

Day-4

Minikube start:

```
> kubelet: 73.81 MiB / 73.81 MiB [-----] 100.00% 102.54 KiB p/s 12m
17s

  ▪ 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! kubectrl is now configured to use "minikube" cluster and "default"
namespace by default
```

Kubectrl nodes:

```
~/$ kubectl get nodes
NAME               STATUS    ROLES    AGE   VERSION
minikube            Ready     control-plane  31s   v1.32.0
```

To Build docker in Backend

```
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:

```
~/kubernetes/backend$ minikube image load backend:latest
```

To Build Docker in Frontend:

```
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:

```
/kubernetes/frontend$ minikube image load frontend:latest
```

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

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

Create an pods and svc

```
~/kubernetes/k8s$ kubectl get pods
NAME                READY   STATUS    RESTARTS   AGE
backend-8rdzg       1/1     Running   0           63s
frontend-5txnb      1/1     Running   0           54s
~/kubernetes/k8s$ kubectl get svc
NAME                TYPE                CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
backend             ClusterIP           10.110.154.68    <none>         5000/TCP          87s
frontend            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

```
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","price":49.99},{ "id":4,"name":"Tablet","price":199.99}]
/ # exit
```

To run the frontend

```
~/kubernetes/k8s$ minikube service frontend-service --url 127.0.0.1:42597
```

are using a Docker driver on linux, the terminal needs to be open to run it.

Output

The screenshot shows a web browser at 127.0.0.1:42597 displaying a page titled "Welcome to Our Store" with a "Loading..." status. Below the browser, the Chrome DevTools Network tab is open, showing a timeline of two requests:

Name	Status	Type	Initiator	Size	Time	Fulfilled by
127.0.0.1	200	document	Other	1.1 kB	38 ms	
products		fetch	(index):9	0 B	2.76 s	

At the bottom of the Network tab, a summary bar indicates: 2 requests, 1.1 kB transferred, 814 B resources, Finish: 2.84 s, DOMContentLoaded: 78 ms, Load: 85 ms.

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