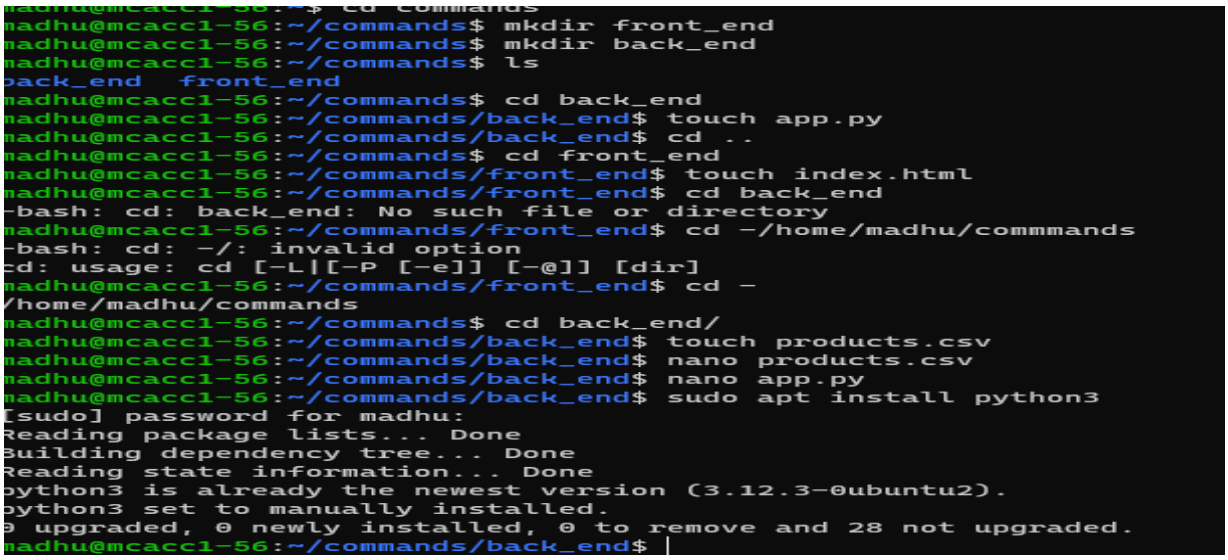
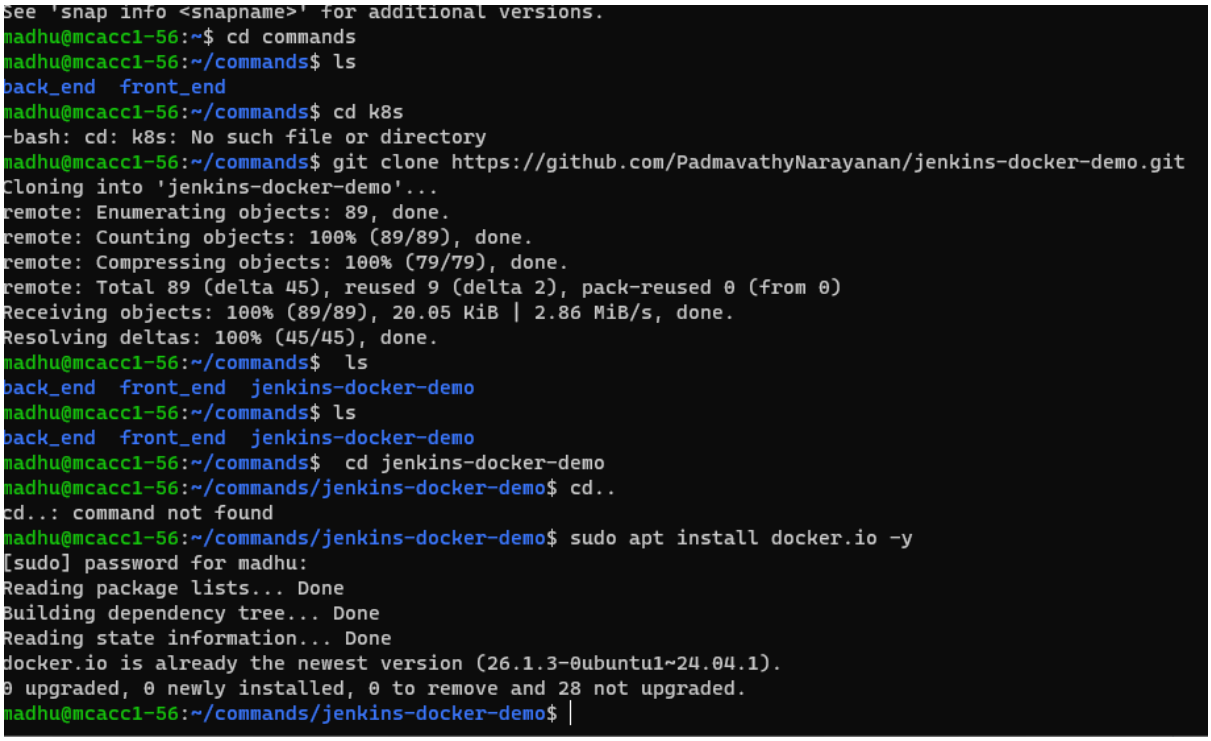
DEVOPS

DAY-3

To create a one Directory



To install python:



After Creating Python, Create an **Backend** and store the files,

App.py

```
#Stop Docker and Restart

systemctl stop docker
systemctl start docker

#Kill All Kubernetes, Minikube, and Docker Processes

pkill -f docker
pkill -f minikube
pkill -f kubectl
pkill -f containerd

#Cleanup Docker System
docker system prune -a --volumes -f

#commands to stop all ip's utilising 8080
sudo netstat -tulnp | grep ":8080"
sudo kill -9 <PID>
madhu@mcacc1-56:~/commands/jenkins-docker-demo/kubernetes$ kubectl delete all --all --force --grace-period=0
Command 'kubectl' not found, but can be installed with:
sudo snap install kubectl
madhu@mcacc1-56:~/commands/jenkins-docker-demo/kubernetes$ kubectl delete namespace kube-system --force --grace-period=0
Command 'kubectl' not found, but can be installed with:
sudo snap install kubectl
madhu@mcacc1-56:~/commands/jenkins-docker-demo/kubernetes$ sudo snap install ctl
[sudo] password for madhu:
error: snap "ctl" not found
madhu@mcacc1-56:~/commands/jenkins-docker-demo/kubernetes$ sudo snap install kubectl --classic
2025-03-21T04:14:37Z INFO Waiting for automatic snapd restart...
Automatically connect eligible plugs and slots of snap "snapd"                                                    /
```

Docker

```
ajay@Sparkajay:~/fullstack/backend$ cat Dockerfile
FROM python:3.11
WORKDIR /app
COPY requirement.txt .
RUN pip install --no-cache-dir -r requirement.txt
COPY . .

EXPOSE 7000
CMD ["python", "app.py"]
```

Requirements.txt:

```
ajay@Sparkajay:~/fullstack/backend$ cat requirements.txt
flask
pandas
```

docker-compose.yml

```
ajay@Sparkajay:~/fullstack/backend$ cat docker-compose.yml
version: '3.8'

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

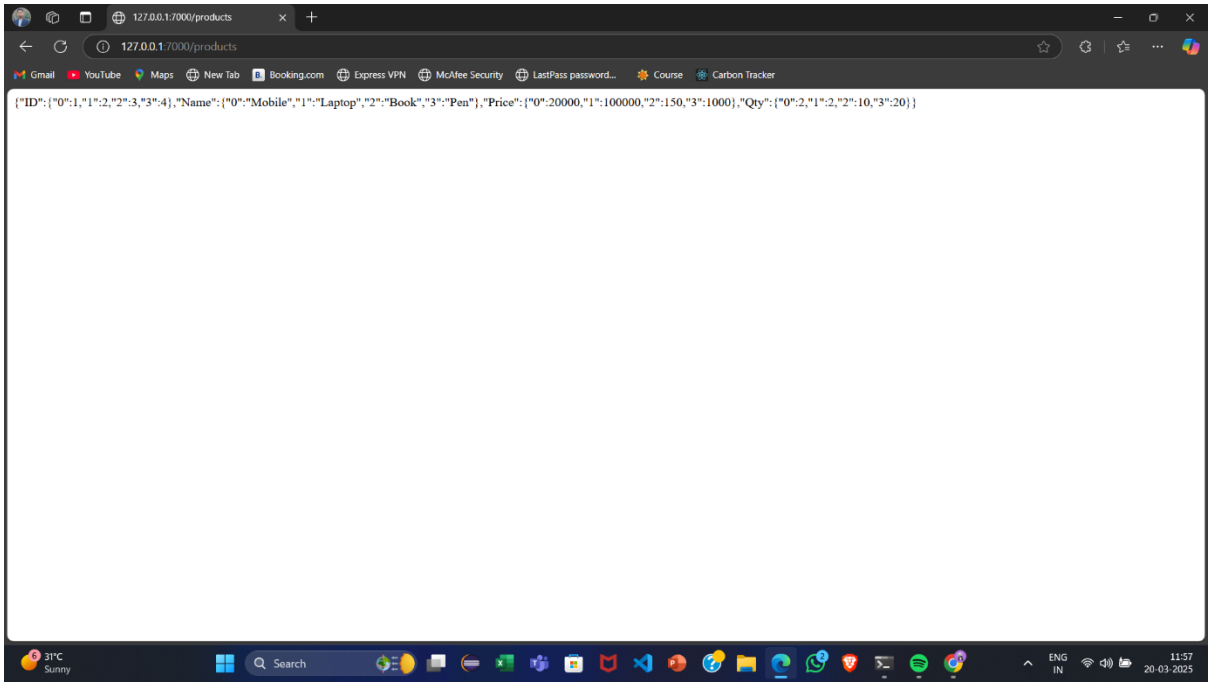
then build Docker and Run the Docker

```
ajay@Sparkajay:~/fullstack/backend$ sudo docker build -t test .
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  6.144kB
Step 1/7 : FROM python:3.11
 ---> 18c0f2265fd9
Step 2/7 : WORKDIR /app
 ---> Using cache
 ---> c74d97a78594
Step 3/7 : COPY requirements.txt .
 ---> Using cache
 ---> 21653e467847
Step 4/7 : RUN pip install --no-cache-dir -r requirements.txt
 ---> Using cache
 ---> 9227532eeb2e
Step 5/7 : COPY . .
 ---> Using cache
 ---> 2437319d1f68
Step 6/7 : EXPOSE 7000
 ---> Using cache
 ---> e744ca585d79
Step 7/7 : CMD ["python", "app.py"]
 ---> Using cache
 ---> 9980d41a1051
Successfully built 9980d41a1051
Successfully tagged test:latest
ajay@Sparkajay:~/fullstack/backend$ sudo docker run -d -p 7000:7000 test
12bda684ff5afd737a286a5cf51d08c9cab605b2fb5a5238ef08bfe99237749c
ajay@Sparkajay:~/fullstack/backend$ sudo docker logs 12bda684ff5afd737a286a5cf51d08c9cab605b2fb5a5238ef08bfe99237749c
 * 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
```

To see the Output, goto the website and put the URL:

then it will display the Json format in website



In **frontend**, Create the files like

index.html

```
ajay@Sparkajay:~/fullstack/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>
```

Dockerfile

```
ajay@Sparkajay:~/fullstack/frontend$ cat Dockerfile
FROM nginx:alpine
COPY index.html /usr/share/nginx/html/index.html
```

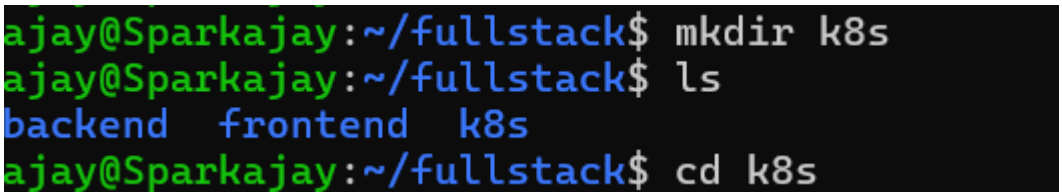
Build the Docker image in Frontend

```
ajay@Sparkajay:~/fullstack/frontend$ sudo docker build -t frontend:latest .
[sudo] password for ajay:
Sorry, try again.
[sudo] password for ajay:
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
 ---> 3aa76d667572
Successfully built 3aa76d667572
Successfully tagged frontend:latest
```

To create an k8s folder in fullstack using mkdir command:

For deployment use Kubernetes



Deployment.yaml (for Backend)

```
ajay@Sparkajay:~/fullstack/k8s$ nano deployment.yaml
ajay@Sparkajay:~/fullstack/k8s$ cat 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
```

Deployment.yaml (for Frontend)

```
ajay@Sparkajay:~/fullstack/k8s$ nano frontend-deployment.yaml
ajay@Sparkajay:~/fullstack/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
```

Service.yaml:

For **frontend-** nodeport **backend-**cluster IP

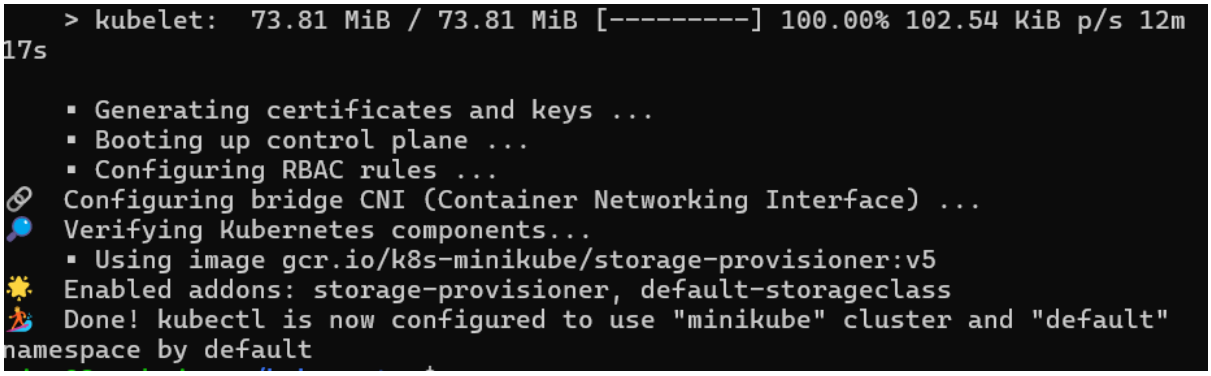
```
ajay@Sparkajay:~/fullstack/k8s$ cat service.yaml
apiVersion: v1
kind: Service
metadata:
  name: backend-service
spec:
  selector:
    app: backend
  ports:
    - protocol: TCP
      port: 7000
      targetPort: 7000
  type: ClusterIP

apiVersion: v1
kind: Service
metadata:
  name: frontend-service
spec:
  selector:
    app: frontend
  ports:
    - protocol: TCP
      port: 3000
      targetPort: 3000
  type: NodePort
```

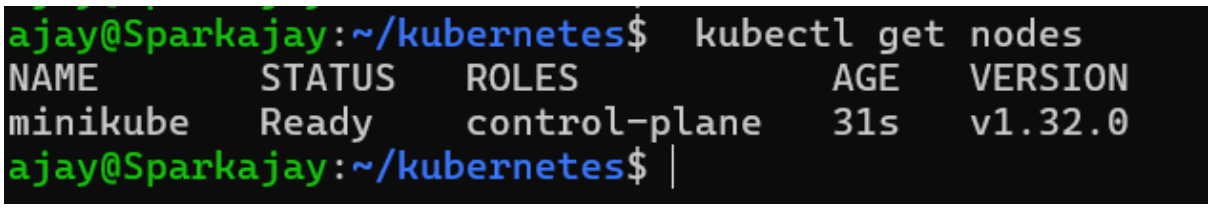
Configmap.yaml:

```
ajay@Sparkajay:~/fullstack/k8s$ cat configmap.yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: backend-config
data:
  DATABASE_FILE: "/backend/products.csv"
```

Minikube start:



Kubectl nodes:



To Build docker in Backend

```
ajay@Sparkajay:~/kubernetes$ cd backend/
ajay@Sparkajay:~/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
 ---> 859d4a0f1fd8
Step 2/6 : WORKDIR /app
 ---> Using cache
 ---> ae27c81ec929
Step 3/6 : COPY requirements.txt .
 ---> Using cache
 ---> 9f03d572763d
Step 4/6 : RUN pip install -r requirements.txt
 ---> Using cache
 ---> 18b868f8c6c4
Step 5/6 : COPY . .
 ---> Using cache
 ---> d85a885ee39d
Step 6/6 : CMD ["python", "app.py"]
 ---> Using cache
 ---> d0cff2fe7bb0
Successfully built d0cff2fe7bb0
```

Minikube for backend:

```
ajay@Sparkajay:~/kubernetes/backend$ minikube image load backend:latest
```

To Build Docker in Frontend:

```
ajay@Sparkajay:~/kubernetes$ cd backend/
ajay@Sparkajay:~/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
 ---> 859d4a0f1fd8
Step 2/6 : WORKDIR /app
 ---> Using cache
 ---> ae27c81ec929
Step 3/6 : COPY requirements.txt .
 ---> Using cache
 ---> 9f03d572763d
Step 4/6 : RUN pip install -r requirements.txt
 ---> Using cache
 ---> 18b868f8c6c4
Step 5/6 : COPY . .
 ---> Using cache
 ---> d85a885ee39d
Step 6/6 : CMD ["python", "app.py"]
 ---> Using cache
```

Minikube for frontend:

```
ajay@Sparkajay:~/kubernetes/frontend$ minikube image load frontend:latest
```

To create a Deployment file for Kubernetes for frontend, Backend, service.yaml,

```
ajay@Sparkajay:~/kubernetes/backend$ cd ..
ajay@Sparkajay:~/kubernetes$ cd k8s/
ajay@Sparkajay:~/kubernetes/k8s$ kubectl apply -f backend-deployment.yaml --validate=false
deployment.apps/backend created
ajay@Sparkajay:~/kubernetes/k8s$ kubectl apply -f frontend-deployment.yaml --validate=false
deployment.apps/frontend created
ajay@Sparkajay:~/kubernetes/k8s$ kubectl apply -f service.yaml --validate=false
service/backend-service created
service/frontend-service created
ajay@Sparkajay:~/kubernetes/k8s$ kubectl apply -f configmap.yaml --validate=false
configmap/backend-config created
ajay@Sparkajay:~/kubernetes/k8s$ kubectl get pods
NAME                       READY   STATUS    RESTARTS   AGE
backend-dfd8d5579-8rdzg    1/1     Running   0          63s
frontend-6cfd7c46-5txnb    1/1     Running   0          54s
ajay@Sparkajay:~/kubernetes/k8s$ kubectl get svc
NAME               TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
backend-service    ClusterIP   10.110.154.68   <none>        5000/TCP         87s
frontend-service   NodePort    10.98.250.114   <none>        3000:32434/TCP   87s
kubernetes         ClusterIP   10.96.0.1       <none>        443/TCP          12m
ajay@Sparkajay:~/kubernetes/k8s$ minikube service frontend-service --url
http://127.0.0.1:42597
!  Because you are using a Docker driver on linux, the terminal needs to be open to run it.
```

Create an pods and svc

```
ajay@Sparkajay:~/kubernetes/k8s$ kubectl get pods
NAME                       READY   STATUS    RESTARTS   AGE
backend-dfd8d5579-8rdzg    1/1     Running   0          63s
frontend-6cfd7c46-5txnb    1/1     Running   0          54s
ajay@Sparkajay:~/kubernetes/k8s$ kubectl get svc
NAME               TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
backend-service    ClusterIP   10.110.154.68   <none>        5000/TCP         87s
frontend-service   NodePort    10.98.250.114   <none>        3000:32434/TCP   87s
kubernetes         ClusterIP   10.96.0.1       <none>        443/TCP          12m
```

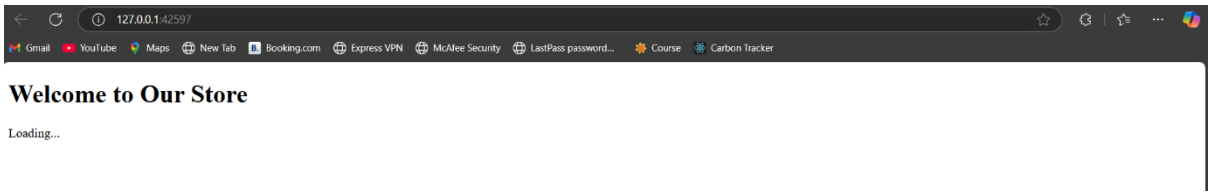
We can see the Output by using Curl command

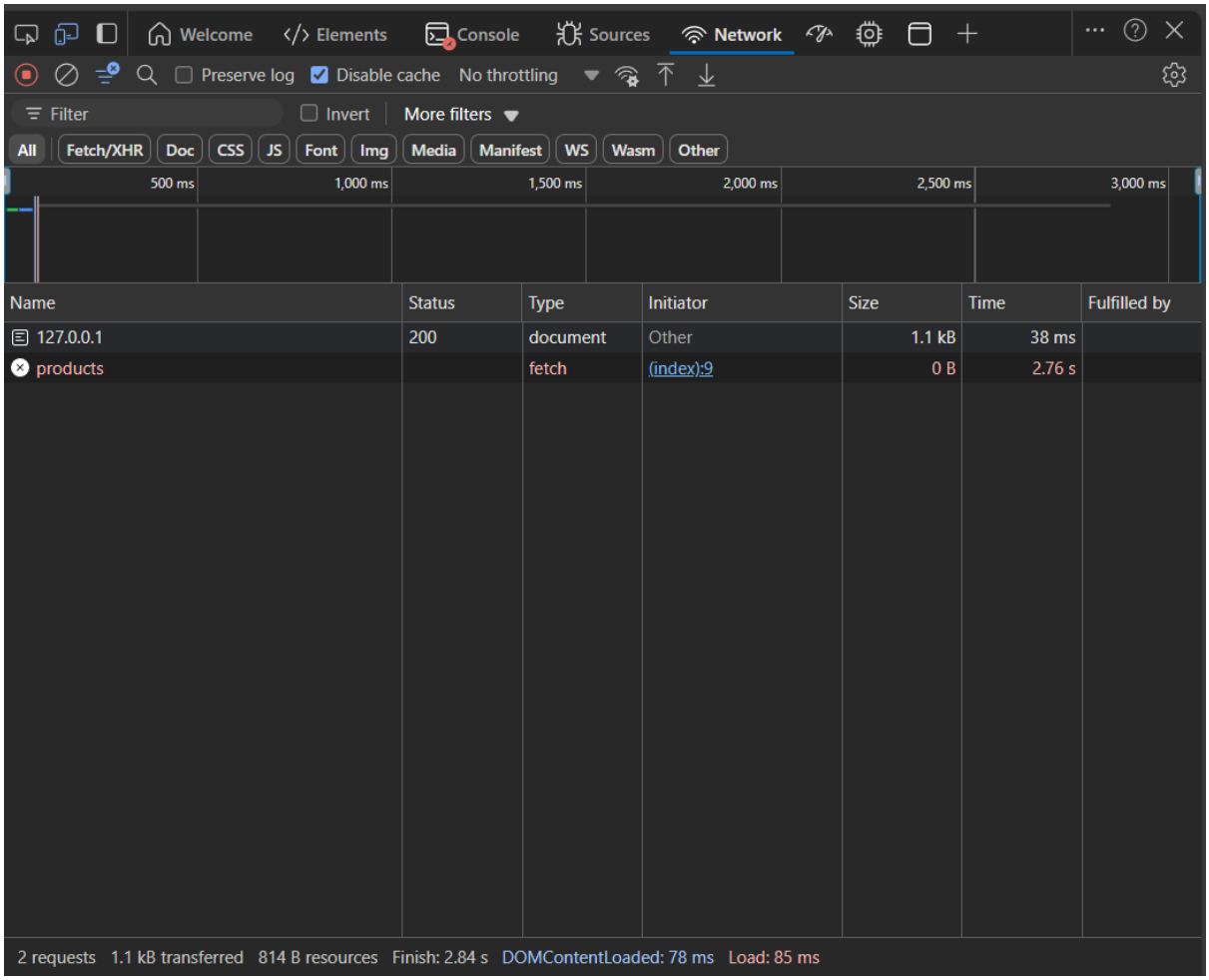
```
ajay@Sparkajay:~/kubernetes/k8s$ kubectl run test-pod --image=alpine --restart=Never -it -- sh
If you don't see a command prompt, try pressing enter.
/ # apk add curl
fetch https://dl-cdn.alpinelinux.org/alpine/v3.21/main/x86_64/APKINDEX.tar.gz
fetch https://dl-cdn.alpinelinux.org/alpine/v3.21/community/x86_64/APKINDEX.tar.gz
(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":"Headphone
s","price":49.99},{"id":4,"name":"Tablet","price":199.99}]
/ # exit
ajay@Sparkajay:~/kubernetes/k8s$
```

To run the frontend

```
ajay@Sparkajay:~/kubernetes/k8s$ minikube service frontend-service --url
http://127.0.0.1:42597
❗  Because you are using a Docker driver on linux, the terminal needs to be open to run it.
```

Output





Note : Since, we are expected this kind of output, because we are running this frontend in localhost.