DAY 3: KUBERNETES

Troubleshooting Minikube and Docker Issues

1. Cloning the Repository

Run the following command to clone the repository:

```
git clone https://github.com/PadmavathyNarayanan/kubernetes.git
```

2. Deleting and Purging Minikube

To reset Minikube, run:

```
minikube delete --all --purge
```

This deletes all profiles and removes the Minikube directory.

3. Restarting Docker

Stop and restart Docker using:

```
sudo systemctl stop docker
sudo systemctl start docker
```

If there are permission issues, try killing Docker processes:

```
sudo pkill -f docker
```

4. Cleaning Up Docker Containers

Kill all running containers:

```
docker kill $(docker ps -q)
```

Remove all stopped containers:

```
docker rm -f $(docker ps -aq)
```

5. Pruning Unused Docker Data

To free up space, remove unused Docker objects:

```
sudo docker system prune -a --volumes –f
```

6. Checking for Processes Running on Port 8080

Check which process is using port 8080:

```
sudo netstat -tulnp | grep ":8080"
```

Kill the process by replacing <PID> with the actual process ID:

```
sudo kill -9 <PID>
```

If the process restarts immediately with a new PID, repeat the above step.

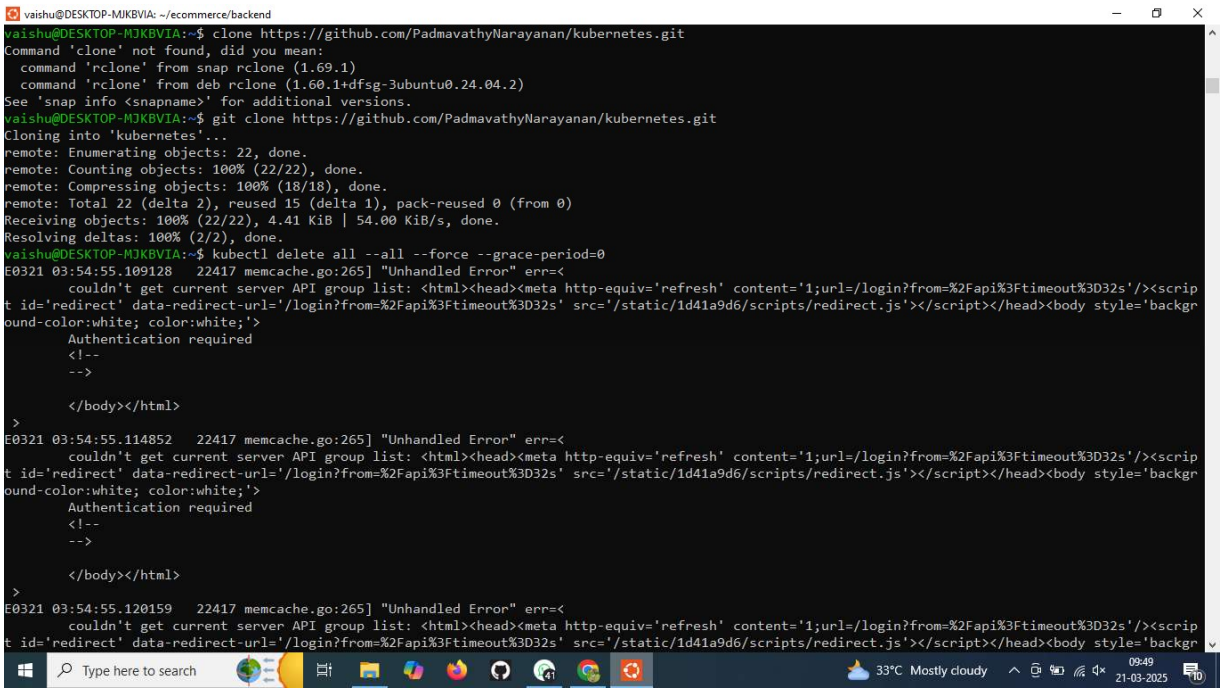
7. Final Steps

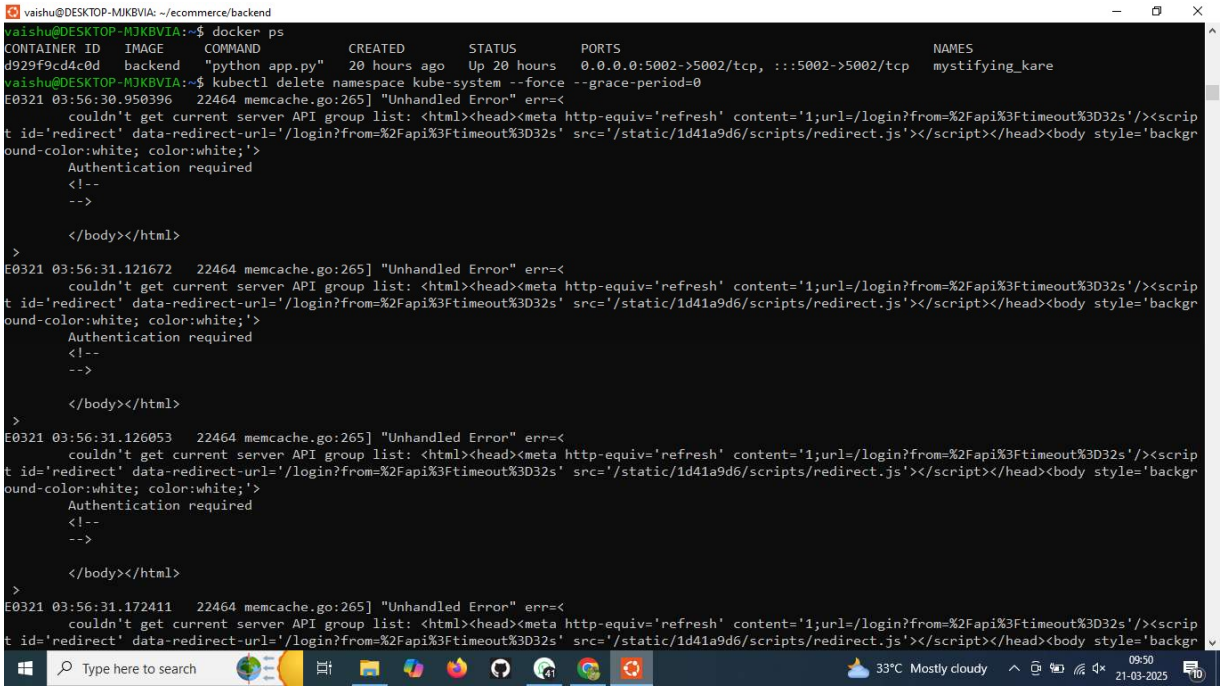
After resolving the issues, restart Minikube:

```
minikube start
```

Conclusion

By following these steps, you can effectively troubleshoot Minikube and Docker-related issues, ensuring a smooth development and deployment process. Always verify that Docker is running correctly and that no conflicting processes are occupying essential ports before starting Minikube.





Docker, Minikube, and Kubernetes Setup on Ubuntu

Prerequisites

- Ubuntu 24.04 (or any other supported version)
- Internet connection
- sudo privileges

Step 1: Update System Packages

sudo apt update

Step 2: Install Docker

```
sudo apt install docker.io -y
```

Verify installation:

```
docker --version
```

Enable and start Docker:

```
sudo systemctl start docker
sudo systemctl enable docker
```

Step 3: Add User to Docker Group

To run Docker without sudo:

```
sudo usermod -aG docker $USER
newgrp docker
```

Verify Docker is running:

```
docker ps
```

Step 4: Install Minikube

Download and install Minikube:

```
curl -LO
```

https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64

sudo install minikube-linux-amd64 /usr/local/bin/minikube

Verify installation:

minikube version

Step 5: Install Kubectl

curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"

chmod +x kubectl

sudo mv kubectl /usr/local/bin/

Verify installation:

kubectl version –client

Step 6: Start Minikube

minikube start

If the download fails, restart Docker:

sudo systemctl restart docker

Verify the Minikube node:

kubectl get nodes

Step 7: Check Running Containers

To list running Docker containers:

```
docker ps
```

Troubleshooting

1. Minikube Fails to Start

If Minikube fails due to missing images:

```
minikube delete --all --purge
minikube start
```

2. Port Conflicts

If ports (e.g., 8080) are already in use:

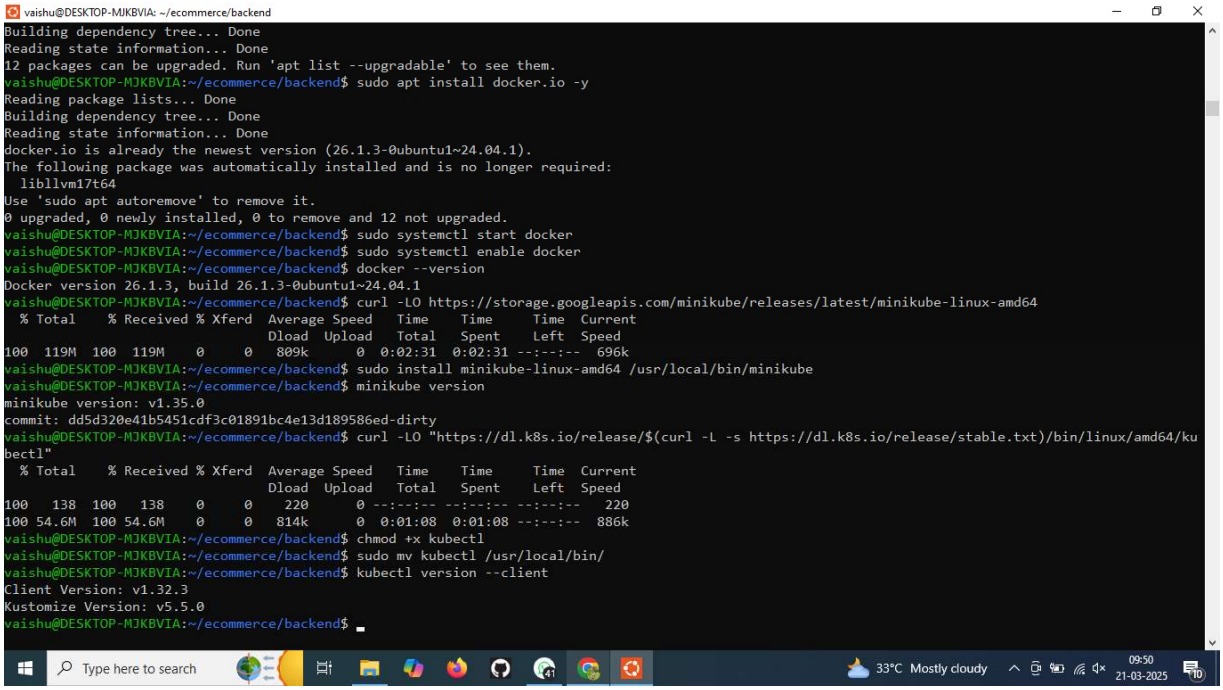
```
sudo netstat -tulnp | grep ":8080"
sudo kill -9 <PID>
```

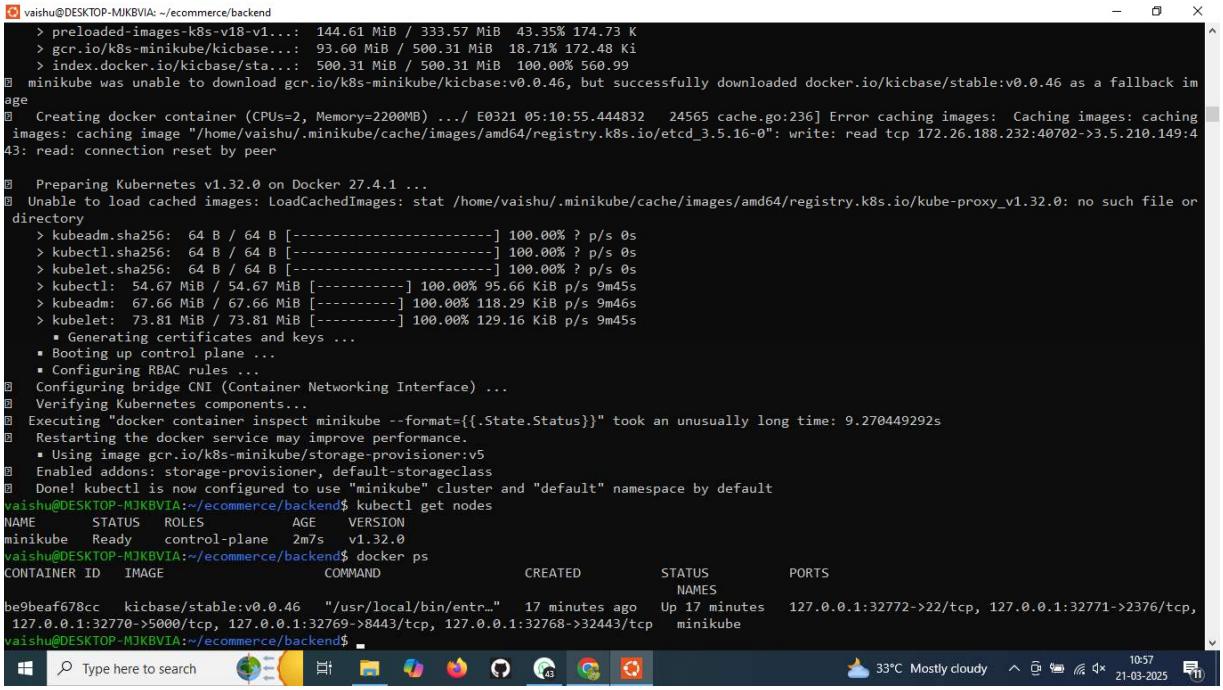
3. Restart Services

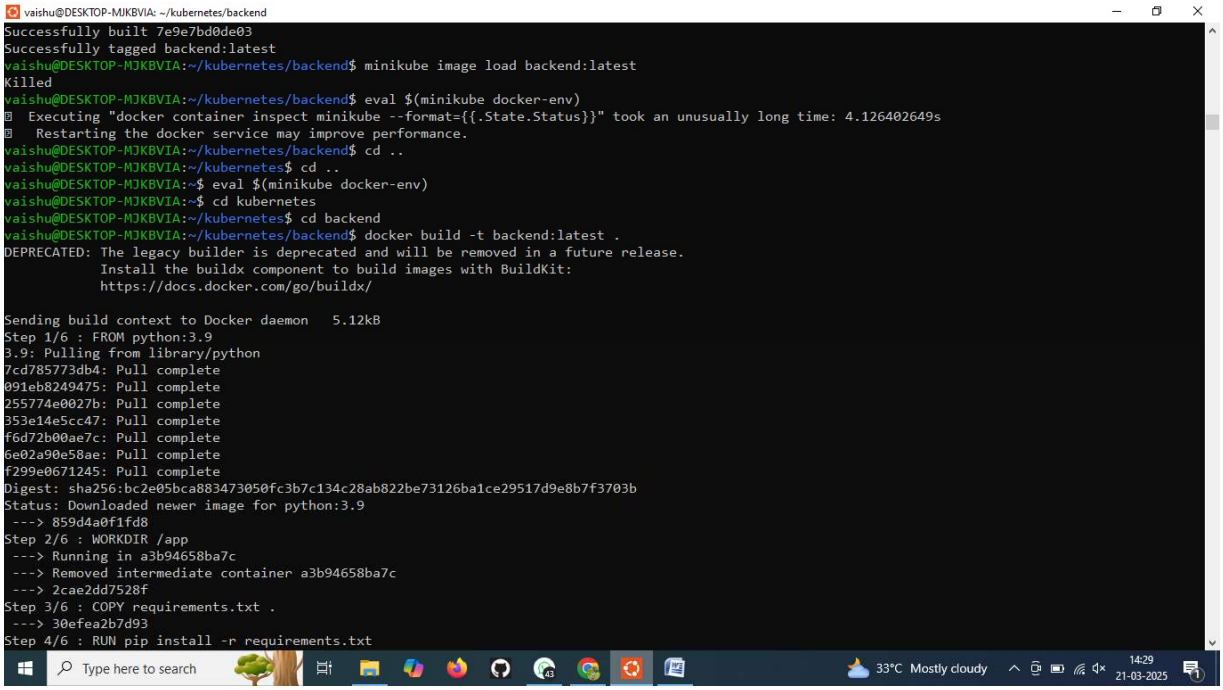
```
sudo systemctl restart docker
minikube stop && minikube delete --all --purge
minikube start
```

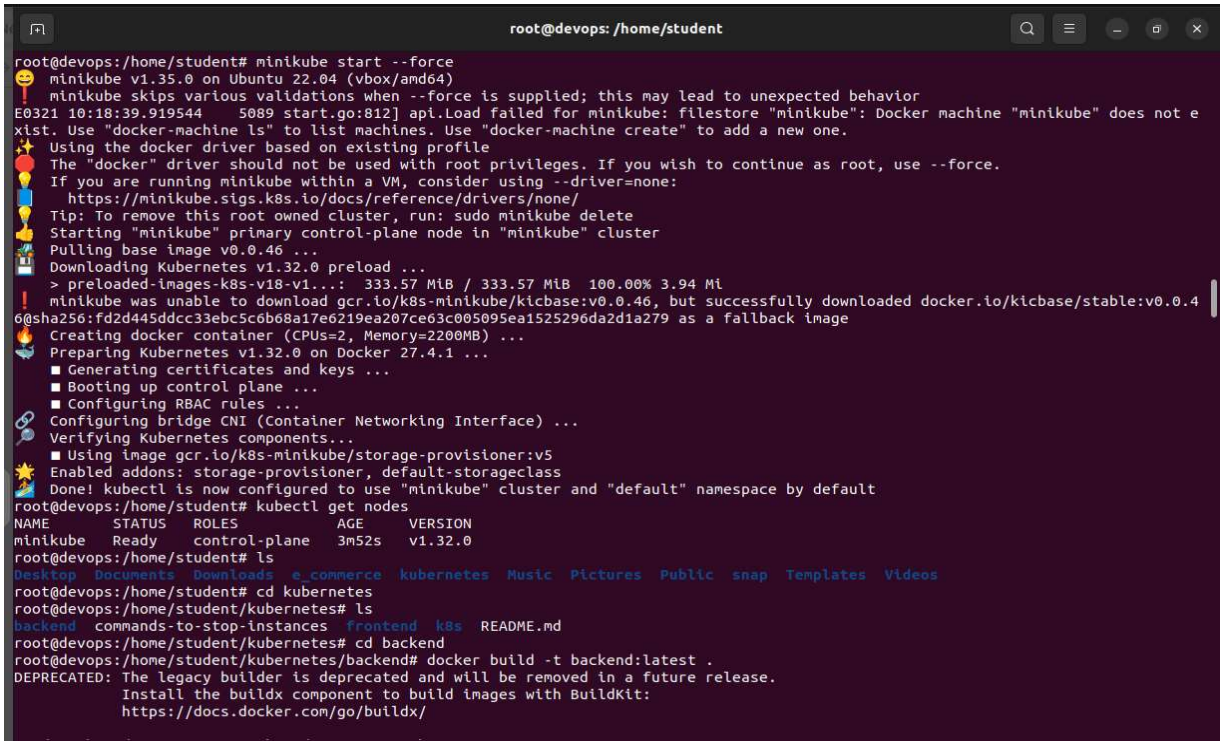
Conclusion

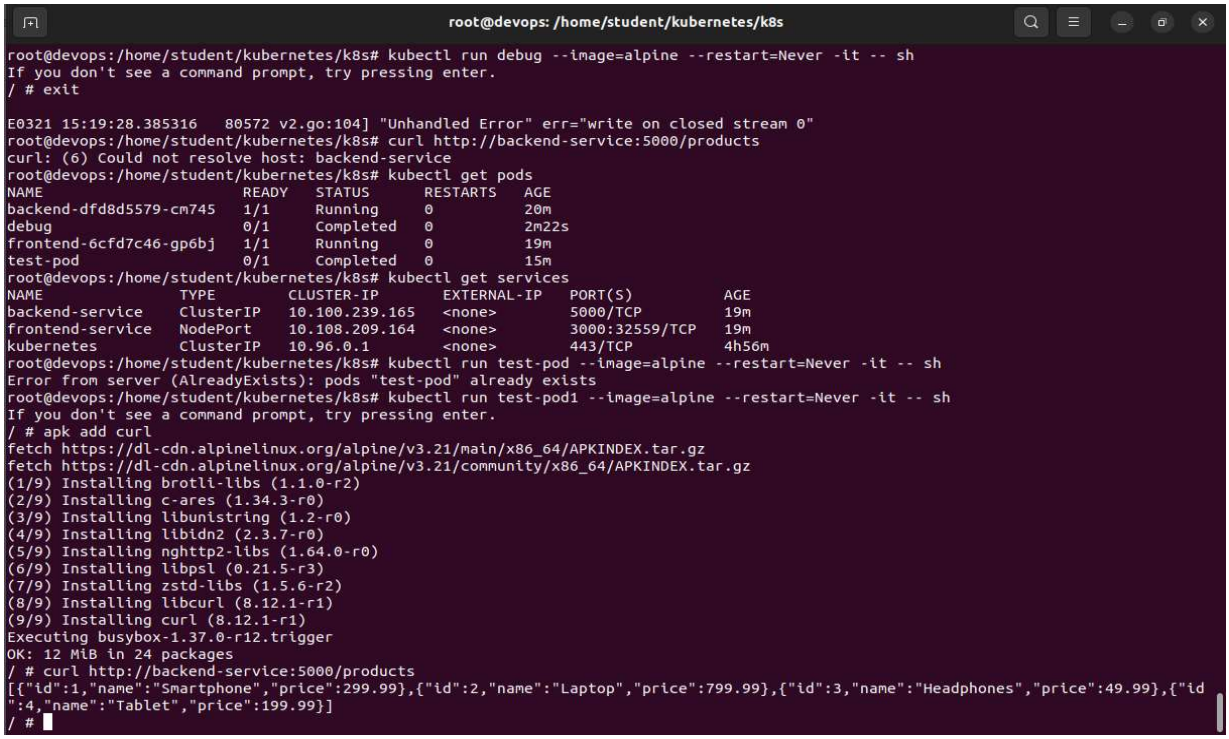
You have successfully set up Docker, Minikube, and Kubernetes on Ubuntu. Now you can deploy and manage containerized applications efficiently!






vaishu@DESKTOP-MJKBVIA: ~/kubernetes/backend
Successfully built 7e9e7bd0de03
Successfully tagged backend:latest
vaishu@DESKTOP-MJKBVIA:~/kubernetes/backend$ minikube image load backend:latest
Killed
vaishu@DESKTOP-MJKBVIA:~/kubernetes/backend$ eval $(minikube docker-env)
Executing "docker container inspect minikube --format={{.State.Status}}" took an unusually long time: 4.126402649s
Restarting the docker service may improve performance.
vaishu@DESKTOP-MJKBVIA:~/kubernetes/backend$ cd ..
vaishu@DESKTOP-MJKBVIA:~/kubernetes$ cd ..
vaishu@DESKTOP-MJKBVIA:~$ eval $(minikube docker-env)
vaishu@DESKTOP-MJKBVIA:~$ cd kubernetes
vaishu@DESKTOP-MJKBVIA:~/kubernetes$ cd backend
vaishu@DESKTOP-MJKBVIA:~/kubernetes/backend$ docker build -t backend:latest .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
Install the buildx component to build images with BuildKit:
https://docs.docker.com/go/buildx/
Sending build context to Docker daemon 5.12kB
Step 1/6 : FROM python:3.9
3.9: Pulling from library/python
7cd785773db4: Pull complete
091eb8249475: Pull complete
255774e0027b: Pull complete
353e14e5cc47: Pull complete
f6d72b00ae7c: Pull complete
6e02a90e58ae: Pull complete
f299e0671245: Pull complete
Digest: sha256:bc2e05bca883473050fc3b7c134c28ab822be73126ba1ce29517d9e8b7f3703b
Status: Downloaded newer image for python:3.9
---> 859d4a0f1fd8
Step 2/6 : WORKDIR /app
---> Running in a3b94658ba7c
---> Removed intermediate container a3b94658ba7c
---> 2cae2dd7528f
Step 3/6 : COPY requirements.txt .
---> 30efea2b7d93
Step 4/6 : RUN pip install -r requirements.txt
Type here to search
33°C Mostly cloudy
14:29
21-03-2025


root@devops: /home/student
root@devops:/home/student# minikube start --force
minikube v1.35.0 on Ubuntu 22.04 (vbox/amd64)
minikube skips various validations when --force is supplied; this may lead to unexpected behavior
E0321 10:18:39.919544 5089 start.go:812] api.Load failed for minikube: filestore "minikube": Docker machine "minikube" does not exist. Use "docker-machine ls" to list machines. Use "docker-machine create" to add a new one.
Using the docker driver based on existing profile
The "docker" driver should not be used with root privileges. If you wish to continue as root, use --force.
If you are running minikube within a VM, consider using --driver=none:
https://minikube.sigs.k8s.io/docs/reference/drivers/none/
Tip: To remove this root owned cluster, run: sudo minikube delete
Starting "minikube" primary control-plane node in "minikube" cluster
Pulling base image v0.0.46 ...
Downloading Kubernetes v1.32.0 preload ...
> preloaded-images-k8s-v18-v1...: 333.57 MiB / 333.57 MiB 100.00% 3.94 Mi
minikube was unable to download gcr.io/k8s-minikube/kicbase:v0.0.46, but successfully downloaded docker.io/kicbase/stable:v0.0.46@sha256:fd2d445ddcc33ebc5c6b68a17e6219ea207ce63c005095ea1525296da2d1a279 as a fallback image
Creating docker container (CPUs=2, Memory=2200MB) ...
Preparing Kubernetes v1.32.0 on Docker 27.4.1 ...
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
root@devops:/home/student# kubectl get nodes
NAME STATUS ROLES AGE VERSION
minikube Ready control-plane 3m52s v1.32.0
root@devops:/home/student# ls
Desktop Documents Downloads e_commerce kubernetes Music Pictures Public snap Templates Videos
root@devops:/home/student# cd kubernetes
root@devops:/home/student/kubernetes# ls
backend commands-to-stop-instances frontend k8s README.md
root@devops:/home/student/kubernetes# cd backend
root@devops:/home/student/kubernetes/backend# docker build -t backend:latest .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
Install the buildx component to build images with BuildKit:
https://docs.docker.com/go/buildx/



Kubernetes Backend Service Debugging

Commands Run

1. **Start a Debug Pod**

   kubectl run debug --image=alpine --restart=Never -it -- sh

2. **Check Backend Service Connectivity**

   curl http://backend-service:5000/products

3. **List Running Pods**

   kubectl get pods

4. **List Services**

   kubectl get services

5. **Start a Test Pod for Debugging**

   kubectl run test-pod1 --image=alpine --restart=Never -it -- sh

6. **Install Curl in Alpine Linux**

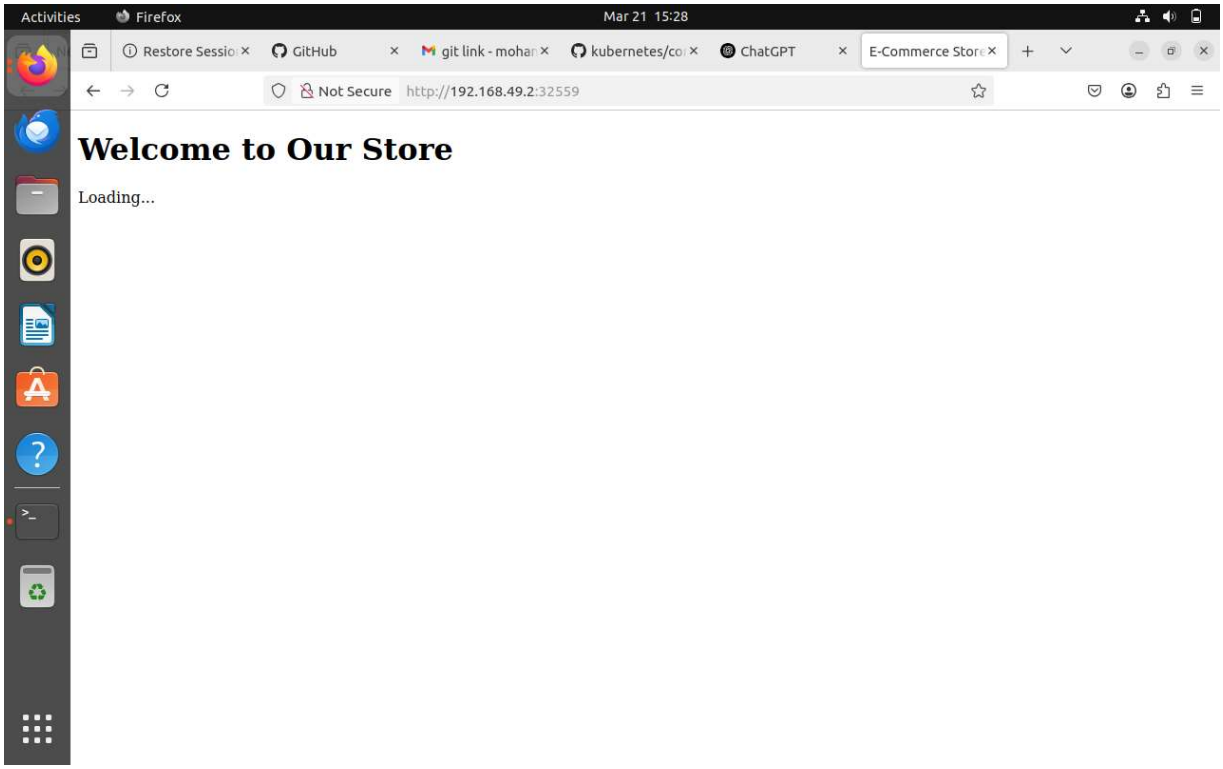
   apk add curl

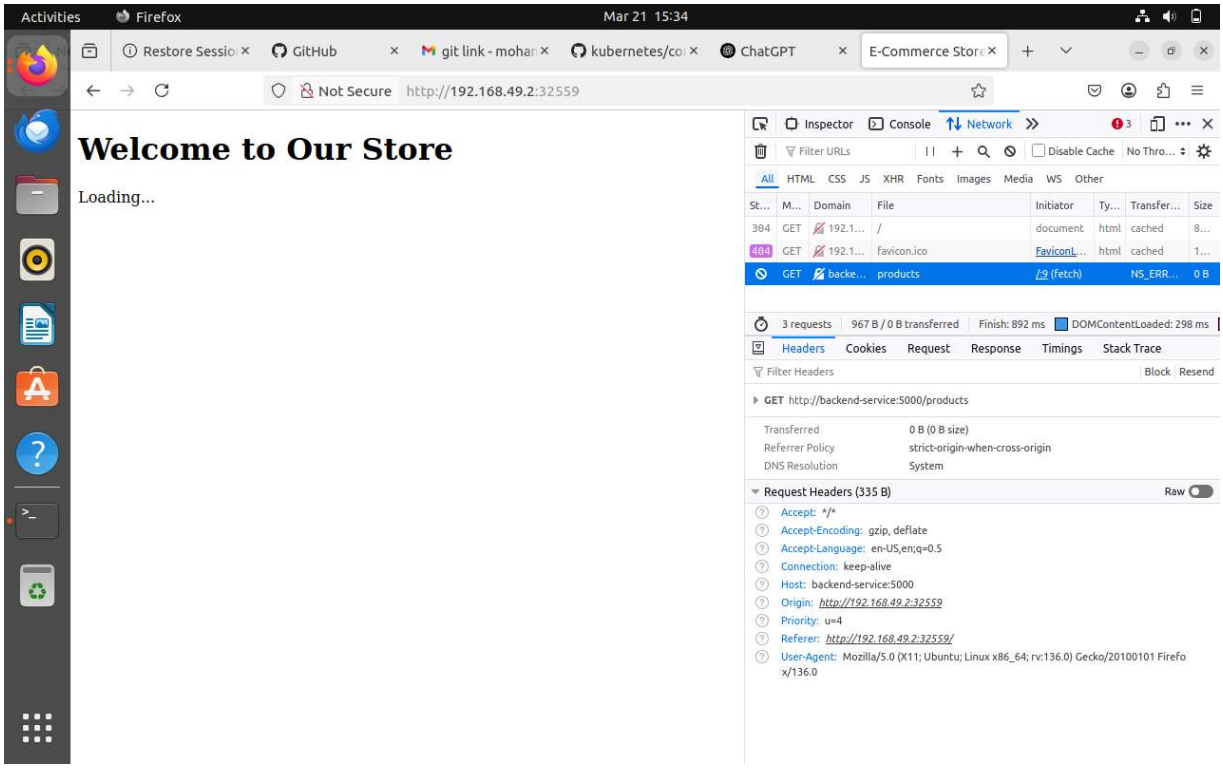
7. **Test API Response from Backend**

   curl http://backend-service:5000/products

Conclusion

The backend service was initially unreachable due to a DNS resolution issue. However, after verifying the pod and service configurations, it was confirmed that the backend service was correctly deployed and responding within the cluster. The issue may be related to the frontend's inability to resolve the backend service name correctly. Possible solutions include checking DNS settings, ensuring proper service discovery, and updating the frontend to use the correct internal service name or ClusterIP.


Activities
Firefox
Mar 21 15:28
Restore Sessio
GitHub
git link - mohar
kubernetes/co
ChatGPT
E-Commerce Store
Not Secure
http://192.168.49.2:32559
Welcome to Our Store
Loading...


Activities
Firefox
Mar 21 15:34
Restore Sessio
GitHub
git link - mohar
kubernetes/co
ChatGPT
E-Commerce Store
Not Secure
http://192.168.49.2:32559
Welcome to Our Store
Loading...
Inspector
Console
Network
Filter URLs
Disable Cache
All HTML CSS JS XHR Fonts Images Media WS Other
St... M... Domain File Initiator Ty... Transfer... Size
304 GET 192.1... / document html cached 8...
404 GET 192.1... favicon.ico FaviconL... html cached 1...
GET backe... products (fetch) NS_ERR... 0 B
3 requests 967 B / 0 B transferred Finish: 892 ms DOMContentLoaded: 298 ms
Headers Cookies Request Response Timings Stack Trace
Filter Headers Block Resend
GET http://backend-service:5000/products
Transferred 0 B (0 B size)
Referrer Policy strict-origin-when-cross-origin
DNS Resolution System
Request Headers (335 B) Raw
Accept: */*
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.5
Connection: keep-alive
Host: backend-service:5000
Origin: http://192.168.49.2:32559
Priority: u=4
Referer: http://192.168.49.2:32559/
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:136.0) Gecko/20100101 Firefox/136.0