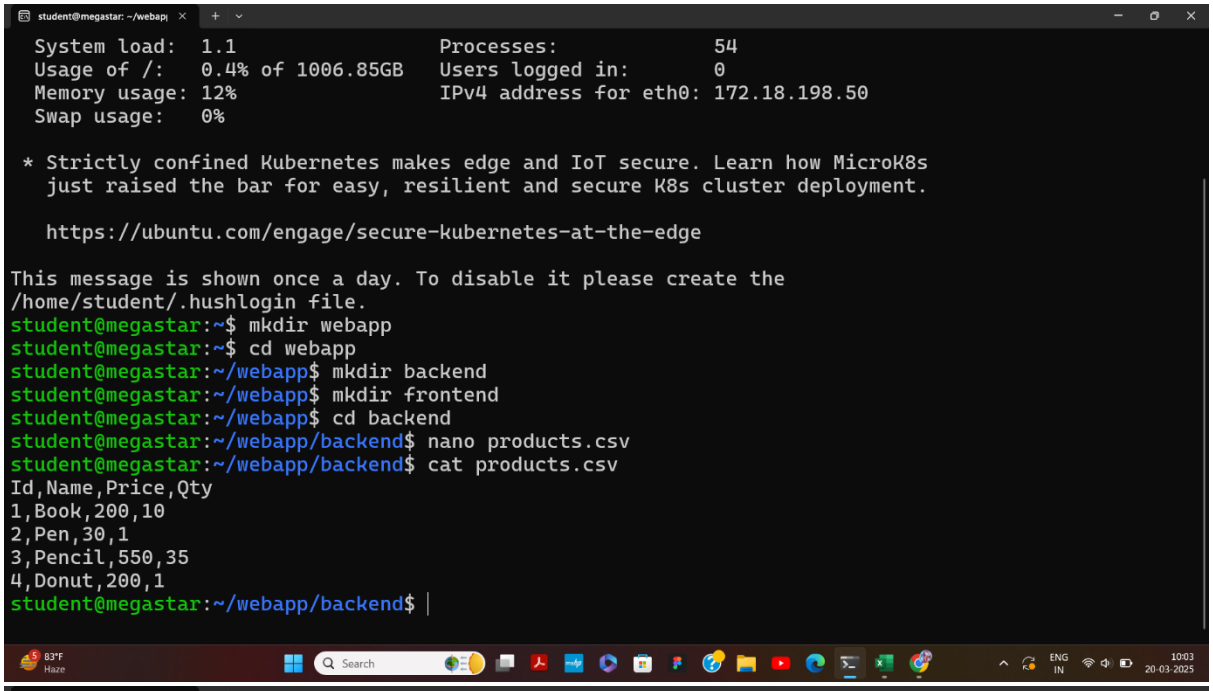
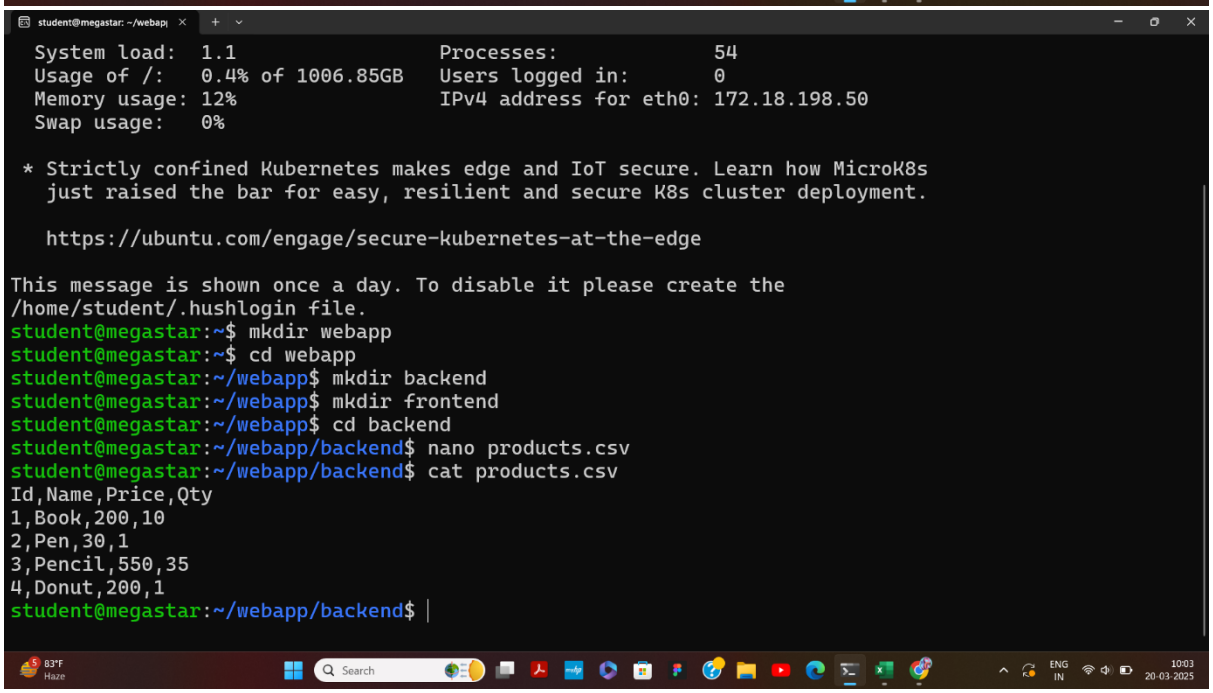
Day 3 and Day 4: Kubernetes

Create a directory ‘Web app’ and its required folders and files

Create a **products.csv** file and **app.py**





```
student@mcacc1-6:~/e-commerce/backend$ cat app.py
import pandas as pd
from flask import Flask

app = Flask(__name__)

@app.route("/products", method=['GET'])
def read_data():
        df = pd.read_csv("./products.csv")
        print(df.head())
        json_data = df.to_json()
        print(json_data)
        return json_data

if __name__ == "__main__":
        app.run(host="0.0.0.0", port=7000)
```

Ensure that the CSV file is read and correctly parsed into **JSON format**.

To verify the **available port numbers**

```
Successfully installed Jinja2-3.1.6 MarkupSafe-3.0.2 Werkzeug-3.1.3 blinker-1.9.0 click-8.1.8 flask-3.1.0 itsdangerous-2.2.0
WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system package manager. It is recommended to use a virtual environment instead: https://pip.pypa.io/warnings/venv

[notice] A new release of pip is available: 24.0 -> 25.0.1
[notice] To update, run: pip install --upgrade pip
 ---> Removed intermediate container e991af9c5d82
 ---> 0f89a5b62654
Step 5/7 : COPY . .
 ---> 1a38ff877e73
Step 6/7 : EXPOSE 7000
 ---> Running in 174f5af258d2
 ---> Removed intermediate container 174f5af258d2
 ---> da35386fca27
Step 7/7 : CMD ["python","app.py"]
 ---> Running in 8d40b29e7ca5
 ---> Removed intermediate container 8d40b29e7ca5
 ---> 0978980e4ec3
Successfully built 0978980e4ec3
Successfully tagged backend:latest
student@megastar:~/webapp/backend$ sudo docker images
REPOSITORY                TAG       IMAGE ID       CREATED          SIZE
backend                   latest    0978980e4ec3   18 seconds ago   1.03GB
sugumar3232/docker-app    latest    943c48aa9e31   20 hours ago     1.04GB
test                      latest    77d678a7a322   40 hours ago     1.03GB
<none>                    <none>    c54a35aa3814   40 hours ago     1.03GB
<none>                    <none>    1e948401f938   40 hours ago     1.03GB
<none>                    <none>    a1c9c7632711   43 hours ago     1.03GB
python                    3.11      18c0f2265fd9   3 months ago     1.01GB
student@megastar:~/webapp/backend$
```

Create **requirements.txt** file

The `requirements.txt` file is used in Python projects to list all the dependencies (packages) that the application needs to run.

Create **docker-compose.yml** file

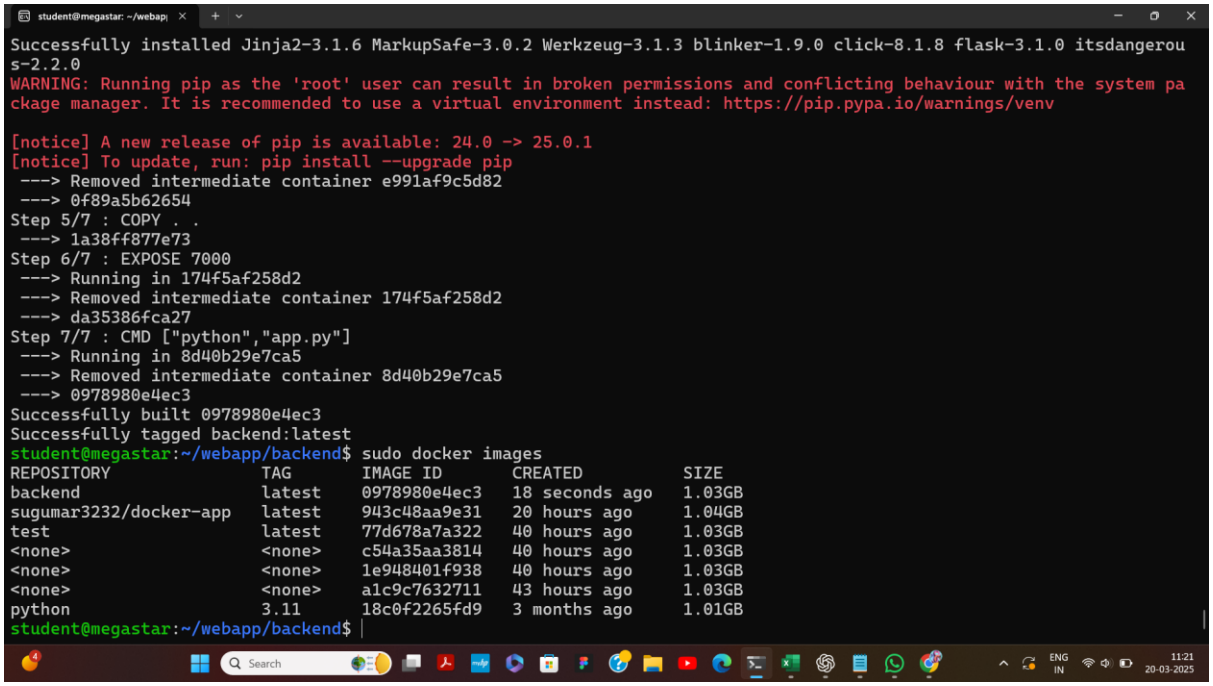
`docker-compose.yml` is a YAML configuration file used to define and run multi-container Docker applications.

```
student@mcacc1-6:~/e-commerce/backend$ nano docker-compose.yml
student@mcacc1-6:~/e-commerce/backend$ cat docker-compose.yml
version: '3.8'

services:
  web:
    build: .
    ports:
      - "7000:7000"
    volumes:
      - .:/app
    restart: always
```

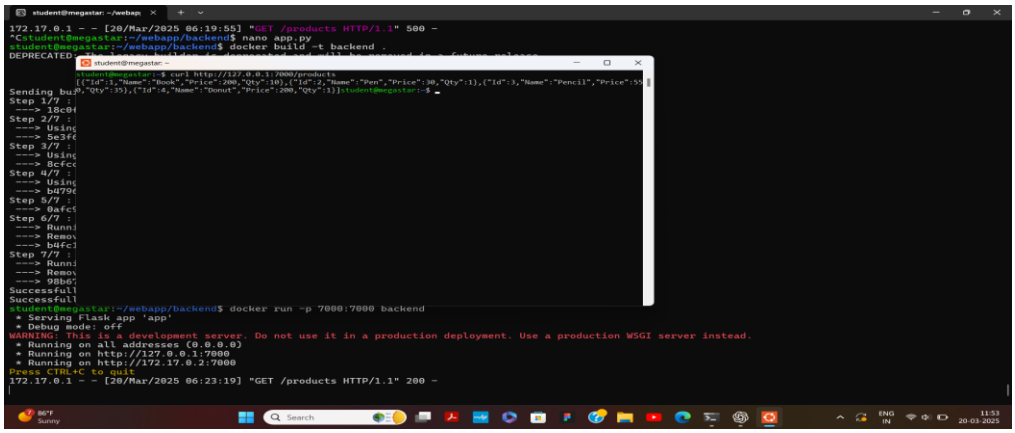
Build Docker image

Sudo docker build -t backend:latest

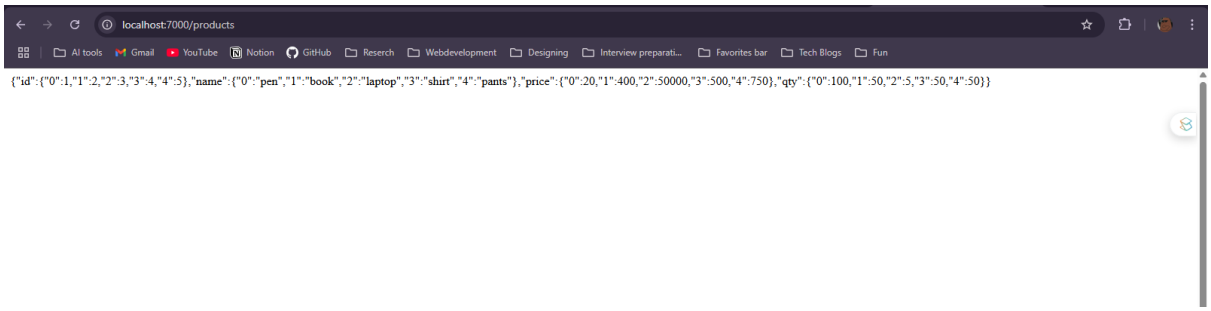


Run the docker:

sudo docker run -d -p 7000:7000 backend:latest
sudo docker logs <Generated number>



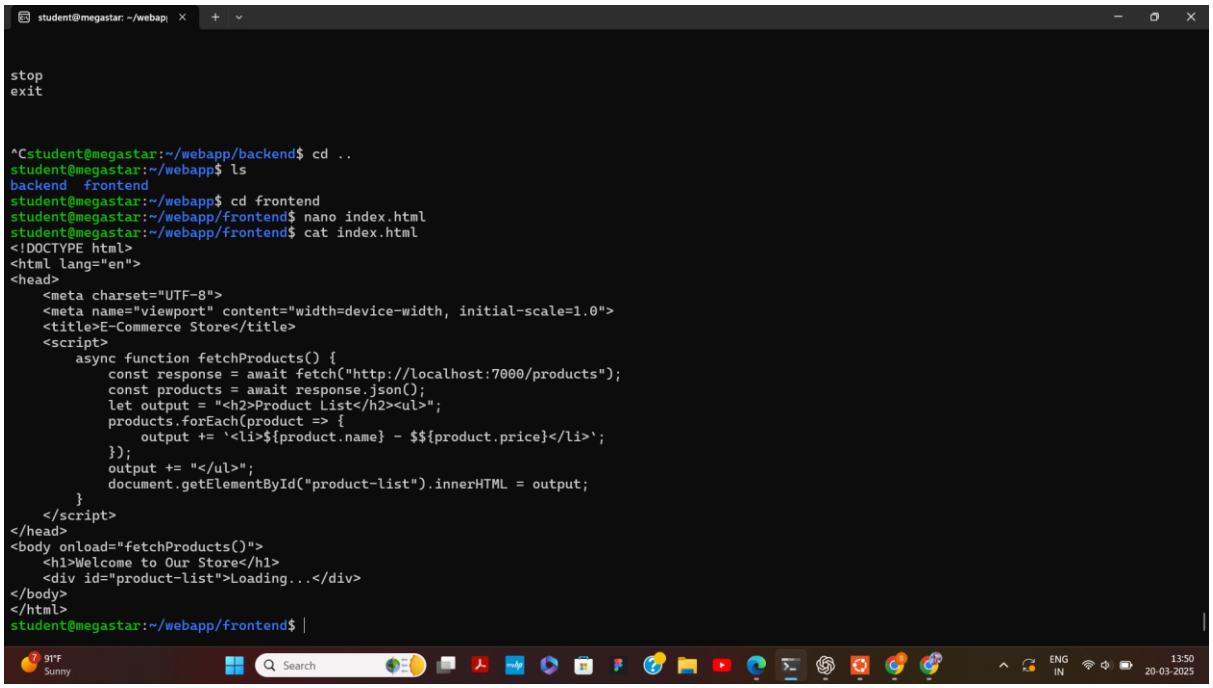
Run the application in the 7000/products



The JSON data is displayed at our port: 7000/products.

Create a container in frontend

Create **index.html file** and **Dockerfile**



Build the image using the command:

sudo docker build -t frontend:latest.

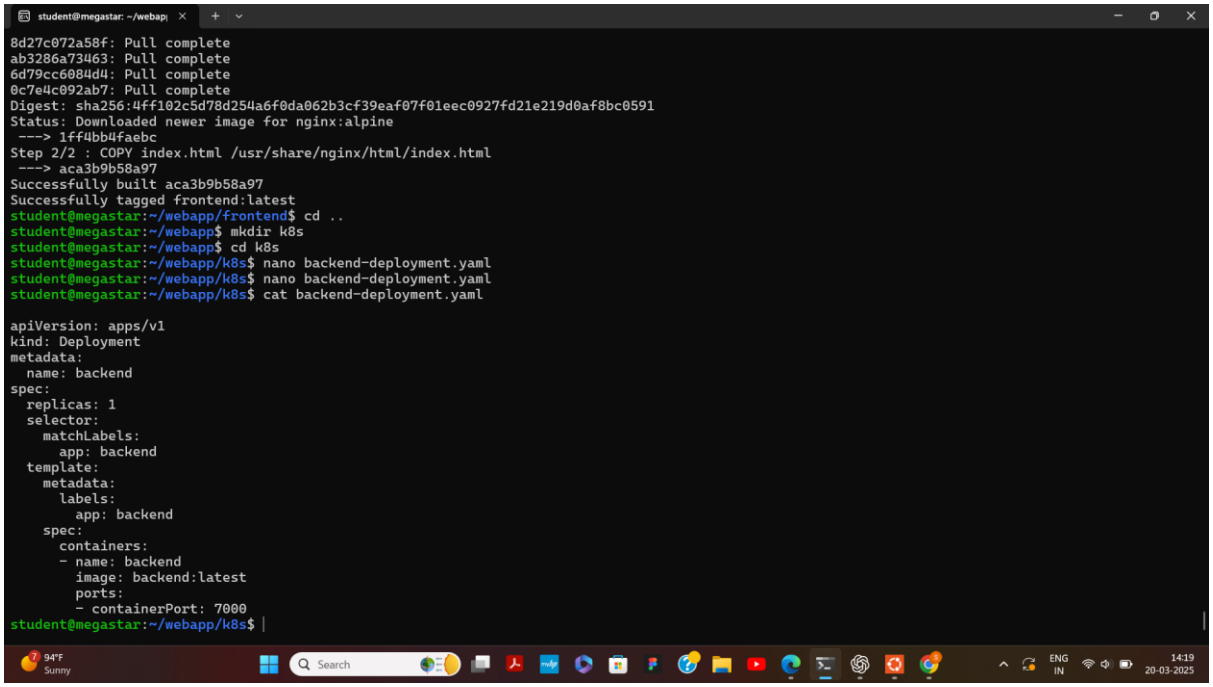
```
student@mcacc1-6:~/e-commerce/frontend$ sudo docker build -t frontend:latest .
[sudo] password for student:
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  3.584kB
Step 1/2 : FROM nginx:alpine
alpine: Pulling from library/nginx
f18232174bc9: Pull complete
ccc35e35d420: Pull complete
43f2ec460bdf: Pull complete
984583bcf083: Pull complete
8d27c072a58f: Pull complete
ab3286a73463: Pull complete
6d79cc6084d4: Pull complete
0c7e4c092ab7: Pull complete
Digest: sha256:4ff102c5d78d254a6f0da062b3cf39eaf07f01eec0927fd21e219d0af8bc0591
Status: Downloaded newer image for nginx:alpine
 ---> 1ff4bb4faebc
Step 2/2 : COPY index.html /usr/share/nginx/html/index.html
 ---> ef6c27374482
Successfully built ef6c27374482
Successfully tagged frontend:latest
```

Kubernetes Deployment YAML Files

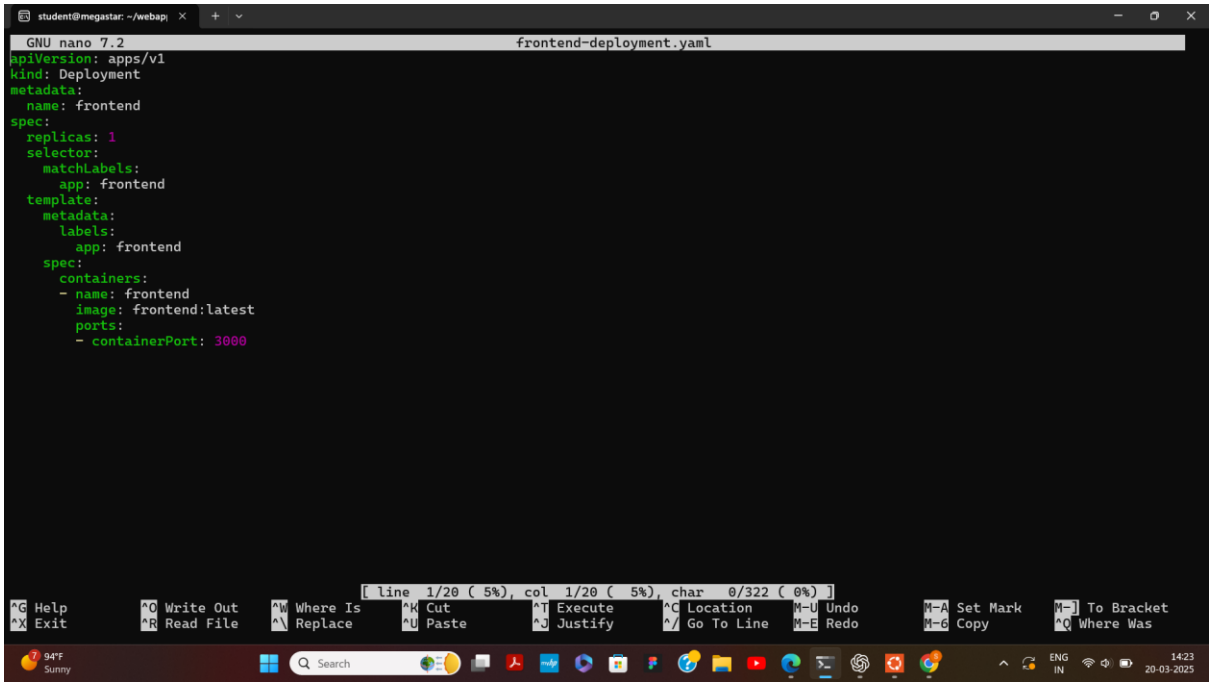
Create **backend-deployment.yaml file** and **frontend-deployment.yaml** in a folder k8s

These files define how our application should be deployed in the cluster.



Create **service.yaml** file

It exposes our application within or outside the cluster.

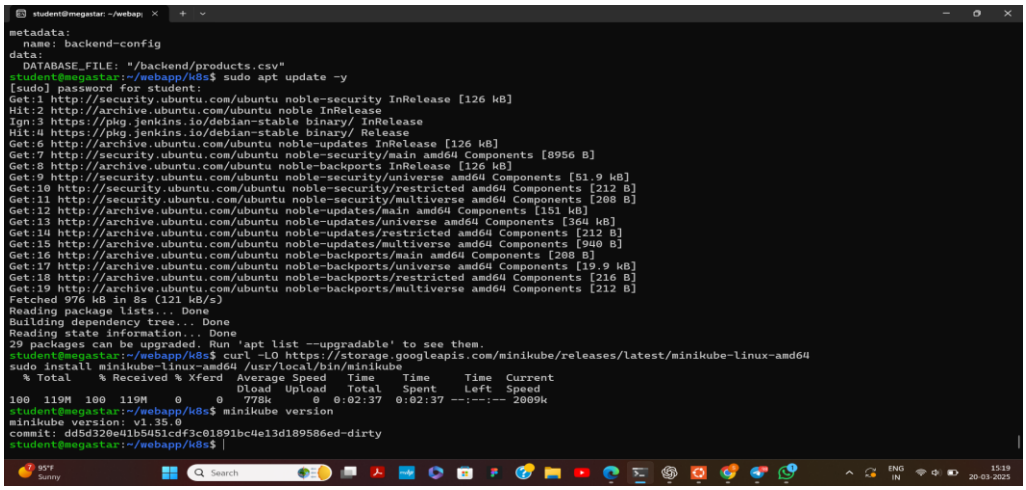


Create **configmap.yaml** file

Stores configuration data as key-value pairs.

Install minikube

`Minikube` is a tool that allows you to run a Kubernetes cluster locally on our machine. It is designed for developers who want to test and experiment with Kubernetes without needing a full-scale cloud-based cluster.



Install kubectl

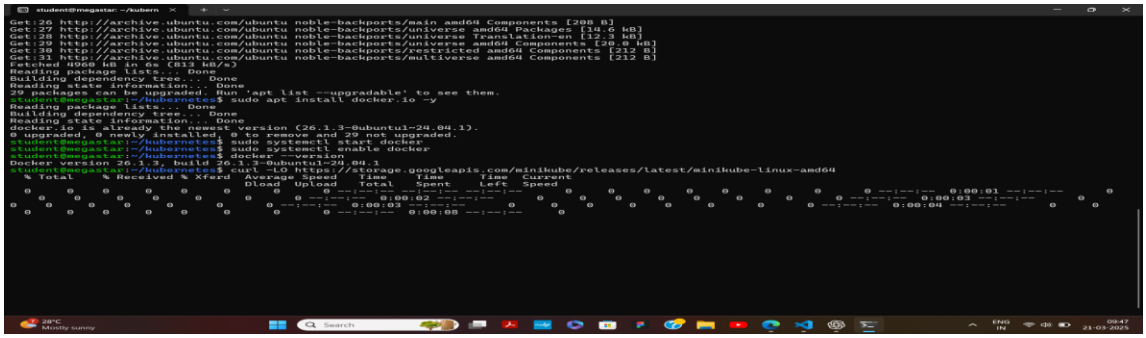
`kubectl` is the command-line tool used to interact with a Kubernetes cluster. It allows you to deploy applications, inspect and manage cluster resources, and troubleshoot issues.

Grant permission for **kubectl**

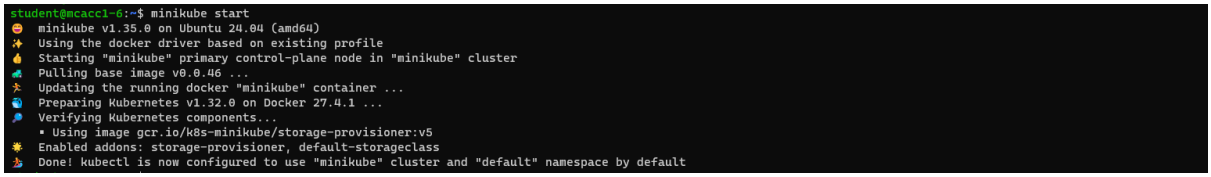
chmod +x kubectl

Move to kubectl to root

Check the **minikube** and **kubectl** installed properly



Start minicube: **minikube start**



Verify minikube is running

Load the image to the minikube

Befor loading images

Perform this commend: eval $(minikube docker-env)

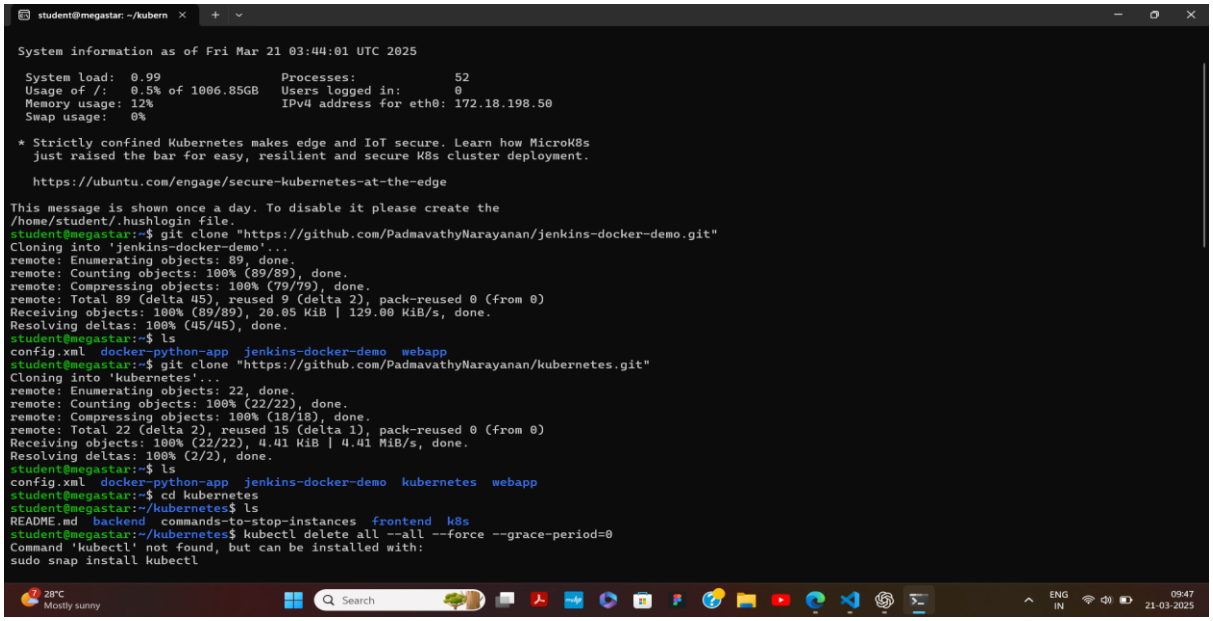
minikube image load frontend:latest

minikube image load backend:latest

Check the images are loaded

```
student@mcacc1-6:~/kubernetes/backend$ docker images | grep backend
backend                      latest    2c8028c02a4e   27 hours ago   1.17GB
student@mcacc1-6:~/kubernetes/backend$ cd ../frontend/
student@mcacc1-6:~/kubernetes/frontend$ docker images | grep frontend
frontend                     latest    ef6c27374482   24 hours ago   47.9MB
```

Commands are used to deploy your application components (backend and frontend), expose them through a service, and provide them with the necessary configuration via a ConfigMap.



These commands are used to list and inspect the running resources in your Kubernetes cluster:

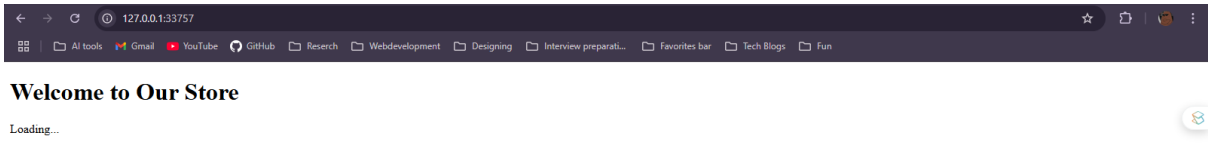
kubectl get pods

kubectl get svc

```
student@mcacc1-6:~/kubernetes/k8s$ kubectl get pods
NAME                       READY   STATUS    RESTARTS   AGE
backend-dfd8d5579-xz2xp    1/1     Running   0          3m46s
frontend-6cfd7c46-dsj9c    1/1     Running   0          3m14s
student@mcacc1-6:~/kubernetes/k8s$ kubectl get svc
NAME               TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
backend-service    ClusterIP   10.104.89.56     <none>        5000/TCP         3m12s
frontend-service   NodePort    10.105.136.172   <none>        3000:30520/TCP   3m12s
kubernetes         ClusterIP   10.96.0.1        <none>        443/TCP          3h53m
```

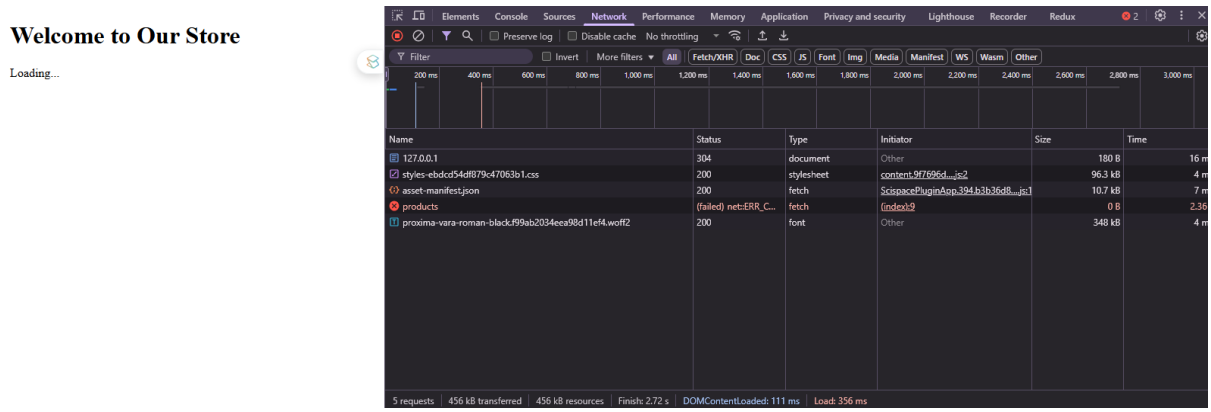
To test Frontend

```
student@mcacc1-6:~/kubernetes/k8s$ minikube service frontend-service --url
http://127.0.0.1:37341
❗  Because you are using a Docker driver on linux, the terminal needs to be open to run it.
```



To Test backend

```
student@mcacc1-6:~/kubernetes/k8s$ kubectl run test-pod --image=alpine --restart=Never -it -- sh
If you don't see a command prompt, try pressing enter.
/ # kubectl get pod test-pod
sh: kubectl: not found
/ # kubectl exec -it test-pod -- sh
sh: kubectl: not found
/ # apk add curl
fetch https://dl-cdn.alpinelinux.org/alpine/v3.21/main/x86_64/APKINDEX.tar.gz
fetch https://dl-cdn.alpinelinux.org/alpine/v3.21/community/x86_64/APKINDEX.tar.gz
(1/9) Installing brotli-libs (1.1.0-r2)
(2/9) Installing c-ares (1.34.3-r0)
(3/9) Installing libunistring (1.2-r0)
(4/9) Installing libidn2 (2.3.7-r0)
(5/9) Installing nghttp2-libs (1.64.0-r0)
(6/9) Installing libpsl (0.21.5-r3)
(7/9) Installing zstd-libs (1.5.6-r2)
(8/9) Installing libcurl (8.12.1-r1)
(9/9) Installing curl (8.12.1-r1)
Executing busybox-1.37.0-r12.trigger
OK: 12 MiB in 24 packages
/ # curl http://backend-service:5000/products
{"id":{"0":1,"1":2,"2":3,"3":4,"4":5},"name":{"0":"pen","1":"book","2":"laptop","3":"shirt","4":"pants"},"price":{"0":20,"1":400,"2":50000,"3":500,"4":750},"qty":{"0":100,"1"
:50,"2":5,"3":50,"4":50}}/ #
```



Note: We expect this kind of output because we are running this frontend on localhost.

— COMPLETED —