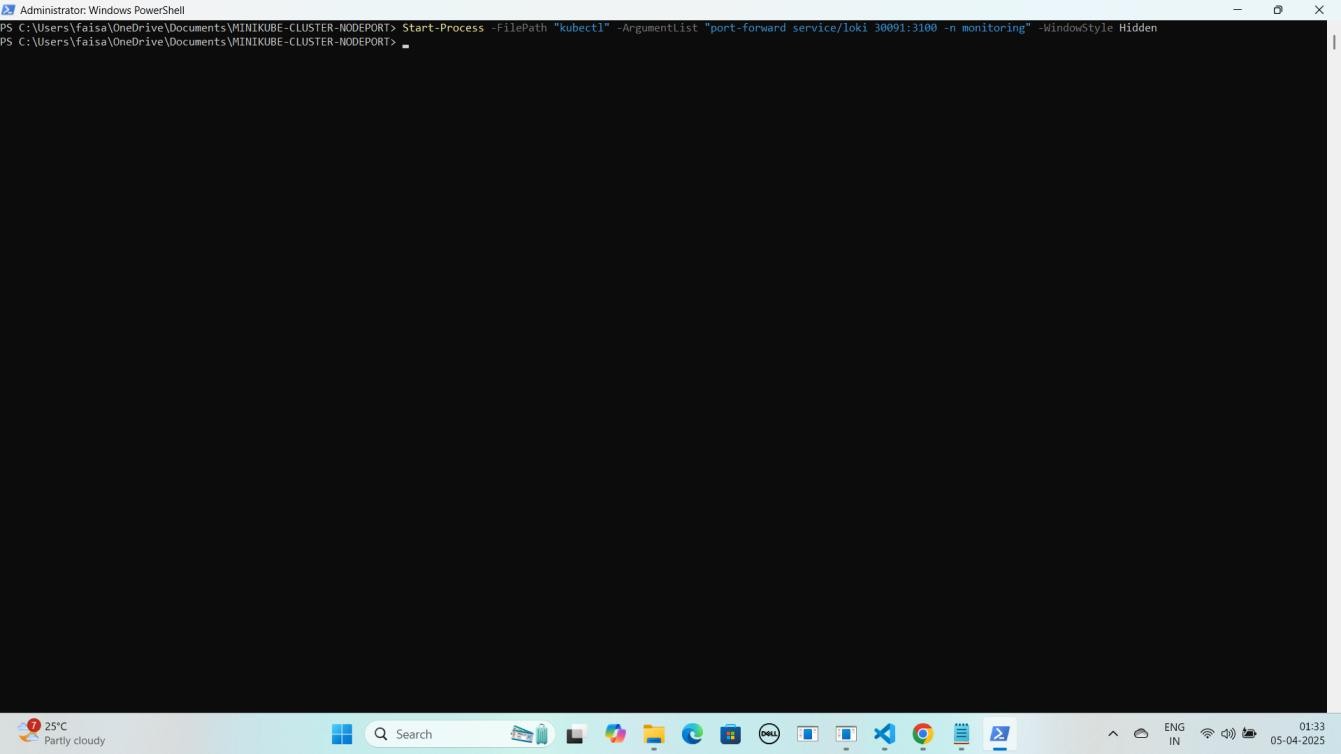
YE KUCH ISTARHA LAGEGA



#### Port Forwarding Karne ke liye ye command run karein

Start-Process -FilePath "kubectl" -ArgumentList "port-forward service/loki 30091:3100 -n monitoring" -WindowStyle Hidden

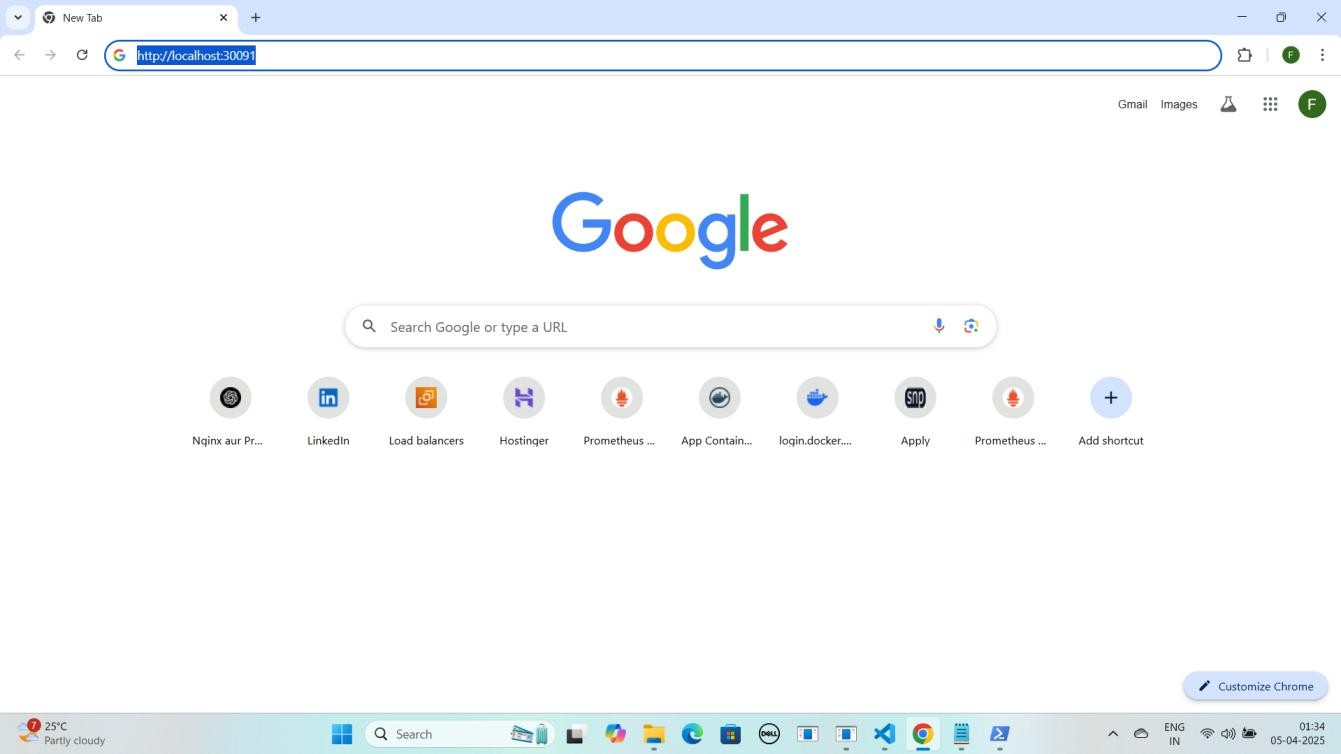
YE KUCH ISTARHA LAGEGA

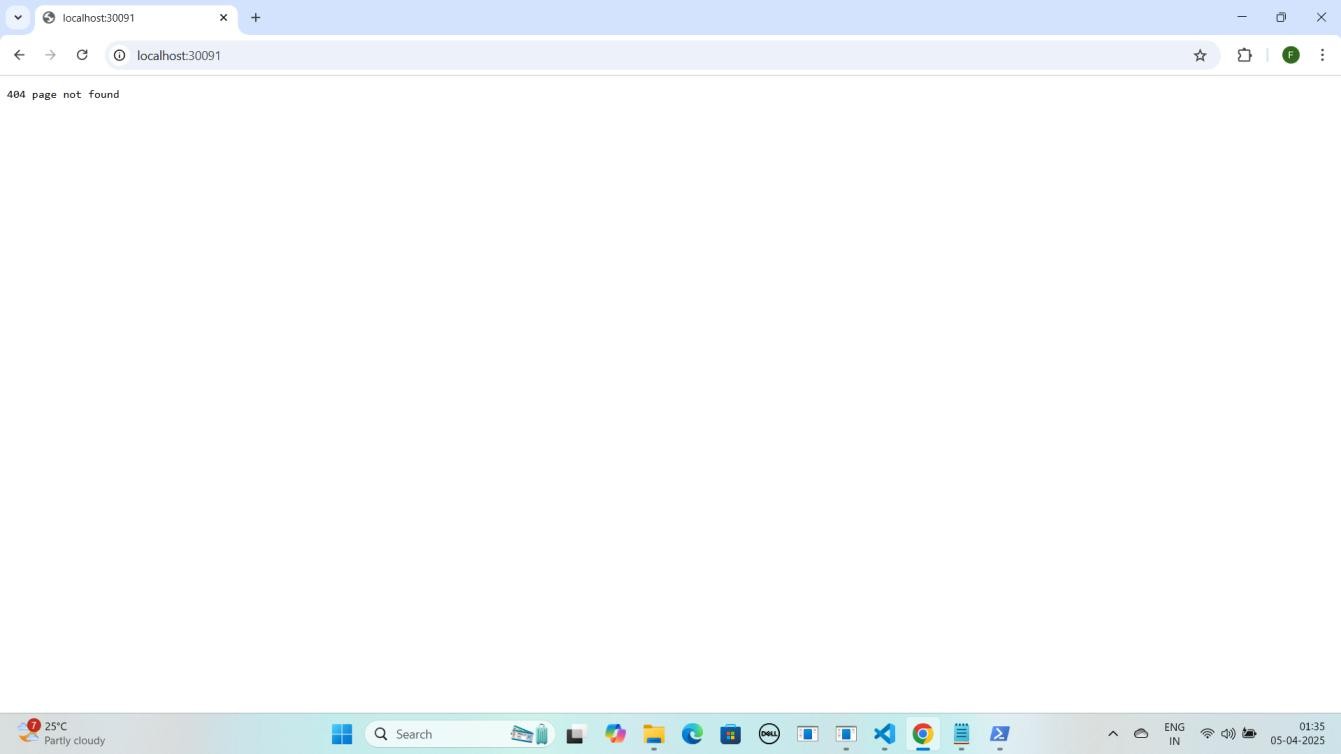


#### NOTE: Ab aapka Loki expose ho chuka hai. Ab apne local system par Loki ko NodePort ke saath browser me run kariye. Jaise ki mere case me kuch aisa hoga

**Loki:** [http://localhost:30091](n)

YE KUCH ISTARHA LAGEGA



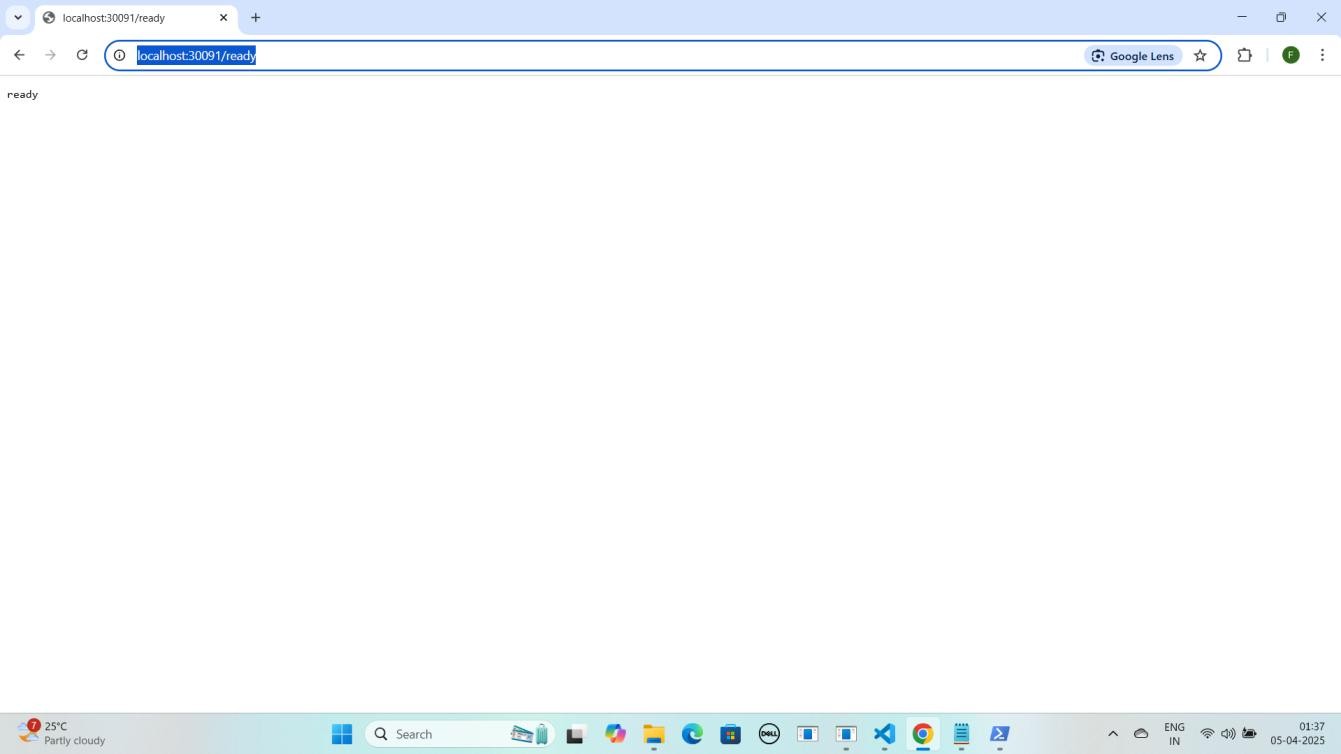


1. Lekin aapko **404 page not found** dikhayga to **/ready** se **check** kariye

**loki** ko jaise ki mere case kuch asisa hoga.

[http://localhost:30091/ready](h)

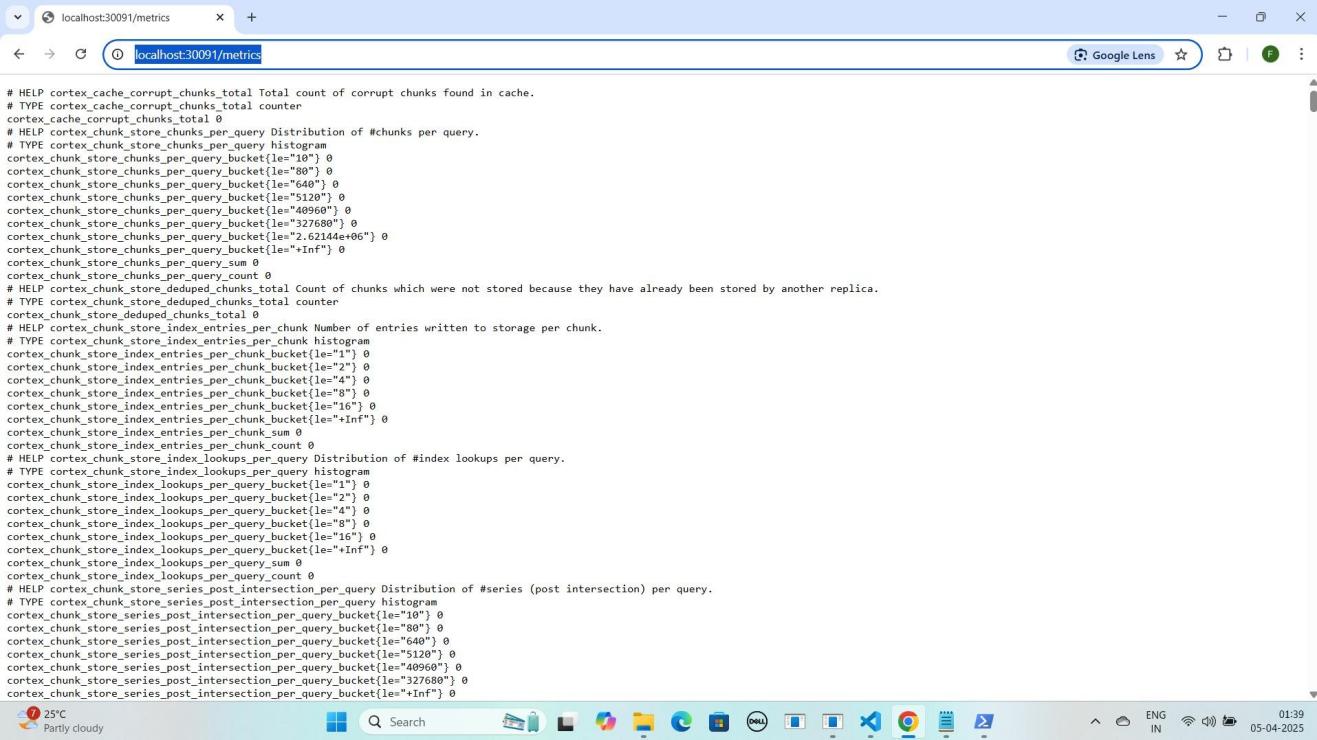
YE KUCH ISTARHA LAGEGA



1. Agar aapko **ready** show karraha hai to iska matlab **loki successfully** run horaha hai ab **loki metrics loggs** collect karraha hai **check** karne ke liye

**/metrics** run kariye jaise ki mere case me kuch aisa hoga. [http://localhost:30091/metrics](hm)

YE KUCH ISTARHA LAGEGA



### Step 7: grafana-deployment.yaml File Ka Kaam

Yeh **file Grafana** ko deploy karne ke liye use hoti hai, taake hum **Prometheus metrics** aur **Loki logs** ko **visualize** kar sakein. Isme **NodePort Service** bhi use ki gayi hai, jisse **Grafana port 30080** par **expose** hoga.

JAISE KI:-

apiVersion: apps/v1 kind: Deployment metadata:

name: grafana namespace: monitoring

spec:

replicas: 1 selector:

matchLabels:

app: grafana template:

metadata:

labels:

app: grafana spec:

containers:

* name: grafana

image: grafana/grafana:8.3.0 ports:

* + containerPort: 3000 env:
  + name: GF\_SECURITY\_ADMIN\_PASSWORD

value: "admin"

---

apiVersion: v1 kind: Service metadata:

name: grafana namespace: monitoring

spec:

type: NodePort # Change the type from ClusterIP to NodePort ports:

- port: 80

targetPort: 3000

nodePort: 30080 # Specify the NodePort (e.g., 30080, any unused port in the range 30000-32767)

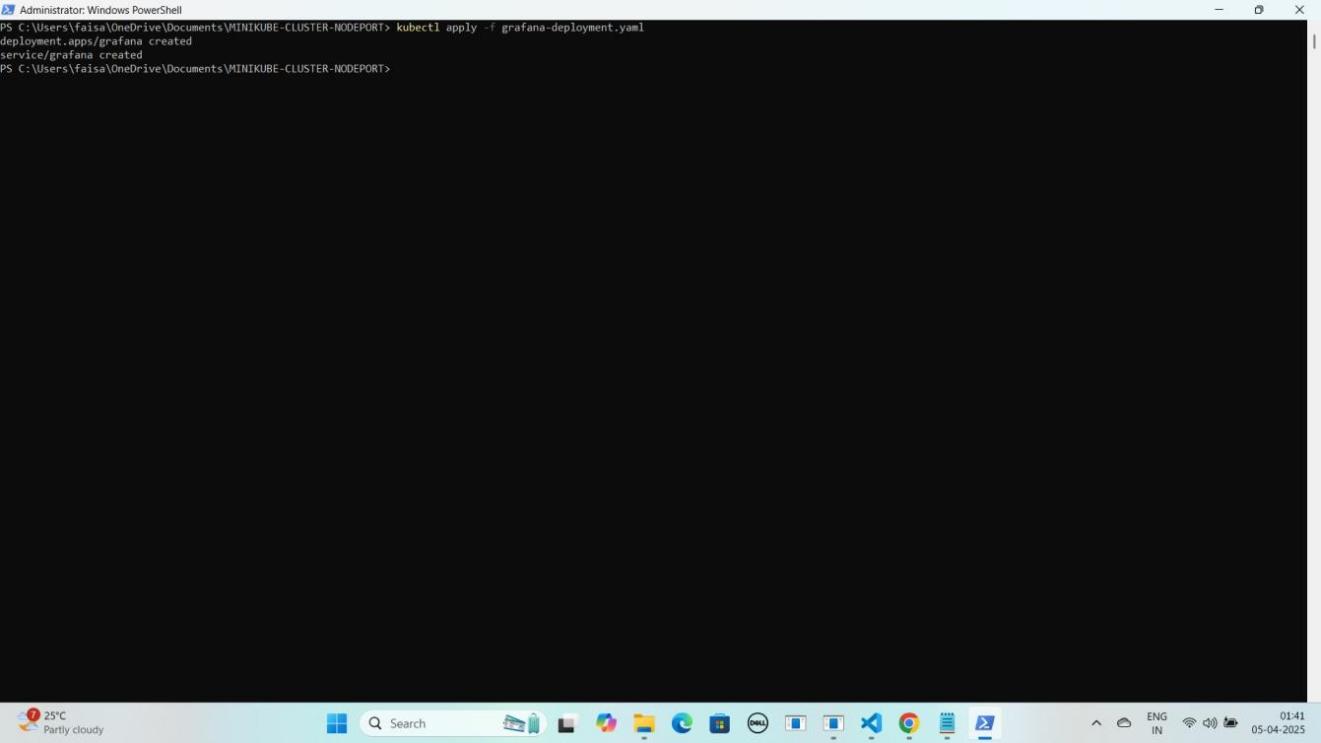
selector:

app: grafana

### Grafana Deployment Apply Karo

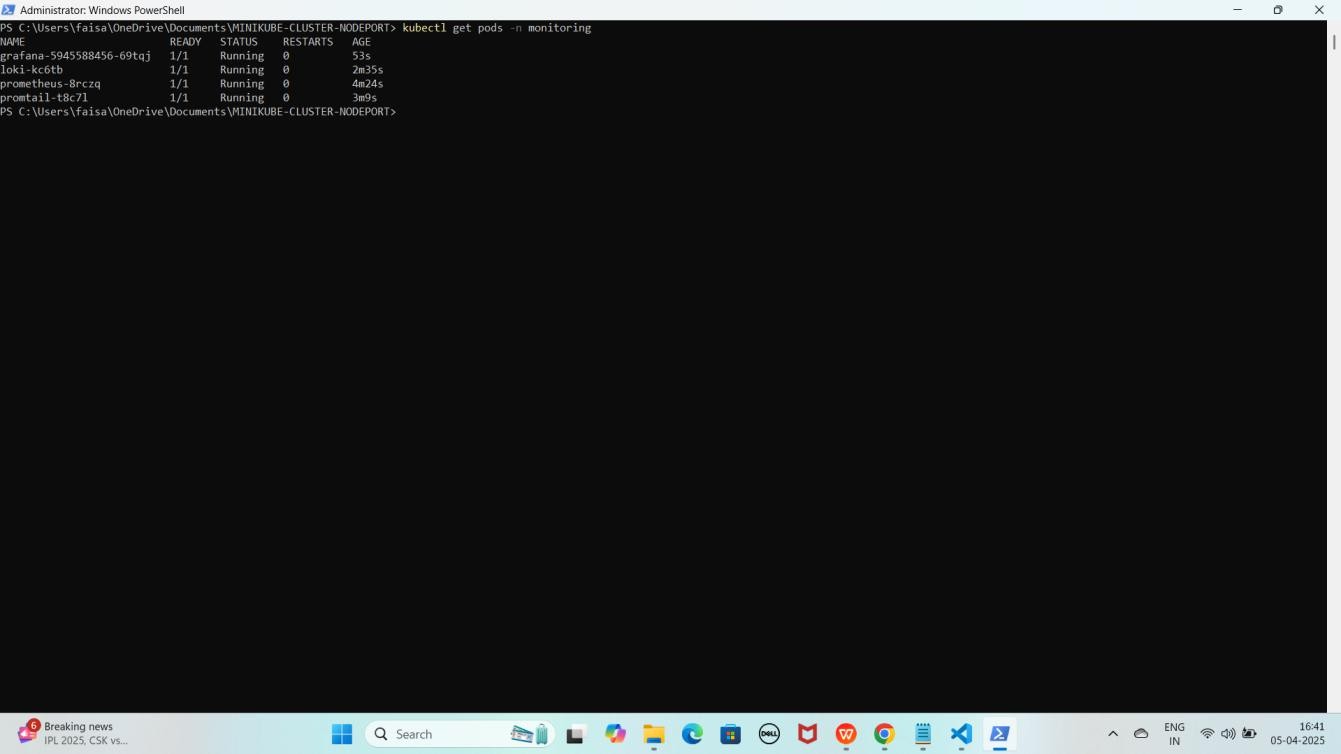
kubectl apply -f grafana-deployment.yaml

YE KUCH ISTARHA LAGEGA



#### Pods check karne ke liye ye command run kariye

kubectl get pods -n monitoring

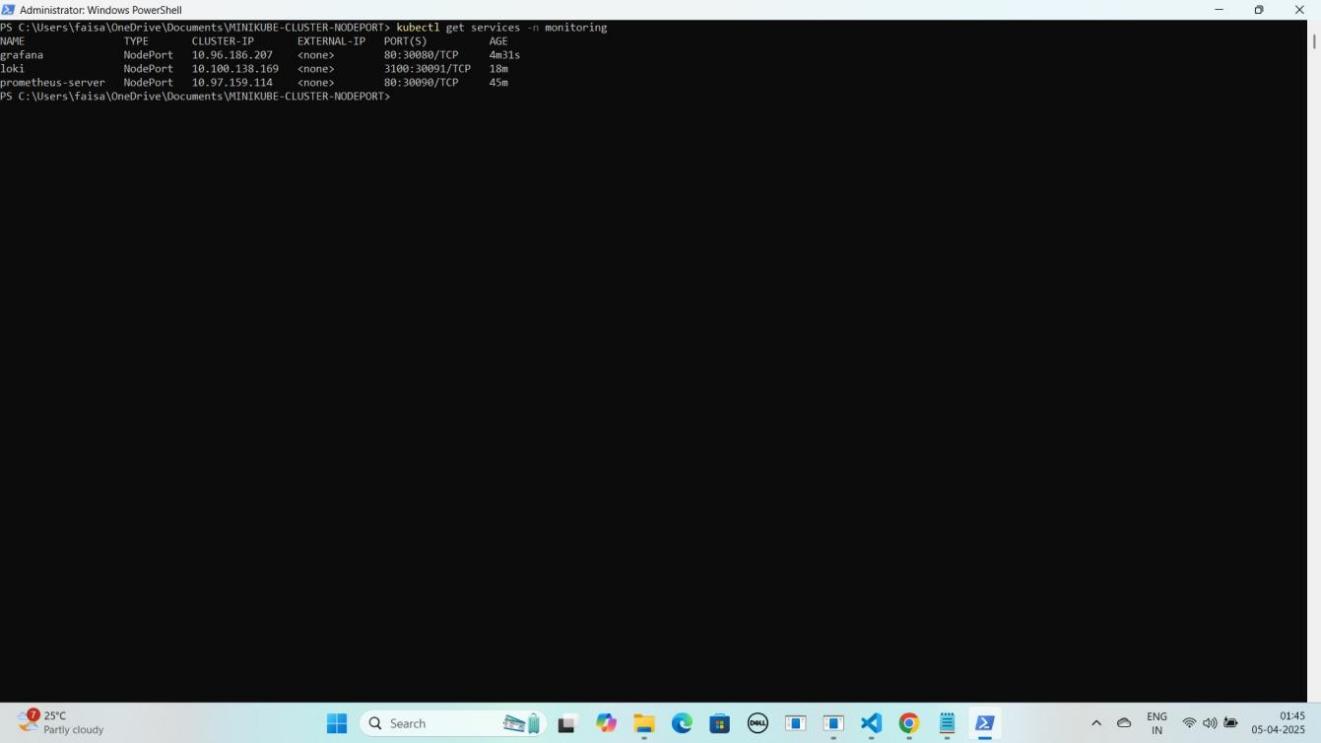
YE KUCH ISTARHA LAGEGA

#### NOTE: Agar STATUS Running show karraha hai to sab kuch sahi hai

1. **Serives check karne ke liye ye commad run kariye**

kubectl get services -n monitoring

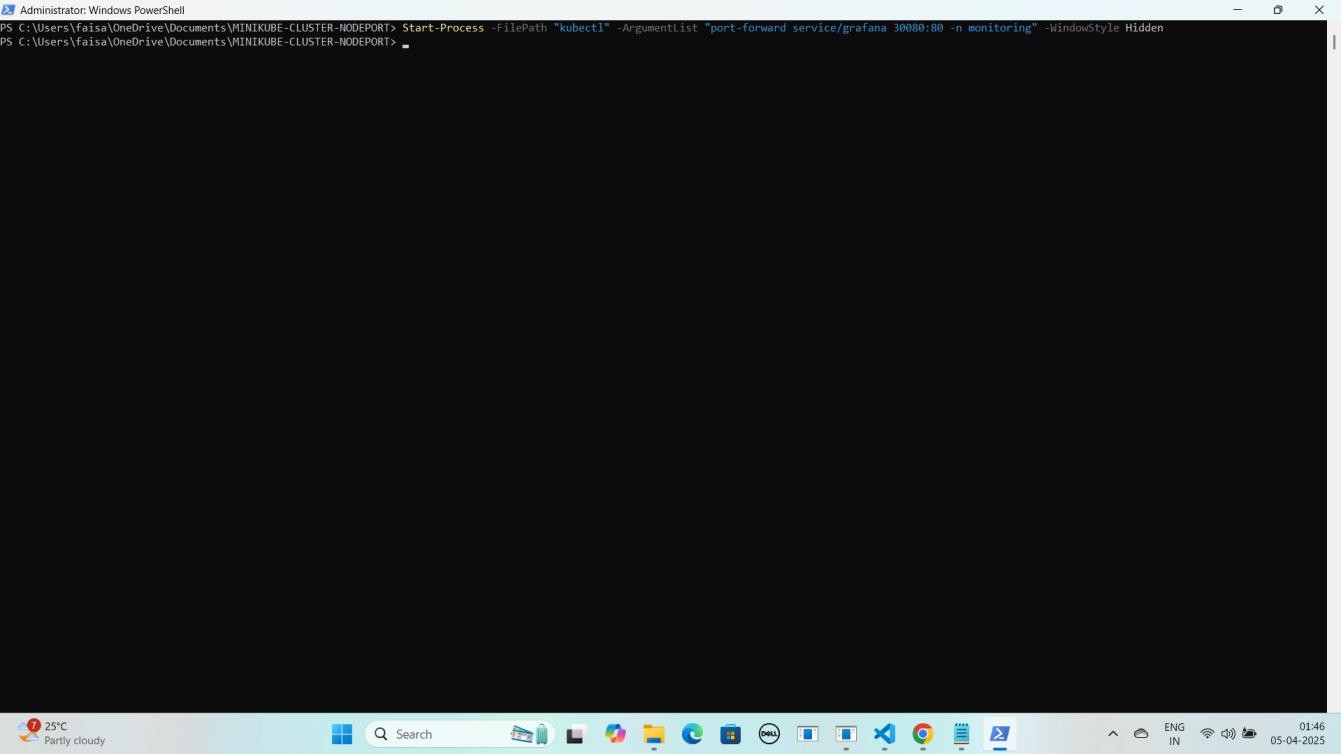
YE KUCH ISTARHA LAGEGA



#### Port Forwarding Karne ke liye ye command run karein

Start-Process -FilePath "kubectl" -ArgumentList "port-forward service/grafana 30080:80 -n monitoring" -WindowStyle Hidden

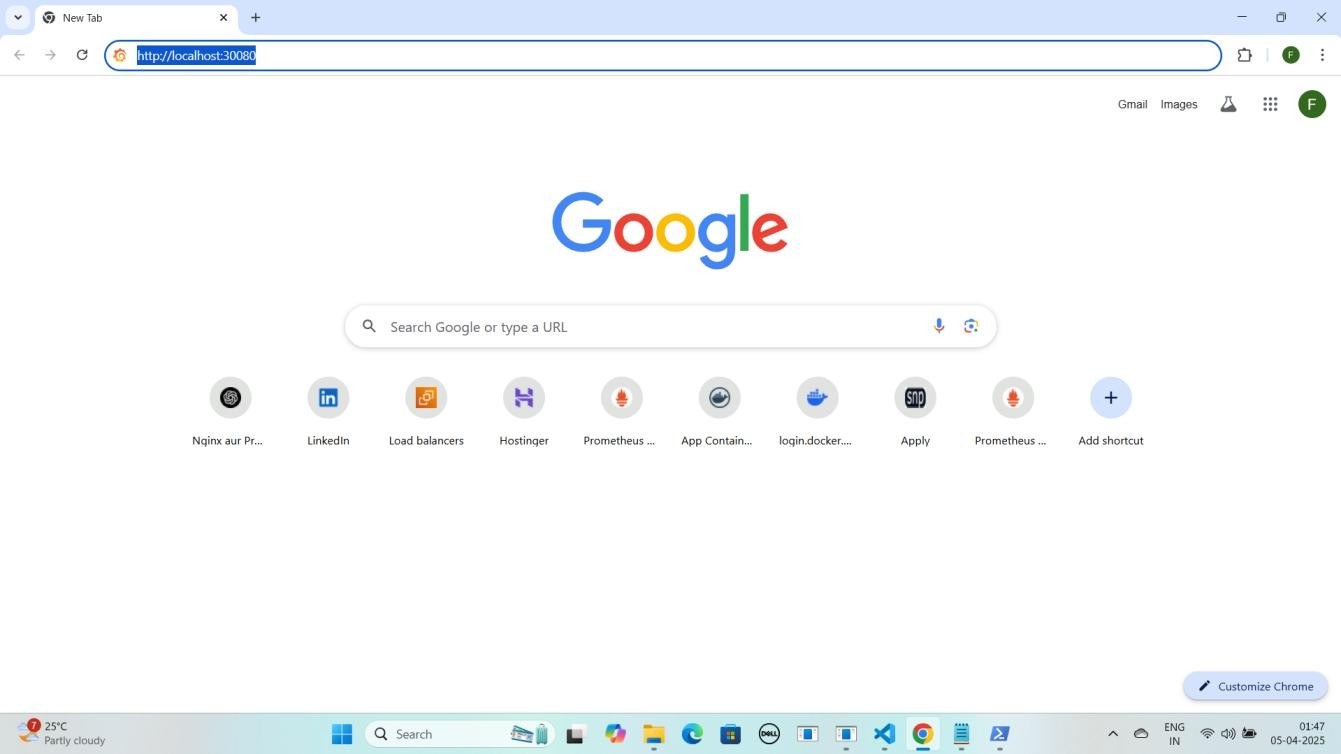
YE KUCH ISTARHA LAGEGA

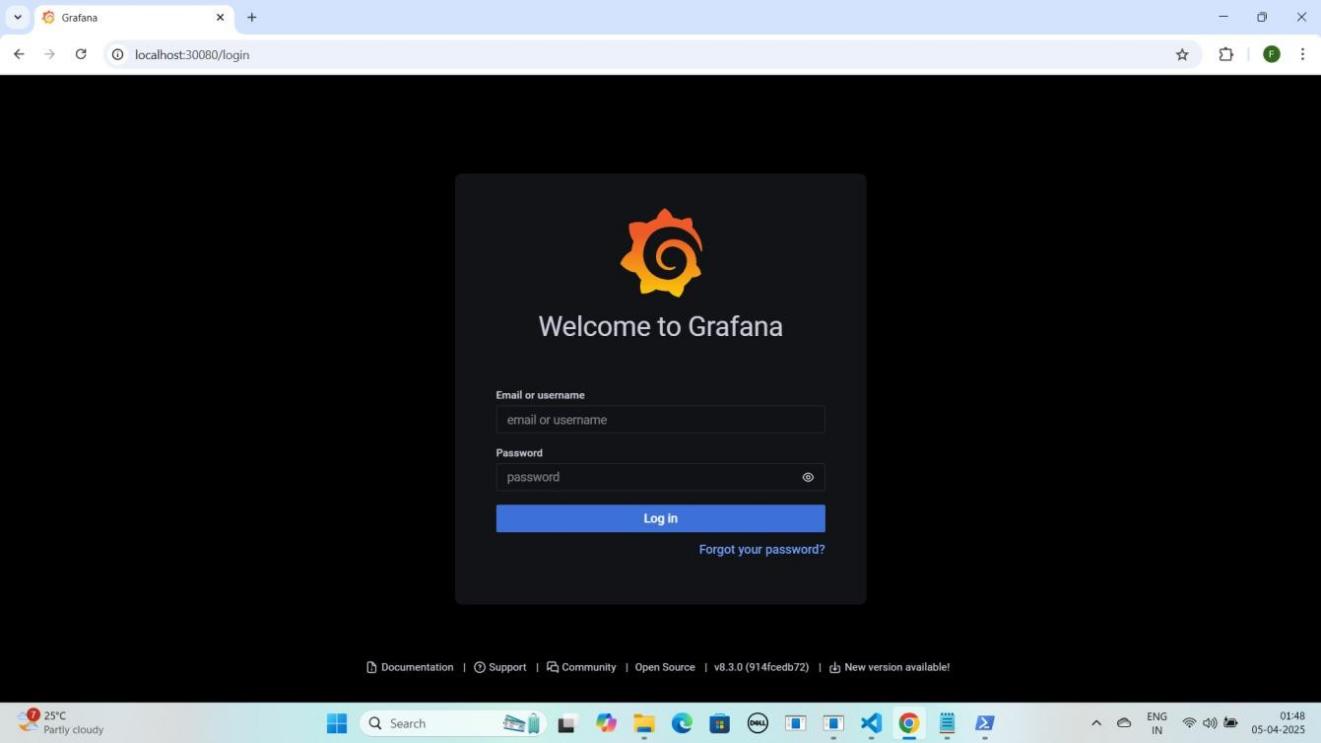


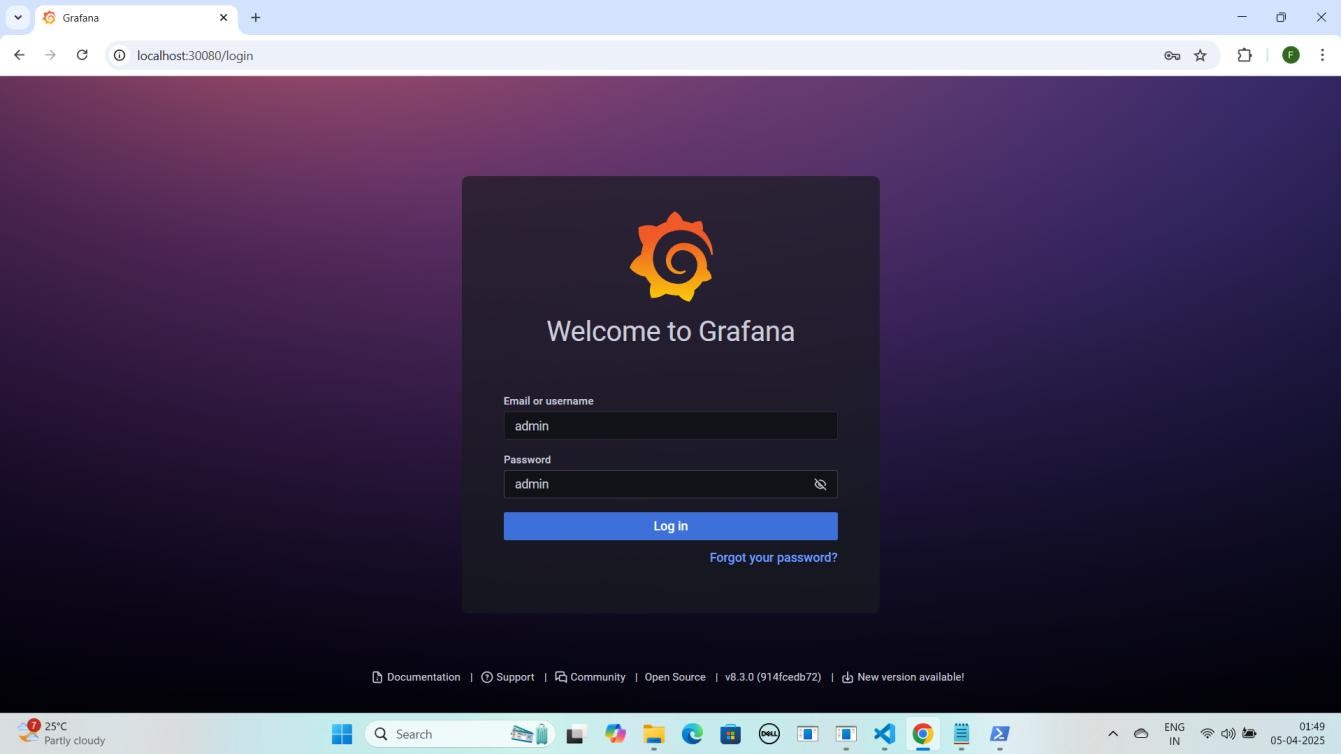
#### NOTE: Ab aapka Grafana expose ho chuka hai. Ab apne local system par Grafana ko NodePort ke saath browser me run kariye. Jaise ki mere case me kuch aisa hoga

**Grafana:** [http://localhost:30080](d)

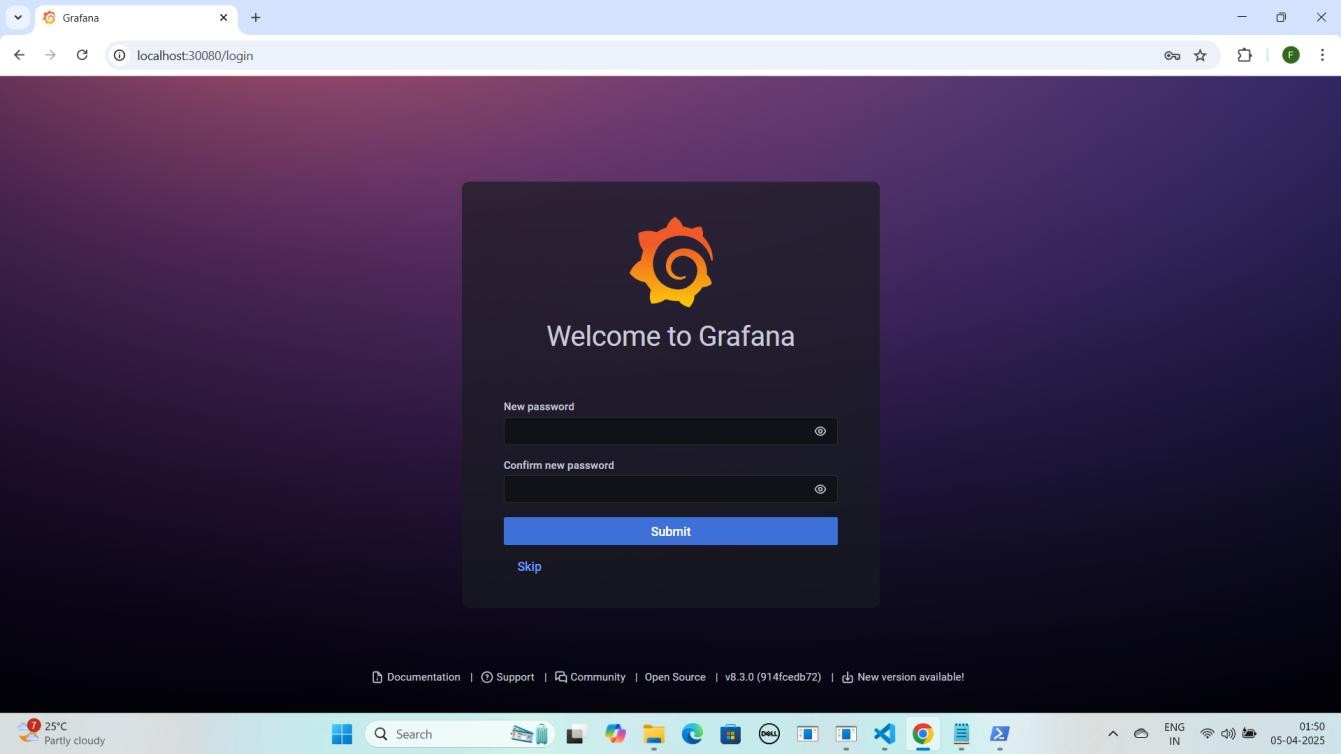
YE KUCH ISTARHA LAGEGA

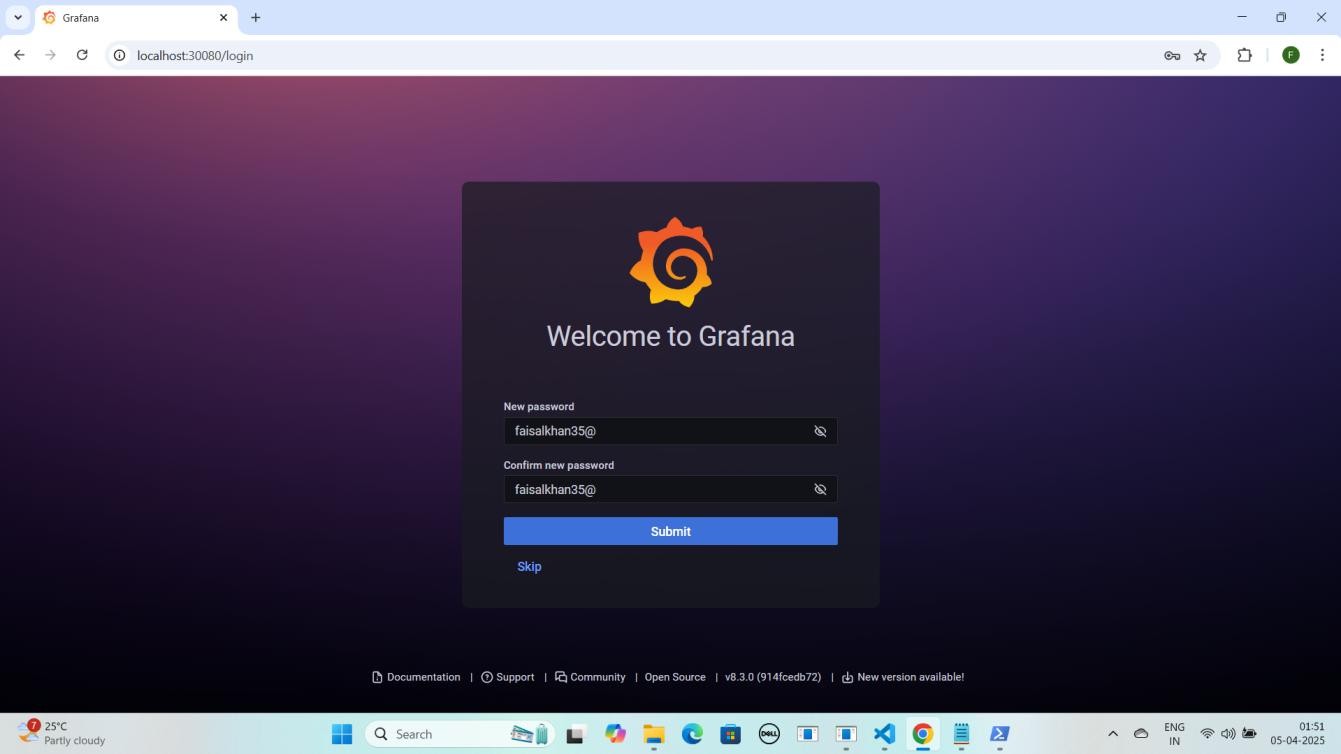






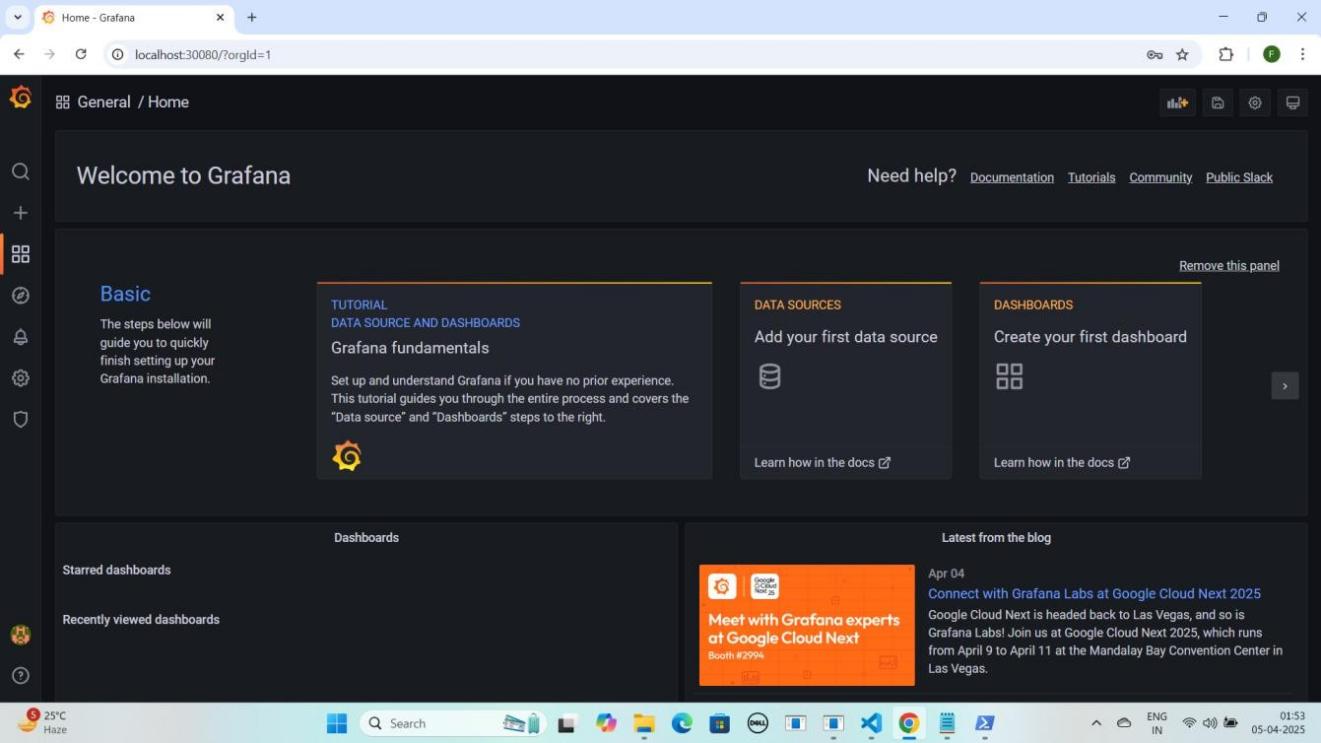
**NOTE : Grafana browser me run hone ke baad aapse login ke liye username aur password maangega. Default username aur password admin hota hai. Login ke baad aap apna password change kar sakte hain.**



****

**NOTE : Is step par aapko apne pasand ka naya password set karsakte hai.**

**Yaha mai faisalkhan35@ password set kar raha hoon.Password type karne ke baad Submit par click kariye**



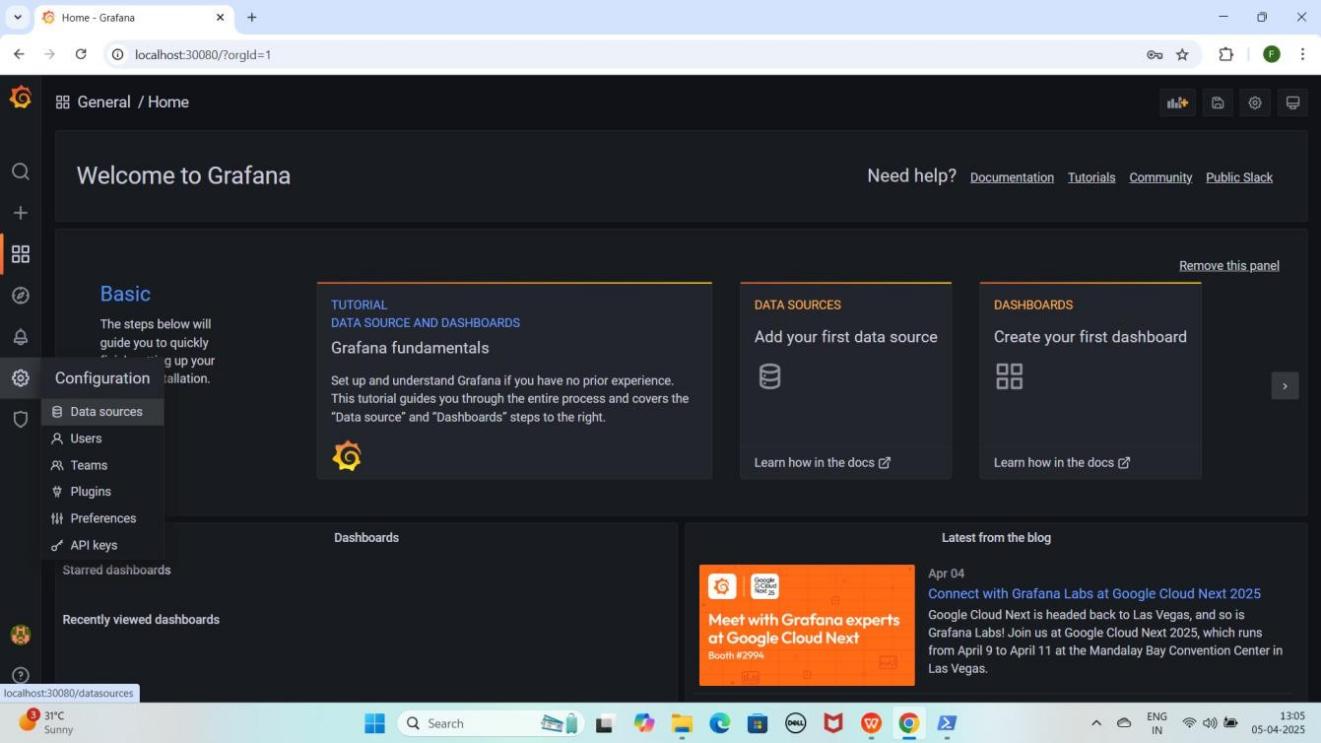
### Step 8: Grafana Mein Prometheus Data Source Add Karna, Queries Add Karna Aur Dashboard Create Karna

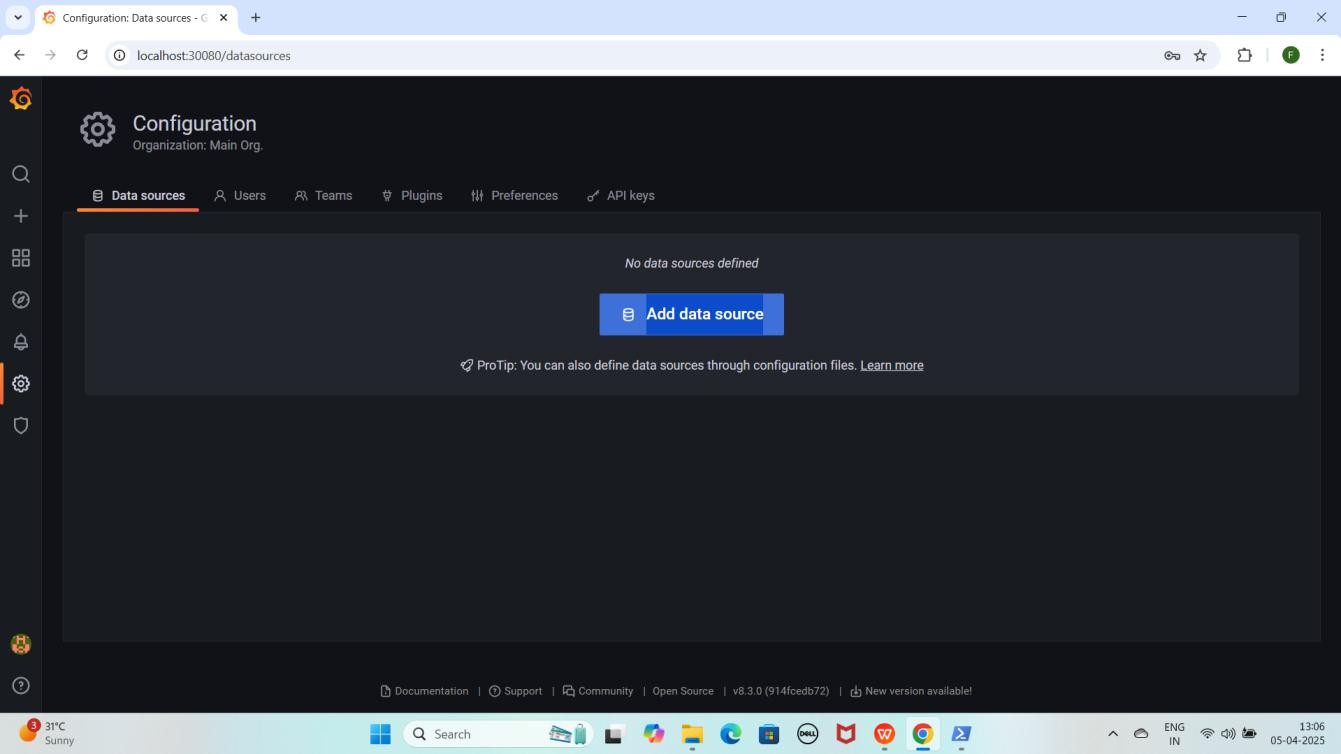
* **Grafana dashboard** open karo.
* **Configuration** pe **click** karo, phir **Data Sources select** karo.
* **Add data source** pe **click** karo.
* **Prometheus** select karo.
* minikube service prometheus-server -n monitoring --url command **Minikube VM** ki **IP** deta hai, Is **IP** ko **Grafana** me **datasource URL** me use karo.
  + **Mere case me yeh kuch is tarah hogi [http://192.168.59.118:30090](h)**

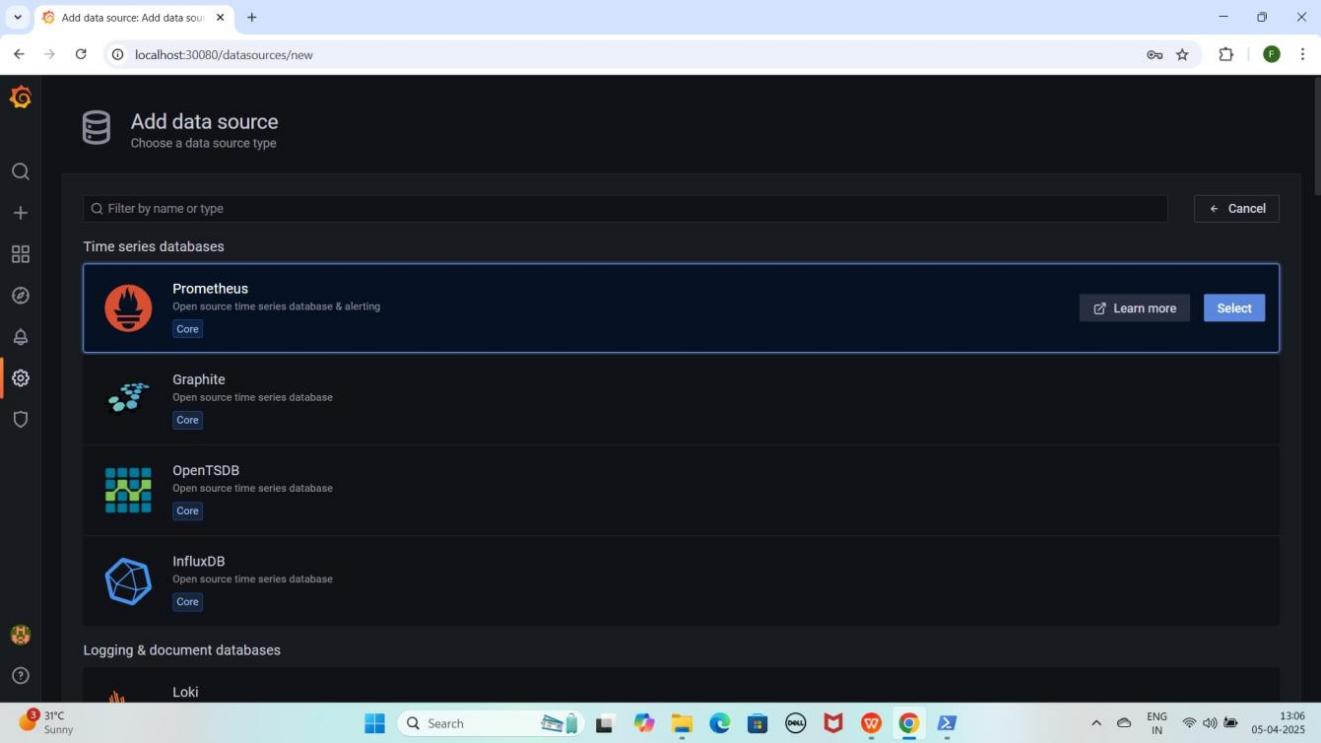
#### Aapke case me IP different ho sakti hai.

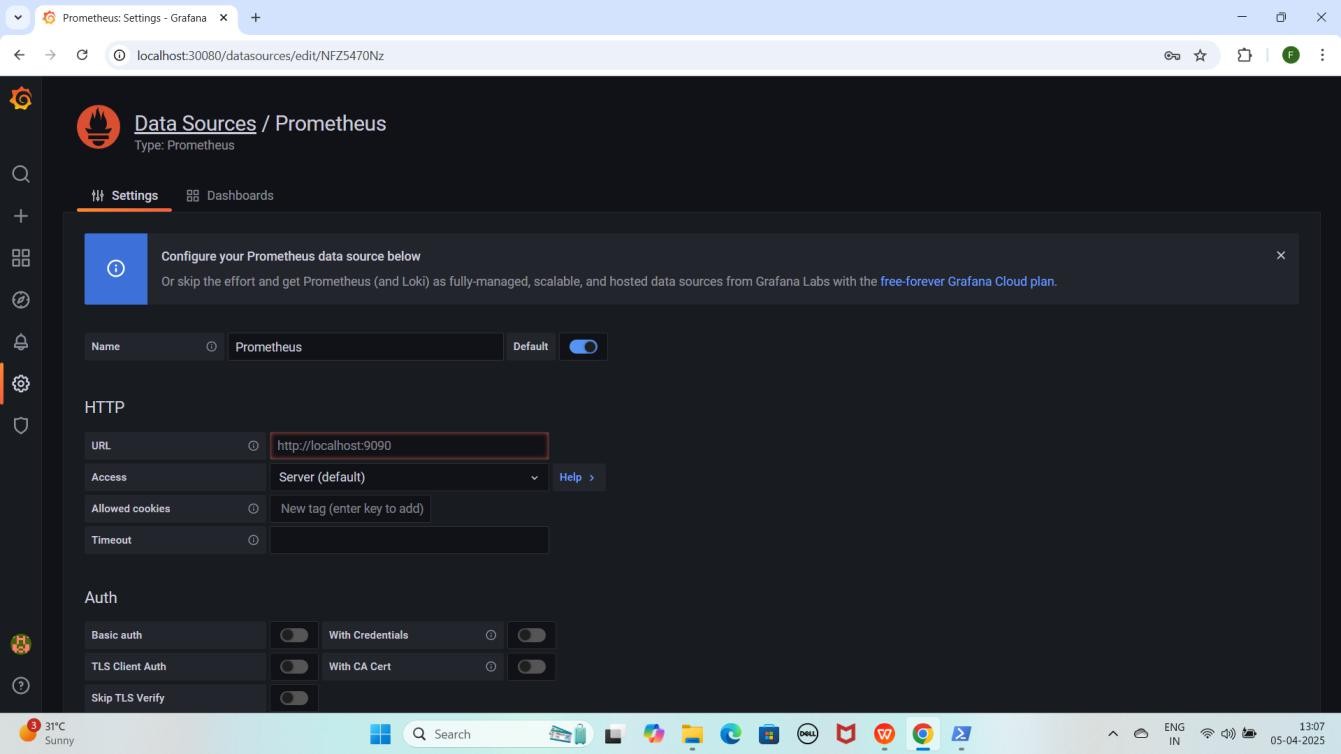
* **HTTPS section** me **URL** box me **paste** karo.
* Neeche scroll karo aur **"Save & Test"** pe **click** karo.

YE KUCH ISTARHA LAGEGA



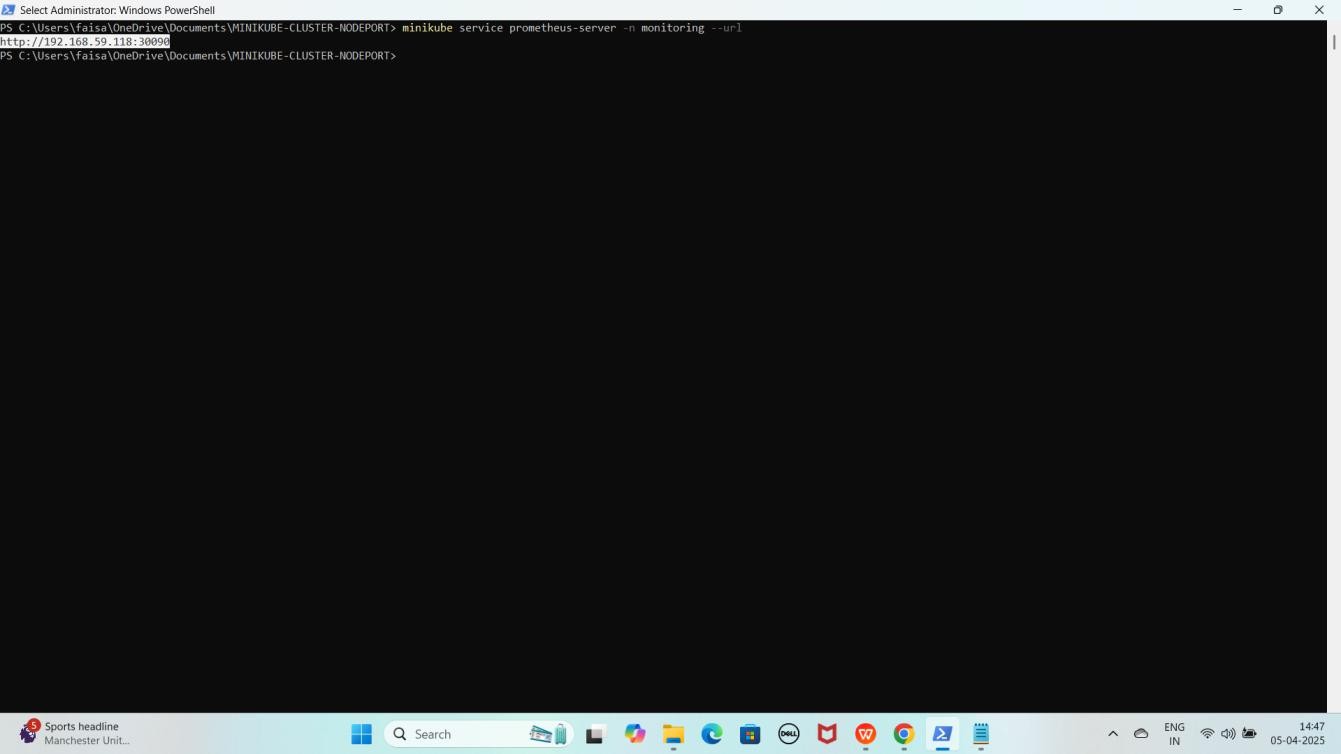


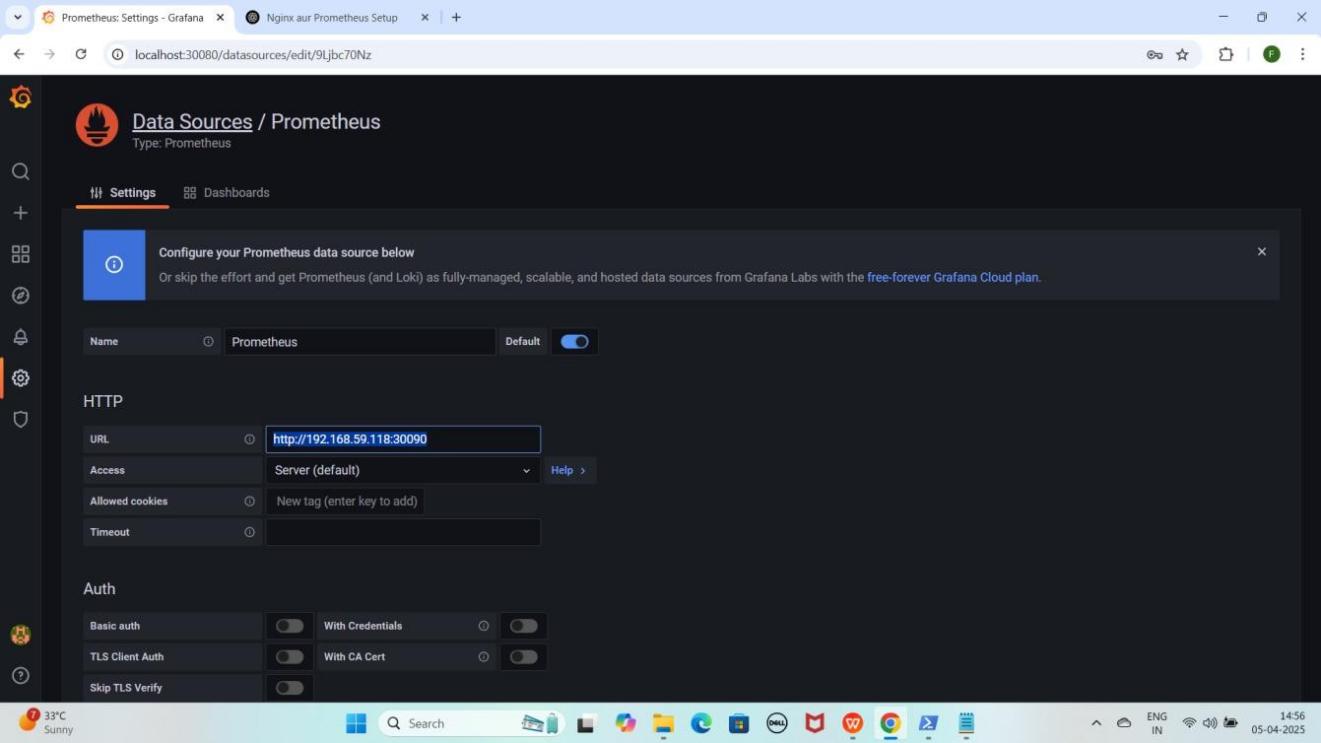


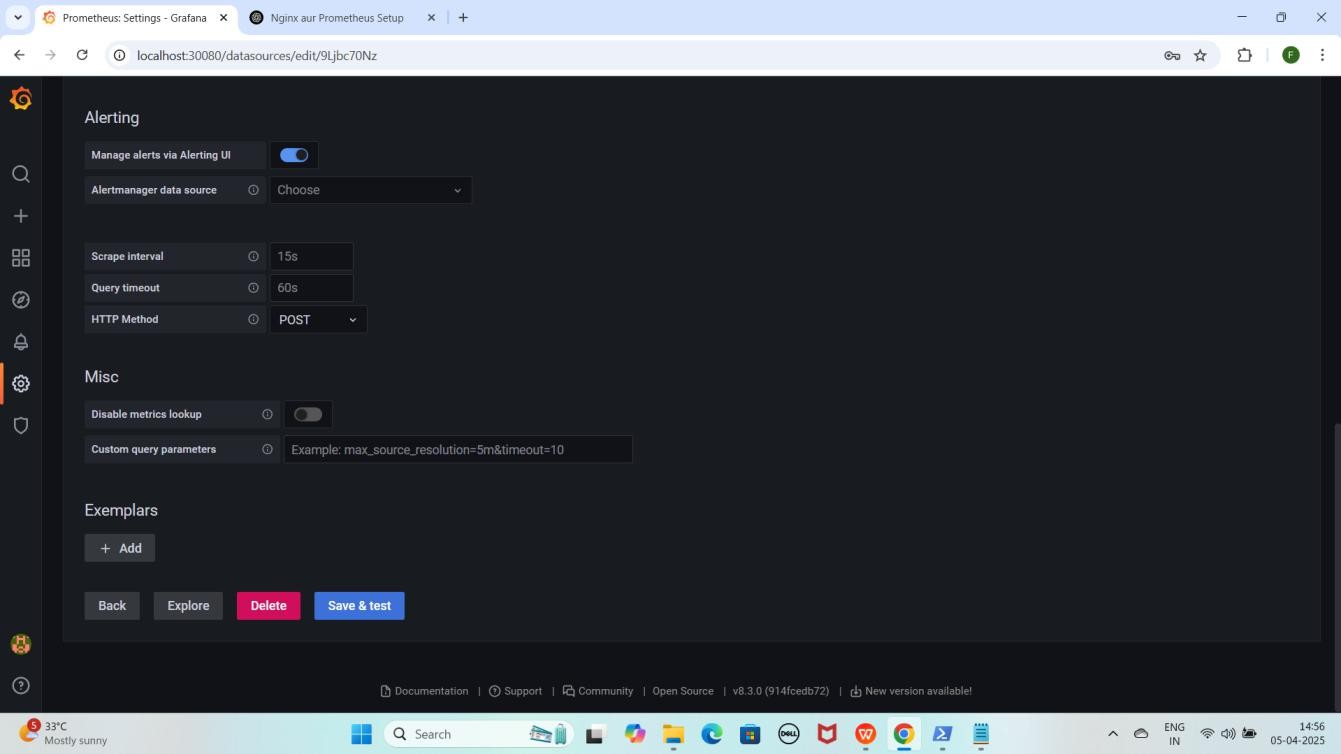


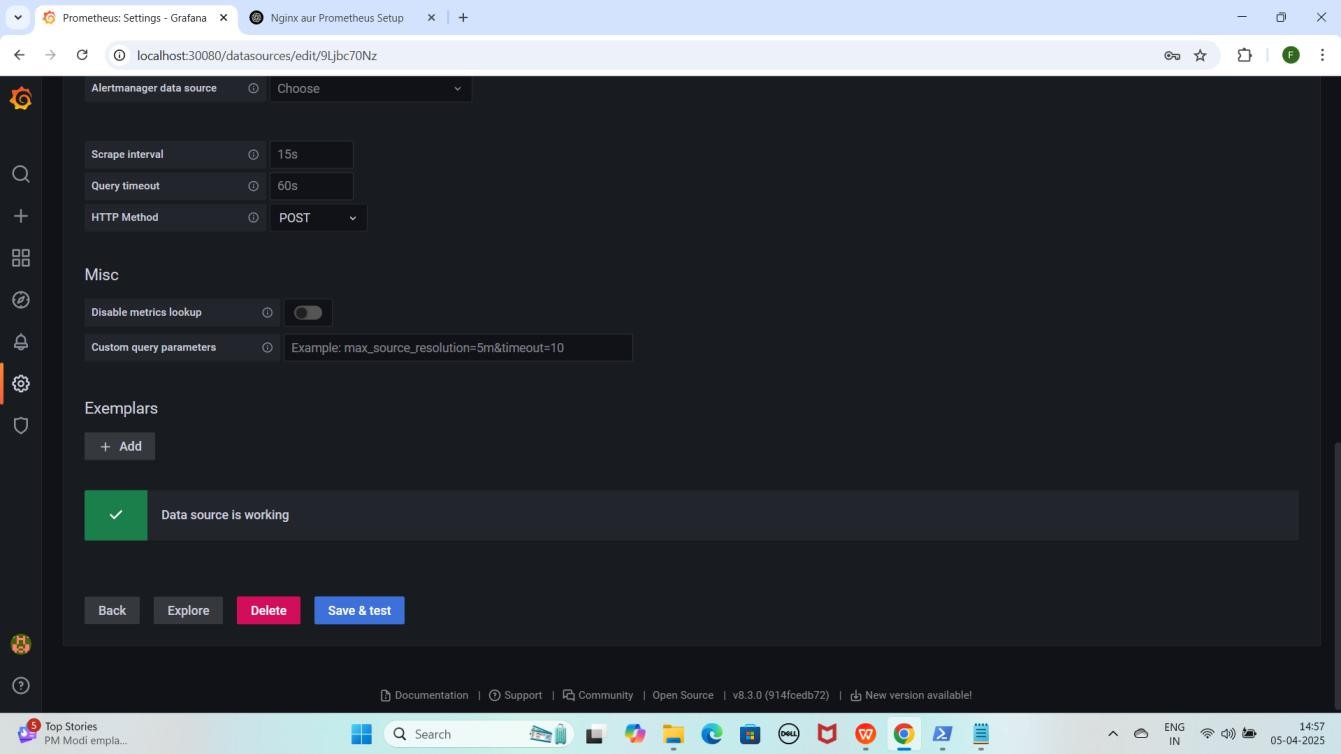
#### NOTE: Minikube VirtualBox me run ho raha ho, to Grafana me localhost kaam nahi karega. Uski jagah minikube ip use karenge kyunki localhost Grafana ke container ka hota hai, host machine ka nahi.

1. **Proper URLs ke liye, Prometheus datasource add karne ke liye yeh command run karein**

minikube service prometheus-server -n monitoring --url YE KUCH ISTARHA LAGEGA



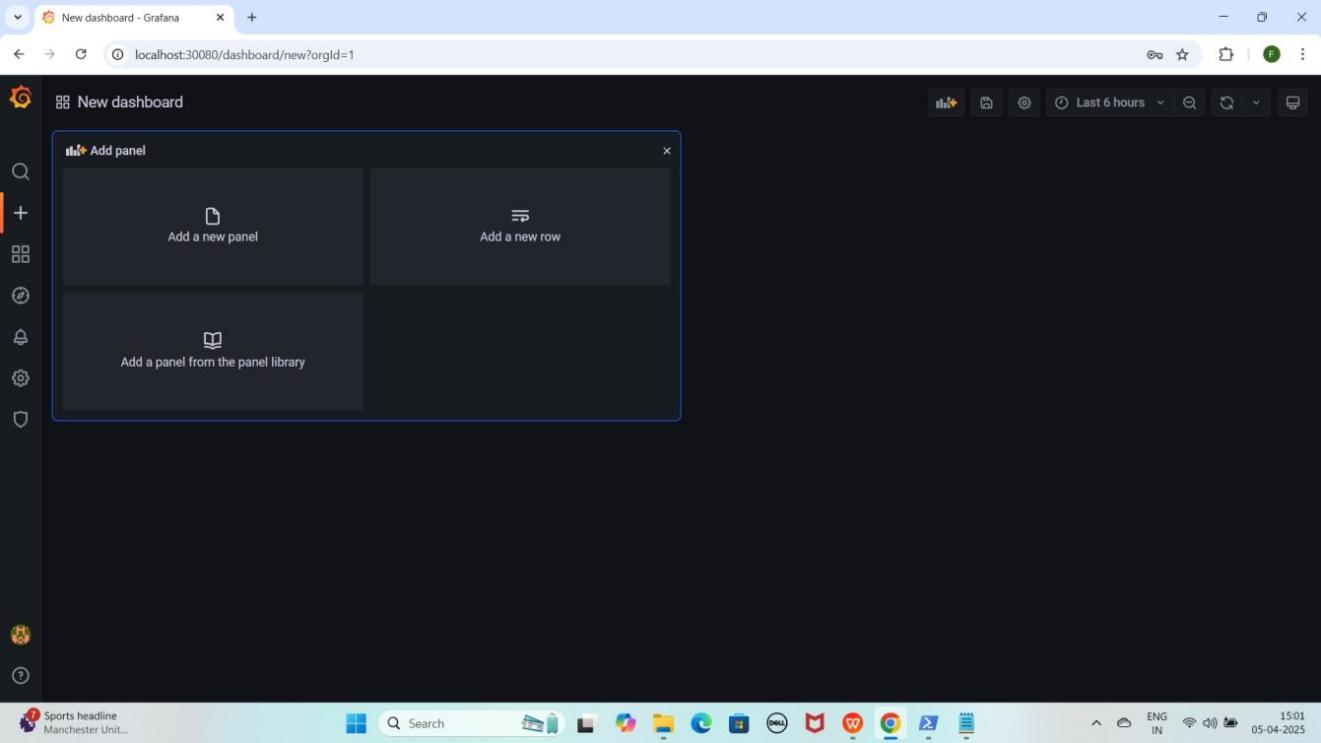
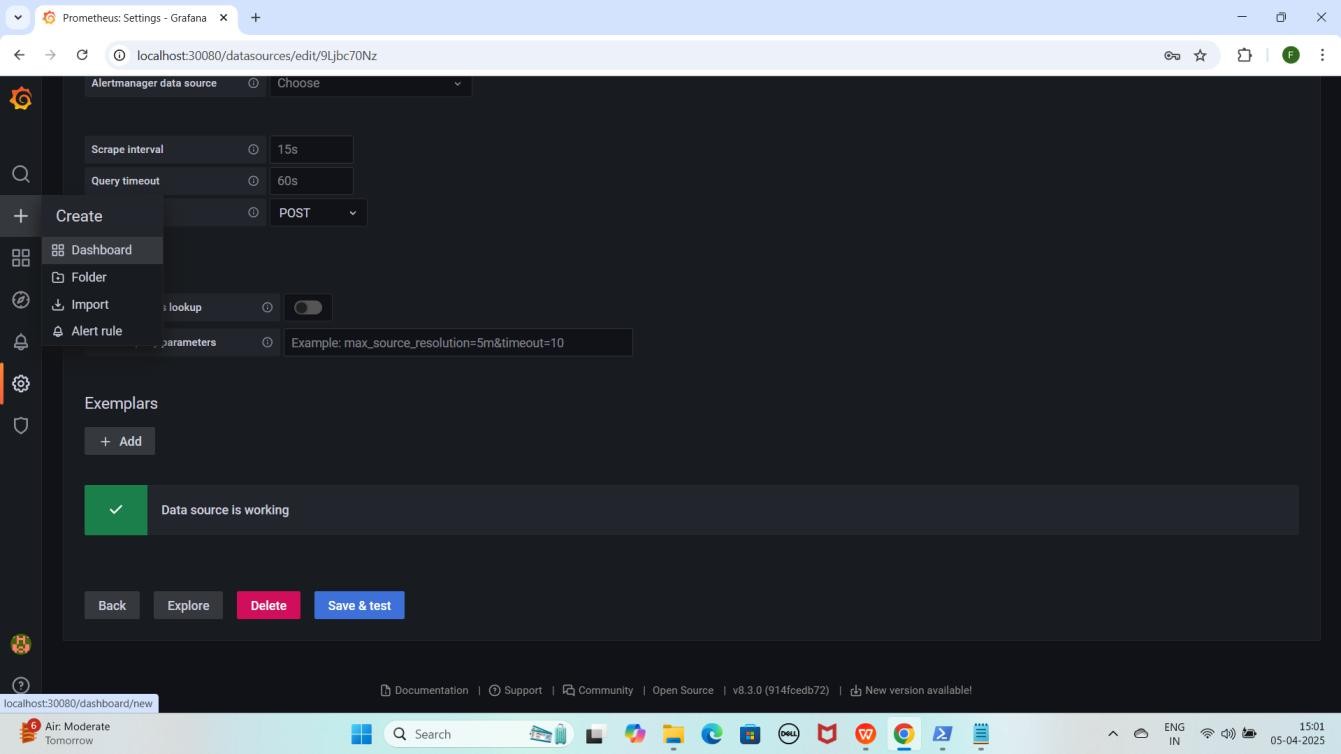


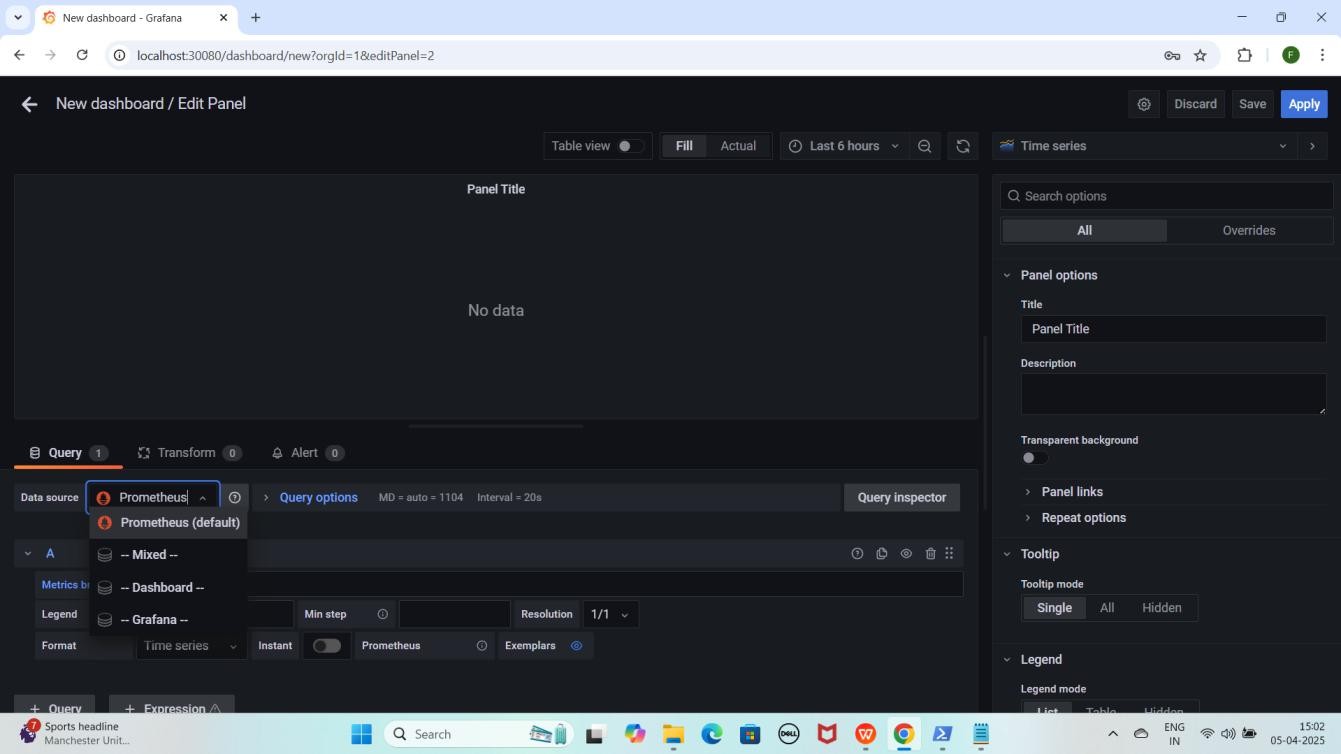


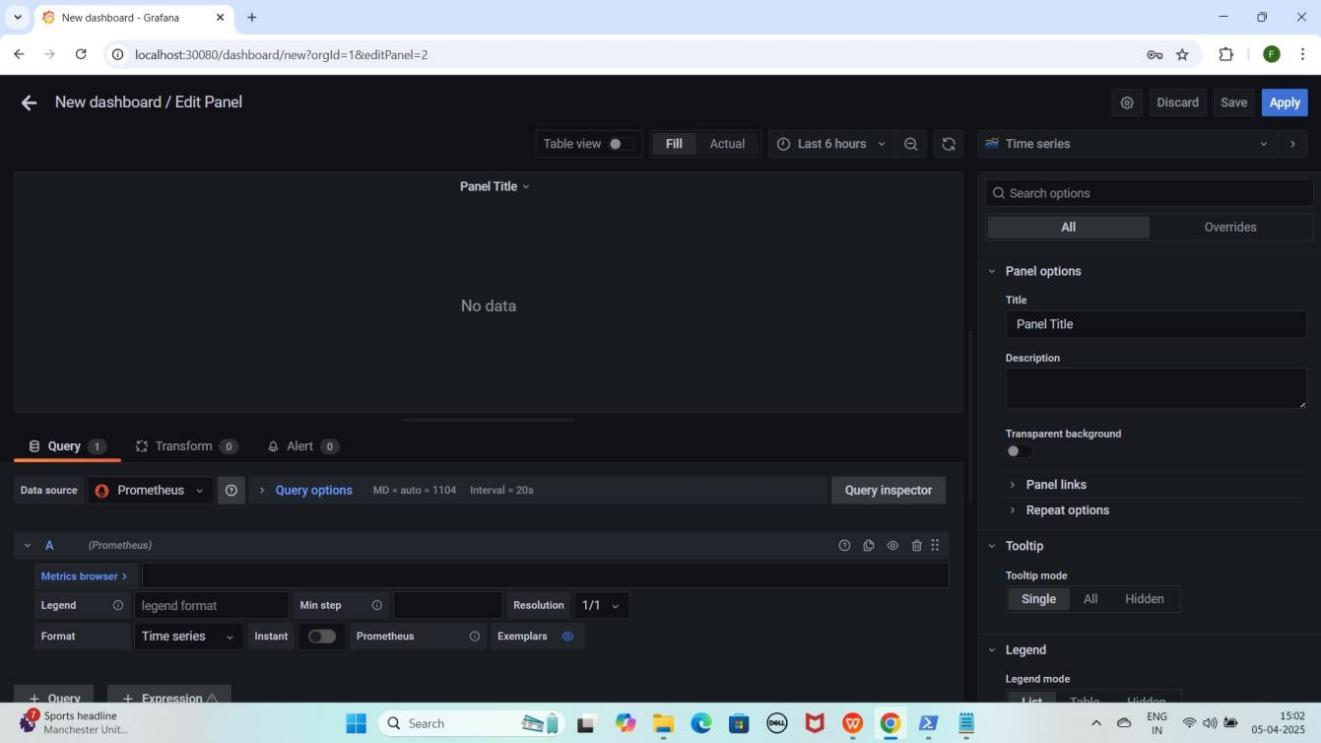
#### NOTE: Save & test pe click karne ke baad aapko Data source is working karke pop up aayega

* + **Prometheus ke Metrics visualize** karne ke liye **"+" (plus) icon** pe click karo.
  + **Dashboard** select karo aur **"Add a new panel"** pe click karo.
  + **Query section** me **Data Source ko "Prometheus"** select karo.

YE KUCH ISTARHA LAGEGA





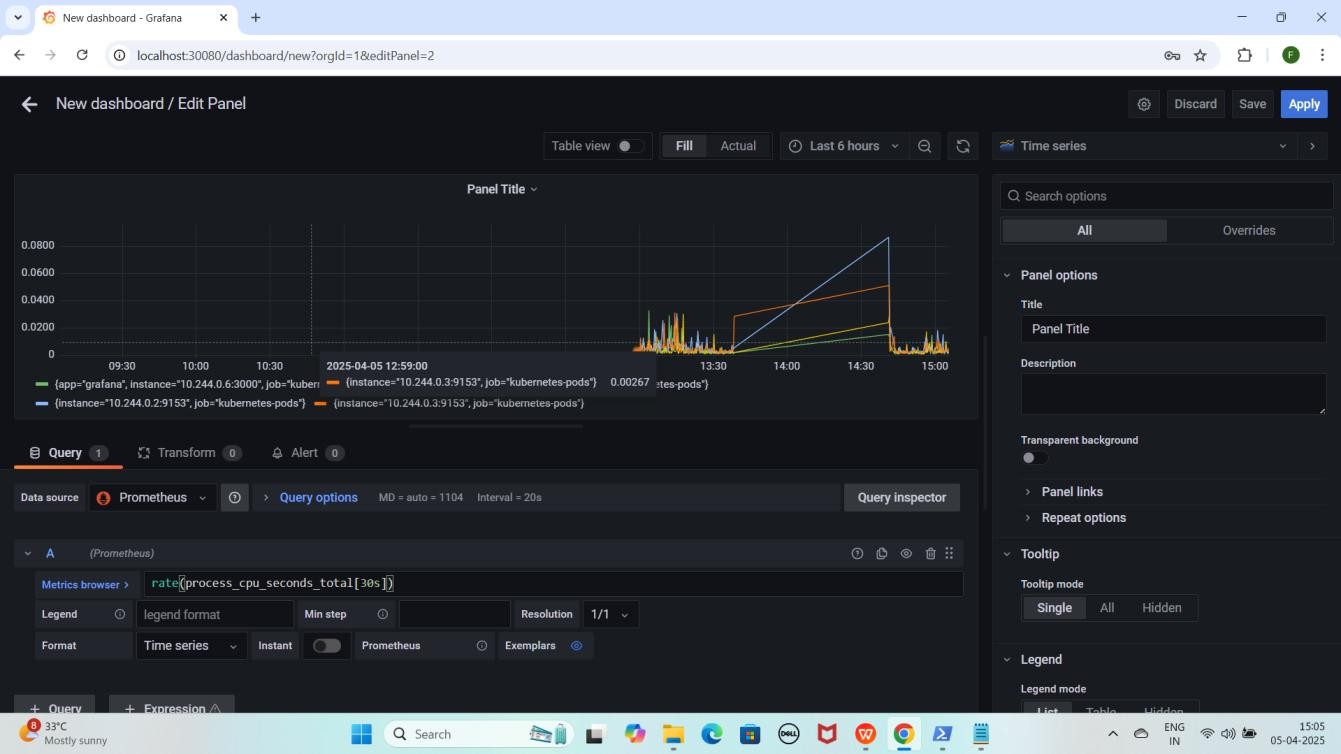


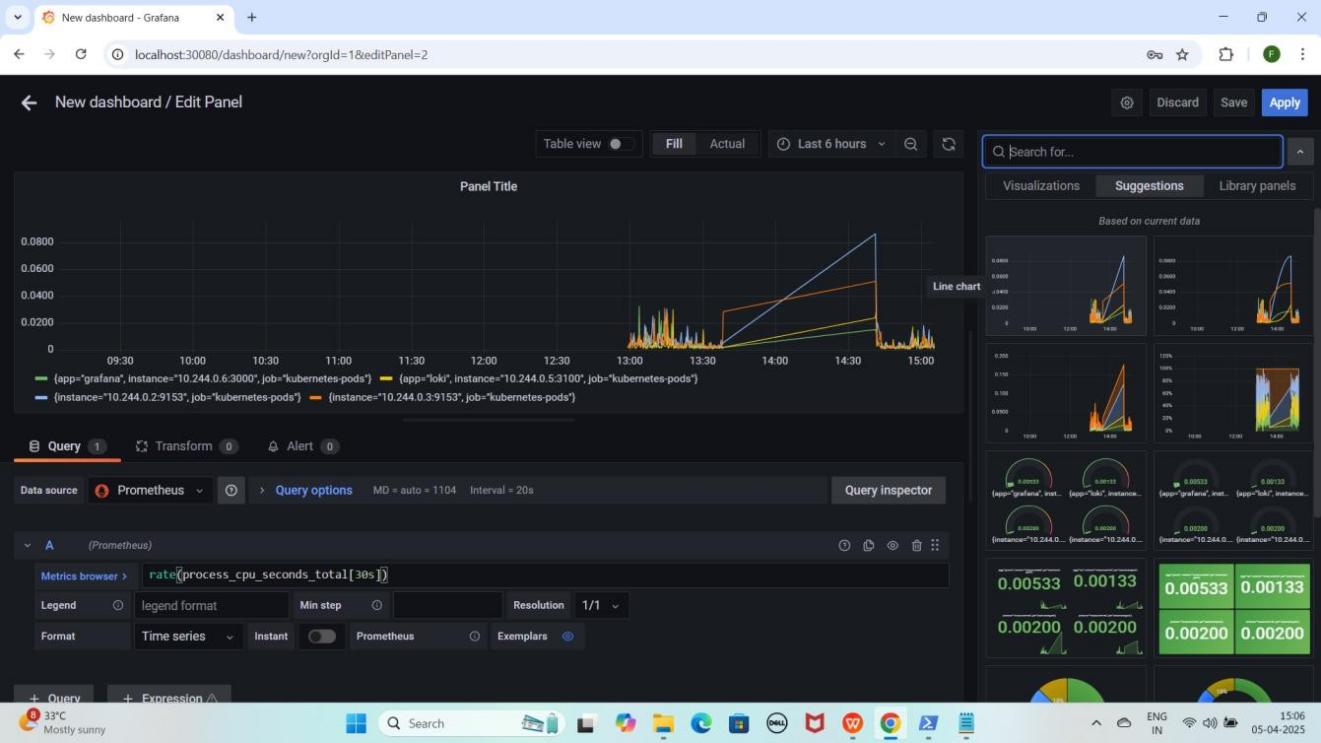
#### Prometheus ko data source select karne ke baad, CPU Usage dekhne ke liye Metric Browser me yeh query paste karein

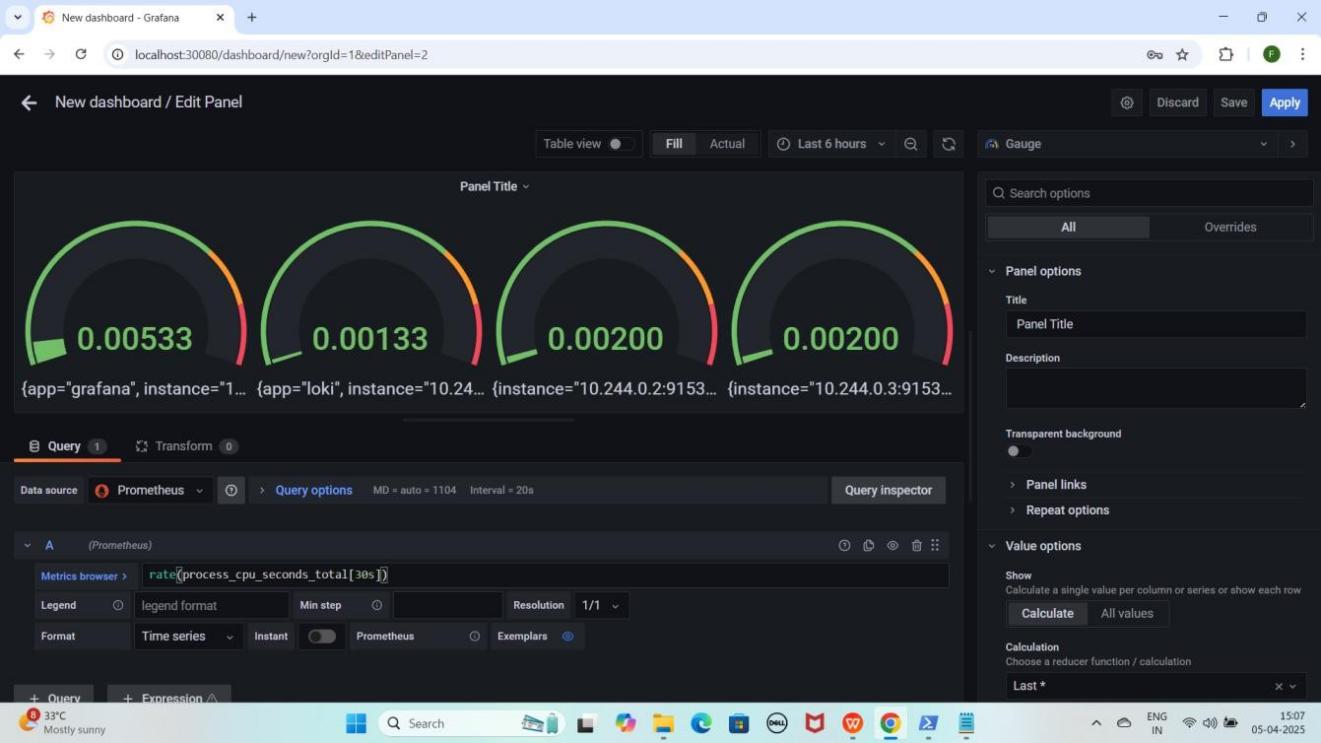
rate(process\_cpu\_seconds\_total[30s])

* + **"Time Series"** pe click karein.
  + **Suggestions** me se **"Dashboard"** select karein jaise maine select kiya hai.

YE KUCH ISTARHA LAGEGA



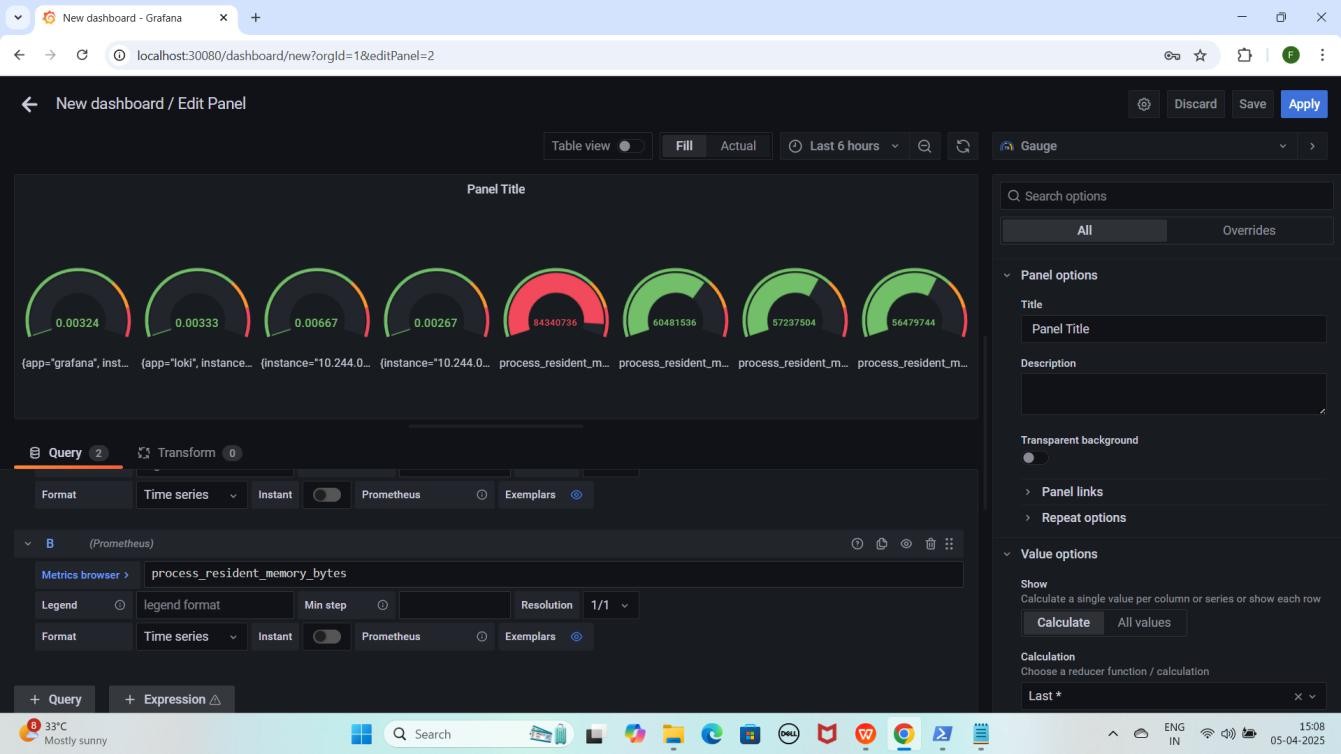




#### Memory Usage dekhne ke liye "+ Query" par click karein.

**Yeh query paste karein taake memory usage dekh sakein**

process\_resident\_memory\_bytes

YE KUCH ISTARHA LAGEGA

**NOTE : Itna karne ke baad aap 30 sec ke interval par CPU usage, current Memory Usage ko Monitor kar sakte hain.**

### Step 9: Grafana Mein Loki Data Source Add Karna Aur Dashboard Create Karna

* + **Grafana** open karo aur **Settings** icon pe click karo.
  + **Data Sources** select karo.
  + **Add data source** pe click karo.
  + Search bar me **"Loki"** likho aur **Loki** select karo.
  + minikube service loki -n monitoring --url command Minikube VM ki IP deta hai, Is IP ko Grafana me datasource URL me use karo:

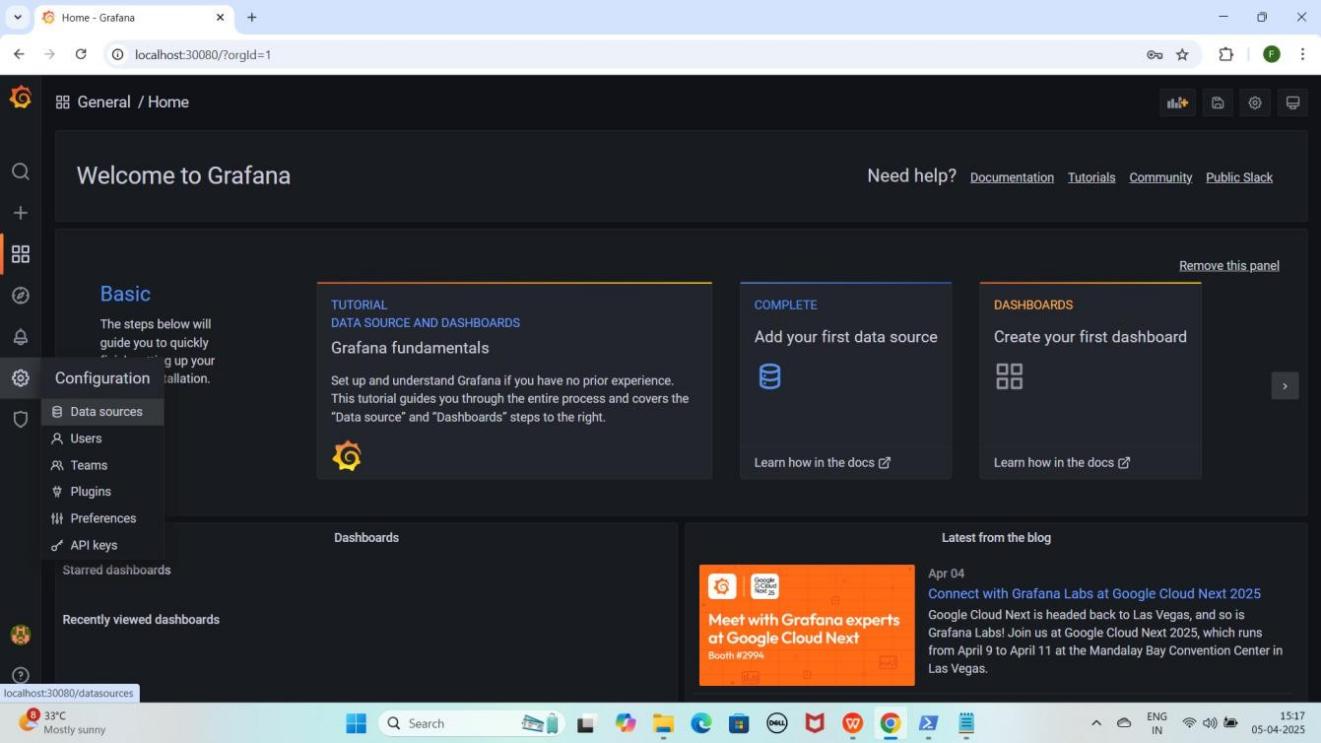
#### Mere case me yeh kuch is tarah hogi:

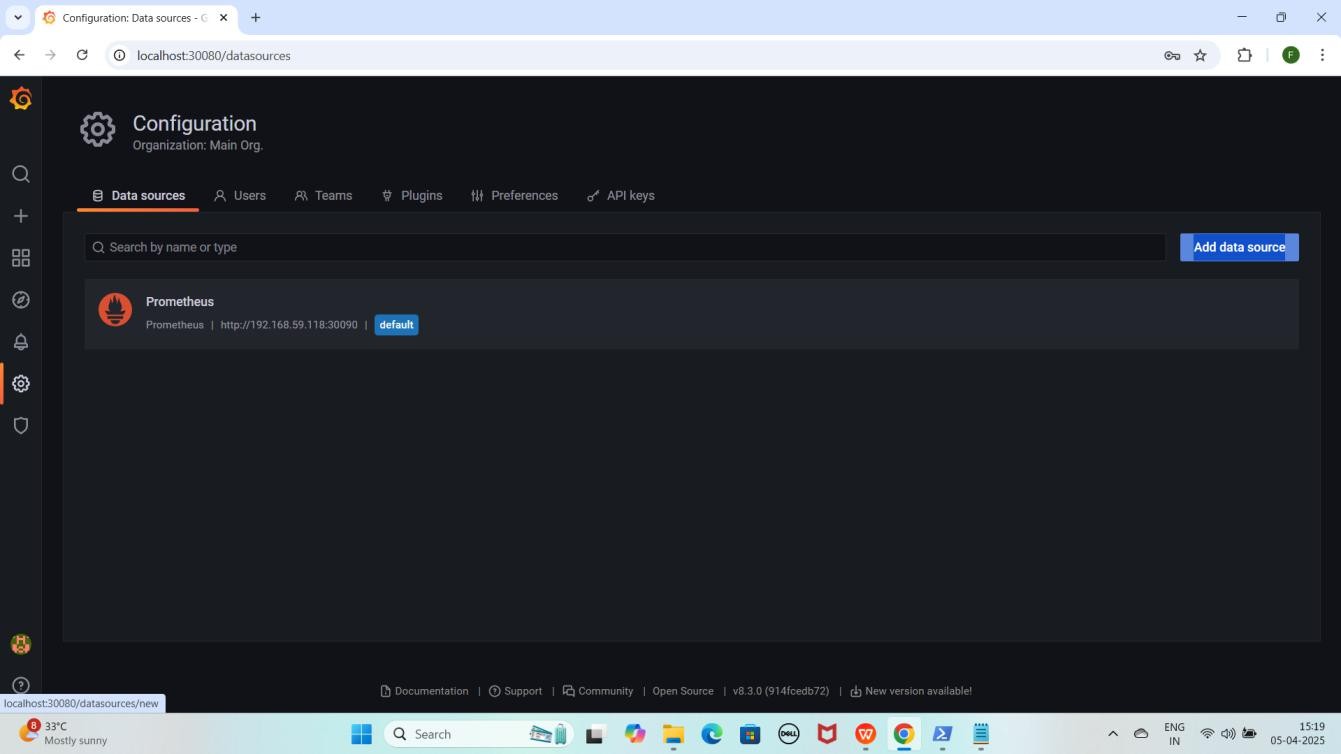
[http://192.168.59.118:30091](h)

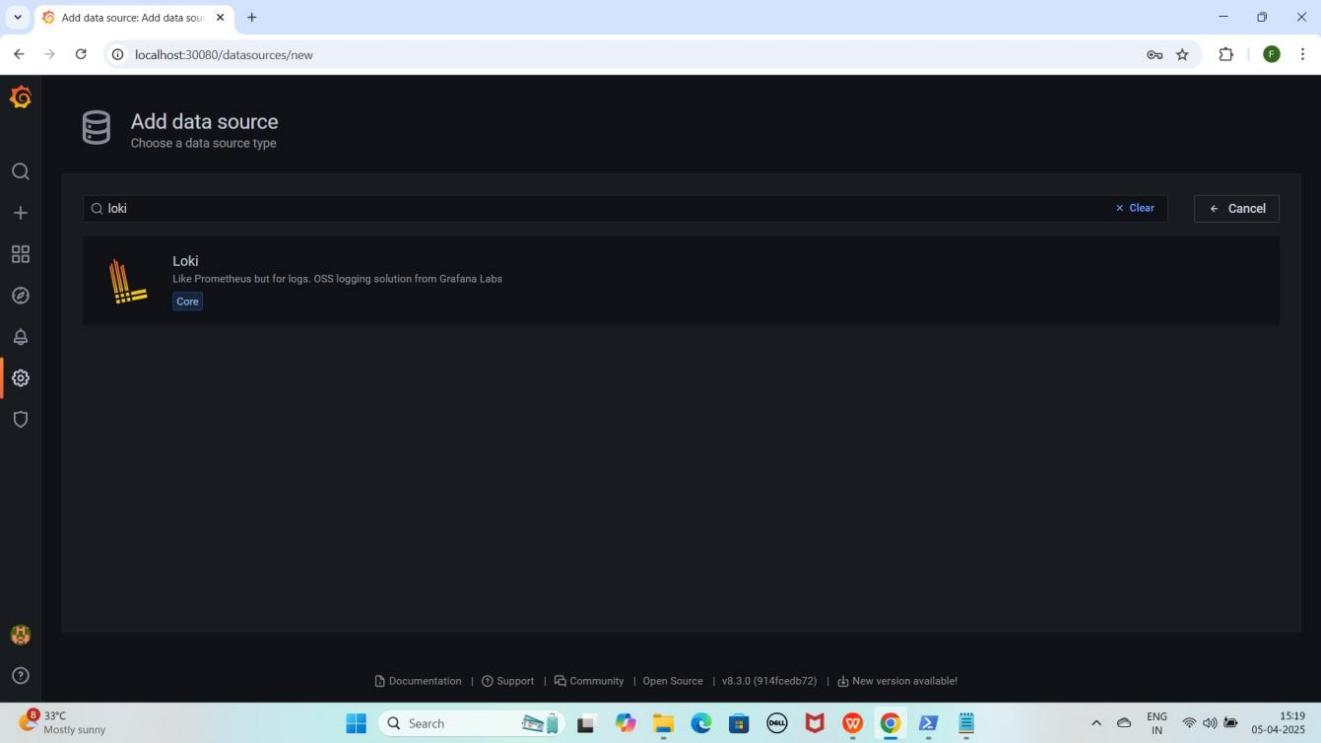
#### Aapke case me IP different ho sakti hai.

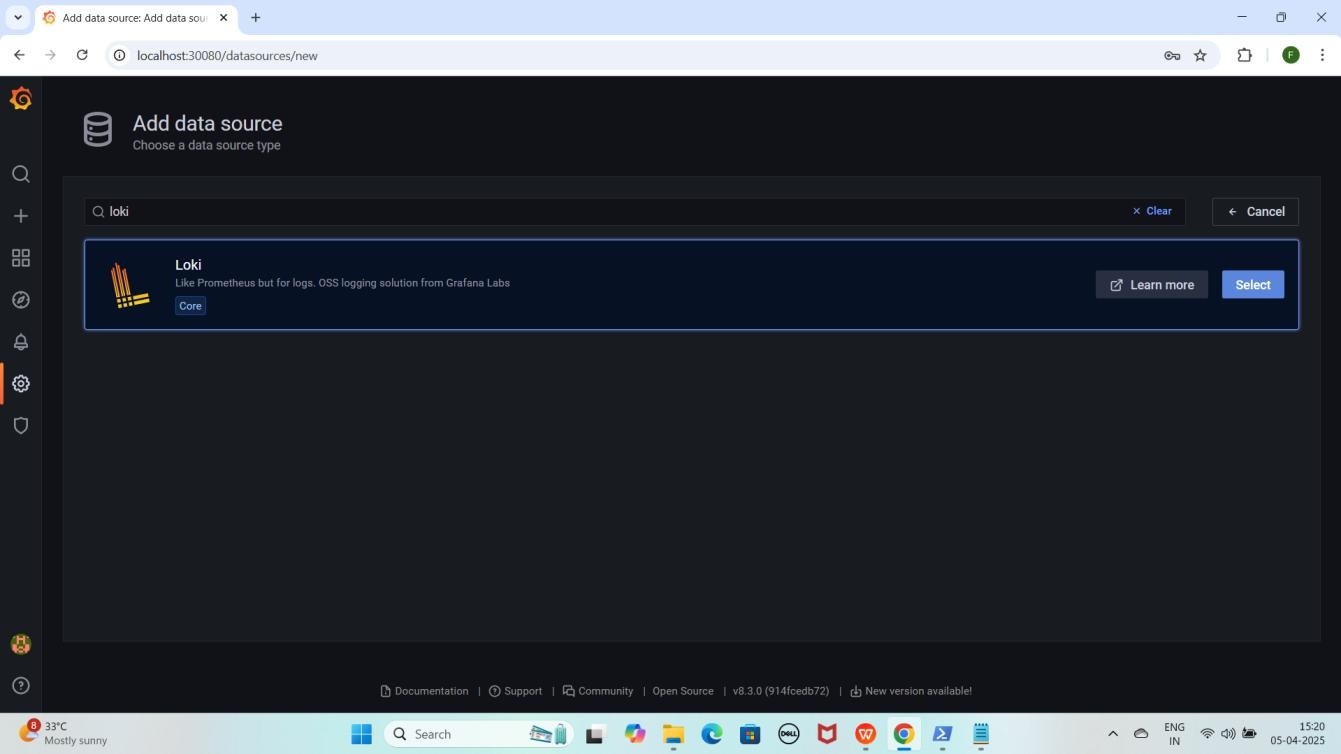
* + **HTTPS section** me URL box me paste karo.
  + Neeche scroll karo aur **"Save & Test"** pe click karo.

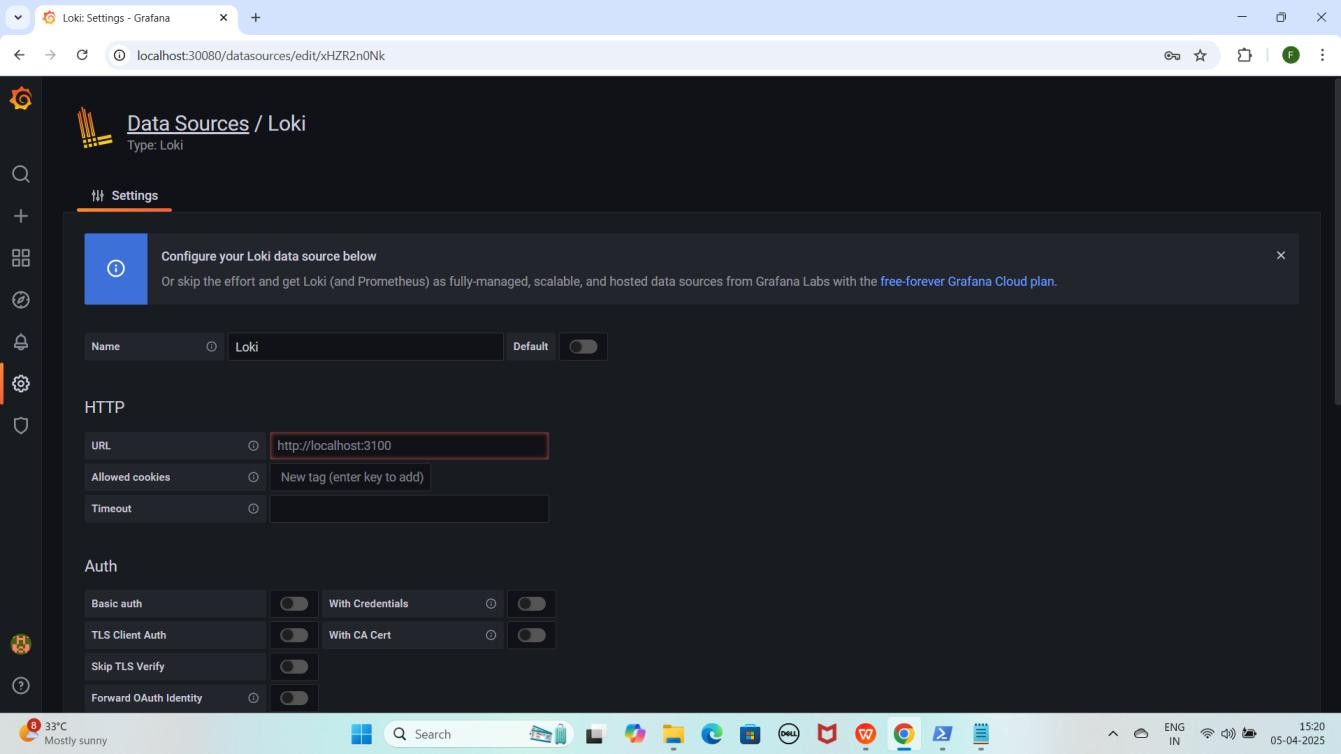
YE KUCH ISTARHA LAGEGA









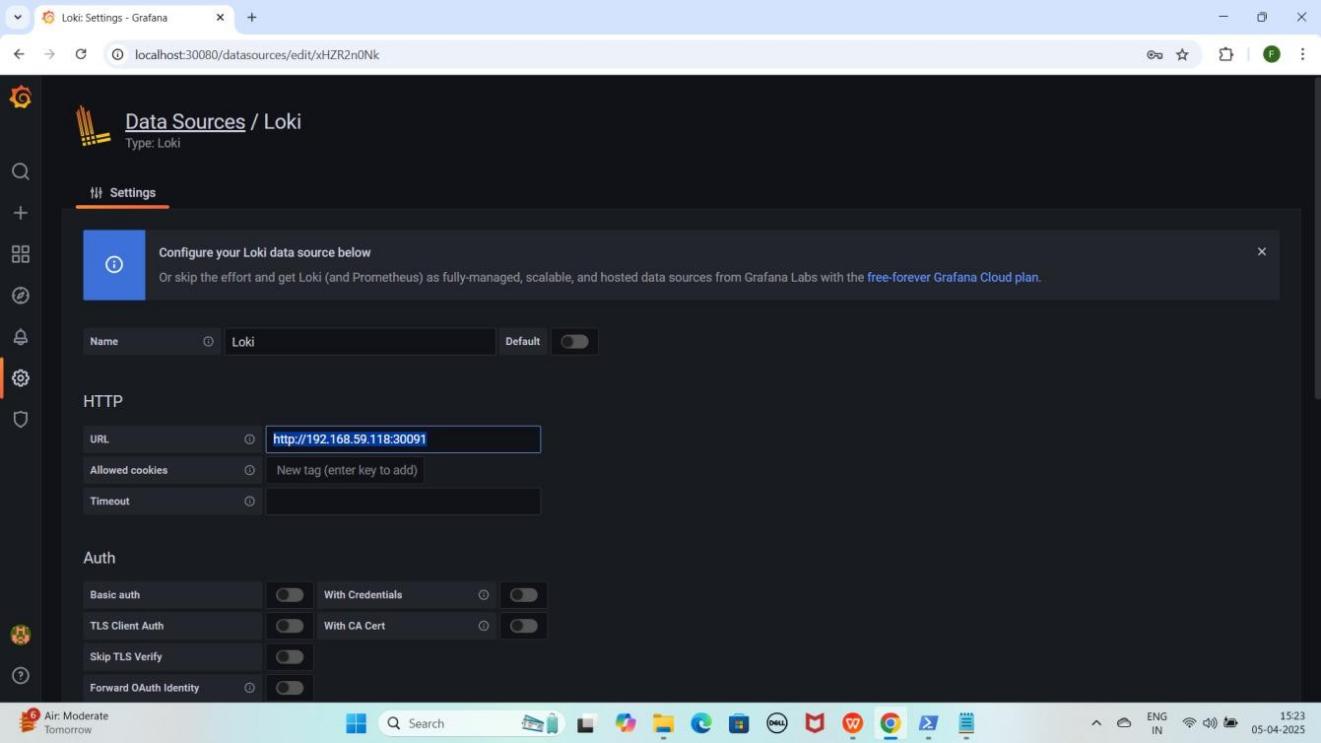
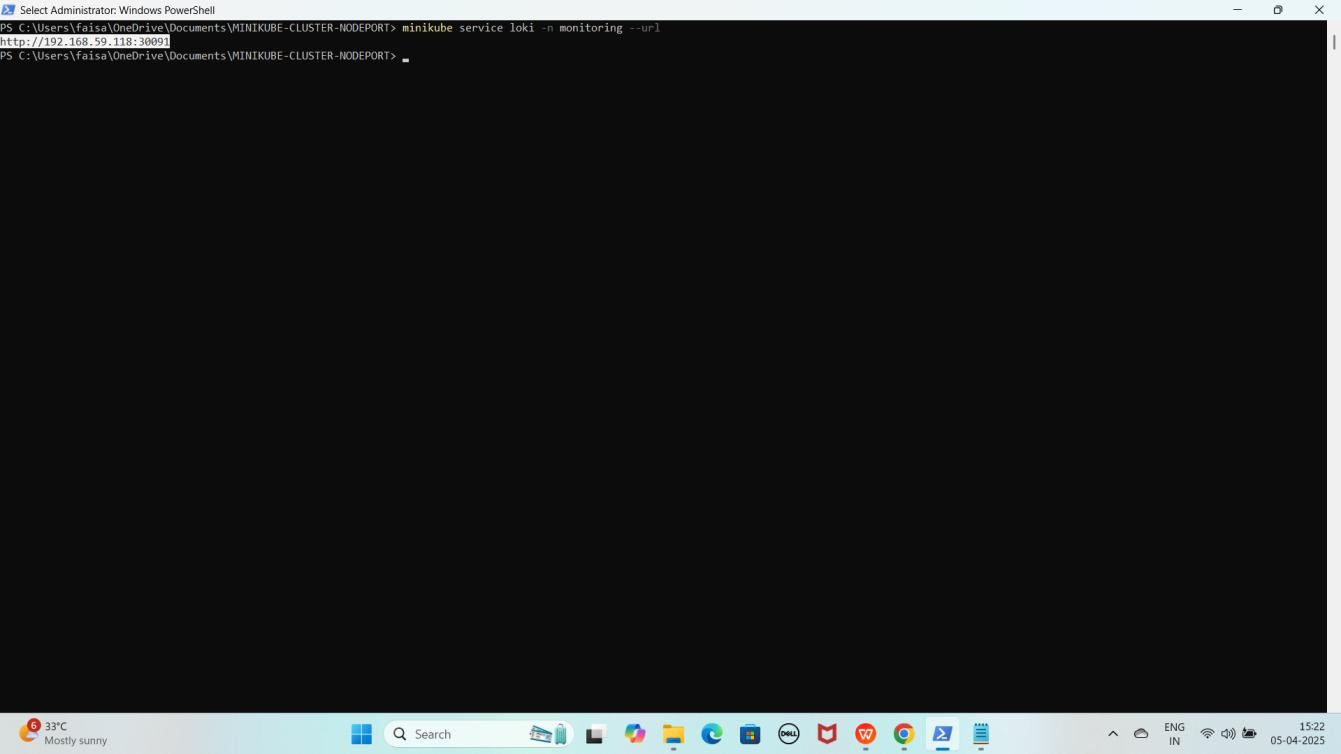


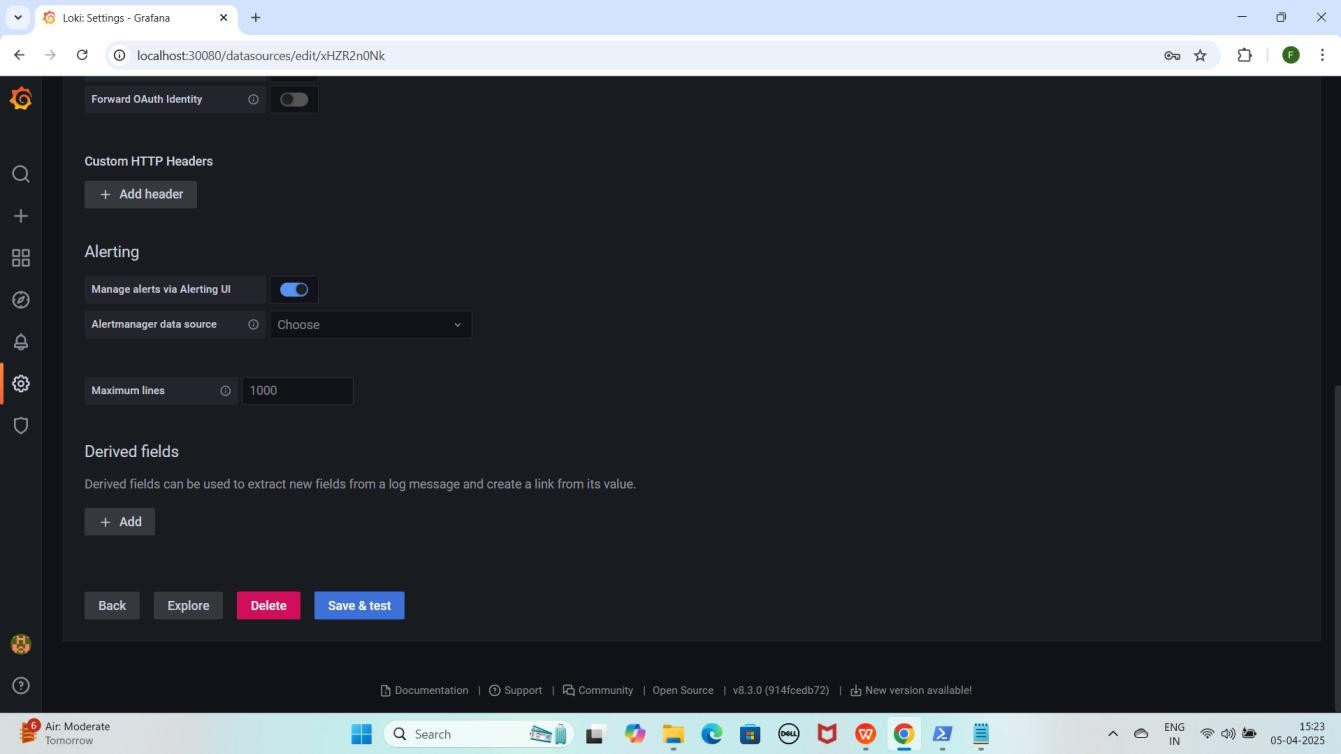
#### NOTE: Minikube VirtualBox me run ho raha ho, to Grafana me localhost kaam nahi karega. Uski jagah minikube ip use karenge kyunki localhost Grafana ke container ka hota hai, host machine ka nahi.

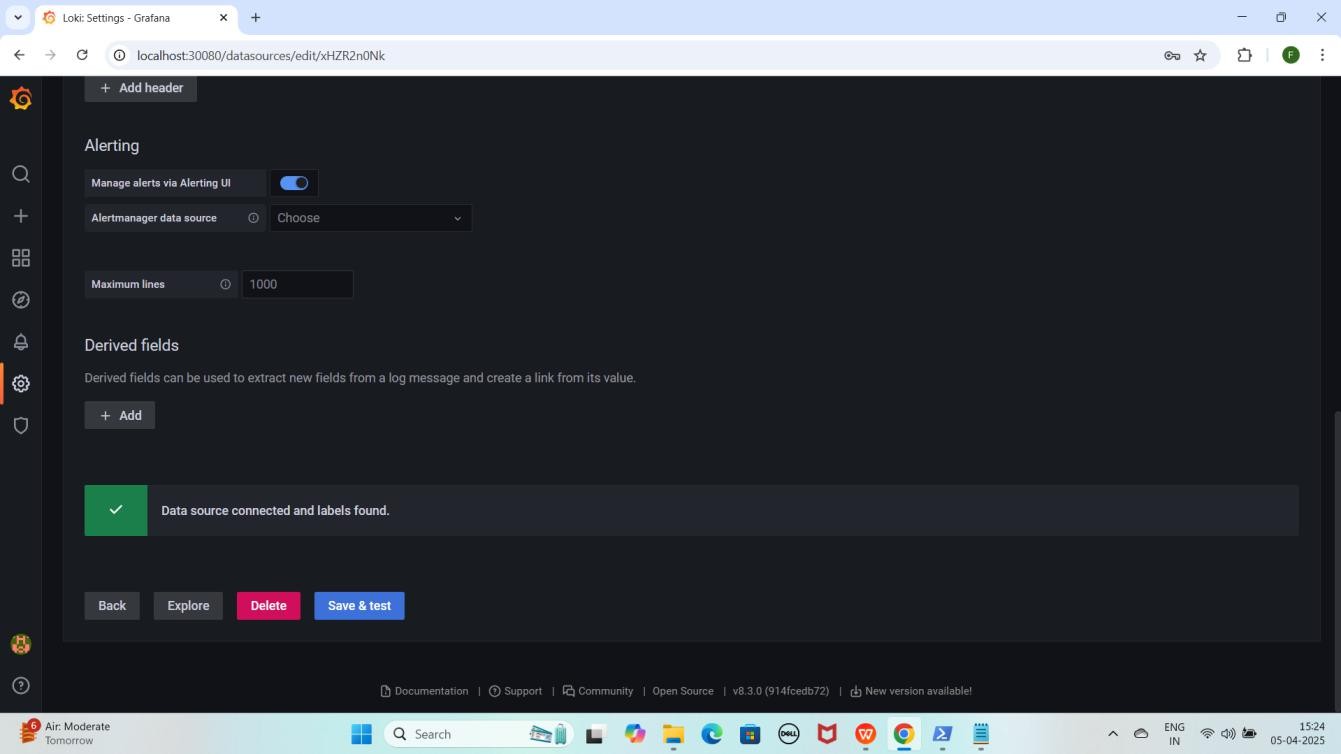
1. **Proper URLs ke liye, Loki datasource add karne ke liye yeh command run karein**

minikube service loki -n monitoring --url

YE KUCH ISTARHA LAGEGA



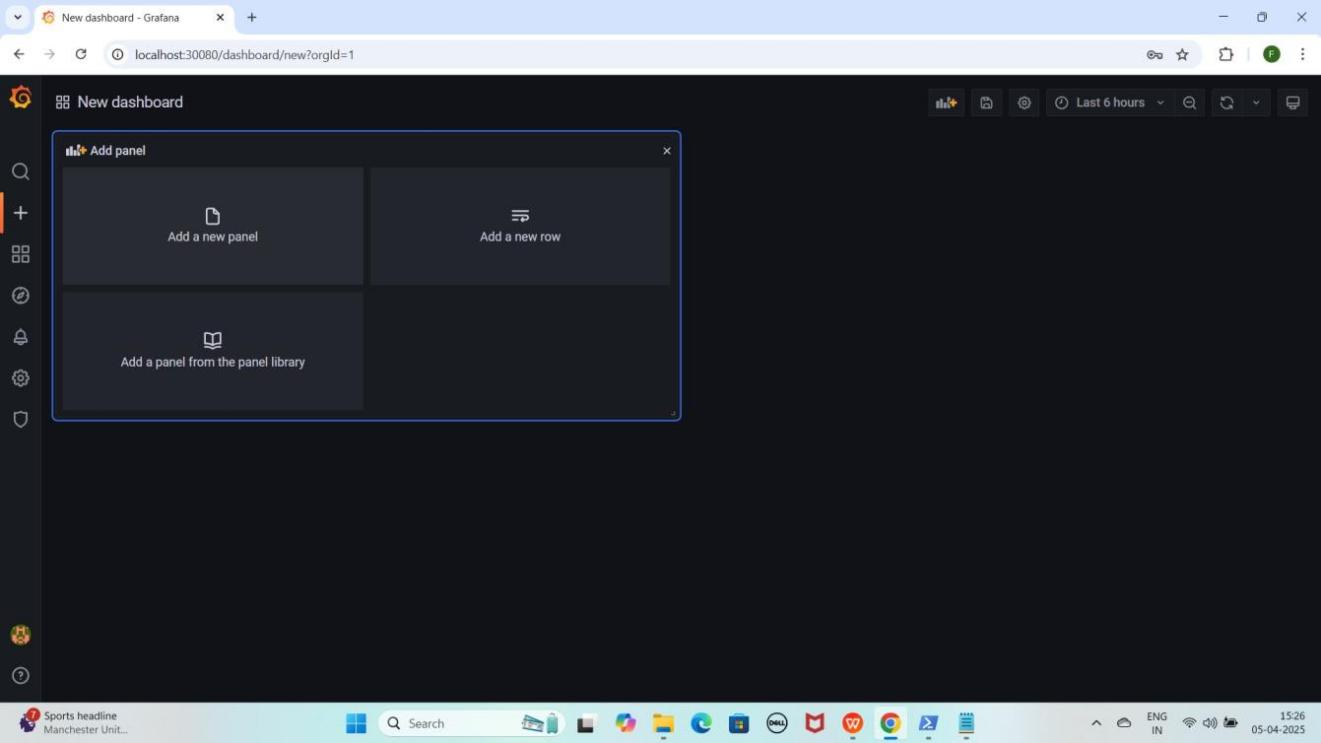
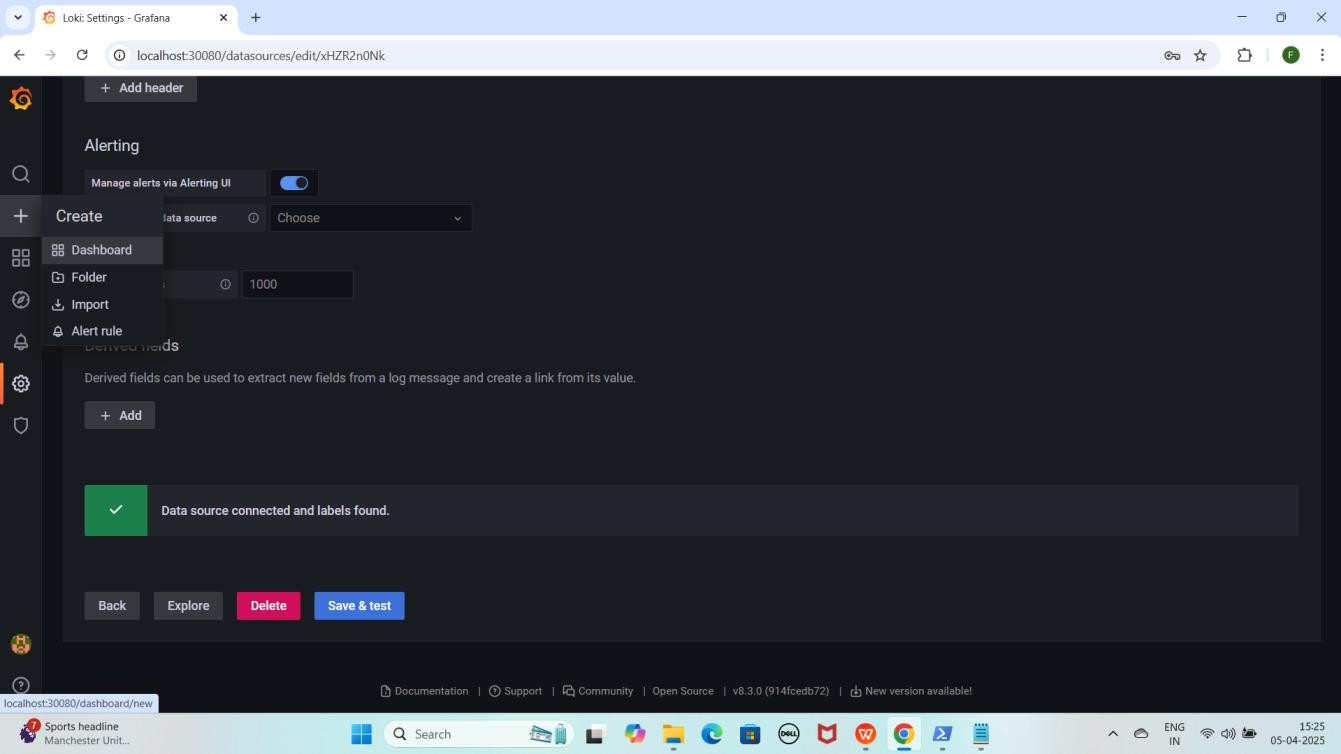


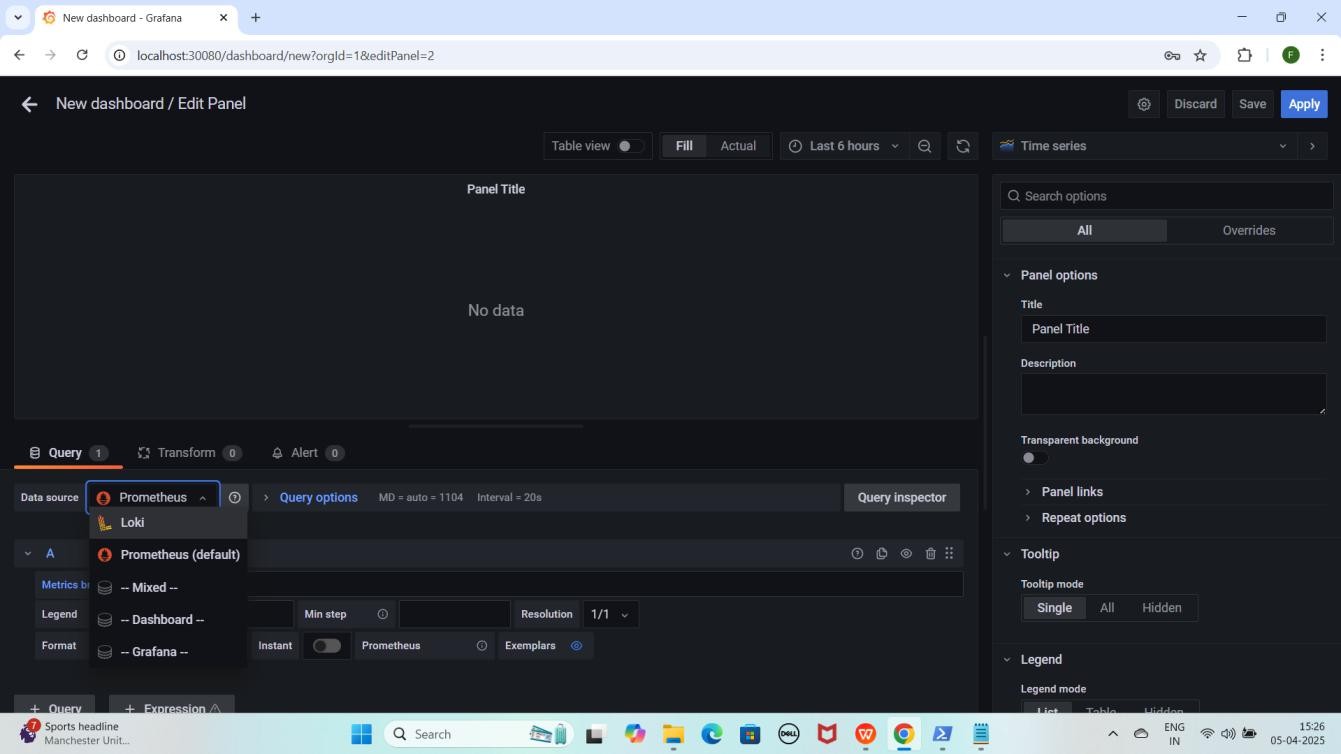


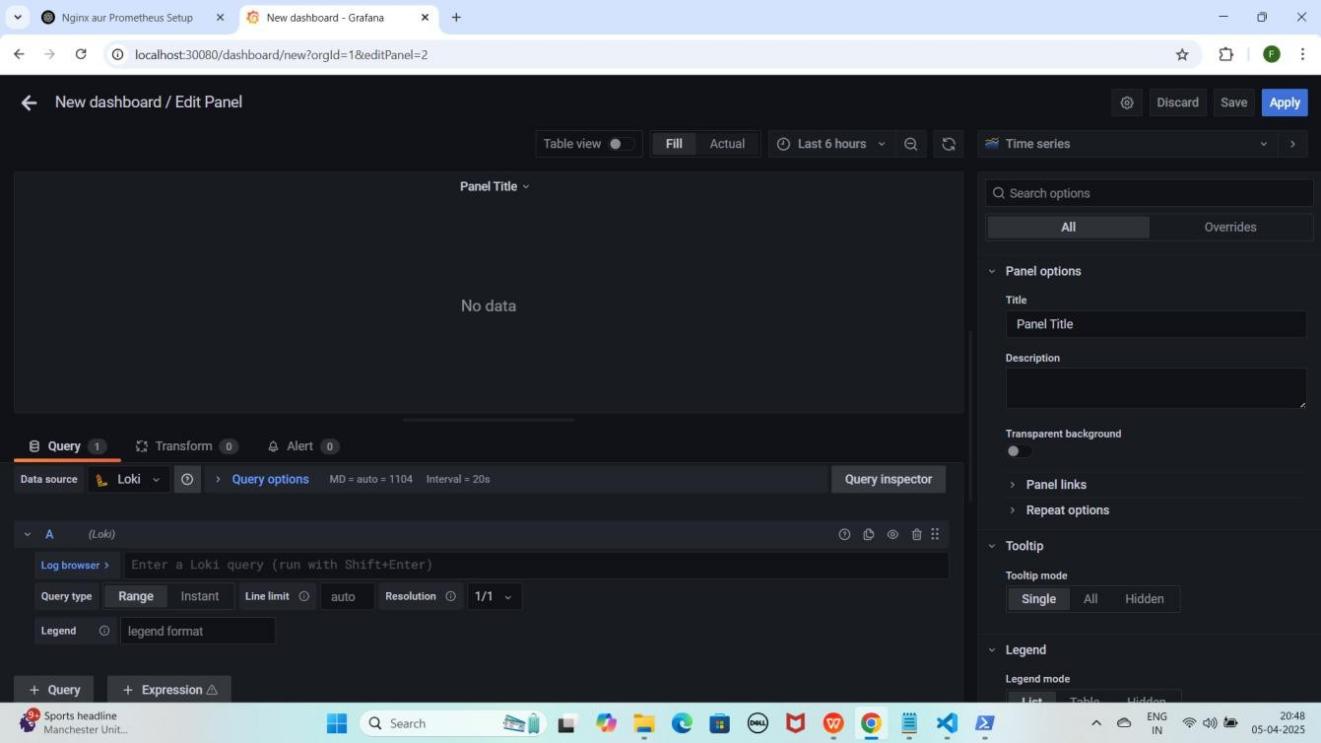
#### NOTE : Save & test pe click karne ke baad aapko Data source connected and label found karke pop up aayega

* + **Loki ke Loggs visualize** karne ke liye **"+" (plus) icon** pe click karo.
  + **Dashboard** select karo aur **"Add a new panel"** pe click karo.
  + **Query section** me **Data Source ko "Loki"** select karo.

YEKUCH ISTARHA LAGEGA

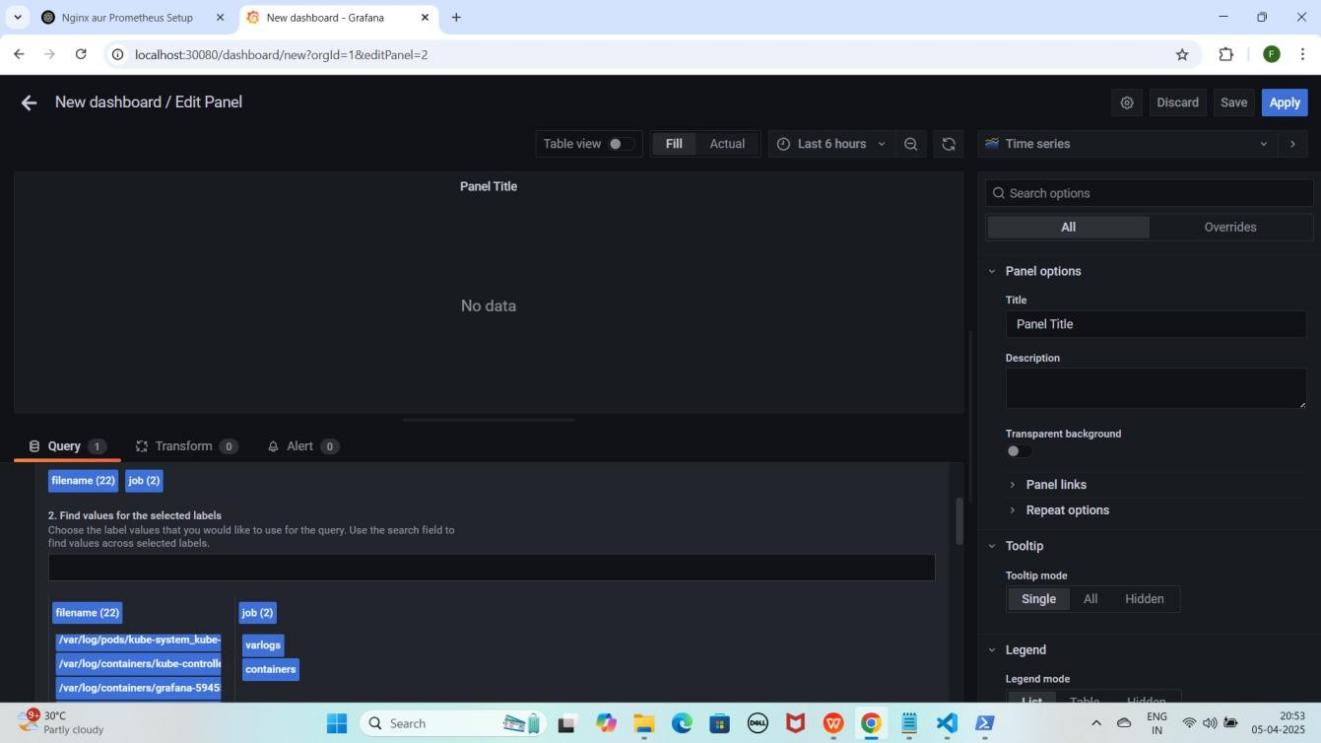
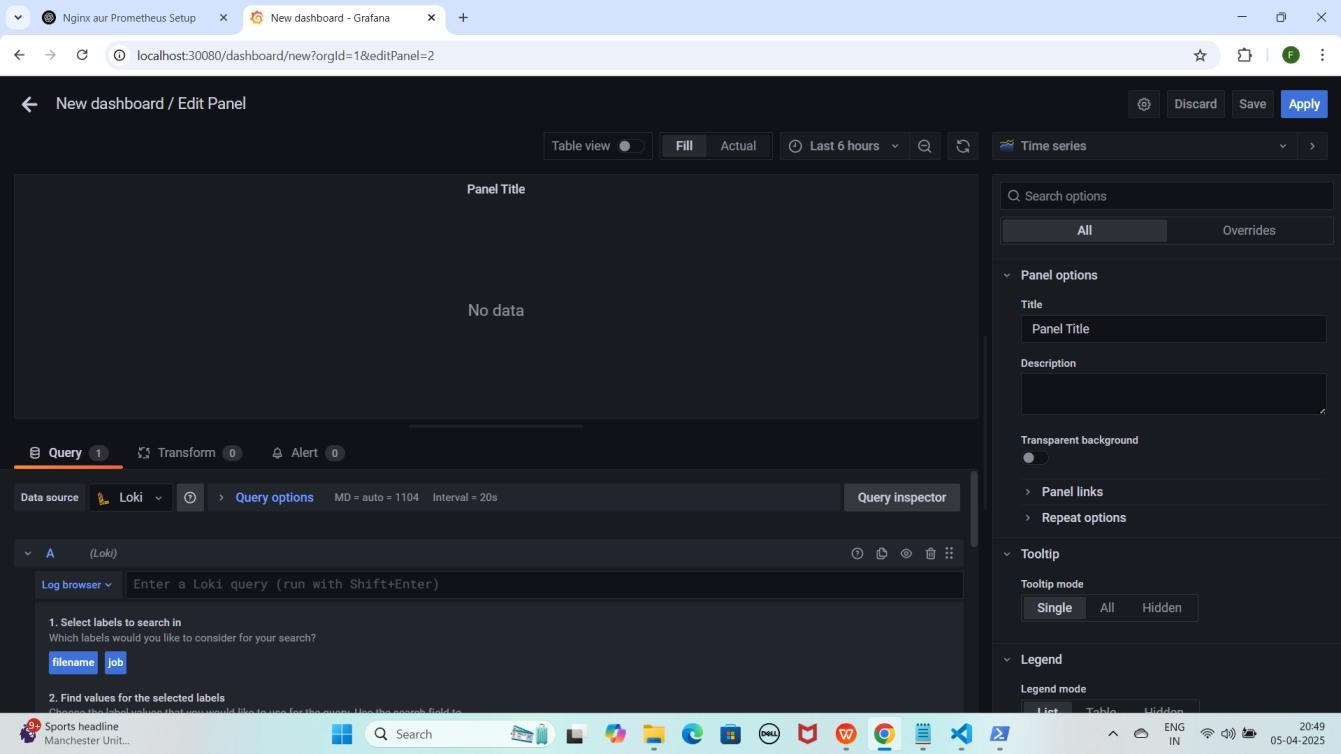


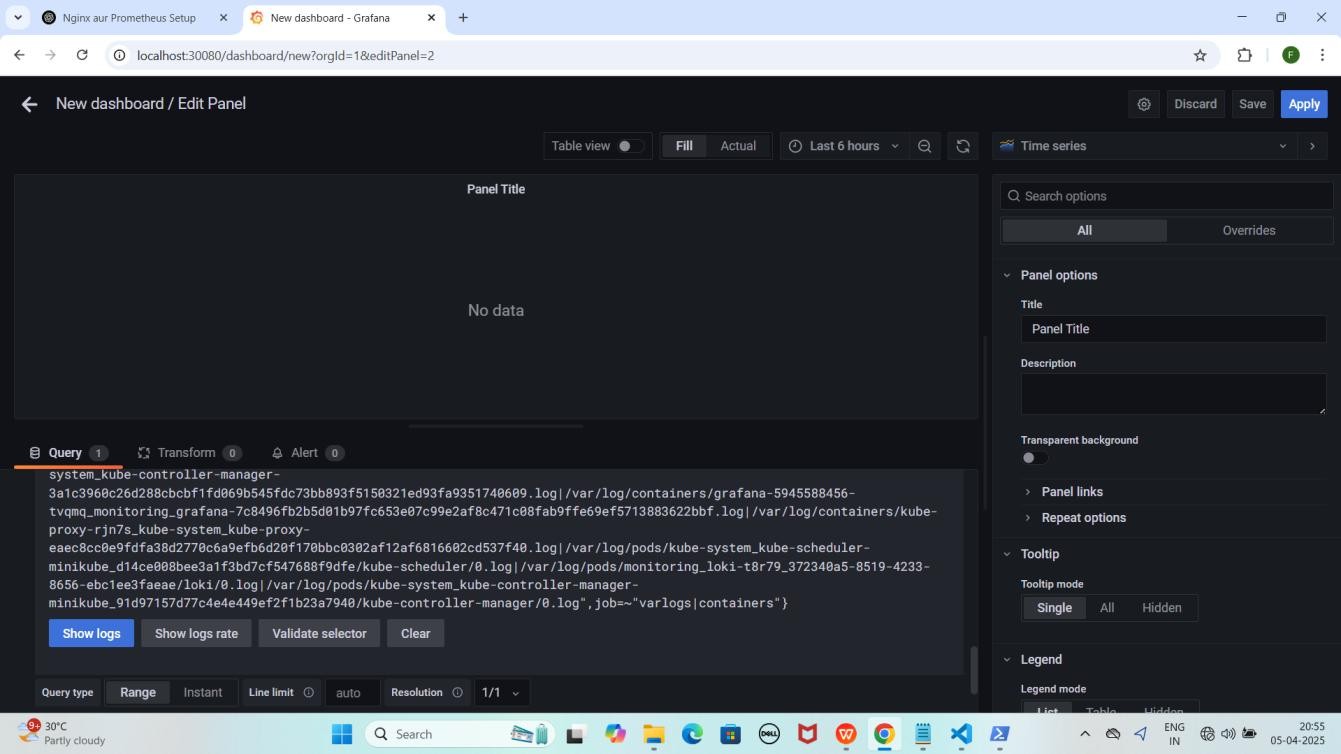


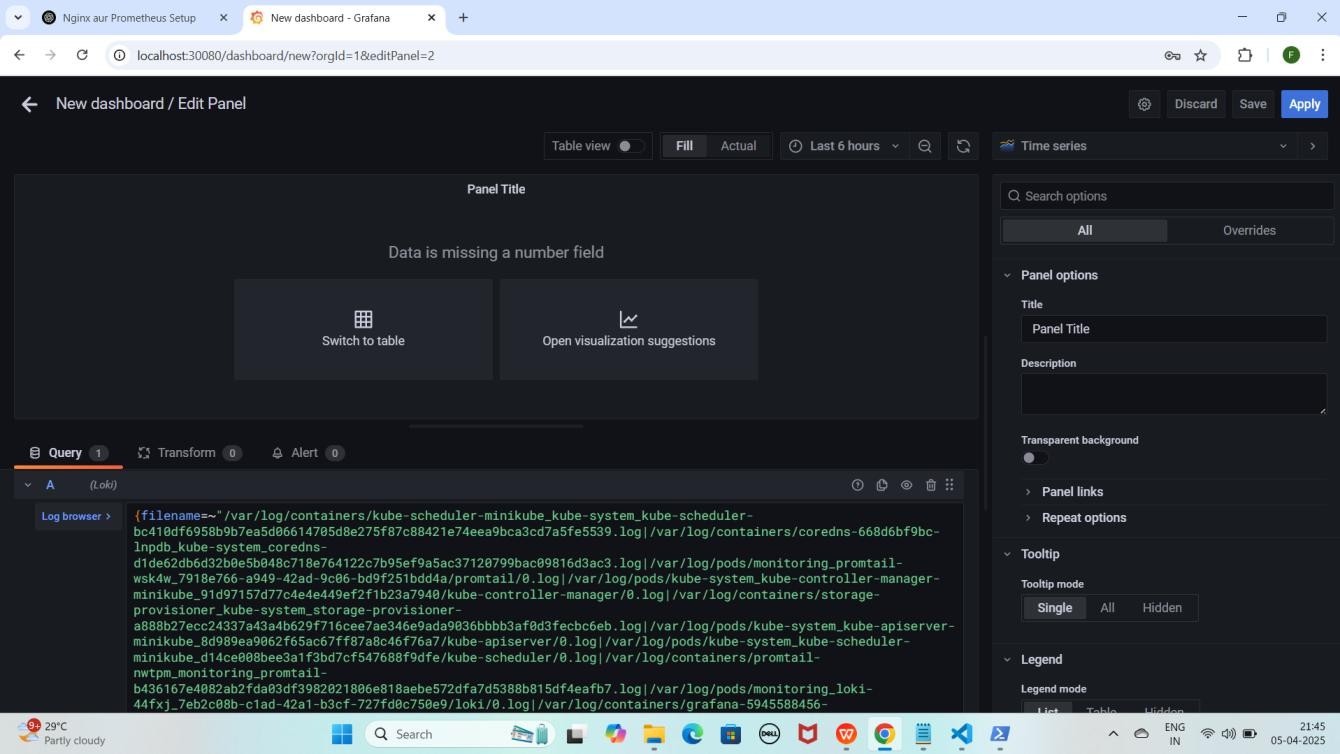


1. **"Log Browser"** pe click karo.
2. Pehle **"filename"** select karo or uske logs bhi, phir **"job"** (jaise ki **varlogs and containers**) select karo or uske logs bhi select karo.
3. Neeche **scroll** karo aur **"Show logs"** pe click karo.

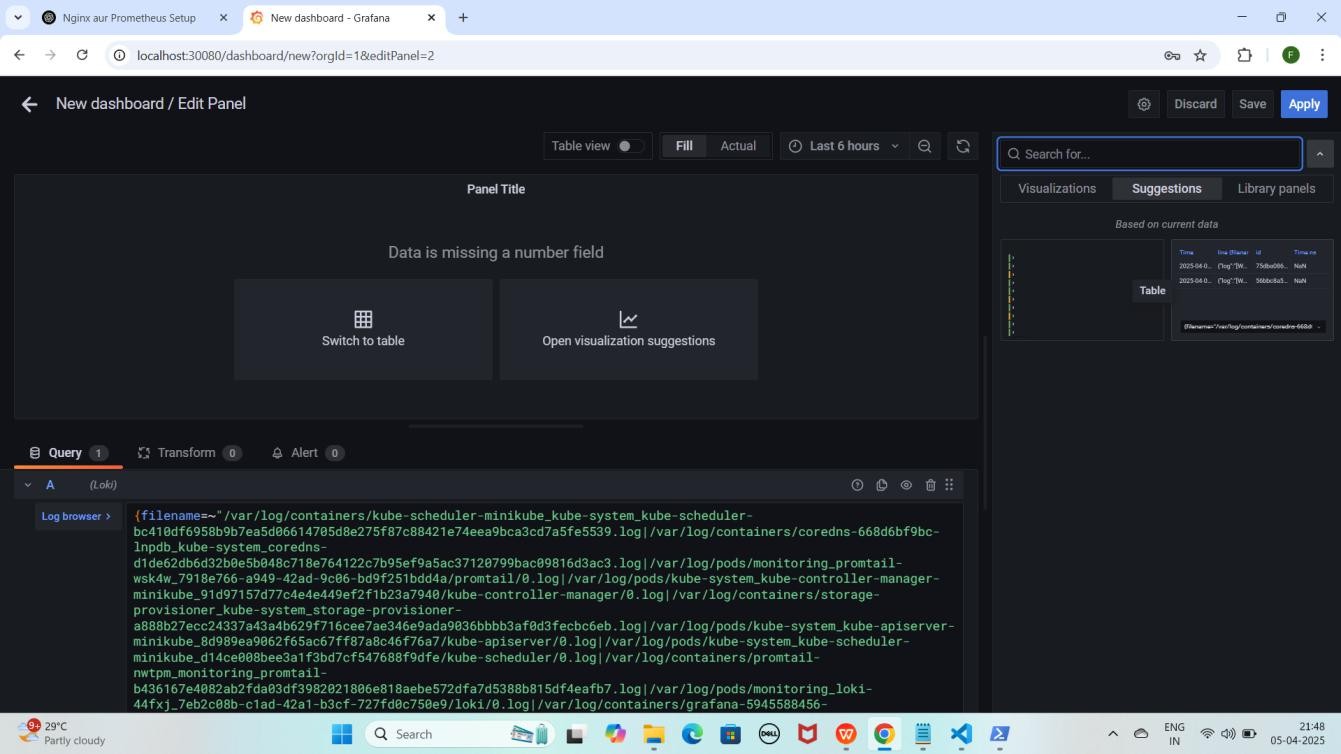
YEKUCH ISTARHA LAGEGA

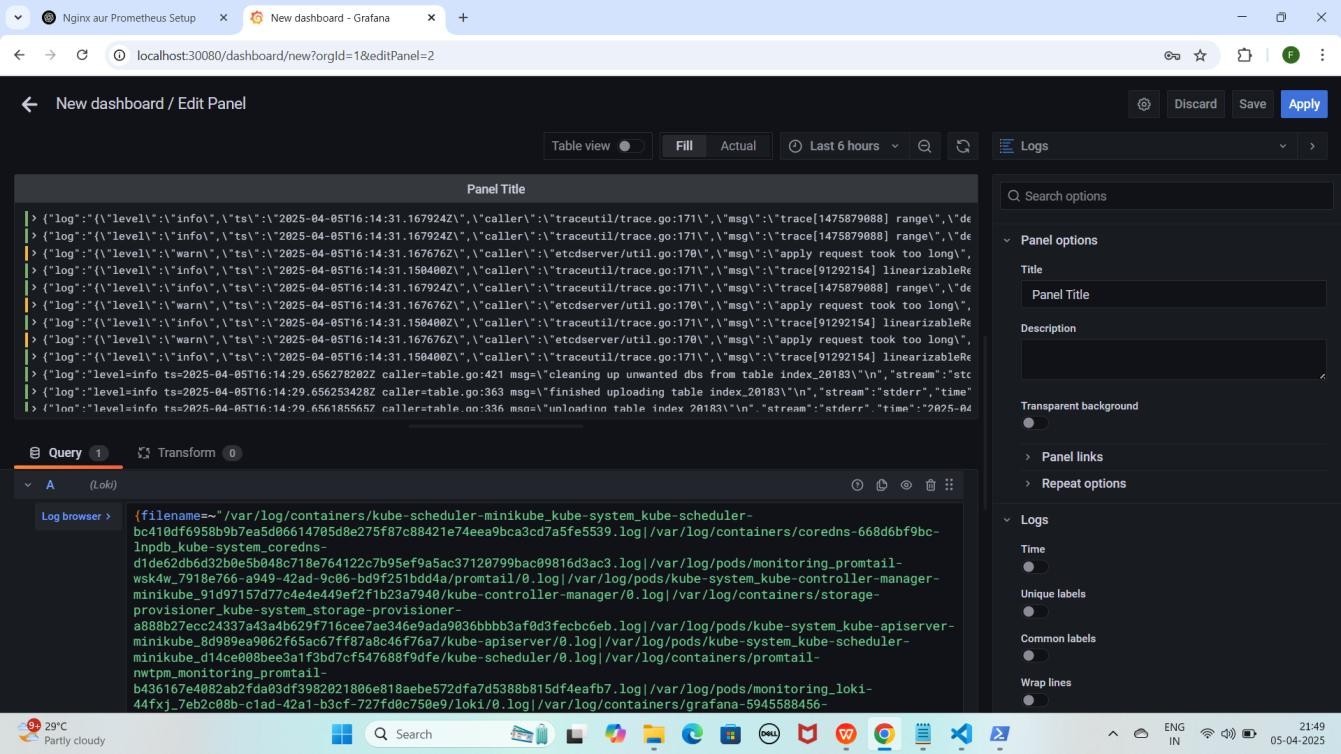






1. **Time Series** pe click karo
2. **Suggestions** me jao aur "Dashboard" select karo Jaise ki maine kiya hai YEKUCH ISTARHA LAGEGA





#### NOTE : Is setup se hum cluster kel saare Logs Monitor kar sakte hain.

**Recommend : Har Step aur Command ko meri Snapshots se Match karein taake Confirm ho sake ke sab kuch sahi tarah se Execute hua hai.**

**Project Video Demo : [https://youtu.be/KTkrwvuFGGU](g)**

# [\*](g)\*\*\*\*Commands Used in This Project\*\*\*\*\*

## PART 1: Minikube Cluster Creation and Setup

#### Start Minikube Cluster using VirtualBox driver with 4GB RAM and 2 CPUs

minikube start --driver=virtualbox --memory=4000 --cpus=2 --force

#### Check Minikube Status

minikube status

#### Verify Node is Ready

kubectl get nodes

## Part 2: Nginx pods and Service Deployment

#### Deploy NGINX with Replicas

kubectl apply -f nginx-deployment.yaml

#### Expose NGINX using NodePort Service

kubectl apply -f nginx-nodeport-service.yaml

Start-Process -FilePath "kubectl" -ArgumentList "port-forward service/nginx- service 30007:80" -WindowStyle Hidden

#### Verify Deployments & Services

kubectl get pods kubectl get services

## Part 3: MySQL Secrets and ConfigMap Configuration

#### Generate Base64 Encodings For Secrets

echo -n Faisal Khan | base64 echo -n Faisalkhan35@ | base64

echo -n 'RmFpc2FsIEtoYW4=' | base64 --decode

echo -n 'RmFpc2Fsa2hhbjM1QA==' | base64 --decode

#### Apply Secrets and ConfigMaps

kubectl apply -f mysql-secrets.yaml kubectl apply -f mysql-configmap.yaml

#### Apply RBAC (Role-Based Access Control)

kubectl apply -f service-account.yaml kubectl apply -f role.yaml

kubectl apply -f rolebinding.yaml

#### Verify Pods

kubectl get pods

## Part 4: MySQL Database Pods and Services Deployment

#### Deploy MySQL Services & Persistent Volumes

kubectl apply -f mysql-headless-service.yaml kubectl apply -f mysql-pv.yaml

kubectl apply -f mysql-pv-2.yaml kubectl apply -f mysql-pv-3.yaml

#### Deploy MySQL StatefulSet

kubectl apply -f mysql-statefulset.yaml

#### Verify Deployments & Services

kubectl get pods kubectl get services

## Part 5: Accessing MySQL Database in Minikube Cluster

#### Access MySQL Pod

kubectl exec -it mysql-0 -- /bin/bash

#### Login to MySQL

mysql -u root -p

#### Create & Verify Database

CREATE DATABASE faisal\_db; SHOW DATABASES;

## Part 6: Monitoring Using Prometheus and Loki With Grafana

#### Create Monitoring Namespace

kubectl create namespace monitoring

#### Deploy Prometheus Components

kubectl apply -f prometheus-daemonset.yaml kubectl apply -f prometheus-rbac.yaml

kubectl apply -f prometheus-nodeport-service.yaml

Start-Process -FilePath "kubectl" -ArgumentList "port-forward service/prometheus-server 30090:80 -n monitoring" -WindowStyle Hidden

#### Verify Prometheus Pods & Services

kubectl get pods -n monitoring kubectl get services -n monitoring

#### Important Prometheus Metrics

rate(process\_cpu\_seconds\_total[30s]) process\_resident\_memory\_bytes

#### Deploy Loki & Promtail

kubectl apply -f promtail-daemonset.yaml kubectl apply -f loki-daemonset.yaml kubectl apply -f loki-nodeport-service.yaml

Start-Process -FilePath "kubectl" -ArgumentList "port-forward service/loki 30091:3100 -n monitoring" -WindowStyle Hidden

#### Verify Loki & Promtail Pods & Services

kubectl get pods -n monitoring kubectl get services -n monitoring

#### Deploy Grafana

kubectl apply -f grafana-deployment.yaml

Start-Process -FilePath "kubectl" -ArgumentList "port-forward service/grafana 30080:80 -n monitoring" -WindowStyle Hidden

#### Verify Grafana Pods & Services

kubectl get pods -n monitoring kubectl get services -n monitoring