Day 4: Kubernetes

Create a directory ‘e-commerce’ and its required folders and files

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

```
jegadeep@LAPTOP-EBJ40AHO:~/fullstack/backend$ nano products.csv
jegadeep@LAPTOP-EBJ40AHO:~/fullstack/backend$ cat products.csv
ID, Name, Price, Qty
1,IPHONE,48000,3
2,DELL,80000,9
3,NOTEBOOK,75000,8
4,HPPAVILION,79000,14
```

```
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)
```

Install the **pandas** library:

```
sudo apt install python3-pandas
[sudo] password for student:
```

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

To verify the **available port numbers**

```
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 127.0.0.53:domain       0.0.0.0:*               LISTEN      104/systemd-resolve
tcp        0      0 localhost:34075         0.0.0.0:*               LISTEN      238/containerd
tcp        0      0 0.0.0.0:http            0.0.0.0:*               LISTEN      208/nginx: master p
tcp        0      0 10.255.255.254:domain   0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.54:domain       0.0.0.0:*               LISTEN      104/systemd-resolve
tcp6       0      0 [::]:http               [::]:*                  LISTEN      208/nginx: master p
tcp6       0      0 [::]:http-alt           [::]:*                  LISTEN      151/java
udp        0      0 127.0.0.54:domain       0.0.0.0:*                           104/systemd-resolve
udp        0      0 127.0.0.53:domain       0.0.0.0:*                           104/systemd-resolve
udp        0      0 10.255.255.254:domain   0.0.0.0:*                           -
udp        0      0 localhost:323           0.0.0.0:*                           -
udp6       0      0 ip6-localhost:323       [::]:*                              -
Active UNIX domain sockets (only servers)
Proto RefCnt Flags       Type       State         I-Node   PID/Program name     Path
unix  2      [ ACC ]     STREAM     LISTENING     25614    2/init               /run/WSL/2_interop
unix  2      [ ACC ]     STREAM     LISTENING     19471    -                    /run/WSL/1_interop
unix  2      [ ACC ]     SEQPACKET  LISTENING     20867    -                    /mnt/wslg/weston-notify.sock
unix  2      [ ACC ]     STREAM     LISTENING     27649    -                    /var/run/dbus/system_bus_socket
unix  2      [ ACC ]     STREAM     LISTENING     24587    -                    /mnt/wslg/runtime-dir/wayland-0
unix  2      [ ACC ]     STREAM     LISTENING     24588    -                    /tmp/.X11-unix/X0
unix  2      [ ACC ]     STREAM     LISTENING     18603    104/systemd-resolve  /run/systemd/resolve/io.systemd.Resolve
unix  2      [ ACC ]     STREAM     LISTENING     18604    104/systemd-resolve  /run/systemd/resolve/io.systemd.Resolve.Monitor
unix  2      [ ACC ]     STREAM     LISTENING     19509    -                    /mnt/wslg/runtime-dir/pulse/native
unix  2      [ ACC ]     STREAM     LISTENING     23832    -                    /mnt/wslg/PulseAudioRDPSource
unix  2      [ ACC ]     STREAM     LISTENING     19682    1/init               /run/apport.socket
unix  2      [ ACC ]     STREAM     LISTENING     19684    1/init               /run/dbus/system_bus_socket
unix  2      [ ACC ]     STREAM     LISTENING     19685    1/init               /run/docker.sock
unix  2      [ ACC ]     STREAM     LISTENING     31887    864/systemd          /run/user/1000/systemd/private
unix  2      [ ACC ]     STREAM     LISTENING     19687    1/init               /run/snapd.socket
unix  2      [ ACC ]     STREAM     LISTENING     31894    864/systemd          /run/user/1000/bus
unix  2      [ ACC ]     STREAM     LISTENING     19688    1/init               /run/snapd-snap.socket
unix  2      [ ACC ]     STREAM     LISTENING     19690    1/init               /run/uuidd/request
unix  2      [ ACC ]     STREAM     LISTENING     31896    864/systemd          /run/user/1000/gnupg/S.dirmngr
```

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.

```
student@mcacc1-6:~/e-commerce/backend$ nano requirements.txt
student@mcacc1-6:~/e-commerce/backend$ cat requirements.txt
flask
pandas
```

Create **docker-compose.yml** file

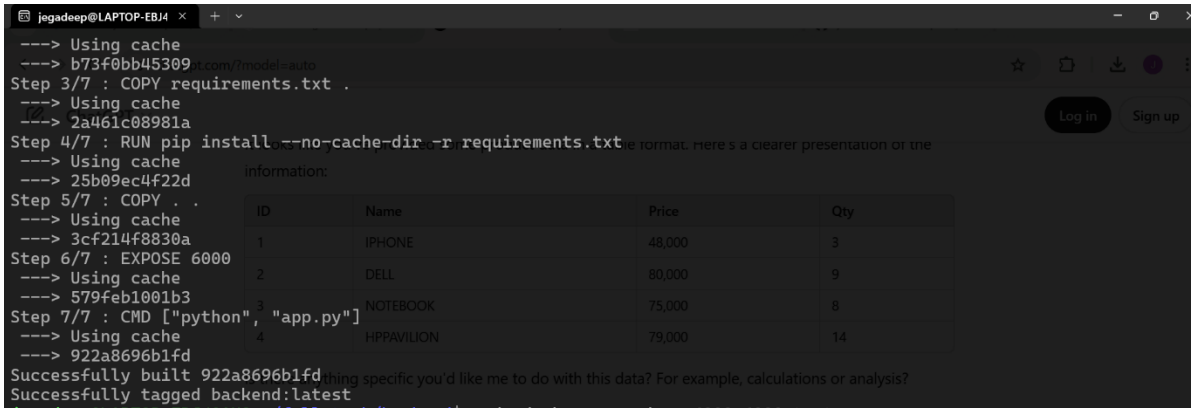
`docker-compose.yml` is a YAML configuration file used to define and run multi-container Docker applications.

```
version: '3.8'

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

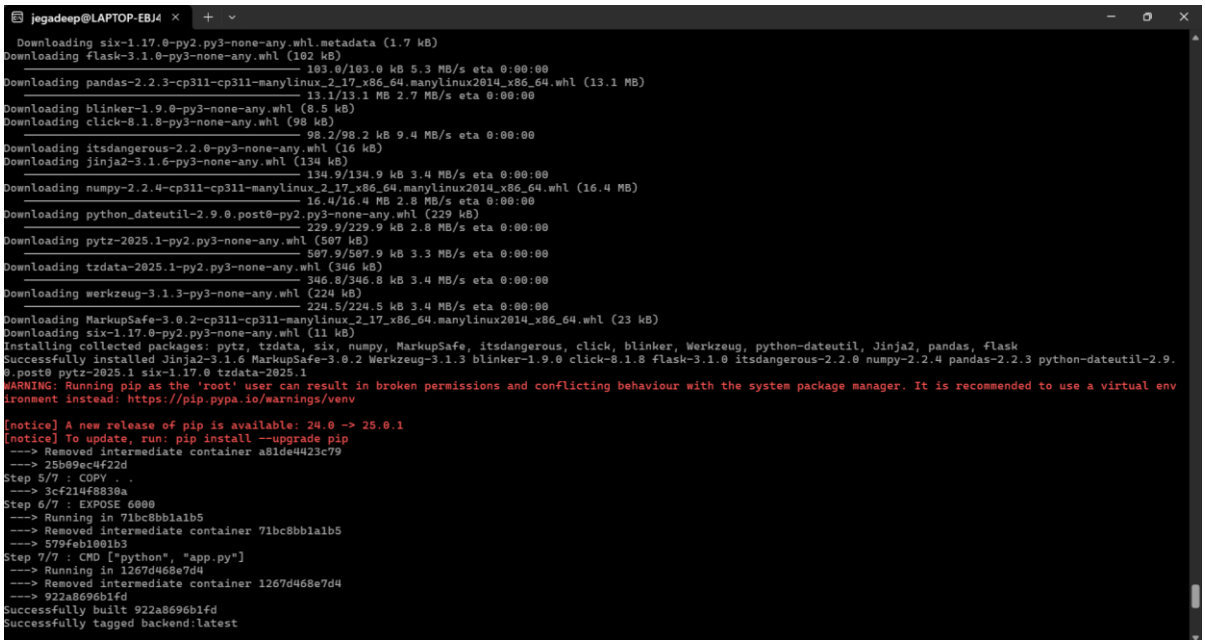
Build Docker image

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: 6000/products.

Create a container in frontend

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

Build the image using the command:

sudo 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
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.

```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
```

Create **service.yaml** file

It exposes our application within or outside the cluster.

```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: 7500
      targetPort: 7500
  type: NodePort
```

Create **configmap.yaml** file

Stores configuration data as key-value pairs.

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: backend-config
data:
  DATABASE_FILE: "/backend/products.csv"
```

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.

```
student@mcacc1-6:~/e-commerce/k8s$ sudo apt update
[sudo] password for student:
Ign:1 https://pkg.jenkins.io/debian-stable binary/ InRelease
Hit:2 https://pkg.jenkins.io/debian-stable binary/ Release
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Hit:5 http://archive.ubuntu.com/ubuntu noble InRelease
Get:6 http://archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/main amd64 Components [8956 B]
Get:8 http://archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:9 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [51.9 kB]
Get:10 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Components [212 B]
Get:11 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [208 B]
Get:12 http://archive.ubuntu.com/ubuntu noble-updates/main amd64 Components [151 kB]
Get:13 http://archive.ubuntu.com/ubuntu noble-updates/universe amd64 Components [364 kB]
Get:14 http://archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Components [212 B]
Get:15 http://archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Components [940 B]
Get:16 http://archive.ubuntu.com/ubuntu noble-backports/main amd64 Components [208 B]
Get:17 http://archive.ubuntu.com/ubuntu noble-backports/universe amd64 Components [19.9 kB]
Get:18 http://archive.ubuntu.com/ubuntu noble-backports/restricted amd64 Components [216 B]
Get:19 http://archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 Components [212 B]
Fetched 976 kB in 2s (412 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
28 packages can be upgraded. Run 'apt list --upgradable' to see them.
student@mcacc1-6:~/e-commerce/k8s$ docker -v
Docker version 26.1.3, build 26.1.3-0ubuntu1~24.04.1
student@mcacc1-6:~/e-commerce/k8s$ sudo apt install docker.io -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
docker.io is already the newest version (26.1.3-0ubuntu1~24.04.1).
0 upgraded, 0 newly installed, 0 to remove and 28 not upgraded.
student@mcacc1-6:~/e-commerce/k8s$ docker -v
Docker version 26.1.3, build 26.1.3-0ubuntu1~24.04.1
student@mcacc1-6:~/e-commerce/k8s$ curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
 25  119M   25 30.2M    0     0  1335k      0  0:01:31  0:00:23  0:01:08 1276k
```

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.

```
student@mcacc1-6:~/e-commerce/k8s$ curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   138  100   138    0     0    424      0 --:--:-- --:--:-- --:--:--   425
 71 54.6M   71 39.3M    0     0   671k      0  0:01:23  0:00:59  0:00:24  594k
```

Grant permission for **kubectl**

chmod +x kubectl

Move to kubectl to root

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

```
student@mcacc1-6:~$ kubectl version
Client Version: v1.32.3
Kustomize Version: v5.5.0
Error from server (Forbidden): <html><head><meta http-equiv='refresh' content='1;url=/login?from=%2Fversion%3Ftimeout%3D32s'/><script id='redirect' data-redirect-url='/login?
from=%2Fversion%3Ftimeout%3D32s' src='/static/dad96ebf/scripts/redirect.js'></script></head><body style='background-color:white; color:white;'>
Authentication required
<!--
-->

</body></html>
student@mcacc1-6:~$ minikube version
minikube version: v1.35.0
commit: dd5d320e41b5451cdf3c01891bc4e13d189586ed-dirty
```

Start minicube: **minikube start**

```
jegadeep@LAPTOP-EBJ40AHO:~/kubernetes$         minikube start

😄  minikube v1.35.0 on Ubuntu 24.04 (amd64)
✨  Using the docker driver based on existing profile
👍  Starting "minikube" primary control-plane node in "minikube"
cluster
🚜  Pulling base image v0.0.46 ...
🏃  Updating the running docker "minikube" container ...



🐳  Preparing Kubernetes v1.32.0 on Docker 27.4.1 ...
🔎  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
```

Verify minikube is running

```
jegadeep@LAPTOP-EBJ40AHO:~/kubernetes$ kubectl get nodes
NAME       STATUS   ROLES           AGE    VERSION
minikube   Ready    control-plane   2m3s   v1.32.0
```

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

```
jegadeep@LAPTOP-EBJ40AHO:~/kubernetes/frontend$ docker images | grep frontend
frontend                                latest     2de8e5a499fc   12 minutes ago   47.9MB
jegadeep@LAPTOP-EBJ40AHO:~/kubernetes/frontend$ minikube image load frontend:latest
jegadeep@LAPTOP-EBJ40AHO:~/kubernetes/frontend$ kubectl apply -f backend-deployment.yaml
```

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.

```
jegadeep@LAPTOP-EBJ40AHO:~/kubernetes$ cd k8s
jegadeep@LAPTOP-EBJ40AHO:~/kubernetes/k8s$ kubectl apply -f backend-deployment.yaml
deployment.apps/backend unchanged
jegadeep@LAPTOP-EBJ40AHO:~/kubernetes/k8s$ kubectl apply -f frontend-deployment.yaml
deployment.apps/frontend created
jegadeep@LAPTOP-EBJ40AHO:~/kubernetes/k8s$ kubectl apply -f service.yaml
service/backend-service unchanged
service/frontend-service unchanged
jegadeep@LAPTOP-EBJ40AHO:~/kubernetes/k8s$ kubectl apply -f configmap.yaml
configmap/backend-config unchanged
```

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

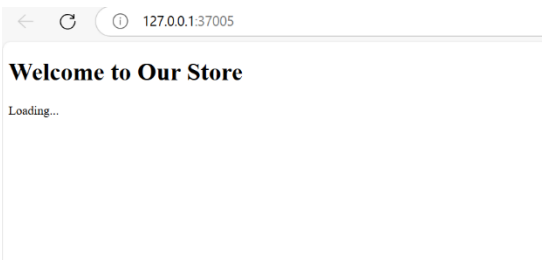
kubectl get pods

kubectl get svc

```
jegadeep@LAPTOP-EBJ40AHO:~/kubernetes/k8s$ kubectl get pods
NAME                       READY   STATUS    RESTARTS   AGE
backend-dfd8d5579-pbs5x    1/1     Running   0          12m
frontend-6cfd7c46-8dj8j    1/1     Running   0          39s
jegadeep@LAPTOP-EBJ40AHO:~/kubernetes/k8s$ kubectl get svc
NAME               TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
backend-service    ClusterIP   10.106.0.242     <none>        5000/TCP         12m
frontend-service   NodePort    10.106.226.159   <none>        3000:30172/TCP   12m
kubernetes         ClusterIP   10.96.0.1        <none>        443/TCP          3h33m
jegadeep@LAPTOP-EBJ40AHO:~/kubernetes/k8s$
```

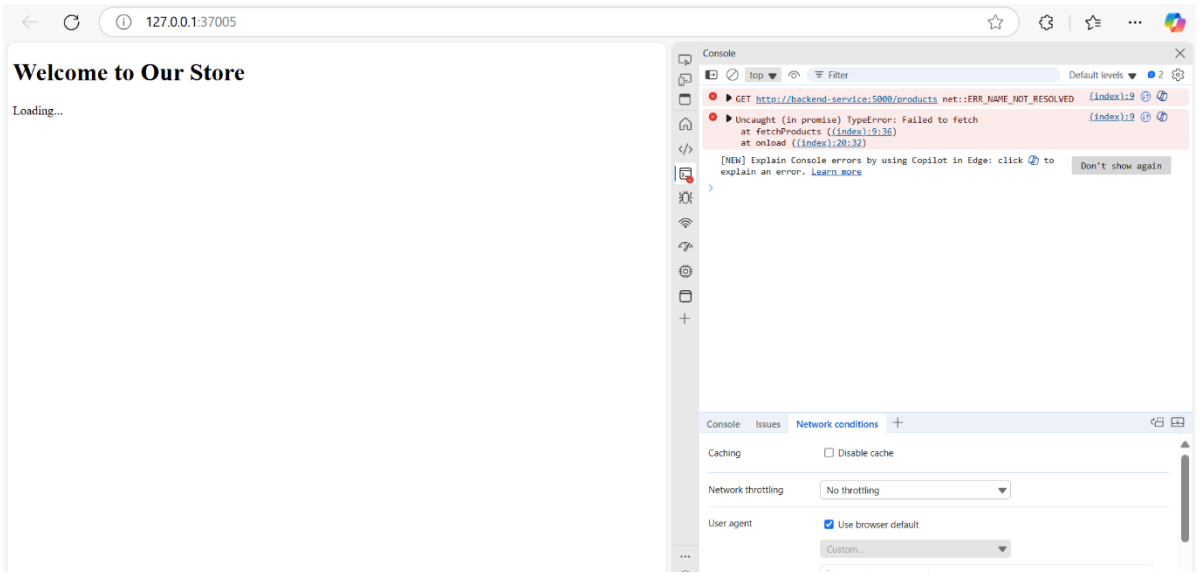
To test Frontend

```
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

```
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 —