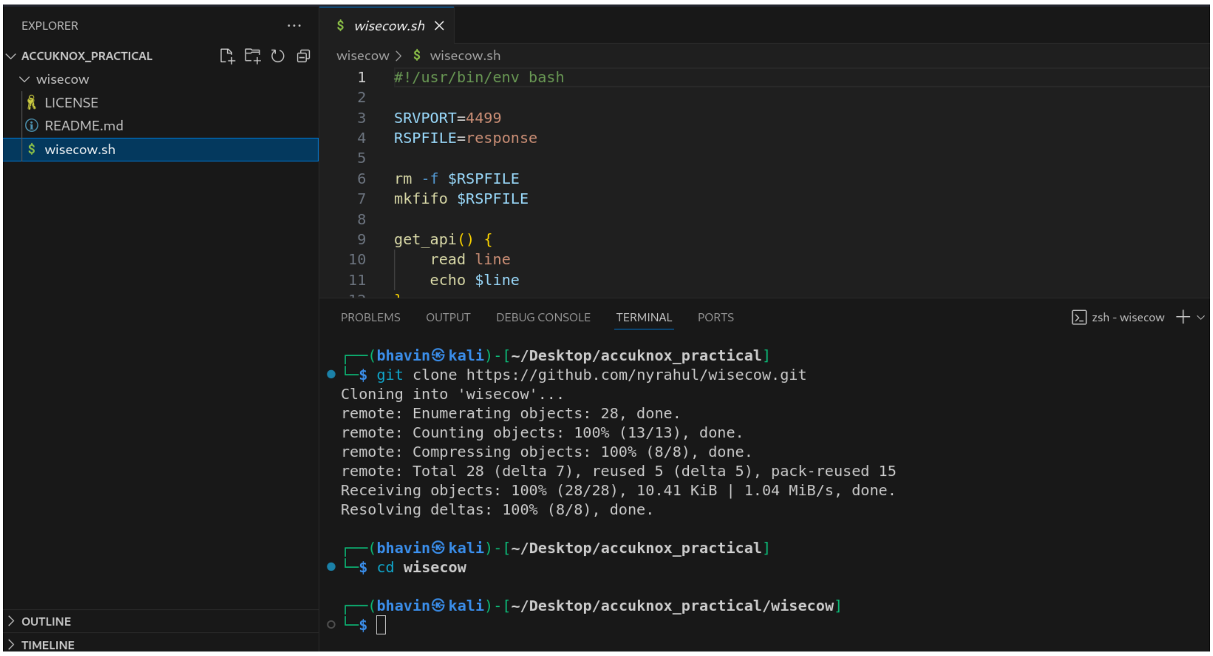
Bhavin_Accuknox_Practical

Practical 1 - Containerisation and Deployment of Wisecow Application on Kubernetes

Steps -

1. Cloning Repository: Cloned the repository onto the system and accessed it using Visual Studio Code IDE.



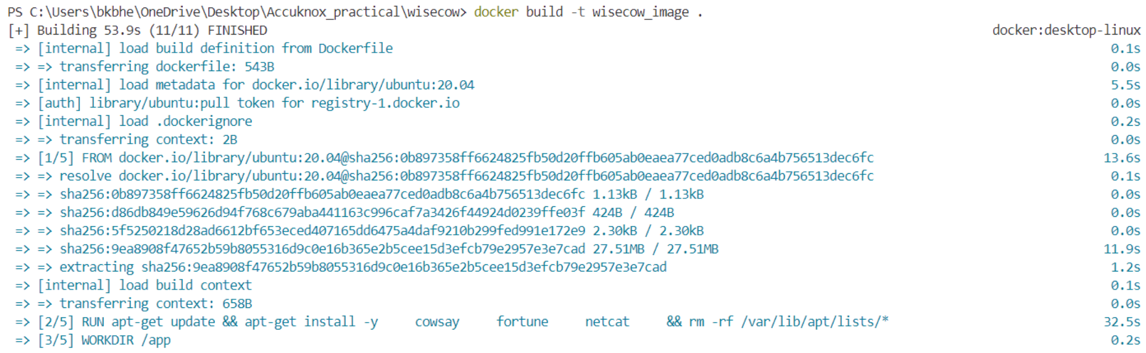
```bash
#!/usr/bin/env bash

SRVPORT=4499
RSPFILE=response

rm -f $RSPFILE
mkfifo $RSPFILE

get_api() {
    read line
    echo $line
```

```
┌──(bhavin㉿kali)-[~/Desktop/accuknox_practical]
└─$ git clone https://github.com/nyrahul/wisecow.git
Cloning into 'wisecow'...
remote: Enumerating objects: 28, done.
remote: Counting objects: 100% (13/13), done.
remote: Compressing objects: 100% (8/8), done.
remote: Total 28 (delta 7), reused 5 (delta 5), pack-reused 15
Receiving objects: 100% (28/28), 10.41 KiB | 1.04 MiB/s, done.
Resolving deltas: 100% (8/8), done.

┌──(bhavin㉿kali)-[~/Desktop/accuknox_practical]
└─$ cd wisecow

┌──(bhavin㉿kali)-[~/Desktop/accuknox_practical/wisecow]
└─$
```

Fig. 1 - Clone Repo and access using VS Code IDE

2. Building Docker Image: Built the Docker image using a Dockerfile. The Dockerfile was written based on the programming language used, such as bash, so an Ubuntu

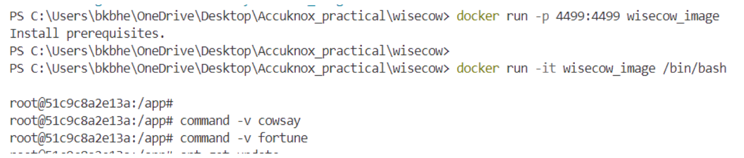
image was used as the base image. Initially, the build failed with the error '/usr/bin/env: 'bash\r': No such file or directory'. To resolve this, I opened `wisecow.sh` and changed the line endings from CRLF to LF.



```
PS C:\Users\bkbhe\OneDrive\Desktop\Accuknox_practical\wisecow> docker build -t wisecow_image .
[+] Building 53.9s (11/11) FINISHED                                                                            docker:desktop-linux
 => [internal] load build definition from Dockerfile                                                                           0.1s
 => => transferring dockerfile: 543B                                                                                           0.0s
 => [internal] load metadata for docker.io/library/ubuntu:20.04                                                                5.5s
 => [auth] library/ubuntu:pull token for registry-1.docker.io                                                                  0.0s
 => [internal] load .dockerignore                                                                                              0.2s
 => => transferring context: 2B                                                                                                0.0s
 => [1/5] FROM docker.io/library/ubuntu:20.04@sha256:0b897358ff6624825fb50d20ffb605ab0eaea77ced0adb8c6a4b756513dec6fc         13.6s
 => => resolve docker.io/library/ubuntu:20.04@sha256:0b897358ff6624825fb50d20ffb605ab0eaea77ced0adb8c6a4b756513dec6fc          0.1s
 => => sha256:0b897358ff6624825fb50d20ffb605ab0eaea77ced0adb8c6a4b756513dec6fc 1.13kB / 1.13kB                                 0.0s
 => => sha256:d86db849e59626d94f768c679aba441163c996caf7a3426f44924d0239ffe03f 424B / 424B                                     0.0s
 => => sha256:5f5250218d28ad6612bf653eced407165dd6475a4daf9210b299fed991e172e9 2.30kB / 2.30kB                                 0.0s
 => => sha256:9ea8908f47652b59b8055316d9c0e16b365e2b5cee15d3efcb79e2957e3e7cad 27.51MB / 27.51MB                              11.9s
 => => extracting sha256:9ea8908f47652b59b8055316d9c0e16b365e2b5cee15d3efcb79e2957e3e7cad                                      1.2s
 => [internal] load build context                                                                                              0.1s
 => => transferring context: 658B                                                                                              0.0s
 => [2/5] RUN apt-get update && apt-get install -y     cowsay     fortune     netcat     && rm -rf /var/lib/apt/lists/*       32.5s
 => [3/5] WORKDIR /app                                                                                                         0.2s
```

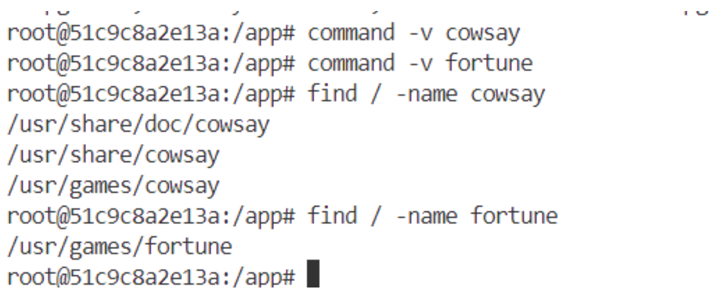
Fig. 2 - Building Docker images

3. Library Path Issue: Faced an issue where the build process and required libraries were installed, but the container's paths did not match. Modified the Dockerfile to link the paths correctly.



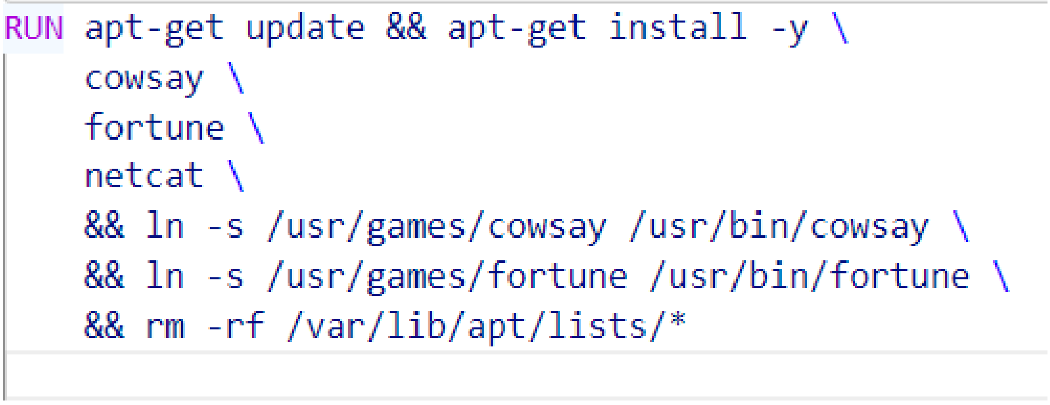
```
PS C:\Users\bkbhe\OneDrive\Desktop\Accuknox_practical\wisecow> docker run -p 4499:4499 wisecow_image
Install prerequisites.
PS C:\Users\bkbhe\OneDrive\Desktop\Accuknox_practical\wisecow>
PS C:\Users\bkbhe\OneDrive\Desktop\Accuknox_practical\wisecow> docker run -it wisecow_image /bin/bash

root@51c9c8a2e13a:/app#
root@51c9c8a2e13a:/app# command -v cowsay
root@51c9c8a2e13a:/app# command -v fortune
root@51c9c8a2e13a:/app# apt-get update
```

Fig. 3 - Debugging Proble by going inside running container



```
root@51c9c8a2e13a:/app# command -v cowsay
root@51c9c8a2e13a:/app# command -v fortune
root@51c9c8a2e13a:/app# find / -name cowsay
/usr/share/doc/cowsay
/usr/share/cowsay
/usr/games/cowsay
root@51c9c8a2e13a:/app# find / -name fortune
/usr/games/fortune
root@51c9c8a2e13a:/app# 
```

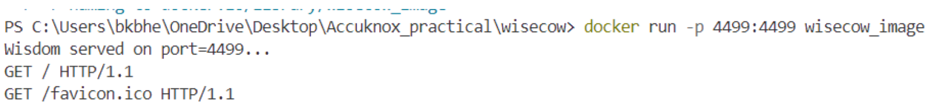
Fig. 4 - Find library installed path and actual code path not matched



```
RUN apt-get update && apt-get install -y \
    cowsay \
    fortune \
    netcat \
    && ln -s /usr/games/cowsay /usr/bin/cowsay \
    && ln -s /usr/games/fortune /usr/bin/fortune \
    && rm -rf /var/lib/apt/lists/*
```

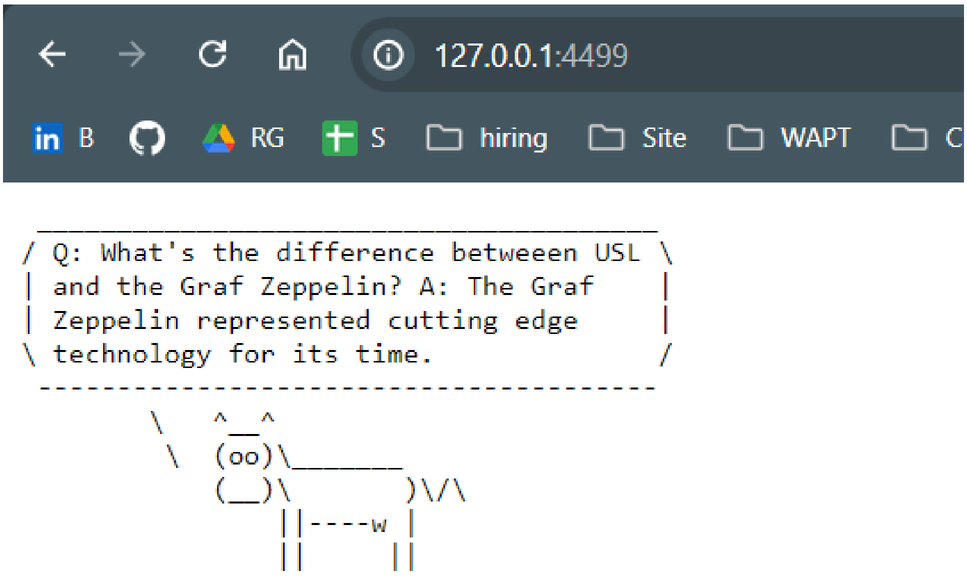
Fig. 4 - To resolve above problem implement symbolic link

4. Successful Container Run: Retried running the container, and this time the web server was accessible as expected.



```
PS C:\Users\bkbhe\OneDrive\Desktop\Accuknox_practical\wisecow> docker run -p 4499:4499 wisecow_image
Wisdom served on port=4499...
GET / HTTP/1.1
GET /favicon.ico HTTP/1.1
```

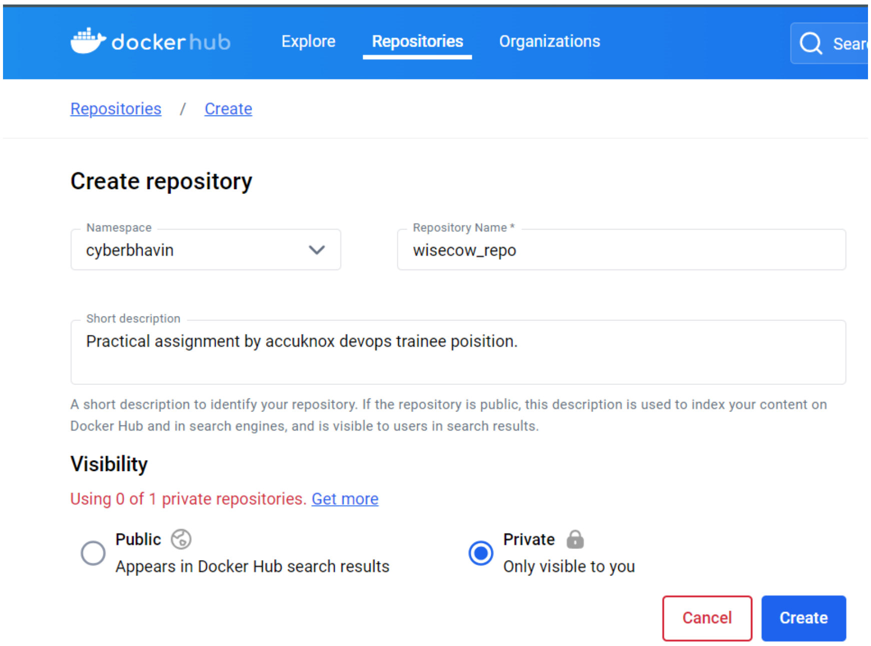
Fig. 5 - Container Running



```
 ______________________________________
/ Q: What's the difference betweeen USL \
| and the Graf Zeppelin? A: The Graf    |
| Zeppelin represented cutting edge     |
\ technology for its time.              /
 --------------------------------------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||
```

Fig. 6 - Accessing Container

5. Pushing Image to DockerHub: After successful local testing, created a private repository on DockerHub and pushed the image there.



Create repository

Namespace: cyberbhavin

Repository Name *: wisecow_repo

Short description: Practical assignment by accuknox devops trainee poisition.

A short description to identify your repository. If the repository is public, this description is used to index your content on Docker Hub and in search engines, and is visible to users in search results.

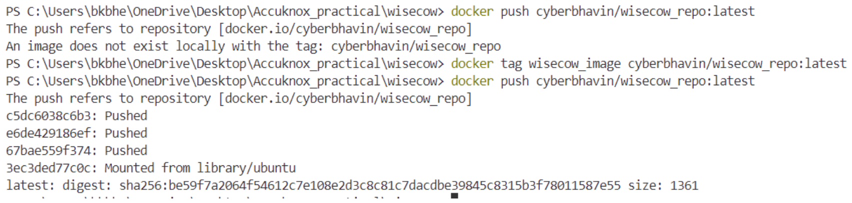
Visibility

Using 0 of 1 private repositories. Get more

○ Public — Appears in Docker Hub search results

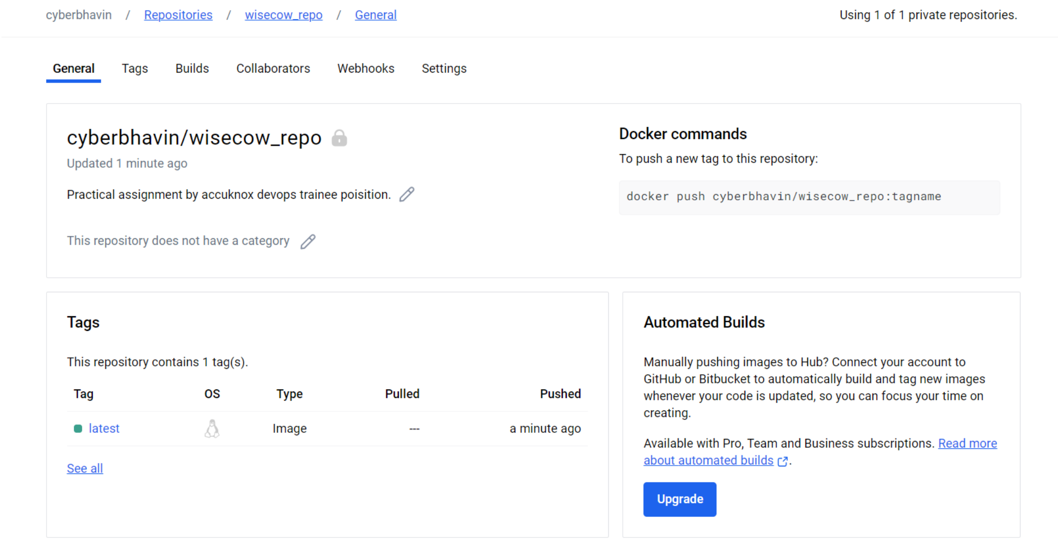
● Private — Only visible to you

Fig. 7 - Create Private repo in docker hub



```
PS C:\Users\bkbhe\OneDrive\Desktop\Accuknox_practical\wisecow> docker push cyberbhavin/wisecow_repo:latest
The push refers to repository [docker.io/cyberbhavin/wisecow_repo]
An image does not exist locally with the tag: cyberbhavin/wisecow_repo
PS C:\Users\bkbhe\OneDrive\Desktop\Accuknox_practical\wisecow> docker tag wisecow_image cyberbhavin/wisecow_repo:latest
PS C:\Users\bkbhe\OneDrive\Desktop\Accuknox_practical\wisecow> docker push cyberbhavin/wisecow_repo:latest
The push refers to repository [docker.io/cyberbhavin/wisecow_repo]
c5dc6038c6b3: Pushed
e6de429186ef: Pushed
67bae559f374: Pushed
3ec3ded77c0c: Mounted from library/ubuntu
latest: digest: sha256:be59f7a2064f54612c7e108e2d3c8c81c7dacdbe39845c8315b3f78011587e55 size: 1361
```

Fig. 8 - Push image to docker hub repo



cyberbhavin / Repositories / wisecow_repo / General

Using 1 of 1 private repositories.

cyberbhavin/wisecow_repo

Updated 1 minute ago

Practical assignment by accuknox devops trainee poisition.

This repository does not have a category

Docker commands

To push a new tag to this repository:

```
docker push cyberbhavin/wisecow_repo:tagname
```

Tags

This repository contains 1 tag(s).

Tag	OS	Type	Pulled	Pushed
latest		Image	---	a minute ago

See all

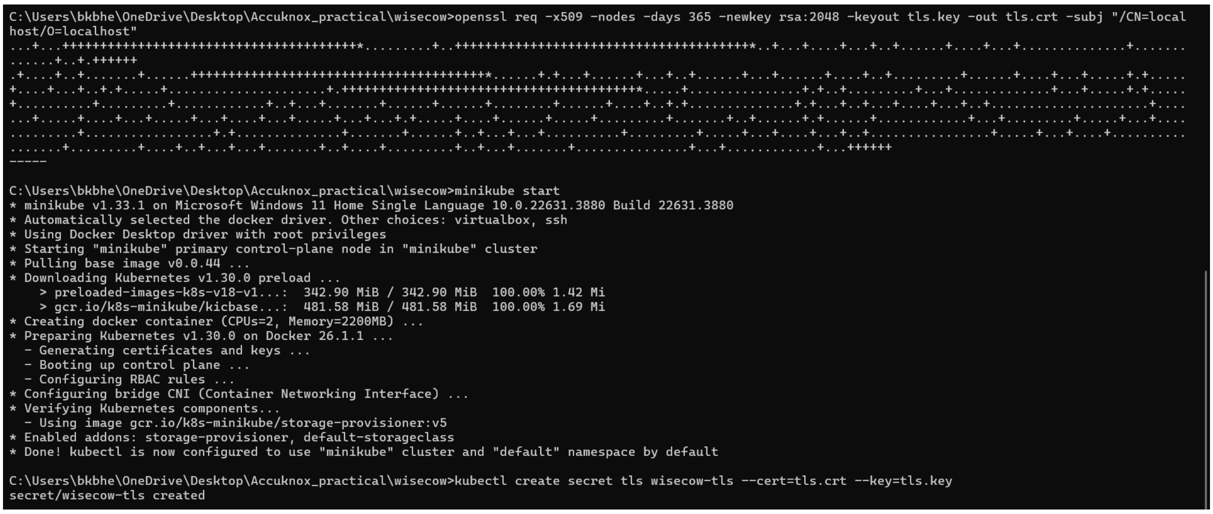
Automated Builds

Manually pushing images to Hub? Connect your account to GitHub or Bitbucket to automatically build and tag new images whenever your code is updated, so you can focus your time on creating.

Available with Pro, Team and Business subscriptions. Read more about automated builds.

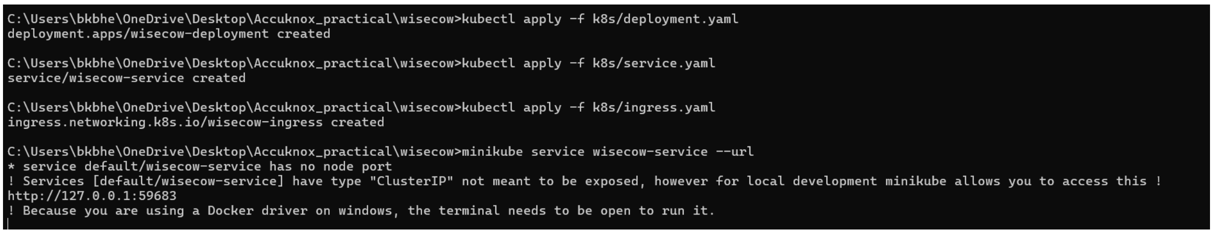
Fig. 9 - Image successfully pushed

6. Implementing TLS: Used OpenSSL to generate certificates and started Minikube to run a local Kubernetes cluster. After starting Minikube, applied the deployment, service, and ingress files. Located the URL to access the application via the service.



```
C:\Users\bkbhe\OneDrive\Desktop\Accuknox_practical\wisecow>openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout tls.key -out tls.crt -subj "/CN=local
host/O=localhost"
...+...+++++++++++++++++++++++++++++++++++++++*.........+..+++++++++++++++++++++++++++++++++++++++*..+...+....+...+..+......+....+...+..............+.......
......+..+.++++++
.+....+..+.......+......+++++++++++++++++++++++++++++++++++++++*......+.+...+......+...+..+......+...+......+....+..+.........+......+....+...+.....+.+.....
+....+...+..+.+.....+....................+.+++++++++++++++++++++++++++++++++++++++*.....+..............+.+..+.........+...+............+...+.....+.+.....
+..........+.........+............+..+...+.......+......+......+........+......+....+..+.+..............+.+...+..+...+....+...+..+.....................+....
...+.....+....+...+.....+...+....+...+.....+...+...+.+.....+....+......+.....+.........+.......+..+.......+.+......+............+..+..........+.....+...+....
.........+.................+.+..............+.......+......+..+...+...+..........+.........+.....+...+....+...+..+................+.....+...+....+..........
.......+.........+....+..+...+...+.......+..+....+.........+..+...+.......+...............+...+............+...++++++
-----

C:\Users\bkbhe\OneDrive\Desktop\Accuknox_practical\wisecow>minikube start
* minikube v1.33.1 on Microsoft Windows 11 Home Single Language 10.0.22631.3880 Build 22631.3880
* Automatically selected the docker driver. Other choices: virtualbox, ssh
* Using Docker Desktop driver with root privileges
* Starting "minikube" primary control-plane node in "minikube" cluster
* Pulling base image v0.0.44 ...
* Downloading Kubernetes v1.30.0 preload ...
    > preloaded-images-k8s-v18-v1...:  342.90 MiB / 342.90 MiB  100.00% 1.42 Mi
    > gcr.io/k8s-minikube/kicbase...:  481.58 MiB / 481.58 MiB  100.00% 1.69 Mi
* Creating docker container (CPUs=2, Memory=2200MB) ...
* Preparing Kubernetes v1.30.0 on Docker 26.1.1 ...
  - Generating certificates and keys ...
  - Booting up control plane ...
  - Configuring RBAC rules ...
* Configuring bridge CNI (Container Networking Interface) ...
* Verifying Kubernetes components...
  - Using image gcr.io/k8s-minikube/storage-provisioner:v5
* Enabled addons: storage-provisioner, default-storageclass
* Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default

C:\Users\bkbhe\OneDrive\Desktop\Accuknox_practical\wisecow>kubectl create secret tls wisecow-tls --cert=tls.crt --key=tls.key
secret/wisecow-tls created
```

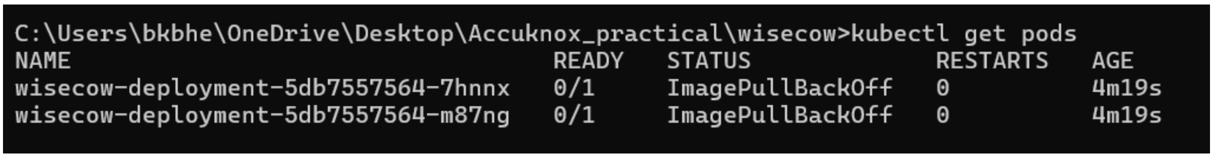
Fig. 10 - Create self-signed certificate



```
C:\Users\bkbhe\OneDrive\Desktop\Accuknox_practical\wisecow>kubectl apply -f k8s/deployment.yaml
deployment.apps/wisecow-deployment created

C:\Users\bkbhe\OneDrive\Desktop\Accuknox_practical\wisecow>kubectl apply -f k8s/service.yaml
service/wisecow-service created

C:\Users\bkbhe\OneDrive\Desktop\Accuknox_practical\wisecow>kubectl apply -f k8s/ingress.yaml
ingress.networking.k8s.io/wisecow-ingress created

C:\Users\bkbhe\OneDrive\Desktop\Accuknox_practical\wisecow>minikube service wisecow-service --url
* service default/wisecow-service has no node port
! Services [default/wisecow-service] have type "ClusterIP" not meant to be exposed, however for local development minikube allows you to access this !
http://127.0.0.1:59683
! Because you are using a Docker driver on windows, the terminal needs to be open to run it.
```

Fig. 11 - Applying manifest files

7. Image Pull Secret: Encountered a `ImagePullBackOff` status for the pods, indicating the repository was private and credentials were not provided. Generated a Kubernetes secret with the DockerHub token for authorization.



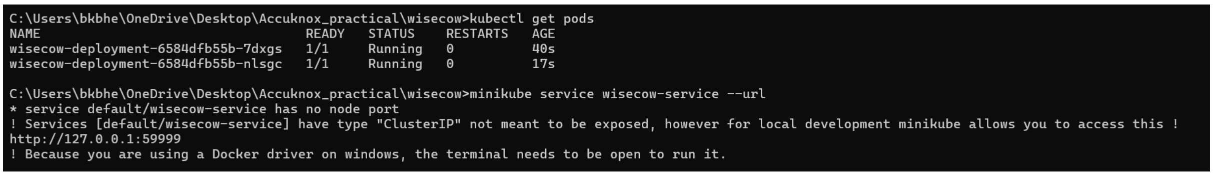
```
C:\Users\bkbhe\OneDrive\Desktop\Accuknox_practical\wisecow>kubectl get pods
NAME                                  READY   STATUS             RESTARTS   AGE
wisecow-deployment-5db7557564-7hnnx   0/1     ImagePullBackOff   0          4m19s
wisecow-deployment-5db7557564-m87ng   0/1     ImagePullBackOff   0          4m19s
```

Fig. 12 - Deubgging Pods

Accuknox_Practical_Token	Read, Write, Delete	Active	Manual	Jul 22, 2024 at 10:39:56	Never	⋮

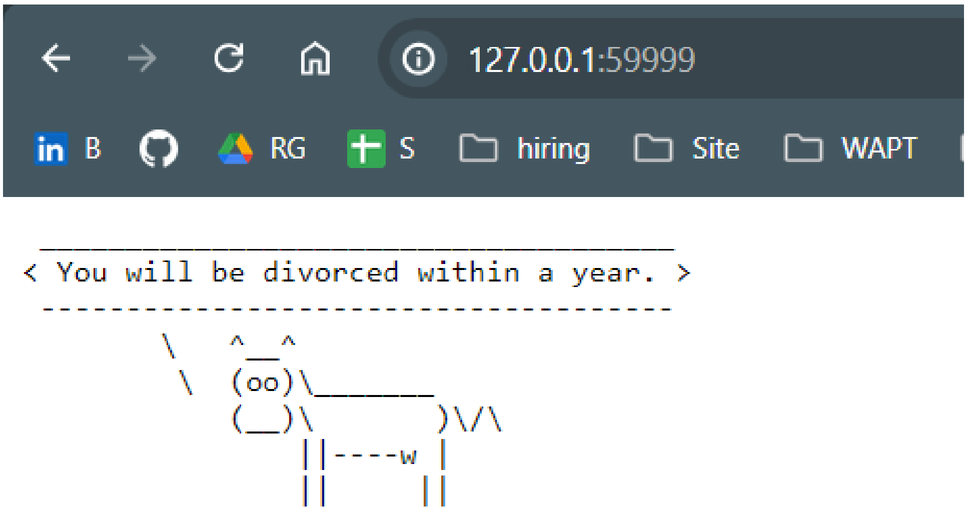
```
kubectl create secret docker-registry regcred --docker-server
```

8. Accessing Private Repository: Once the secret was created and attached to the cluster, it could access the private repository and use it in the deployment, allowing the pod to run successfully.



```
C:\Users\bkbhe\OneDrive\Desktop\Accuknox_practical\wisecow>kubectl get pods
NAME                                 READY   STATUS    RESTARTS   AGE
wisecow-deployment-6584dfb55b-7dxgs   1/1     Running   0          40s
wisecow-deployment-6584dfb55b-nlsgc   1/1     Running   0          17s

C:\Users\bkbhe\OneDrive\Desktop\Accuknox_practical\wisecow>minikube service wisecow-service --url
* service default/wisecow-service has no node port
! Services [default/wisecow-service] have type "ClusterIP" not meant to be exposed, however for local development minikube allows you to access this !
http://127.0.0.1:59999
! Because you are using a Docker driver on windows, the terminal needs to be open to run it.
```

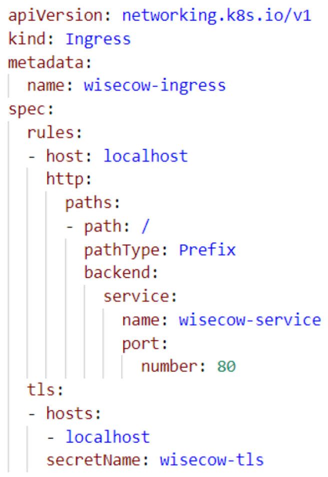
Fig. 13 - Error Resolved and Service started running



```
 _____________________________________
< You will be divorced within a year. >
 -------------------------------------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||
```

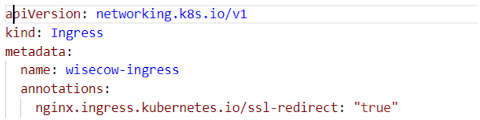
Fig. 14 - Accessing service by url mentioned in minikube

9. TLS Configuration Issue: Encountered an issue with the TLS certificate due to missing Nginx annotations in the ingress file, preventing access to the application via HTTPS. Resolved the issue by consulting documentation and various platforms, then reapplied the `ingress.yml` file.



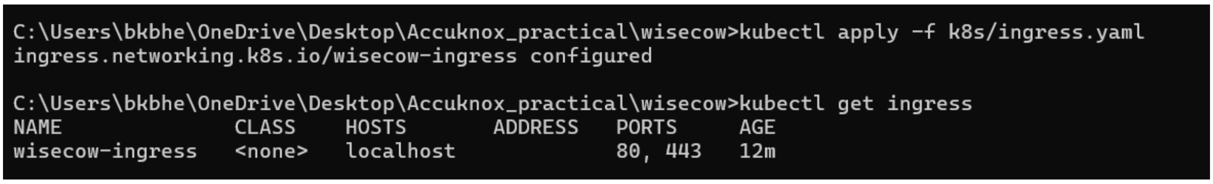
```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: wisecow-ingress
spec:
  rules:
  - host: localhost
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: wisecow-service
            port:
              number: 80
  tls:
  - hosts:
    - localhost
    secretName: wisecow-tls
```

Fig. 15 - File before annotations



```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: wisecow-ingress
  annotations:
    nginx.ingress.kubernetes.io/ssl-redirect: "true"
```

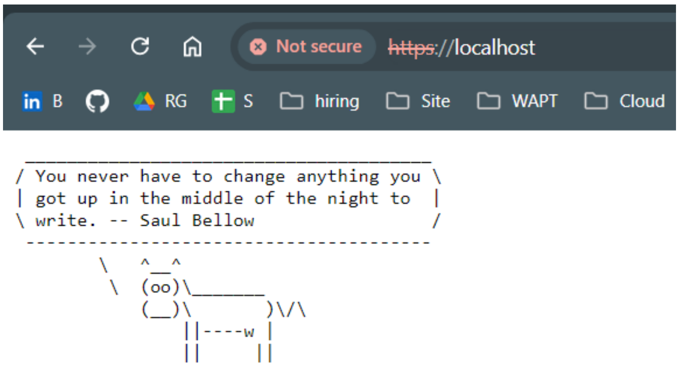
Fig. 16 - Annotations changes



```
C:\Users\bkbhe\OneDrive\Desktop\Accuknox_practical\wisecow>kubectl apply -f k8s/ingress.yaml
ingress.networking.k8s.io/wisecow-ingress configured

C:\Users\bkbhe\OneDrive\Desktop\Accuknox_practical\wisecow>kubectl get ingress
NAME              CLASS    HOSTS       ADDRESS   PORTS     AGE
wisecow-ingress   <none>   localhost             80, 443   12m
```

Fig. 17 - Ingress inforamtion

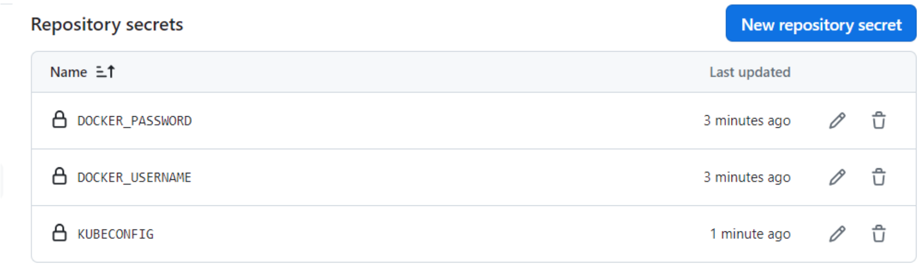
10. Successful HTTPS Access: Successfully accessed the server with HTTPS and TLS as required, with a self-signed certificate, resulting in a warning when running locally.



```
 ________________________________________
/ You never have to change anything you \
| got up in the middle of the night to  |
\ write. -- Saul Bellow                 /
 ----------------------------------------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||
```

Fig.18 - Access service using https

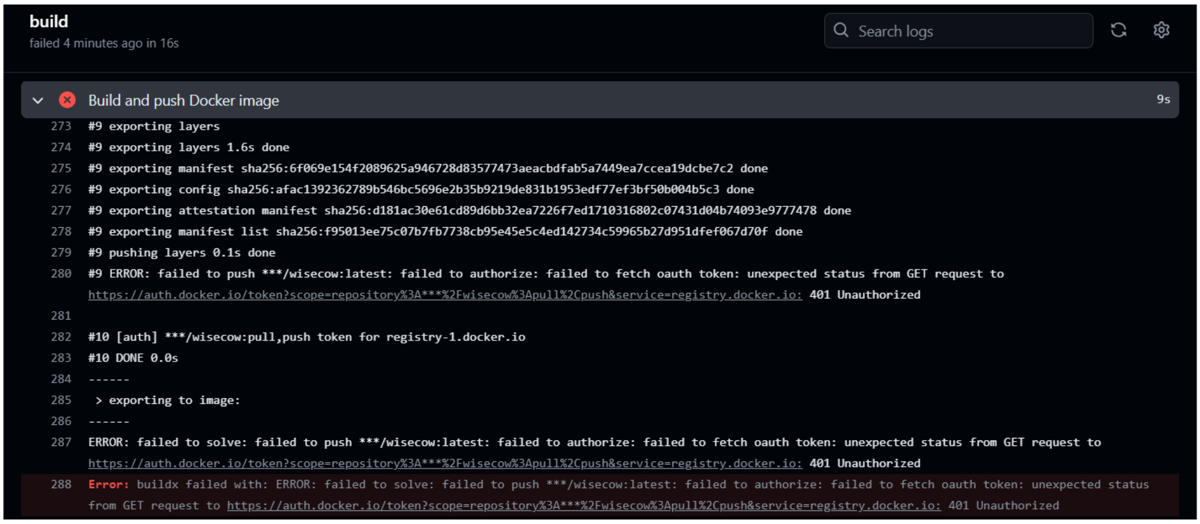
11. Building CI/CD Pipeline: Began building the CI/CD pipeline. Stored sensitive credentials using GitHub secrets instead of writing them in the file.



Name	Last updated
DOCKER_PASSWORD	3 minutes ago
DOCKER_USERNAME	3 minutes ago
KUBECONFIG	1 minute ago

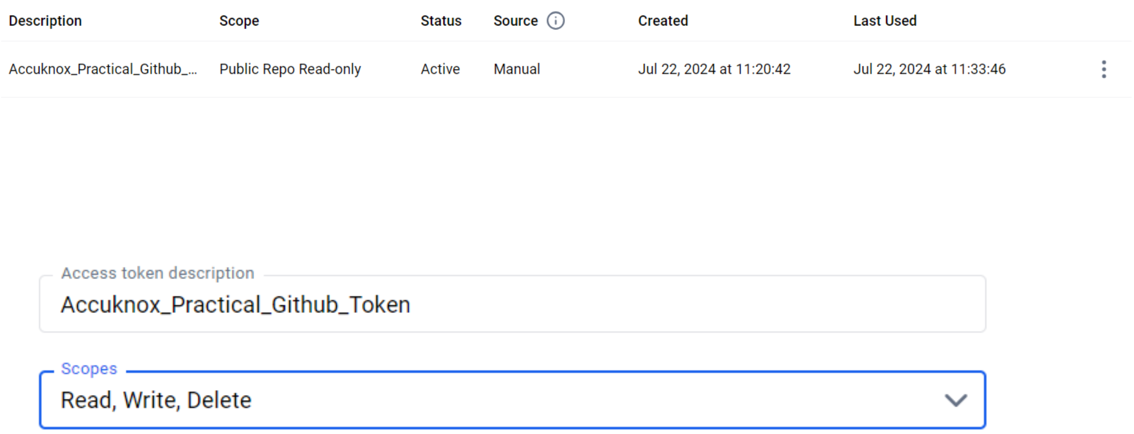
Fig. 19 - Store secrets in github

12. GitHub Actions: While pushing the image, initially provided only public access to the repository, causing a 401 error. Corrected the permissions on DockerHub, resolving the issue.



```
#9 exporting layers
#9 exporting layers 1.6s done
#9 exporting manifest sha256:6f069e154f2089625a946728d83577473aeacbdfab5a7449ea7ccea19dcbe7c2 done
#9 exporting config sha256:afac1392362789b546bc5696e2b35b9219de831b1953edf77ef3bf50b004b5c3 done
#9 exporting attestation manifest sha256:d181ac30e61cd89d6bb32ea7226f7ed1710316802c07431d04b74093e9777478 done
#9 exporting manifest list sha256:f95013ee75c07b7fb7738cb95e45e5c4ed142734c59965b27d951dfef067d70f done
#9 pushing layers 0.1s done
#9 ERROR: failed to push ***/wisecow:latest: failed to authorize: failed to fetch oauth token: unexpected status from GET request to
https://auth.docker.io/token?scope=repository%3A***%2Fwisecow%3Apull%2Cpush&service=registry.docker.io: 401 Unauthorized

#10 [auth] ***/wisecow:pull,push token for registry-1.docker.io
#10 DONE 0.0s
------
 > exporting to image:
------
ERROR: failed to solve: failed to push ***/wisecow:latest: failed to authorize: failed to fetch oauth token: unexpected status from GET request to
https://auth.docker.io/token?scope=repository%3A***%2Fwisecow%3Apull%2Cpush&service=registry.docker.io: 401 Unauthorized
Error: buildx failed with: ERROR: failed to solve: failed to push ***/wisecow:latest: failed to authorize: failed to fetch oauth token: unexpected status
from GET request to https://auth.docker.io/token?scope=repository%3A***%2Fwisecow%3Apull%2Cpush&service=registry.docker.io: 401 Unauthorized
```

Fig. 20 - 401 unauthorized Error



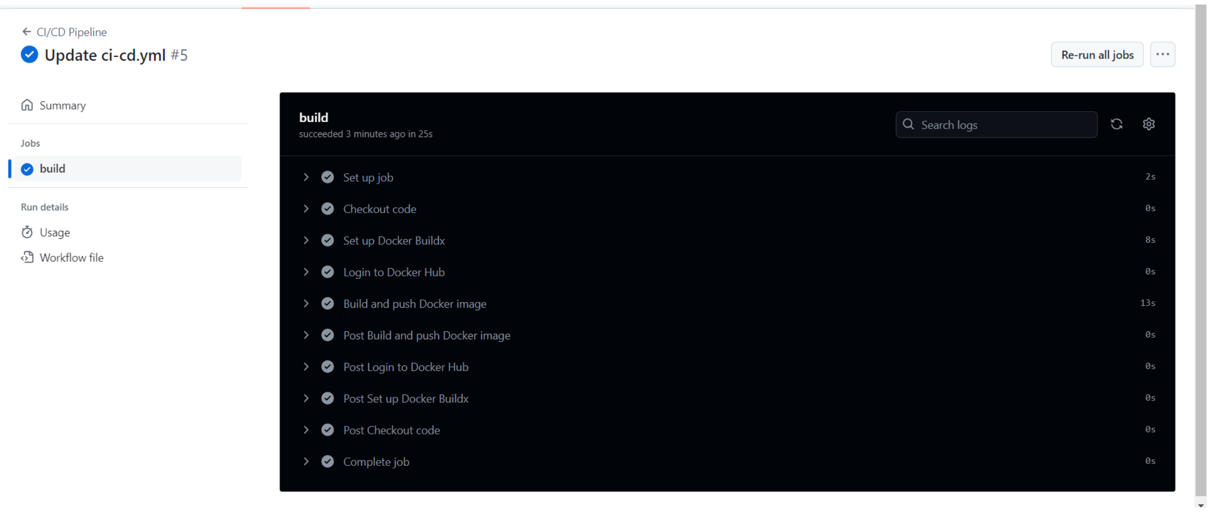
Description	Scope	Status	Source	Created	Last Used
Accuknox_Practical_Github_...	Public Repo Read-only	Active	Manual	Jul 22, 2024 at 11:20:42	Jul 22, 2024 at 11:33:46

Access token description: Accuknox_Practical_Github_Token

Scopes: Read, Write, Delete

Fig. 21 - Find issues in scope that why 401 generated

13. Successful CI/CD Pipeline: Successfully completed the CI/CD pipeline using GitHub Actions.



← CI/CD Pipeline

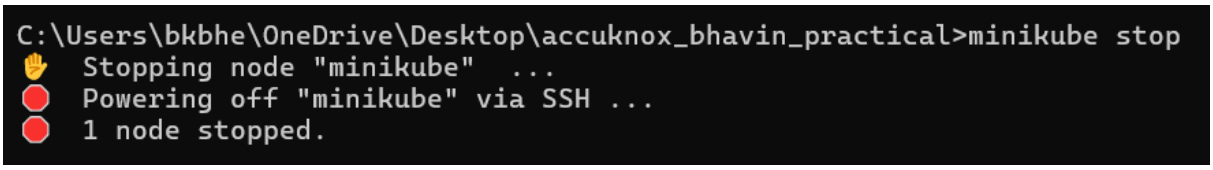
Update ci-cd.yml #5

build
succeeded 3 minutes ago in 25s

```
Set up job                         2s
Checkout code                      0s
Set up Docker Buildx               8s
Login to Docker Hub                0s
Build and push Docker image       13s
Post Build and push Docker image   0s
Post Login to Docker Hub           0s
Post Set up Docker Buildx          0s
Post Checkout code                 0s
Complete job                       0s
```

Fig. 22 - CICD Completed

14. Conclusion: **The task is now complete, providing a moment of relaxation and joy. This project was challenging, especially with configuring TLS certificates and writing Kubernetes files for the first time, but it was a valuable learning experience. My growth looks promising at Accuknox. One point to note is that the task description did not mention using any external cloud providers for hosting, so the system was tested locally after deployment and worked fine. Thank you.**



```
C:\Users\bkbhe\OneDrive\Desktop\accuknox_bhavin_practical>minikube stop
✋  Stopping node "minikube"  ...
🛑  Powering off "minikube" via SSH ...
🛑  1 node stopped.
```

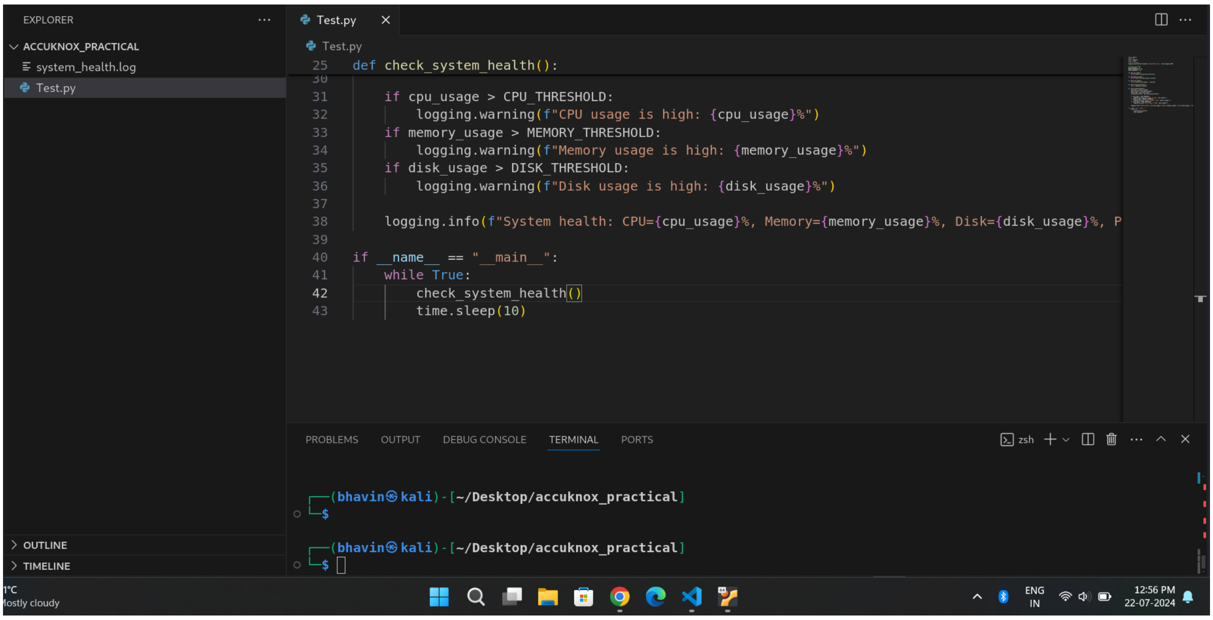
Fig. 23 - Stop Minikube

Practical - 2

Problem - 1 System Health Monitoring Script

Code Breakdown -

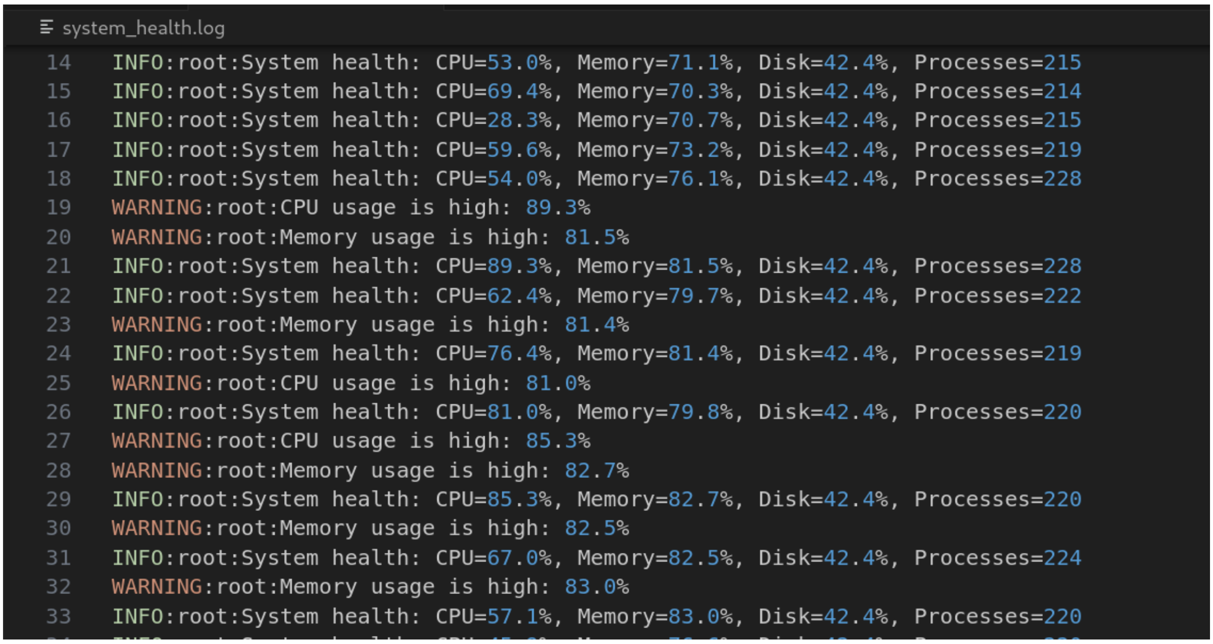
1. **Objective -** Script is designed to monitor system health by checking CPU usage, memory usage, disk space, and running processes, alerting when thresholds are exceeded.
2. **Setup logging -** Configures logging to write messages to a file named `system_health.log` with an information logging level.
3. **Defining Thresholds -** Sets threshold values for CPU, memory, and disk usage at 80%.
4. **Getting CPU, Memory , Disk Usage and No. Running Process**
5. **Checking System Health -** Calls functions to get current system metrics and stores them in variables.
6. **Logging Warnings for High Usage - Logs warnings if CPU, memory, or disk usage exceeds the defined thresholds.**
7. **Logging System Health Information**
8. **Continuously checks system health every 10 seconds and logs the results.**



```python
def check_system_health():

    if cpu_usage > CPU_THRESHOLD:
        logging.warning(f"CPU usage is high: {cpu_usage}%")
    if memory_usage > MEMORY_THRESHOLD:
        logging.warning(f"Memory usage is high: {memory_usage}%")
    if disk_usage > DISK_THRESHOLD:
        logging.warning(f"Disk usage is high: {disk_usage}%")

    logging.info(f"System health: CPU={cpu_usage}%, Memory={memory_usage}%, Disk={disk_usage}%, P

if __name__ == "__main__":
    while True:
        check_system_health()
        time.sleep(10)
```

```
┌──(bhavin㉿kali)-[~/Desktop/accuknox_practical]
└─$

┌──(bhavin㉿kali)-[~/Desktop/accuknox_practical]
└─$ 
```

Fig. - 1 Script for system monitoring



```
INFO:root:System health: CPU=53.0%, Memory=71.1%, Disk=42.4%, Processes=215
INFO:root:System health: CPU=69.4%, Memory=70.3%, Disk=42.4%, Processes=214
INFO:root:System health: CPU=28.3%, Memory=70.7%, Disk=42.4%, Processes=215
INFO:root:System health: CPU=59.6%, Memory=73.2%, Disk=42.4%, Processes=219
INFO:root:System health: CPU=54.0%, Memory=76.1%, Disk=42.4%, Processes=228
WARNING:root:CPU usage is high: 89.3%
WARNING:root:Memory usage is high: 81.5%
INFO:root:System health: CPU=89.3%, Memory=81.5%, Disk=42.4%, Processes=228
INFO:root:System health: CPU=62.4%, Memory=79.7%, Disk=42.4%, Processes=222
WARNING:root:Memory usage is high: 81.4%
INFO:root:System health: CPU=76.4%, Memory=81.4%, Disk=42.4%, Processes=219
WARNING:root:CPU usage is high: 81.0%
INFO:root:System health: CPU=81.0%, Memory=79.8%, Disk=42.4%, Processes=220
WARNING:root:CPU usage is high: 85.3%
WARNING:root:Memory usage is high: 82.7%
INFO:root:System health: CPU=85.3%, Memory=82.7%, Disk=42.4%, Processes=220
WARNING:root:Memory usage is high: 82.5%
INFO:root:System health: CPU=67.0%, Memory=82.5%, Disk=42.4%, Processes=224
WARNING:root:Memory usage is high: 83.0%
INFO:root:System health: CPU=57.1%, Memory=83.0%, Disk=42.4%, Processes=220
```

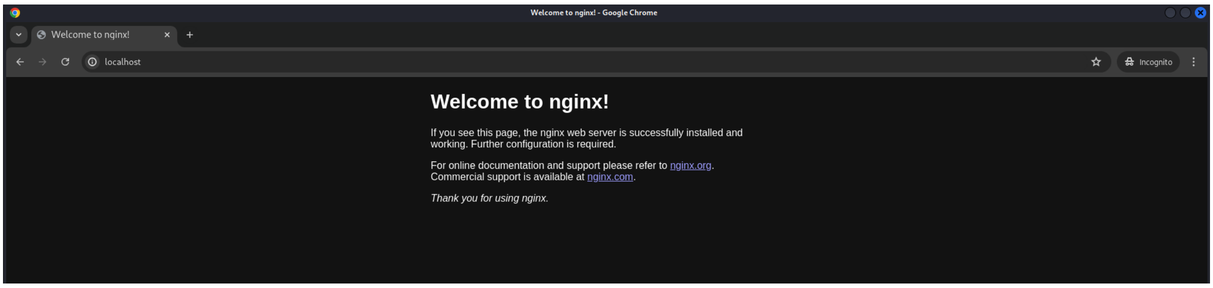
Fig. - 2 Logging cpu, memery threshold when go above 80%

Problem - 2 Log File Analyzer

- Create a script that analyzes web server logs (e.g., Apache, Nginx) for common patterns such as the number of 404 errors, the most requested pages, or IP addresses with the most requests. The script should output a summarized report.

1. Imports the `re` module for regular expressions and `defaultdict` from the collections module to simplify dictionary creation and management.
2. Defines patterns to match IP addresses, status codes, and request URIs in the log lines.
3. Sets the path to the Nginx log file to be analyzed.
4. Creates dictionaries to count occurrences of IP addresses, status codes, and requested pages.
5. Opens the log file for reading and iterates through each line.
6. Uses a regular expression to find IP addresses, Staus codes, requested URLs in each log line and counts their occurrences.

7. Prints the results of the log analysis, including the top 10 IP addresses with the most requests, status code counts, the top 10 most requested pages, and the count of 404 errors.



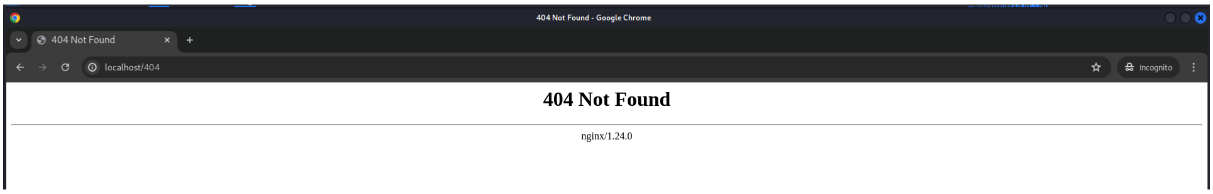
Welcome to nginx!

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.

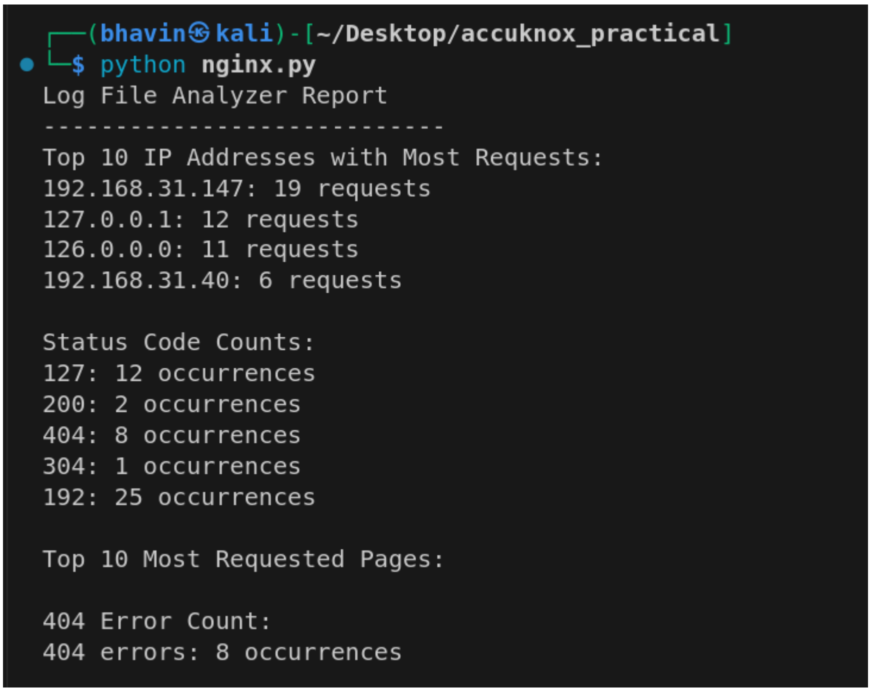
Fig. 1 - Running Nginx server



404 Not Found

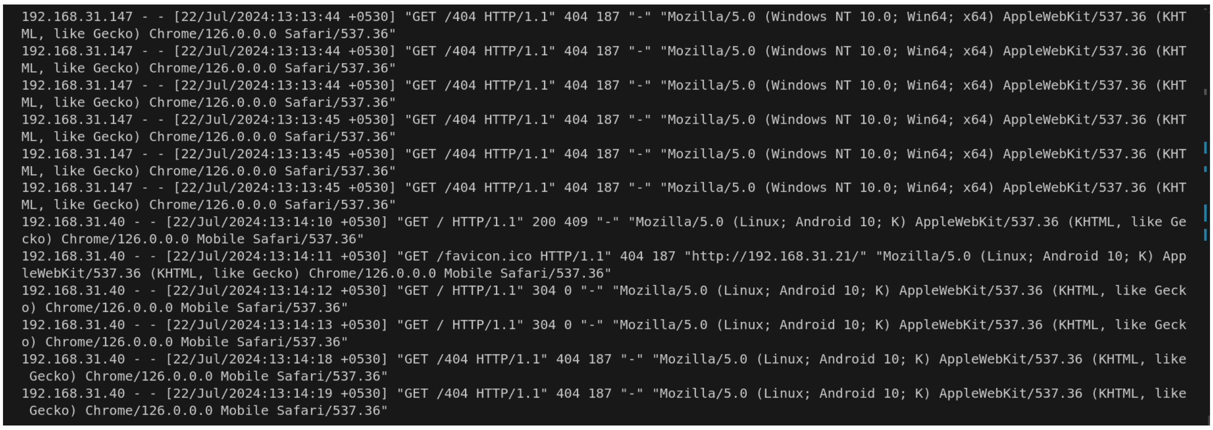
nginx/1.24.0

Fig. 2 - Requesting different endpoint to generate logs



```
┌──(bhavin㉿kali)-[~/Desktop/accuknox_practical]
└─$ python nginx.py
Log File Analyzer Report
----------------------------
Top 10 IP Addresses with Most Requests:
192.168.31.147: 19 requests
127.0.0.1: 12 requests
126.0.0.0: 11 requests
192.168.31.40: 6 requests

Status Code Counts:
127: 12 occurrences
200: 2 occurrences
404: 8 occurrences
304: 1 occurrences
192: 25 occurrences

Top 10 Most Requested Pages:

404 Error Count:
404 errors: 8 occurrences
```

Fig. 3 - Python output report



```
192.168.31.147 - - [22/Jul/2024:13:13:44 +0530] "GET /404 HTTP/1.1" 404 187 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHT
ML, like Gecko) Chrome/126.0.0.0 Safari/537.36"
192.168.31.147 - - [22/Jul/2024:13:13:44 +0530] "GET /404 HTTP/1.1" 404 187 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHT
ML, like Gecko) Chrome/126.0.0.0 Safari/537.36"
192.168.31.147 - - [22/Jul/2024:13:13:44 +0530] "GET /404 HTTP/1.1" 404 187 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHT
ML, like Gecko) Chrome/126.0.0.0 Safari/537.36"
192.168.31.147 - - [22/Jul/2024:13:13:45 +0530] "GET /404 HTTP/1.1" 404 187 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHT
ML, like Gecko) Chrome/126.0.0.0 Safari/537.36"
192.168.31.147 - - [22/Jul/2024:13:13:45 +0530] "GET /404 HTTP/1.1" 404 187 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHT
ML, like Gecko) Chrome/126.0.0.0 Safari/537.36"
192.168.31.147 - - [22/Jul/2024:13:13:45 +0530] "GET /404 HTTP/1.1" 404 187 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHT
ML, like Gecko) Chrome/126.0.0.0 Safari/537.36"
192.168.31.40 - - [22/Jul/2024:13:14:10 +0530] "GET / HTTP/1.1" 200 409 "-" "Mozilla/5.0 (Linux; Android 10; K) AppleWebKit/537.36 (KHTML, like Ge
cko) Chrome/126.0.0.0 Mobile Safari/537.36"
192.168.31.40 - - [22/Jul/2024:13:14:11 +0530] "GET /favicon.ico HTTP/1.1" 404 187 "http://192.168.31.21/" "Mozilla/5.0 (Linux; Android 10; K) App
leWebKit/537.36 (KHTML, like Gecko) Chrome/126.0.0.0 Mobile Safari/537.36"
192.168.31.40 - - [22/Jul/2024:13:14:12 +0530] "GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (Linux; Android 10; K) AppleWebKit/537.36 (KHTML, like Geck
o) Chrome/126.0.0.0 Mobile Safari/537.36"
192.168.31.40 - - [22/Jul/2024:13:14:13 +0530] "GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (Linux; Android 10; K) AppleWebKit/537.36 (KHTML, like Geck
o) Chrome/126.0.0.0 Mobile Safari/537.36"
192.168.31.40 - - [22/Jul/2024:13:14:18 +0530] "GET /404 HTTP/1.1" 404 187 "-" "Mozilla/5.0 (Linux; Android 10; K) AppleWebKit/537.36 (KHTML, like
 Gecko) Chrome/126.0.0.0 Mobile Safari/537.36"
192.168.31.40 - - [22/Jul/2024:13:14:19 +0530] "GET /404 HTTP/1.1" 404 187 "-" "Mozilla/5.0 (Linux; Android 10; K) AppleWebKit/537.36 (KHTML, like
 Gecko) Chrome/126.0.0.0 Mobile Safari/537.36"
```

Fig. 4 - Nginx access.log file

Thank you, AccuKnox team, for providing such a challenging task. Whether I am selected or not, what truly matters is that I successfully completed the task and learned a great deal in the process. I appreciate the opportunity to grow and explore various topics. Thank you for taking the time to read this report.