

Name: B. Bharanidharan [24MCR010]

Class: I – MCA – “A”

DAY 3 & 4 - DEVOPS TRAINING

Step 1:

Create the directory as webapp and make the directory frontend and backend and create a nano products.csv file and check in excel whether the format is correct

Step 2: Install the python3-pandas-y libraries into it

```
Select student@CTS-6: ~/webapp/backend
If you wish to install a non-Debian packaged Python application,
it may be easiest to use pipx install xyz, which will manage a
virtual environment for you. Make sure you have pipx installed.

See /usr/share/doc/python3.12/README.venv for more information.

note: If you believe this is a mistake, please contact your Python installation or OS distribution provider. You can override this,
g --break-system-packages.
hint: See PEP 668 for the detailed specification.
student@CTS-6:~/webapp/backend$ python3 -m venv venv
The virtual environment was not created successfully because ensurepip is not
available. On Debian/Ubuntu systems, you need to install the python3-venv
package using the following command.

apt install python3.12-venv

You may need to use sudo with that command. After installing the python3-venv
package, recreate your virtual environment.

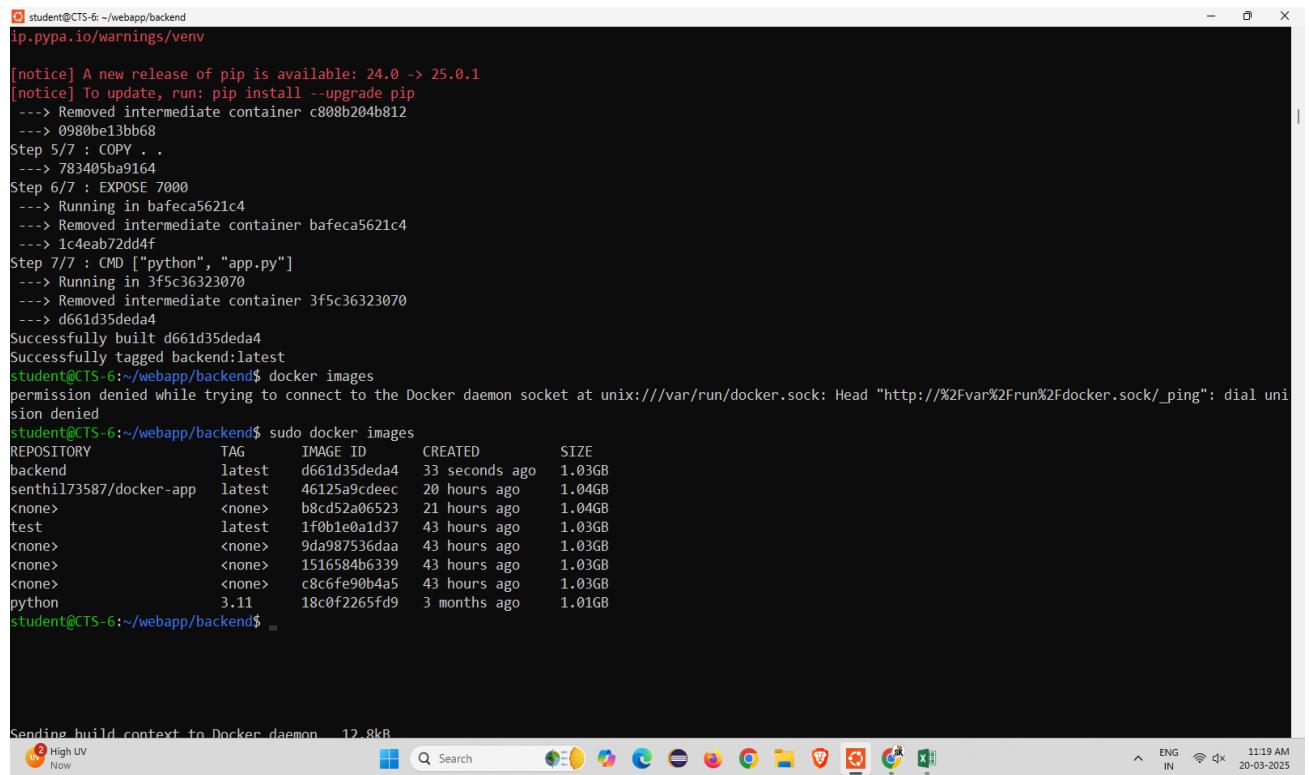
Failing command: /home/student/webapp/backend/venv/bin/python3

student@CTS-6:~/webapp/backend$ source venv/bin/activate
-bash: venv/bin/activate: No such file or directory
student@CTS-6:~/webapp/backend$ sudo apt install python3-pandas -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  btl fonts-lyx isympy-common isympy3 libaec0 libbblass libbblosci libbblosc2-2t64 libgfortran5 libhdf5-103-1t64 libimagequant0 libjbig2dec0 liblqr0 libnsappy1v5 libss2 zlib18.6 libtbbk8.6 libwebpdemux2 libwebpmux3 libxslt1.1 libxxss1 python-matplotlib-data python-odf-do
python3-bottleneck python3-brotli python3-bs4 python3-contourpy python3-cpuinfo python3-csseselect python3-cycler python3-dateutilin
python3-fonttools python3-fs python3-html5lib python3-kiwisolver python3-lxml python3-lz4 python3-matplotlib python3-mpmath python3
python3-openpyxl python3-packaging python3-pandas-lib python3-pil python3-pil.imagetk python3-scipy python3-soupsieve python3-sym

  Sync   Sunny
    Search         
  ^ FNC d x 11:59 AM
  INL 20-03-2025
```

Step 3:

Using sudo docker images command to view list



The terminal window shows the following output:

```
[student@CTS-6:~/webapp/backend]
ip.pyfa.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 c808b204b812
---> 0980be13bb68
Step 5/7 : COPY . .
---> 783405ba9164
Step 6/7 : EXPOSE 7000
---> Running in bafeca5621c4
---> Removed intermediate container bafeca5621c4
---> 1c4eab72dd4f
Step 7/7 : CMD ["python", "app.py"]
---> Running in 3f5c36323070
---> Removed intermediate container 3f5c36323070
---> d661d35deda4
Successfully built d661d35deda4
Successfully tagged backend:latest
student@CTS-6:~/webapp/backend$ docker images
permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Head "http://%2Fvar%2Frun%2Fdocker.sock/_ping": dial unix
: permission denied
student@CTS-6:~/webapp/backend$ sudo docker images
REPOSITORY          TAG      IMAGE ID      CREATED       SIZE
backend             latest   d661d35deda4  33 seconds ago  1.03GB
senthil73587/docker-app  latest   46125a9cdee  20 hours ago  1.04GB
<none>              <none>   b8cd52a06523  21 hours ago  1.04GB
test                latest   1f0b1e0a1d37  43 hours ago  1.03GB
<none>              <none>   9da987536daa  43 hours ago  1.03GB
<none>              <none>   1516584b6339  43 hours ago  1.03GB
<none>              <none>   c8c6fe90b4a5  43 hours ago  1.03GB
python              3.11    18c0f2265fd9  3 months ago  1.01GB
student@CTS-6:~/webapp/backend$ =
```

The file browser window shows a single file named "products.csv".

App.py

```
from flask import Flask
import pandas as pd
app = Flask(__name__)
@app.route("/products", methods=['GET']) # ✓ Fixed 'methods' issue
def read_data():
    df = pd.read_csv("products.csv")
    json_data = df.to_json()
    return json_data
if __name__ == "__main__":
    app.run(host="0.0.0.0", port=7000)
```

Docker File

```
FROM python:3.11

WORKDIR /app

COPY requirements.txt . # Ensure pandas is in requirements.txt
RUN pip install --no-cache-dir -r requirements.txt

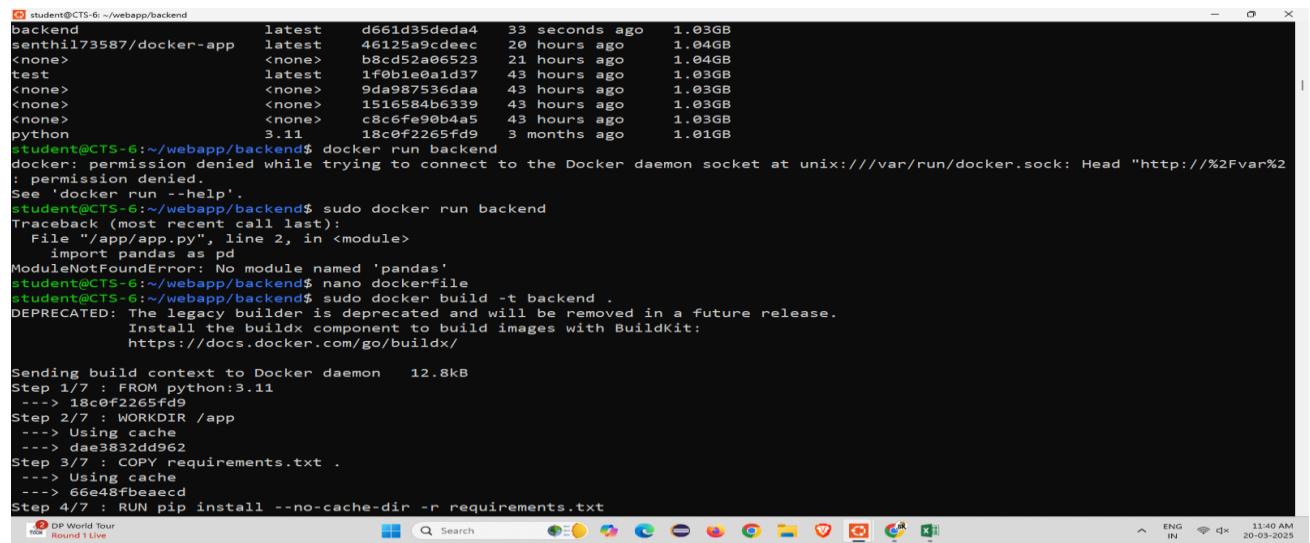
COPY ..

EXPOSE 7000

CMD ["python", "app.py"]
```

Step 4 :

nano dockerfile and nano docker built -t backend . to built the docker.



```
student@CTS-6:~/webapp/backend$ docker run backend
docker: permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Head "http://%2Fvar%2
: Permission denied.
See 'docker run --help'.
student@CTS-6:~/webapp/backend$ sudo docker run backend
Traceback (most recent call last):
  File "/app/app.py", line 2, in <module>
    import pandas as pd
ModuleNotFoundError: No module named 'pandas'
student@CTS-6:~/webapp/backend$ nano dockerfile
student@CTS-6:~/webapp/backend$ sudo docker build -t backend .
DEPRECATION: 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 12.8kB
Step 1/7 : FROM python:3.11
--> 18c0f2265fd9
Step 2/7 : WORKDIR /app
--> Using cache
--> dae3832dd962
Step 3/7 : COPY requirements.txt .
--> Using cache
--> 66e48fbbeacd
Step 4/7 : RUN pip install --no-cache-dir -r requirements.txt
```

Step 5 :

using sudo docker run -p 7000:7000 backend command to run the backend

```
student@CTS-6: ~/webapp/backend
--> Using cache
--> 66e48fbeacd
Step 4/7 : RUN pip install --no-cache-dir -r requirements.txt
--> Using cache
--> e095ae34c3f2
Step 5/7 : COPY . .
--> 700cf7379f1b
Step 6/7 : EXPOSE 7000
--> Running in 8d1749b839b3
--> Removed intermediate container 8d1749b839b3
--> 95a1002980f5
Step 7/7 : CMD ["python", "app.py"]
--> Running in c1f2651321af
--> Removed intermediate container c1f2651321af
--> 2de55ba2d920
Successfully built 2de55ba2d920
Successfully tagged backend:latest
student@CTS-6:~/webapp/backend$ sudo docker run -p 7000:7000 backend
* Serving Flask app 'app'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:7000
* Running on http://172.17.0.2:7000
Press CTRL+C to quit
172.17.0.1 - - [20/Mar/2025 06:10:16] "GET /products HTTP/1.1" 200 -

```

Step 6:

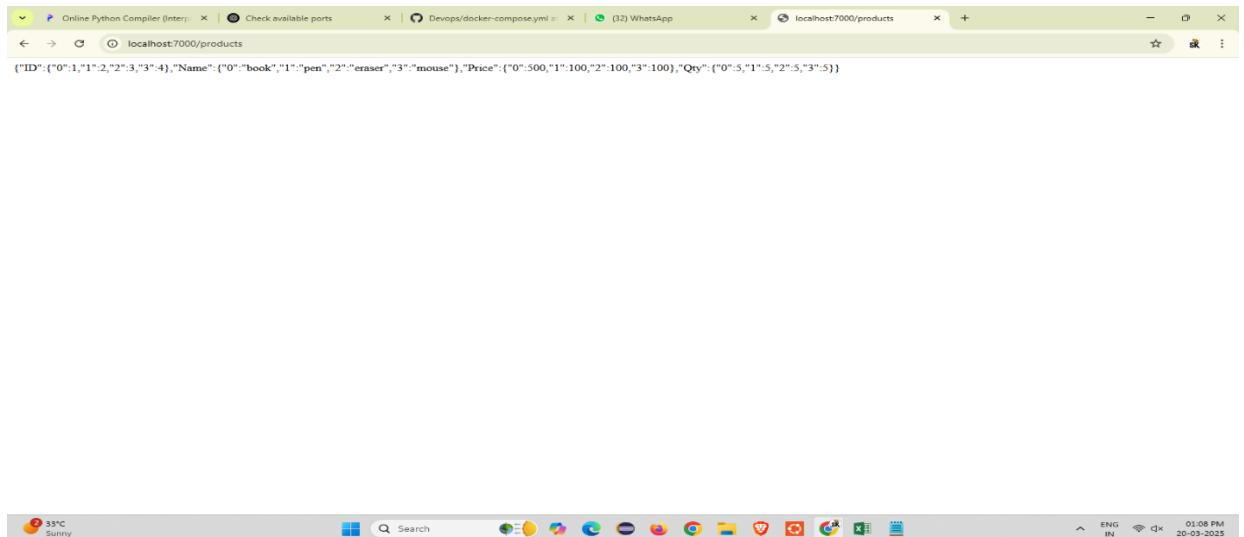
Open new Ubuntu and run as administration and enter the command as curl –X GET <http://localhost:7000/products>

```
student@CTS-6:~/webapp/backend
Step 3/7 : COPY requirements.txt .
--> Using cache
--> 68af149a8d64
Step 4/7 : RUN pip install --no-cache-dir -r requirements.txt
--> Running in 3d356f719bb9
ERROR: Invalid [student@CTS-6:~]
[notice] A new version of curl is available: 8.2.0
[notice] To update:
{"ID":("0":1,"1":2,"2":3,"3":4),"Name":("0":"book","1":"pen","2":"eraser","3":"mouse"),"Price":{"0":500,"1":100,"2":100,
"3":100),"Qty":{"0":5,"1":5,"2":5,"3":5}}student@CTS-6:~$ 
The command '
student@CTS-6:
student@CTS-6:
DEPRECATED: Th
Ir
ht

Sending build
Step 1/7 : FRC
--> 18c0f226
Step 2/7 : WOF
--> Using ca
--> dae3832c
Step 3/7 : COF
--> Using ca
--> 66e48fbe
Step 4/7 : RUN
--> Using cache
--> e095ae34c3f2
Step 5/7 : COPY . .
--> 700cf7379f1b
Step 6/7 : EXPOSE 7000
--> Running in 8d1749b839b3
--> Removed intermediate container 8d1749b839b3

```

Step 7: Go to the browser and enter the url it displays the backend the backend



Step 8:

Change the repository to frontend and create a nano index.html file and enter the code

```
* Running on http://172.17.0.2:7000
Press CTRL+C to quit
172.17.0.1 - - [20/Mar/2025 06:10:16] "GET /products HTTP/1.1" 200 -
172.17.0.1 - - [20/Mar/2025 06:11:18] "GET /products HTTP/1.1" 200 -
172.17.0.1 - - [20/Mar/2025 07:38:54] "GET /products HTTP/1.1" 200 -
172.17.0.1 - - [20/Mar/2025 07:38:54] "GET /favicon.ico HTTP/1.1" 404 -
^Cstudent@CTS-6:~/webapp/backend$ cd ..
student@CTS-6:~/webapp$ ls
backend frontend
student@CTS-6:~/webapp$ cd frontend
student@CTS-6:~/webapp/frontend$ nano index.html
student@CTS-6:~/webapp/frontend$ cat index.html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>E-Commerce Store</title>
    <script>
        async function fetchProducts() {
            const response = await fetch("http://localhost:7000/products");
            const products = await response.json();
            let output = "<h2>Product List</h2><ul>";
            products.forEach(product => {
                output += `<li>${product.name} - ${product.price}</li>`;
            });
            output += "</ul>";
            document.getElementById("product-list").innerHTML = output;
        }
    </script>
</head>
<body onload="fetchProducts()">
    <h1>Welcome to Our Store</h1>
    <div id="product-list">Loading...</div>
</body>
</html>
student@CTS-6:~/webapp/frontend$ nano index.html
student@CTS-6:~/webapp/frontend$ nano dockerfile
student@CTS-6:~/webapp/frontend$ cat dockerfile
FROM nginx:alpine
COPY index.html /usr/share/nginx/html/index.html
student@CTS-6:~/webapp/frontend$
```

Step 9:

Change the directory and make the directory k8s in that create a nano backend-deployment.yaml and add the code

```
student@CTS-6:~/webapp/k8s$ 
9c/e4c092ab7: Pull complete
Digest: sha256:fb192c5d78d254a6f0da62b3cf39ea0f07f01ec0927fd21e219d0af8bc0591
Status: Downloaded newer image for nginx:alpine
-> 1ffa4b4faebc
Step 2/2 : COPY index.html /usr/share/nginx/html/index.html
--> a68174ecbac
Successfully built a68174ecbac
Successfully tagged frontend:latest
student@CTS-6:~/webapp/frontend$ cd ..
student@CTS-6:~/webapp$ mkdir k8s
student@CTS-6:~/webapp$ cd k8s
student@CTS-6:~/webapp/k8s$ nano backend-deployment.yaml
student@CTS-6:~/webapp/k8s$ cd ..
student@CTS-6:~/webapp$ ls
backend frontend k8s
student@CTS-6:~/webapp$ cd frontend
student@CTS-6:~/webapp/frontend$ nano dockerfile
student@CTS-6:~/webapp/frontend$ ls
dockerfile index.html
student@CTS-6:~/webapp/frontend$ nano index.html
student@CTS-6:~/webapp/frontend$ cd ..
student@CTS-6:~/webapp$ ls
backend frontend k8s
student@CTS-6:~/webapp$ cd k8s
student@CTS-6:~/webapp$ cat backend-deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: backend
spec:
  replicas: 1
  selector:
    matchLabels:
      app: backend
  template:
    metadata:
      labels:
        app: backend
    spec:
      containers:
        - name: backend
          image: backend:latest
          ports:
            - containerPort: 7000
student@CTS-6:~/webapp/k8s$
```

Step 10:

Create another nano frontend-deployment.yaml file and add the code

```
student@CTS-6:~/webapp/k8s$ 
labels:
  app: backend
spec:
  containers:
    - name: backend
      image: backend:latest
      ports:
        - containerPort: 7000
student@CTS-6:~/webapp/k8s$ nano frontend-deployment.yaml
student@CTS-6:~/webapp/k8s$ cat frontend-deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: frontend
spec:
  replicas: 1
  selector:
    matchLabels:
      app: frontend
  template:
    metadata:
      labels:
        app: frontend
    spec:
      containers:
        - name: frontend
          image: frontend:latest
          ports:
            - containerPort: 3000
student@CTS-6:~/webapp/k8s$
```

Step 11:

Create another nano service.yaml and configmap.yaml file and add the code

```
student@CTS-6:~/webapp/k8s$ type: NodePort
student@CTS-6:~/webapp/k8s$ cat service.yaml
apiVersion: v1
kind: Service
metadata:
  name: backend-service
spec:
  selector:
    app: backend
  ports:
  - protocol: TCP
    port: 7000
    targetPort: 7000
  type: ClusterIP

apiVersion: v1
kind: Service
metadata:
  name: frontend-service
spec:
  selector:
    app: frontend
  ports:
  - protocol: TCP
    port: 3000
    targetPort: 3000
  type: NodePort
student@CTS-6:~/webapp/k8s$ nano configmap.yaml
student@CTS-6:~/webapp/k8s$ cat configmap.yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: backend-config
data:
  DATABASE_FILE: "/backend/products.csv"
student@CTS-6:~/webapp/k8s$ _
```

Step 12:

Clone the kubernetes github repository and run the following commands

Step 13 :

Go to the kubernetes backend and frontend directory and type the command to load an image

```
student@CTS-6:~/kubernetes/frontend
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 or run pip as a regular user.
[notice] A new release of pip is available: 23.0.1 -> 25.0.1
[notice] To update, run: pip install --upgrade pip
----> Removed intermediate container 38e940d4c380
----> 76d1bea76aa8
Step 5/6 : COPY . .
----> e791dccfffb5
Step 6/6 : CMD ["python", "app.py"]
----> Running in de3a6ce812db
----> Removed intermediate container de3a6ce812db
----> 089598ec5312
Successfully built 089598ec5312
Successfully tagged backend:latest
student@CTS-6:~/kubernetes/backend$ minikube image load backend:latest
student@CTS-6:~/kubernetes/backend$ ls
app.py dockerfile products.csv requirements.txt
student@CTS-6:~/kubernetes/backend$ cd kubernetes
-bash: cd: kubernetes: No such file or directory
student@CTS-6:~/kubernetes/backend$ cd ../frontend
student@CTS-6:~/kubernetes/frontend$ docker build -t frontend: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 3.584kB
Step 1/2 : FROM nginx:alpine
----> 1ff4bb4faebc
Step 2/2 : COPY index.html /usr/share/nginx/html/index.html
----> bbd017075857
Successfully built bbd017075857
Successfully tagged frontend:latest
student@CTS-6:~/kubernetes/frontend$ minikube image load frontend:latest
student@CTS-6:~/kubernetes/frontend$
```

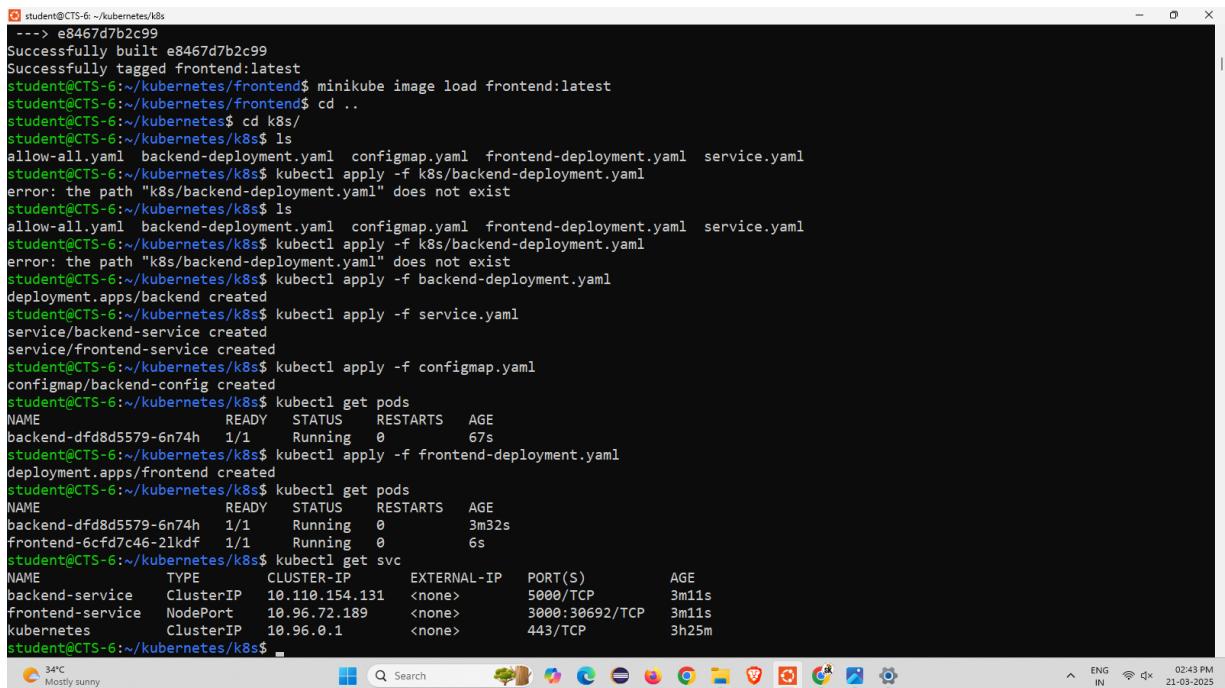
Step 14 : Come back to root directory setup minikube docker-env and build the frontend and backend

```
student@CTS-6:~/kubernetes/backend
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
----> 1ff4bb4faebc
Step 2/2 : COPY index.html /usr/share/nginx/html/index.html
----> bbd017075857
Successfully built bbd017075857
Successfully tagged frontend:latest
student@CTS-6:~/kubernetes/frontend$ minikube image load frontend:latest
student@CTS-6:~/kubernetes/frontend$ cd ..
student@CTS-6:~$ eval $(minikube docker-env)
student@CTS-6:~$ ls
docker-python-app kubernetes webapp
student@CTS-6:~$ cd kubernetes/
student@CTS-6:~/kubernetes$ ls
README.md backend commands-to-stop-instances frontend k8s minikube-linux-amd64
student@CTS-6:~/kubernetes$ cd backend/
student@CTS-6:~/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: Downloading [=====] 16.72MB/48.47MB
091eb8249475: Downloading [=====] 17.68MB/24.01MB
255774e0027b: Downloading [=====] 23.71MB/64.4MB
353e14e5cc47: Waiting
f6d72b0ae7c: Waiting
6e02a00e58ae: Waiting
f299e0671245: Waiting
Dinesh's AIS
Internet access
```

Step 15 :

Open k8s directory and list the files into it and kubectl apply -f commands and initialize get pods , get svc



```
student@CTS-6:~/Kubernetes/frontend$ ---> e8467d7b2c99
Successfully built e8467d7b2c99
Successfully tagged frontend:latest
student@CTS-6:~/Kubernetes/frontend$ minikube image load frontend:latest
student@CTS-6:~/Kubernetes/frontend$ cd ..
student@CTS-6:~/Kubernetes$ cd k8s/
student@CTS-6:~/Kubernetes/k8s$ ls
allow-all.yaml backend-deployment.yaml configmap.yaml frontend-deployment.yaml service.yaml
student@CTS-6:~/Kubernetes/k8s$ kubectl apply -f k8s/backend-deployment.yaml
error: the path "k8s/backend-deployment.yaml" does not exist
student@CTS-6:~/Kubernetes/k8s$ ls
allow-all.yaml backend-deployment.yaml configmap.yaml frontend-deployment.yaml service.yaml
student@CTS-6:~/Kubernetes/k8s$ kubectl apply -f k8s/backend-deployment.yaml
error: the path "k8s/backend-deployment.yaml" does not exist
student@CTS-6:~/Kubernetes/k8s$ kubectl apply -f backend-deployment.yaml
deployment.apps/backend created
student@CTS-6:~/Kubernetes/k8s$ kubectl apply -f service.yaml
service/backend-service created
service/frontend-service created
student@CTS-6:~/Kubernetes/k8s$ kubectl apply -f configmap.yaml
configmap/backend-config created
student@CTS-6:~/Kubernetes/k8s$ kubectl get pods
NAME           READY   STATUS    RESTARTS   AGE
backend-dfd8d5579-6n74h  1/1    Running   0          67s
student@CTS-6:~/Kubernetes/k8s$ kubectl apply -f frontend-deployment.yaml
deployment.apps/frontend created
student@CTS-6:~/Kubernetes/k8s$ kubectl get pods
NAME           READY   STATUS    RESTARTS   AGE
backend-dfd8d5579-6n74h  1/1    Running   0          3m32s
frontend-6cf7c46-21kdf  1/1    Running   0          6s
student@CTS-6:~/Kubernetes/k8s$ kubectl get svc
NAME            TYPE        CLUSTER-IP      EXTERNAL-IP      PORT(S)        AGE
backend-service  ClusterIP  10.110.154.131  <none>         5000/TCP     3m11s
frontend-service NodePort   10.96.72.189   <none>         3000:30692/TCP 3m11s
kubernetes       ClusterIP  10.96.0.1     <none>         443/TCP      3h25m
student@CTS-6:~/Kubernetes/k8s$
```

Step 16 :

using minikube service frontend-service –url it will displays the https:// ip address and using curl command to run in the terminal.

```

student@CTS-6:~/kubernetes/k8s$ kubectl apply -f configmap.yaml
configmap/backend-config created
student@CTS-6:~/kubernetes/k8s$ kubectl get pods
NAME           READY   STATUS    RESTARTS   AGE
backend-dfd8d5579-6n74h   1/1     Running   0          67s
student@CTS-6:~/kubernetes/k8s$ kubectl apply -f frontend-deployment.yaml
deployment.apps/frontend created
student@CTS-6:~/kubernetes/k8s$ kubectl get pods
NAME           READY   STATUS    RESTARTS   AGE
backend-dfd8d5579-6n74h   1/1     Running   0          3m32s
frontend-6cf7c46-21kdf   1/1     Running   0          6s
student@CTS-6:~/kubernetes/k8s$ kubectl get svc
NAME            TYPE      CLUSTER-IP   EXTERNAL-IP   PORT(S)        AGE
backend-service   ClusterIP  10.110.154.131 <none>       5000/TCP      3m11s
frontend-service  NodePort   10.96.72.189  <none>       3000:30692/TCP  3m11s
kubernetes       ClusterIP  10.96.0.1    <none>       443/TCP       3h25m
student@CTS-6:~/kubernetes/k8s$ minikube service frontend-service --url
http://127.0.0.1:39285
! Because you are using a Docker driver on linux, the terminal needs to be open to run it.
student@CTS-6:~/kubernetes/k8s$ kubectl run test-pod --image=alpine --restart=Never -it -- sh
If you don't see a command prompt, try pressing enter.
/ # 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":1,"name":"Smartphone","price":299.99},{"id":2,"name":"Laptop","price":799.99},{"id":3,"name":"Headphones","price":49.99},{"id":4,"name":"Tablet","price":199.99}]
/ #

```

Step 17:

Enter the given ip address in the browser and get the output.

