LAB ACTION PLAN FOR WEEK 10

Working with minikube and Nagios

1. Hands-on practice of creating, running and scaling pods in minikube.
2. Running Nginx server on specified port number by explaining the Nginx monitoring tool
3. Running Nagios server and Understanding the Monitoring tool using Docker.
4. AWS-free Trier account Creation steps
5. Upload the screenshots for the tasks

Kubernetes:

Kubernetes is a tool that automates how we run and manage applications inside the container.

Dockers will only run containers, if in any case the container fails/stopped/killed, the docker will not help us, here is where Kubernetes plays an important role, Kubernetes cluster will be responsible in creating a new container and managing various containers.

POD: In Kubernetes, a Pod is the smallest deployable unit that you can create and manage.

Minikube:

Minikube creates a VM on your local machine and deploys a simple Kubernetes cluster with one node. It's a lightweight implementation.Minikube *is* a version of Kubernetes.

Nagios:

Nagios is an **open-source IT infrastructure monitoring tool**. It monitors

- Serrvers
- Network devices
- Applications and services

It **alerts admnistrators** when issues occur and notifies when they are resolved.

Step 1: Install Prerequisites

Before installing Minikube, ensure the following are installed:

1. **Virtualization Support**:
   - Verify virtualization is enabled:

2. **Hypervisor**:
   - Minikube supports multiple hypervisors (e.g., **Hyper-V**, **VirtualBox**, or **Docker** as a driver).
   - Install one of the following:
     - **Hyper-V** (pre-installed on Windows 10/11 Pro or Enterprise).
     - **Docker Desktop** (if you want to use Docker as the driver).

Step 2: Download Minikube

1. Open a PowerShell or Command Prompt with administrator privileges.
2. Download the latest Minikube executable using this command:

3.
   ```
   curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-installer.exe
   ```
4. Install Minikube by running the installer:

5.
   ```
   .\minikube-installer.exe
   ```

Step 3: Add Minikube to PATH

If Minikube is not automatically added to your PATH during installation:

1. Open **System Properties → Environment Variables**.
2. Add the directory where Minikube is installed (e.g., `C:\Program Files\Minikube`) to your PATH variable.

Step 4: Start Minikube

1. Open a terminal (PowerShell or CMD). Do the following commands
2. Start Minikube with a specified driver (e.g., Hyper-V, Docker, or VirtualBox). For example:
   - **Hyper-V**:

   -
     ```
     minikube start --driver=hyperv
     ```
   - **Docker**:

   -
     ```
     minikube start --driver=docker
     ```
3. Verify Minikube is running:

4.
   ```
   minikube status
   ```

Step 5: Interact with Minikube

kubectl is a command-line tool used in Kubernetes to interact with and manage Kubernetes clusters.

Once Minikube is running:

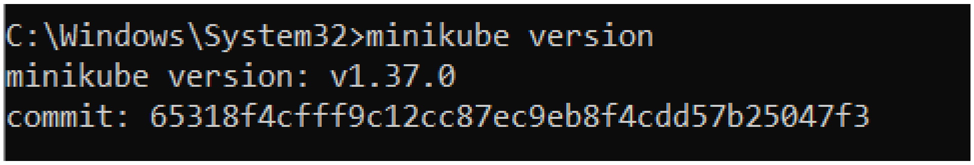
1. Use `kubectl` to interact with the cluster.
   - Install `kubectl` if not already installed:
   - `minikube kubectl -- get pods -A`
   - Or download it separately from the official Kubernetes site.
2. Open the Minikube dashboard (optional):
3. `minikube dashboard`

Optional: Check Your Installation

Run the following to verify the installation:

Minikube version

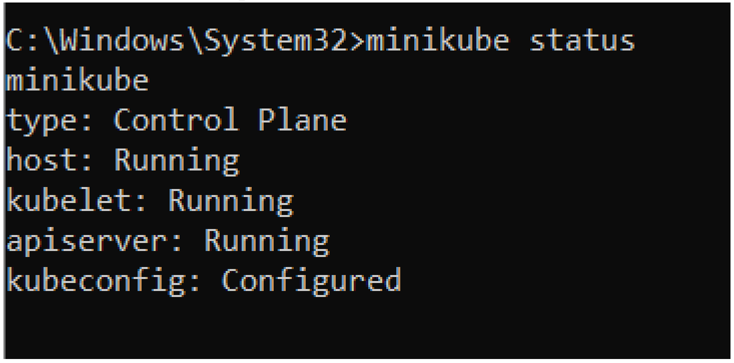
```
kubectl version -client
```



```
C:\Windows\System32>minikube version
minikube version: v1.37.0
commit: 65318f4cfff9c12cc87ec9eb8f4cdd57b25047f3
```

Troubleshooting

1. **If Minikube fails to start**:
   - Ensure your hypervisor (Hyper-V/Docker/VirtualBox) is installed and running.
   - Check the Minikube logs:
   - `minikube logs`
2. **Updating Minikube**:
3. `minikube update-check`
4. `minikube update`



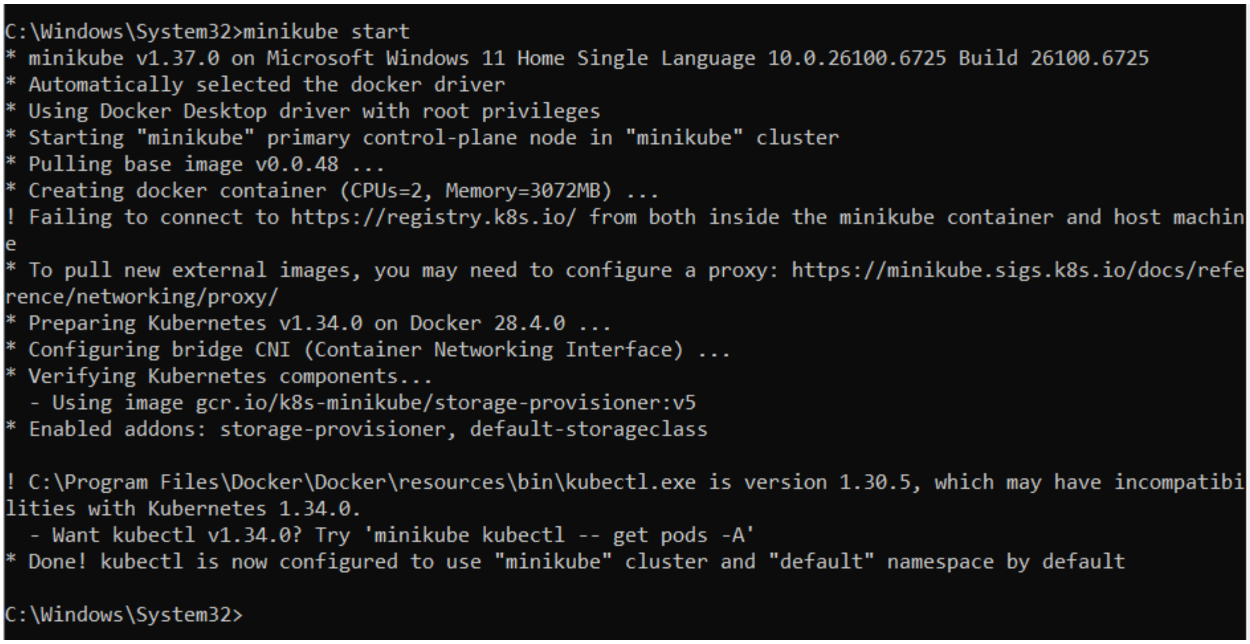
```
C:\Windows\System32>minikube status
minikube
type: Control Plane
host: Running
kubelet: Running
apiserver: Running
kubeconfig: Configured
```

Minikube Automation Steps

Step 1: Start Minikube Cluster

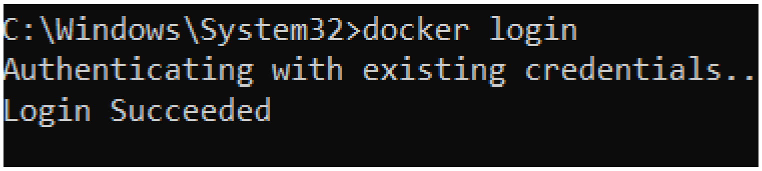
- Open your terminal and run the command:

```
minikube start
```



```
C:\Windows\System32>minikube start
* minikube v1.37.0 on Microsoft Windows 11 Home Single Language 10.0.26100.6725 Build 26100.6725
* Automatically selected the docker driver
* Using Docker Desktop driver with root privileges
* Starting "minikube" primary control-plane node in "minikube" cluster
* Pulling base image v0.0.48 ...
* Creating docker container (CPUs=2, Memory=3072MB) ...
! Failing to connect to https://registry.k8s.io/ from both inside the minikube container and host machin
e
* To pull new external images, you may need to configure a proxy: https://minikube.sigs.k8s.io/docs/refe
rence/networking/proxy/
* Preparing Kubernetes v1.34.0 on Docker 28.4.0 ...
* Configuring bridge CNI (Container Networking Interface) ...
* Verifying Kubernetes components...
  - Using image gcr.io/k8s-minikube/storage-provisioner:v5
* Enabled addons: storage-provisioner, default-storageclass

! C:\Program Files\Docker\Docker\resources\bin\kubectl.exe is version 1.30.5, which may have incompatibi
lities with Kubernetes 1.34.0.
  - Want kubectl v1.34.0? Try 'minikube kubectl -- get pods -A'
* Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default

C:\Windows\System32>
```

```
Docker ps
```



```
C:\Windows\System32>docker login
Authenticating with existing credentials..
Login Succeeded
```

Step 2: Create and Manage Deployment

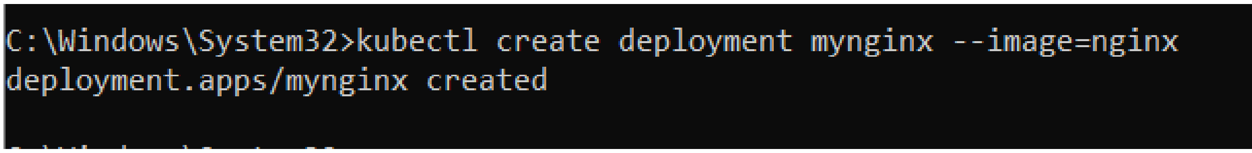
1. **Create an application in Kubernetes:**

   - Command:

     ```
     kubectl create deployment mynginx --image=nginx
     ```

     ```
     if already created then
     ```

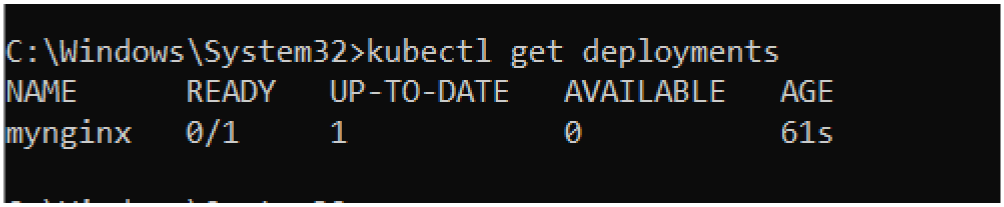
```
kubectl set image deployment/myngnix nginx=nginx:latest
```



```
C:\Windows\System32>kubectl create deployment mynginx --image=nginx
deployment.apps/mynginx created
```

- Verify the deployment using: Kubernetes responds by showing you a list that includes the names of your deployment groups

```
kubectl get deployments
```

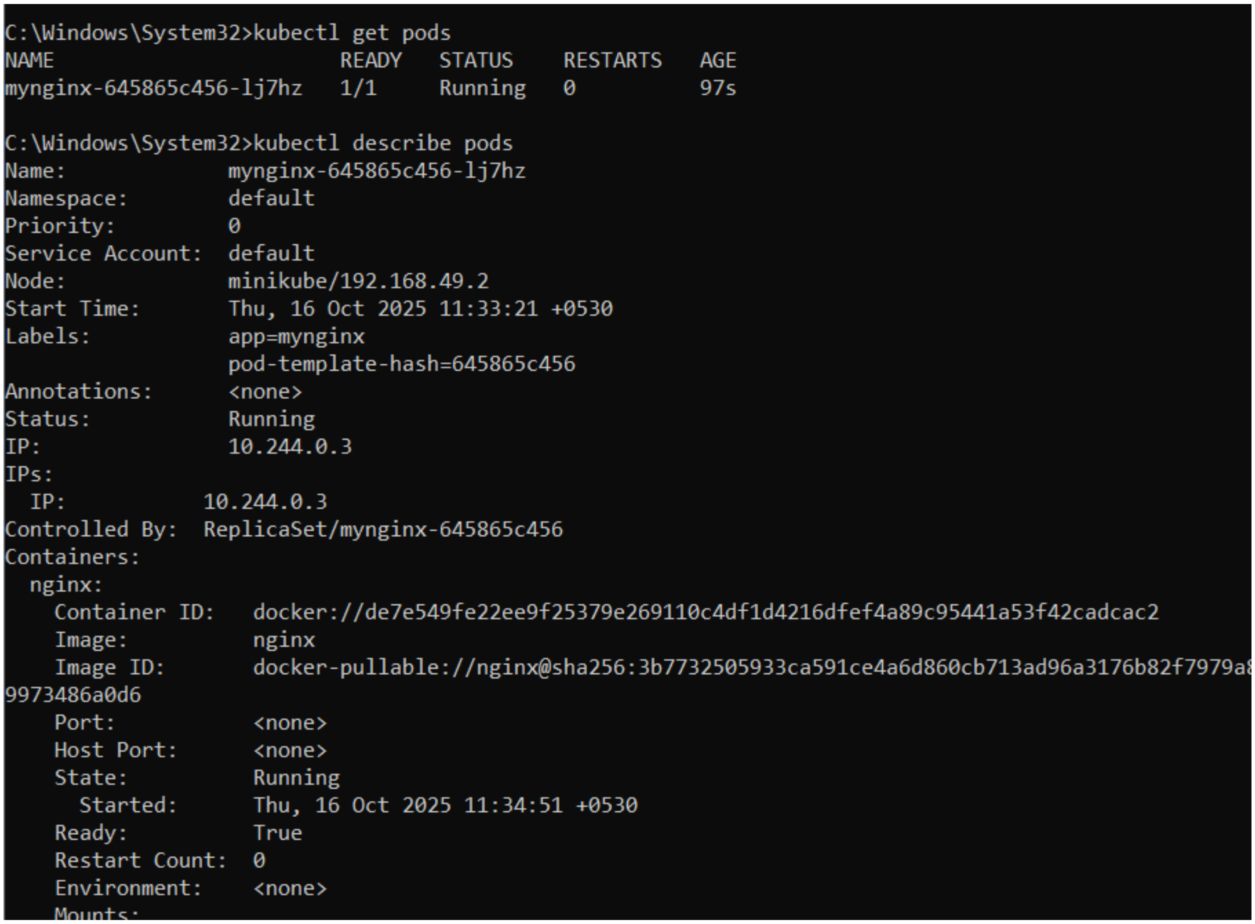


```
C:\Windows\System32>kubectl get deployments
NAME      READY   UP-TO-DATE   AVAILABLE   AGE
mynginx   0/1     1            0           61s
```

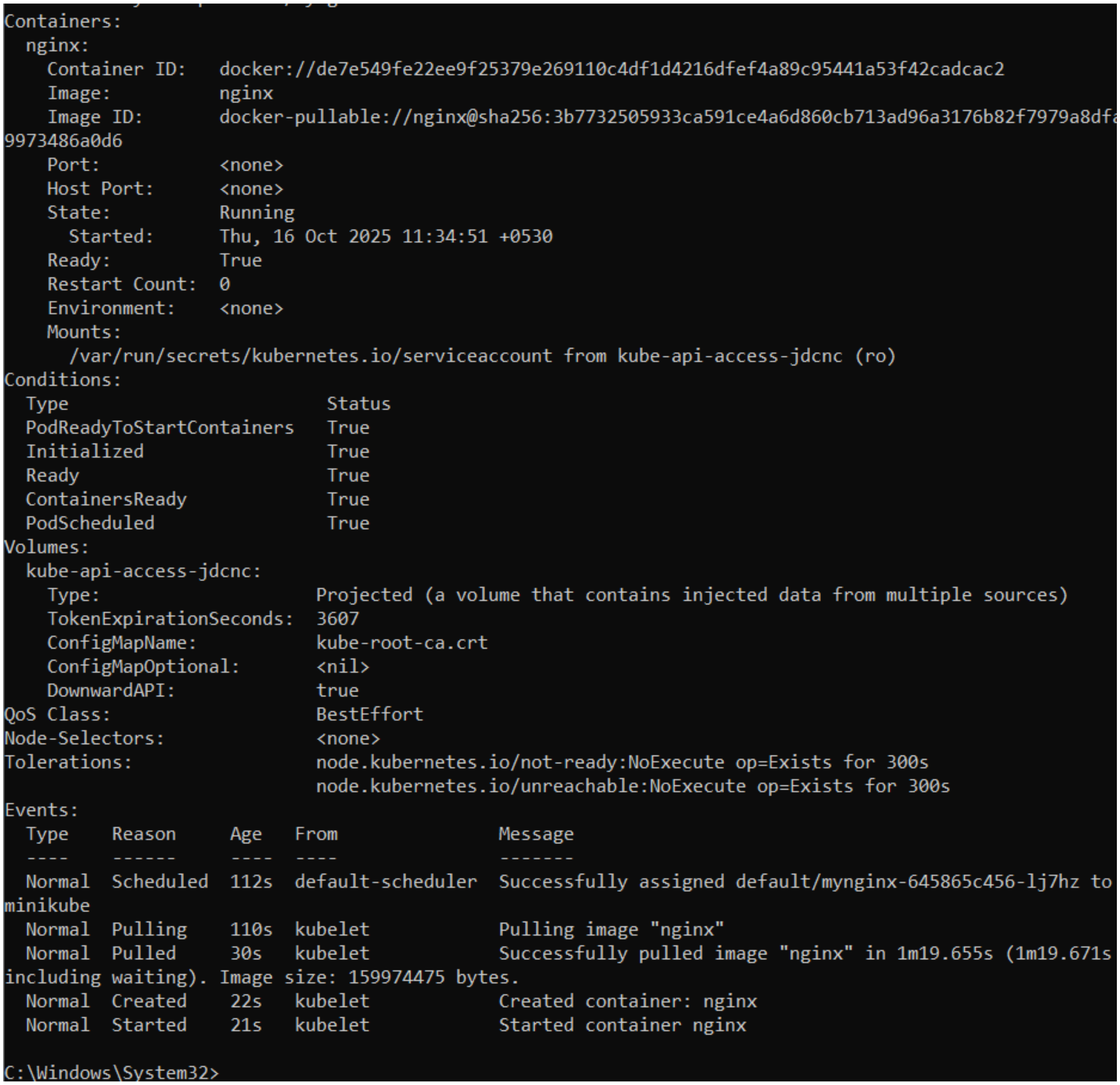
  - Ensure `mynginx` appears in the output.

Check the following commands:

- kubectl get pods
- kubectl describe pods



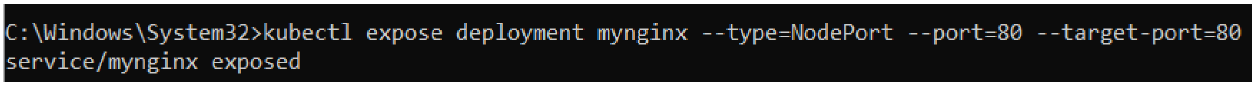
```
C:\Windows\System32>kubectl get pods
NAME                       READY   STATUS    RESTARTS   AGE
mynginx-645865c456-lj7hz   1/1     Running   0          97s

C:\Windows\System32>kubectl describe pods
Name:             mynginx-645865c456-lj7hz
Namespace:        default
Priority:         0
Service Account:  default
Node:             minikube/192.168.49.2
Start Time:       Thu, 16 Oct 2025 11:33:21 +0530
Labels:           app=mynginx
                  pod-template-hash=645865c456
Annotations:      <none>
Status:           Running
IP:               10.244.0.3
IPs:
  IP:           10.244.0.3
Controlled By:  ReplicaSet/mynginx-645865c456
Containers:
  nginx:
    Container ID:   docker://de7e549fe22ee9f25379e269110c4df1d4216dfef4a89c95441a53f42cadcac2
    Image:          nginx
    Image ID:       docker-pullable://nginx@sha256:3b7732505933ca591ce4a6d860cb713ad96a3176b82f7979a8
9973486a0d6
    Port:           <none>
    Host Port:      <none>
    State:          Running
      Started:      Thu, 16 Oct 2025 11:34:51 +0530
    Ready:          True
    Restart Count:  0
    Environment:    <none>
    Mounts:
```



```
Containers:
  nginx:
    Container ID:   docker://de7e549fe22ee9f25379e269110c4df1d4216dfef4a89c95441a53f42cadcac2
    Image:          nginx
    Image ID:       docker-pullable://nginx@sha256:3b7732505933ca591ce4a6d860cb713ad96a3176b82f7979a8dfa
9973486a0d6
    Port:           <none>
    Host Port:      <none>
    State:          Running
      Started:      Thu, 16 Oct 2025 11:34:51 +0530
    Ready:          True
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-jdcnc (ro)
Conditions:
  Type                        Status
  PodReadyToStartContainers   True
  Initialized                 True
  Ready                       True
  ContainersReady             True
  PodScheduled                True
Volumes:
  kube-api-access-jdcnc:
    Type:                    Projected (a volume that contains injected data from multiple sources)
    TokenExpirationSeconds:  3607
    ConfigMapName:           kube-root-ca.crt
    ConfigMapOptional:       <nil>
    DownwardAPI:             true
QoS Class:                   BestEffort
Node-Selectors:              <none>
Tolerations:                 node.kubernetes.io/not-ready:NoExecute op=Exists for 300s
                             node.kubernetes.io/unreachable:NoExecute op=Exists for 300s
Events:
  Type    Reason     Age   From               Message
  ----    ------     ----  ----               -------
  Normal  Scheduled  112s  default-scheduler  Successfully assigned default/mynginx-645865c456-lj7hz to
minikube
  Normal  Pulling    110s  kubelet            Pulling image "nginx"
  Normal  Pulled     30s   kubelet            Successfully pulled image "nginx" in 1m19.655s (1m19.671s
including waiting). Image size: 159974475 bytes.
  Normal  Created    22s   kubelet            Created container: nginx
  Normal  Started    21s   kubelet            Started container nginx

C:\Windows\System32>
```

2. **Expose Deployment as a Service**:
   - Command:

```
kubectl expose deployment mynginx --type=NodePort --port=80 --target-port=80
```

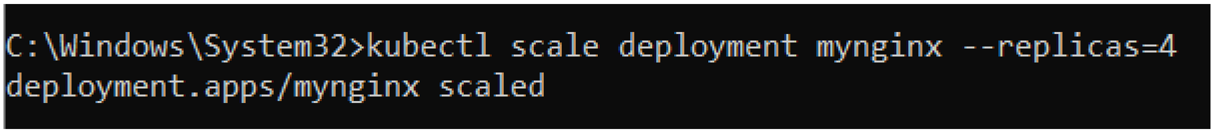


```
C:\Windows\System32>kubectl expose deployment mynginx --type=NodePort --port=80 --target-port=80
service/mynginx exposed
```

Step 3: Scale the Deployment

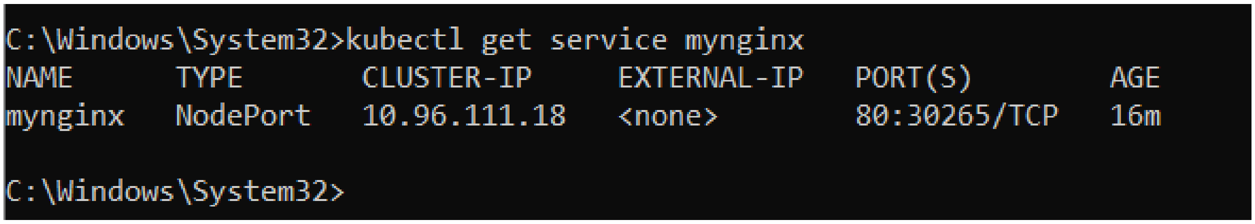
Command:Scales the Nginx deployment to 4 replicas (pods).

```
kubectl scale deployment mynginx --replicas=4
```



```
C:\Windows\System32>kubectl scale deployment mynginx --replicas=4
deployment.apps/mynginx scaled
```

```
kubectl get service myngnix
```



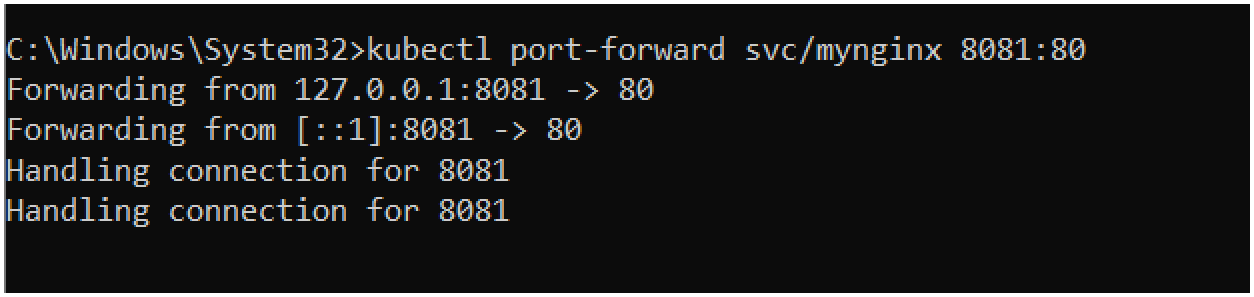
```
C:\Windows\System32>kubectl get service mynginx
NAME      TYPE       CLUSTER-IP     EXTERNAL-IP   PORT(S)        AGE
mynginx   NodePort   10.96.111.18   <none>        80:30265/TCP   16m

C:\Windows\System32>
```

Step 4: Access the Nginx App

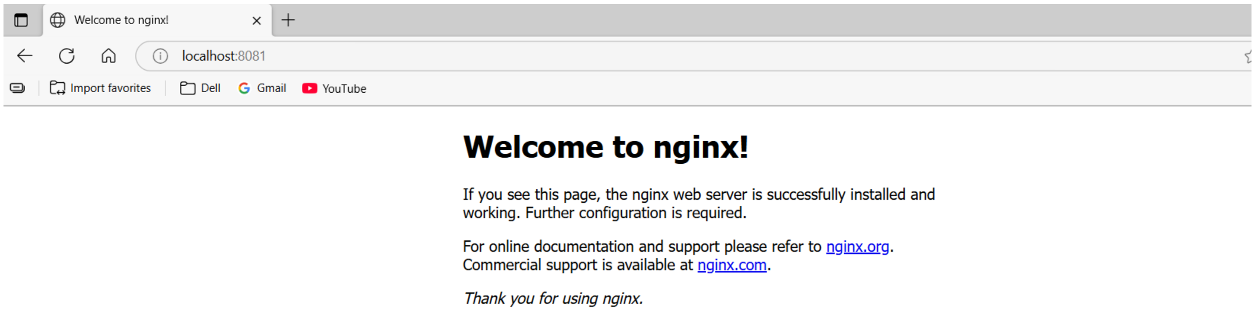
1. **Using Port Forwarding**:
   - Command:

     ```
     kubectl port-forward svc/mynginx 8081:80
     ```

     - Access the app via http://localhost:8081.



```
C:\Windows\System32>kubectl port-forward svc/mynginx 8081:80
Forwarding from 127.0.0.1:8081 -> 80
Forwarding from [::1]:8081 -> 80
Handling connection for 8081
Handling connection for 8081
```



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

If Error, use this option, **Using Minikube Tunnel**:

- Start the tunnel:

  ```
  minikube tunnel
  ```

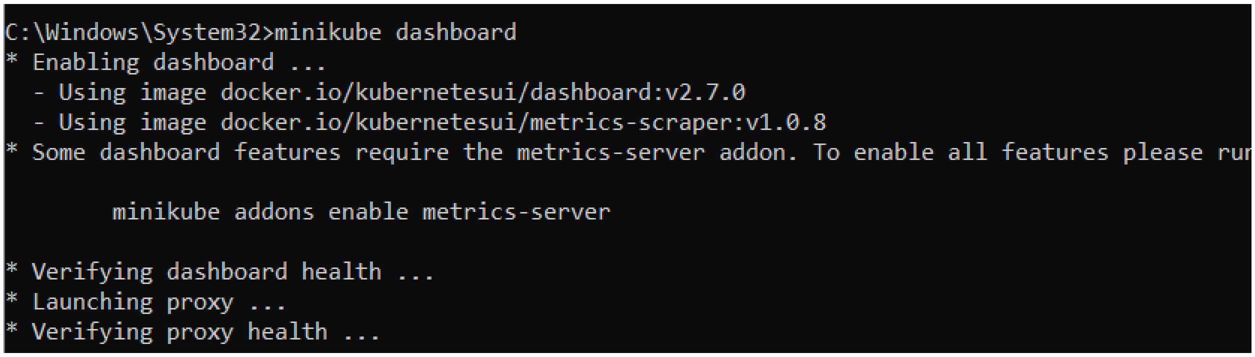
- Retrieve the service URL:

  ```
  minikube service mynginx --url
  ```

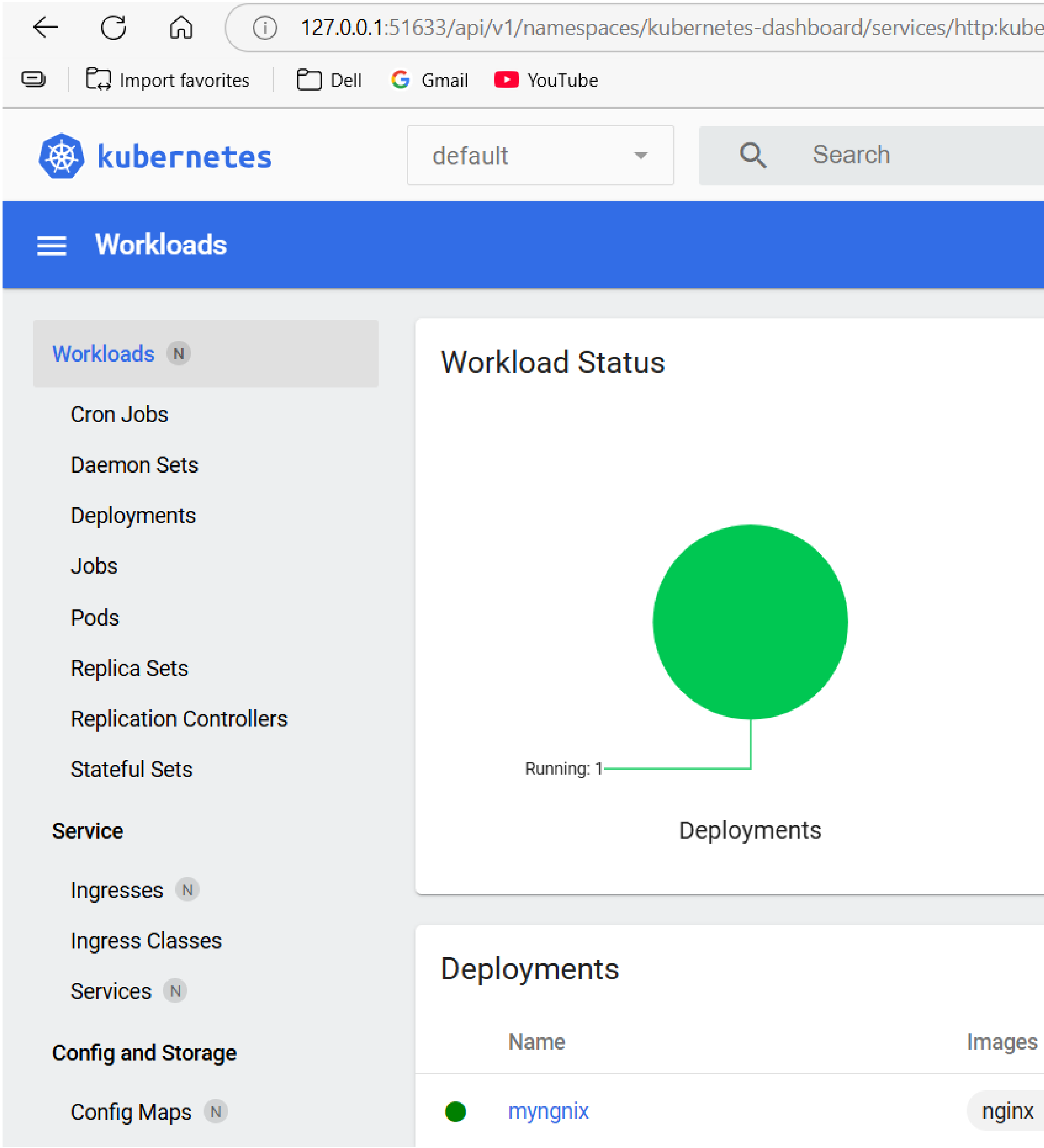
- Open the provided URL in your browser.
- Open the kubernets dashboard

- Open the minikube dashboard

  Minikube dashboard



```
C:\Windows\System32>minikube dashboard
* Enabling dashboard ...
  - Using image docker.io/kubernetesui/dashboard:v2.7.0
  - Using image docker.io/kubernetesui/metrics-scraper:v1.0.8
* Some dashboard features require the metrics-server addon. To enable all features please run

        minikube addons enable metrics-server

* Verifying dashboard health ...
* Launching proxy ...
* Verifying proxy health ...
```



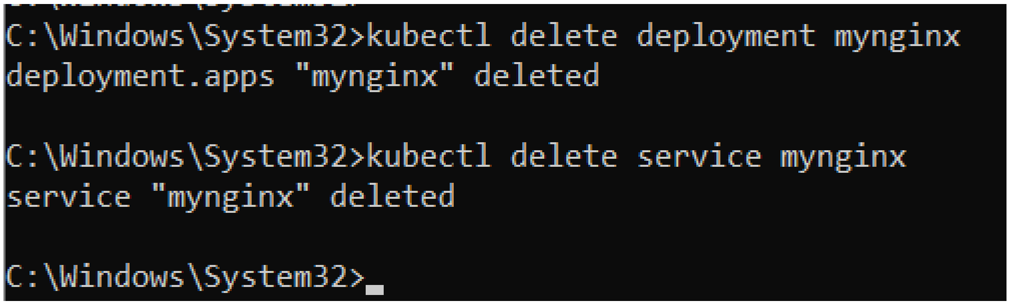
Deployments

	Name	Images
●	myngnix	nginx

Step 5: Stop and Clean Up

1. **Stop Nginx Deployment**:
   - Commands:

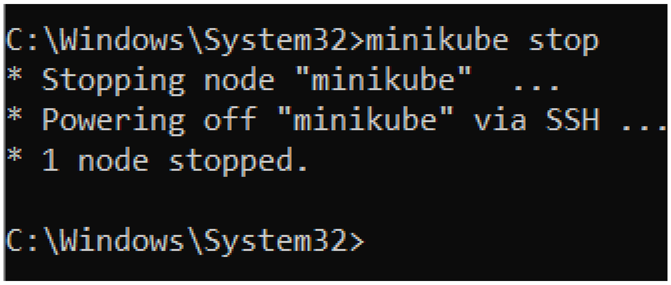
```
kubectl delete deployment mynginx
kubectl delete service mynginx
```



```
C:\Windows\System32>kubectl delete deployment mynginx
deployment.apps "mynginx" deleted

C:\Windows\System32>kubectl delete service mynginx
service "mynginx" deleted

C:\Windows\System32>
```

2. **Stop Minikube (Optional)**:
   - Command:

```
minikube stop
```



```
C:\Windows\System32>minikube stop
* Stopping node "minikube"  ...
* Powering off "minikube" via SSH ...
* 1 node stopped.

C:\Windows\System32>
```

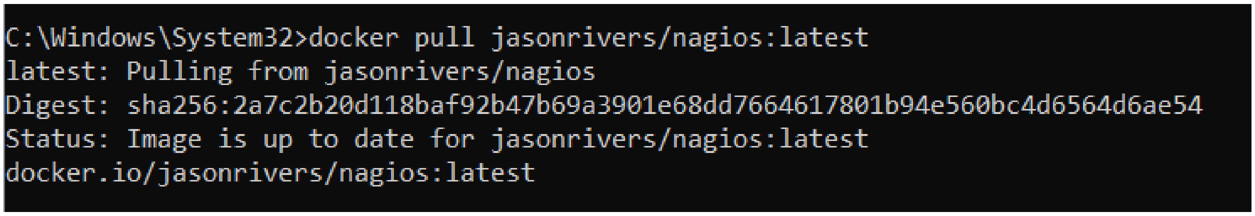
3. **Delete Minikube Cluster (Optional)**:
   - Command:`minikube delete`

Nagios Automation Steps

Step 1: Pull the Nagios Docker Image

- Open a terminal and run:

```
docker pull jasonrivers/nagios:latest
```

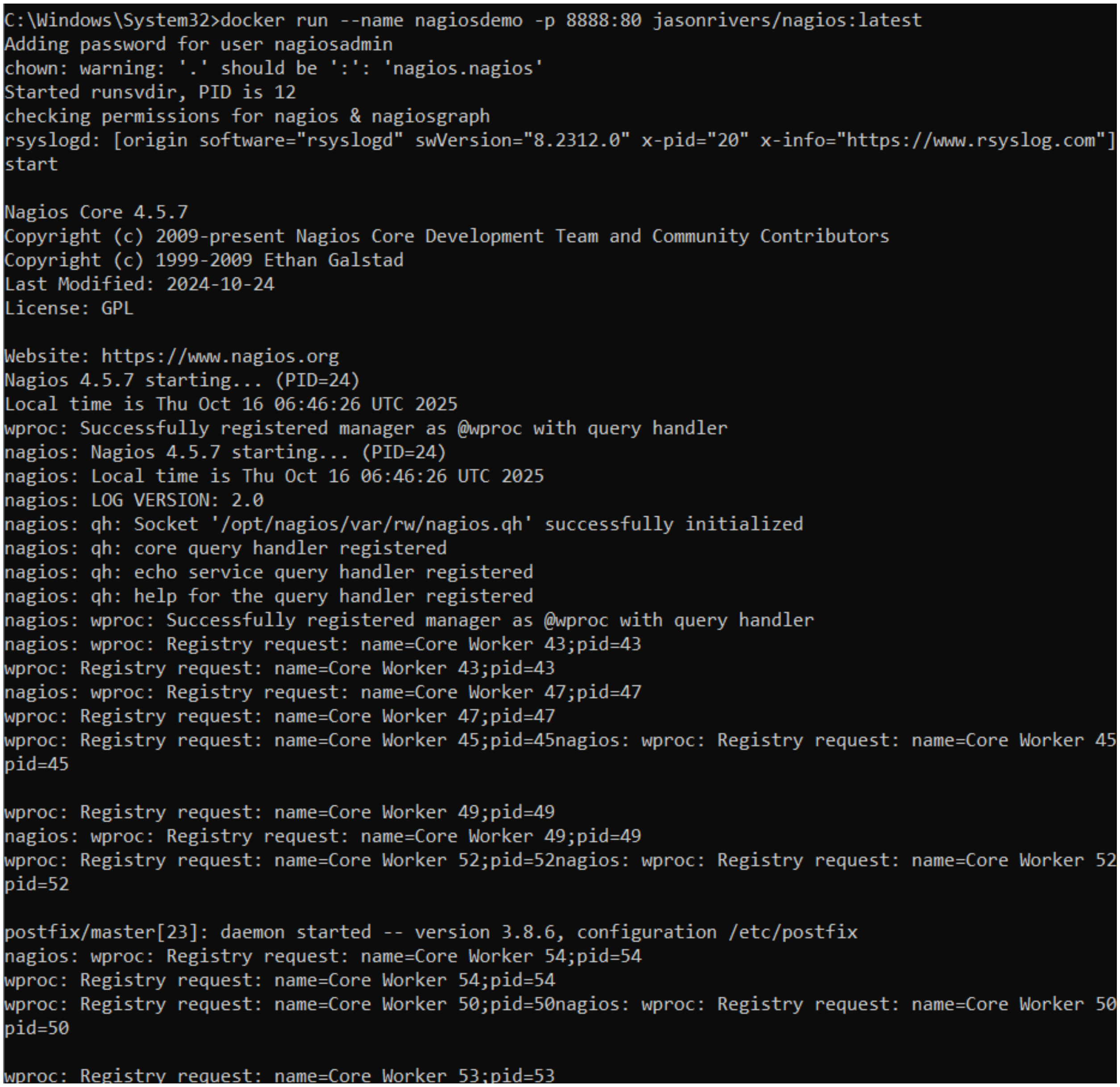


```
C:\Windows\System32>docker pull jasonrivers/nagios:latest
latest: Pulling from jasonrivers/nagios
Digest: sha256:2a7c2b20d118baf92b47b69a3901e68dd7664617801b94e560bc4d6564d6ae54
Status: Image is up to date for jasonrivers/nagios:latest
docker.io/jasonrivers/nagios:latest
```

Step 2: Run Nagios

- Command:

```
docker run --name nagiosdemo -p 8888:80 jasonrivers/nagios:latest
```



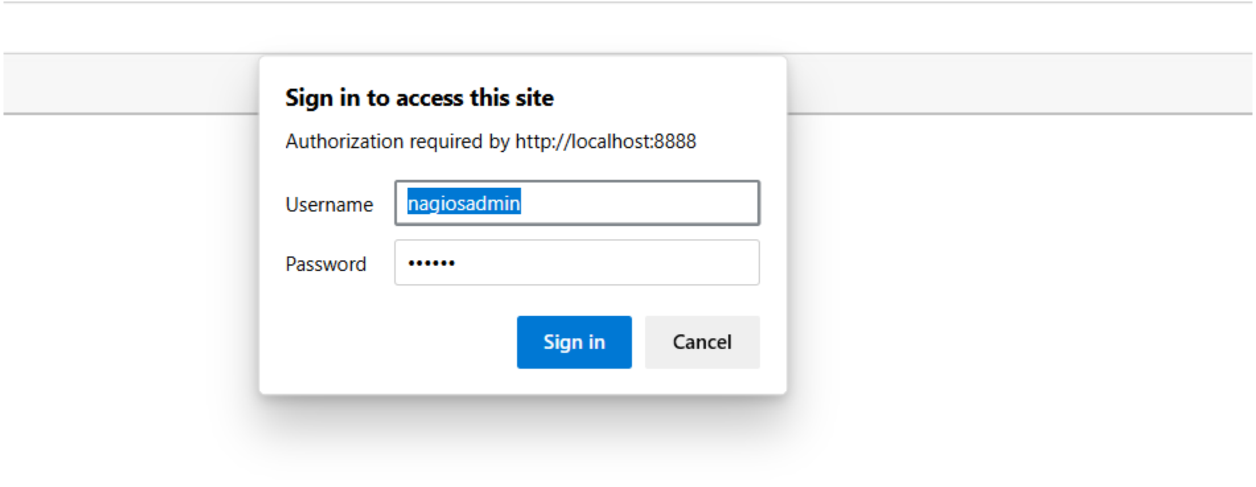
```
C:\Windows\System32>docker run --name nagiosdemo -p 8888:80 jasonrivers/nagios:latest
Adding password for user nagiosadmin
chown: warning: '.' should be ':': 'nagios.nagios'
Started runsvdir, PID is 12
checking permissions for nagios & nagiosgraph
rsyslogd: [origin software="rsyslogd" swVersion="8.2312.0" x-pid="20" x-info="https://www.rsyslog.com"]
start

Nagios Core 4.5.7
Copyright (c) 2009-present Nagios Core Development Team and Community Contributors
Copyright (c) 1999-2009 Ethan Galstad
Last Modified: 2024-10-24
License: GPL

Website: https://www.nagios.org
Nagios 4.5.7 starting... (PID=24)
Local time is Thu Oct 16 06:46:26 UTC 2025
wproc: Successfully registered manager as @wproc with query handler
nagios: Nagios 4.5.7 starting... (PID=24)
nagios: Local time is Thu Oct 16 06:46:26 UTC 2025
nagios: LOG VERSION: 2.0
nagios: qh: Socket '/opt/nagios/var/rw/nagios.qh' successfully initialized
nagios: qh: core query handler registered
nagios: qh: echo service query handler registered
nagios: qh: help for the query handler registered
nagios: wproc: Successfully registered manager as @wproc with query handler
nagios: wproc: Registry request: name=Core Worker 43;pid=43
wproc: Registry request: name=Core Worker 43;pid=43
nagios: wproc: Registry request: name=Core Worker 47;pid=47
wproc: Registry request: name=Core Worker 47;pid=47
wproc: Registry request: name=Core Worker 45;pid=45nagios: wproc: Registry request: name=Core Worker 45
pid=45

wproc: Registry request: name=Core Worker 49;pid=49
nagios: wproc: Registry request: name=Core Worker 49;pid=49
wproc: Registry request: name=Core Worker 52;pid=52nagios: wproc: Registry request: name=Core Worker 52
pid=52

postfix/master[23]: daemon started -- version 3.8.6, configuration /etc/postfix
nagios: wproc: Registry request: name=Core Worker 54;pid=54
wproc: Registry request: name=Core Worker 54;pid=54
wproc: Registry request: name=Core Worker 50;pid=50nagios: wproc: Registry request: name=Core Worker 50
pid=50

wproc: Registry request: name=Core Worker 53;pid=53
```

Step 3: Access Nagios Dashboard

- Open your browser and navigate to:

```
http://localhost:8888
```

  - **Login Credentials**:
    - Username: `nagiosadmin`
    - Password: `nagios`

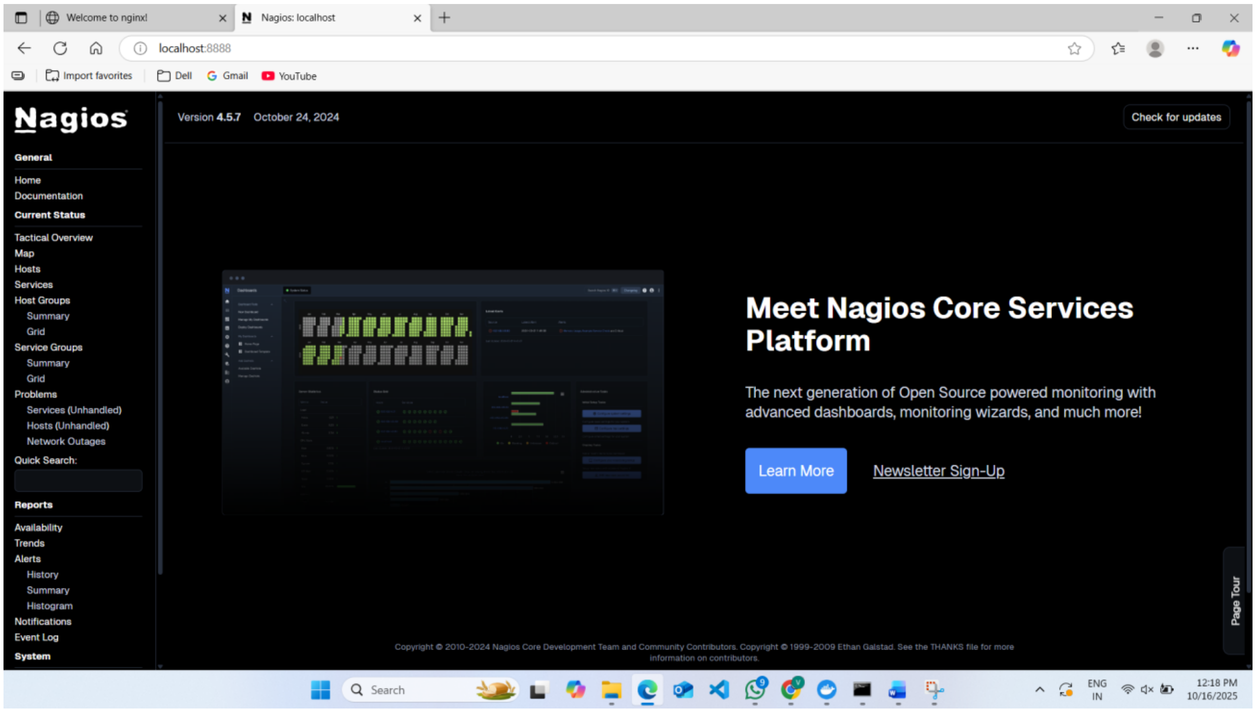


Sign in to access this site

Authorization required by http://localhost:8888

Username: nagiosadmin

Password: ••••••



Version 4.5.7 October 24, 2024

Meet Nagios Core Services Platform

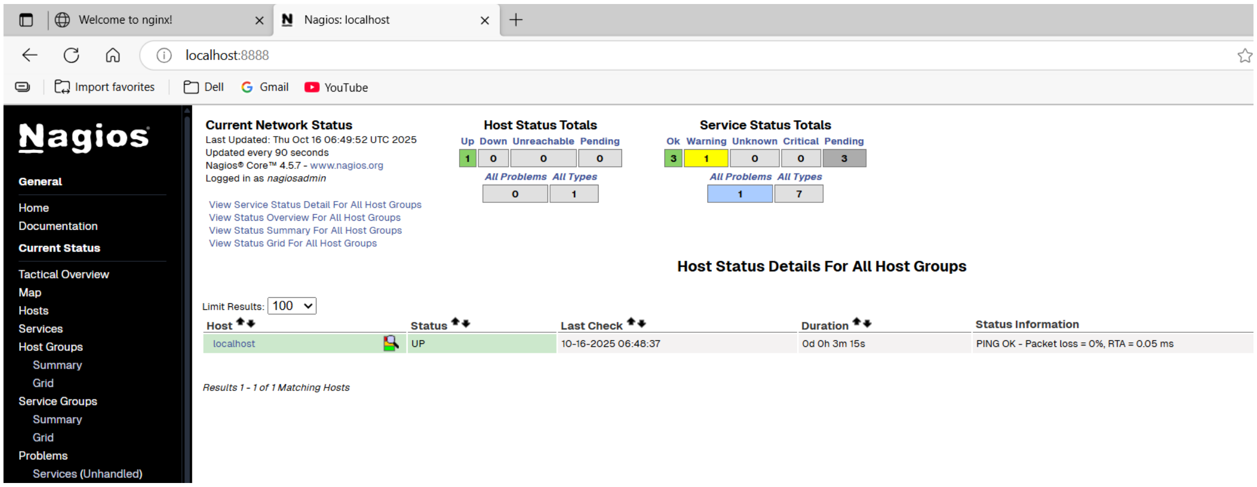
The next generation of Open Source powered monitoring with advanced dashboards, monitoring wizards, and much more!

Copyright © 2010-2024 Nagios Core Development Team and Community Contributors. Copyright © 1999-2009 Ethan Galstad. See the THANKS file for more information on contributors.

- Once logged in, explore:
  - Hosts: View systems being monitored.
  - Services: Check tasks being monitored (e.g., CPU usage).
  - Alerts: Access recent notifications.

Step 4: Monitoring Host Details

1. **Navigate to the Host Information Page**:
    - Select a host from the **Hosts** menu.



Current Network Status
Last Updated: Thu Oct 16 06:49:52 UTC 2025
Updated every 90 seconds
Nagios® Core™ 4.5.7 - www.nagios.org
Logged in as *nagiosadmin*

View Service Status Detail For All Host Groups
View Status Overview For All Host Groups
View Status Summary For All Host Groups
View Status Grid For All Host Groups

Host Status Totals

Up	Down	Unreachable	Pending
1	0	0	0

All Problems	All Types
0	1

Service Status Totals

Ok	Warning	Unknown	Critical	Pending
3	1	0	0	3

All Problems	All Types
1	7

Host Status Details For All Host Groups

Limit Results: 100

Host	Status	Last Check	Duration	Status Information
localhost	UP	10-16-2025 06:48:37	0d 0h 3m 15s	PING OK - Packet loss = 0%, RTA = 0.05 ms

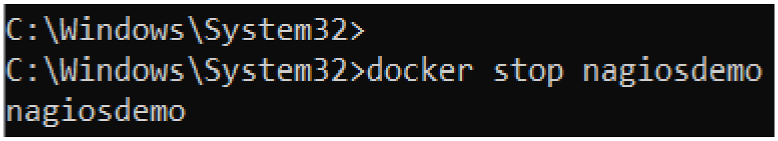
Results 1 - 1 of 1 Matching Hosts

2. **Key Details**:
    - Host Status: Indicates if the system is UP or DOWN.
    - Metrics: View CPU usage, memory status, and network activity.
    - Actions: Reschedule checks, disable notifications, or schedule downtime.

Step 5: Stop and Remove Nagios

1. **Stop the Container**:
    - Command:

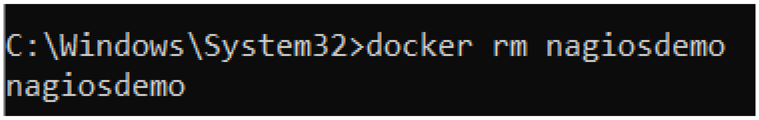
```
docker stop nagiosdemo
```



```
C:\Windows\System32>
C:\Windows\System32>docker stop nagiosdemo
nagiosdemo
```

2. **Delete the Container**:
    - Command:

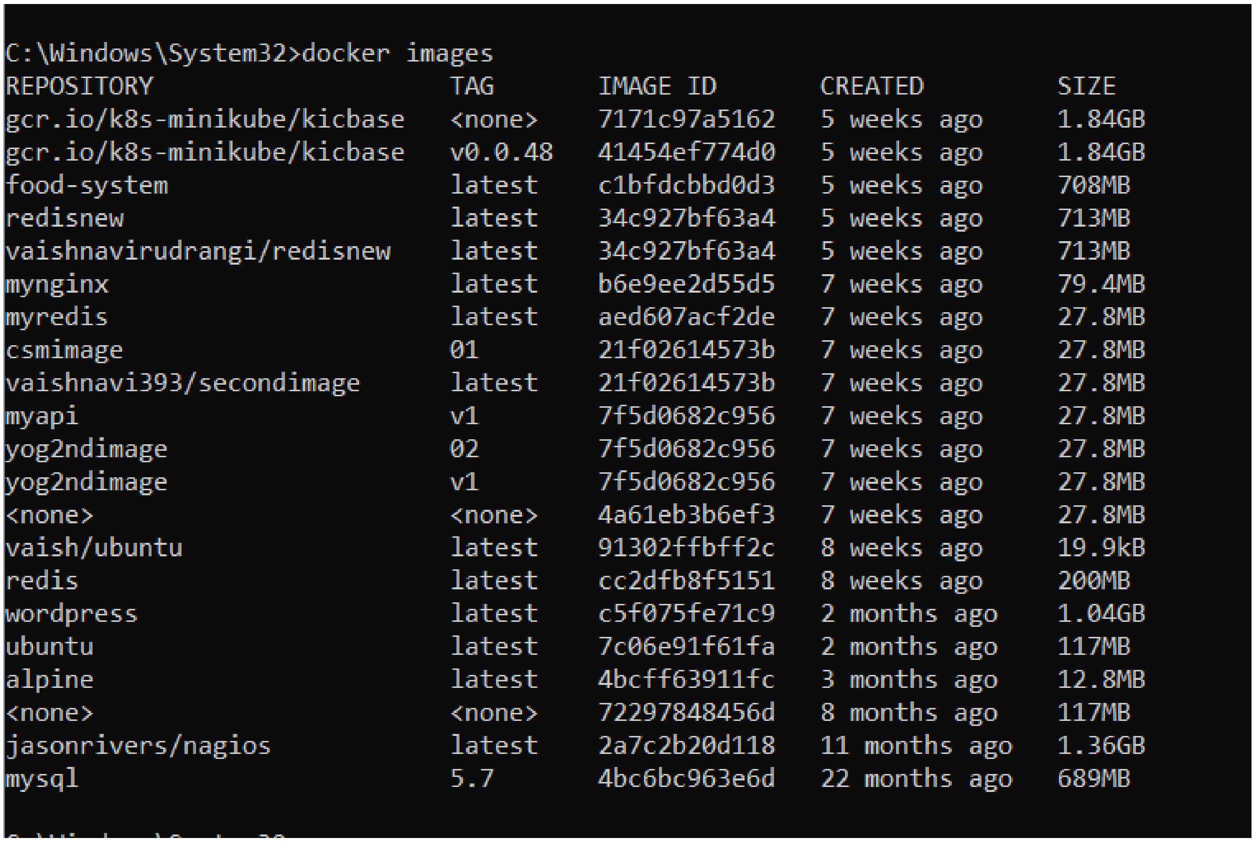
```
docker rm nagiosdemo
```



```
C:\Windows\System32>docker rm nagiosdemo
nagiosdemo
```

3. **Remove the Image** (Optional):
   - List images:

```
docker images
```

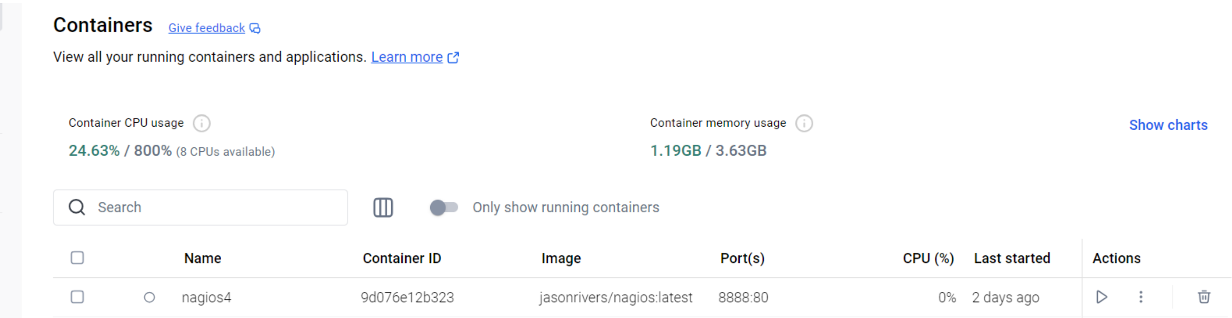


```
C:\Windows\System32>docker images
REPOSITORY                     TAG       IMAGE ID       CREATED         SIZE
gcr.io/k8s-minikube/kicbase    <none>    7171c97a5162   5 weeks ago     1.84GB
gcr.io/k8s-minikube/kicbase    v0.0.48   41454ef774d0   5 weeks ago     1.84GB
food-system                    latest    c1bfdcbbd0d3   5 weeks ago     708MB
redisnew                       latest    34c927bf63a4   5 weeks ago     713MB
vaishnavirudrangi/redisnew     latest    34c927bf63a4   5 weeks ago     713MB
mynginx                        latest    b6e9ee2d55d5   7 weeks ago     79.4MB
myredis                        latest    aed607acf2de   7 weeks ago     27.8MB
csmimage                       01        21f02614573b   7 weeks ago     27.8MB
vaishnavi393/secondimage       latest    21f02614573b   7 weeks ago     27.8MB
myapi                          v1        7f5d0682c956   7 weeks ago     27.8MB
yog2ndimage                    02        7f5d0682c956   7 weeks ago     27.8MB
yog2ndimage                    v1        7f5d0682c956   7 weeks ago     27.8MB
<none>                         <none>    4a61eb3b6ef3   7 weeks ago     27.8MB
vaish/ubuntu                   latest    91302ffbff2c   8 weeks ago     19.9kB
redis                          latest    cc2dfb8f5151   8 weeks ago     200MB
wordpress                      latest    c5f075fe71c9   2 months ago    1.04GB
ubuntu                         latest    7c06e91f61fa   2 months ago    117MB
alpine                         latest    4bcff63911fc   3 months ago    12.8MB
<none>                         <none>    72297848456d   8 months ago    117MB
jasonrivers/nagios             latest    2a7c2b20d118   11 months ago   1.36GB
mysql                          5.7       4bc6bc963e6d   22 months ago   689MB
```

   - Delete the Nagios image:

```
docker rmijasonrivers/nagios:latest
```

4. Observe the docker containers in DockerHub, we can see the latest Nagios Installed running on port:8888



Containers Give feedback

View all your running containers and applications. Learn more

Container CPU usage
24.63% / 800% (8 CPUs available)

Container memory usage
1.19GB / 3.63GB

□		Name	Container ID	Image	Port(s)	CPU (%)	Last started	Actions
□	○	nagios4	9d076e12b323	jasonrivers/nagios:latest	8888:80	0%	2 days ago	▷ ⋮ 🗑

Steps for AWS-free trier account creation

- open https://aws.amazon.com
- click on create AWS account
- Provide email id and name details for AWS account creation then a screen will appear as below
- Enter verification code received in your mail
- Provide your contact details and agree Terms and conditions, then click on create
- Provide billing details, click on verify and continue
- Amazon chagres 2 rs and it will be credit back into your account once verification is over
- Once payment success a screen appear as below, provide your working contact number after selecting country, and click on send SMS after entering the details
- Provide verification code received in your mobile
- Now select basic plan and click on complete sign up
- Now a screen appear as follows then click on AWS management console
- You will navigating to the aws page ..where you need to Select role as Academic/Researcher and interest as DevOps and click on submit
- Sign into AWS console as Root user
- A console screen appears aws will appear

SBQ’s

1. Your Pod keeps restarting repeatedly.

→ Check logs using kubectl logs <pod> --previous and describe the pod to find crash or config errors.

2. A Kubernetes pod is stuck in a "Pending" state. What could be the possible reasons, and how would you troubleshoot it?

→ Usually due to insufficient resources or scheduling issues; check with kubectl describe pod <pod-name>.

3. How would you debug a failed deployment in Kubernetes?

→ Use kubectl describe deployment <name> to check events and kubectl logs for container errors.

4. You have a Kubernetes Deployment with multiple replicas, and some pods are failing health checks. How would you identify the root cause and fix it?

→ Check liveness/readiness probe configurations and inspect logs to find what’s failing.

5. How do you roll back a faulty deployment?

→ Run kubectl rollout undo deployment/<deployment-name>.

6. How do you debug a running Pod?

→ Use kubectl exec -it <pod> -- /bin/sh and kubectl logs <pod> to inspect inside the container.

7. You need to expose a local service externally. How to do?

→ Use kubectl expose deployment <name> --type=LoadBalancer --port=<port>.

8. How to start and stop Nagios?

→ Start with sudo systemctl start nagios and stop with sudo systemctl stop nagios.

9. You installed Nagios but the web interface shows “Unable to connect to Nagios process”. How to resolve this?

→ Ensure the Nagios service is running and check nagios.cfg for a valid lock_file path.

10. You added a new host in Nagios, but it’s not appearing on the web interface. How to check?

→ Verify config using nagios -v /usr/local/nagios/etc/nagios.cfg and restart Nagios.

11. How can you check whether Nagios is running properly?

→ Run sudo systemctl status nagios or ps aux | grep nagios.

12. How do you view Nagios logs in real-time?

→ Use tail -f /usr/local/nagios/var/nagios.log.

13. What are the advantages of using Nagios?

→ Provides proactive monitoring, alerting, performance tracking, and centralized visibility.

Conclusion: In this week we learnt how to deploy pods in minikube and how to monitor our local system using Nagios tool and will know how to create the aws free trier account.